UNIVERSITY OF OKLAHOMA
GRADUATE COLLEGE

# COGNITIVE RADIO SOLUTION FOR IEEE 802.22

A DISSERTATION

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

DOCTOR OF PHILOSOPHY

By

YAHIA TACHWALI
Norman, Oklahoma
2010

COGNTIVE RADIO SOLUTION FOR IEEE 802.22

A DISSERTATION APPROVED FOR THE
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

BY

Dr. Hazem Refai, Chair

Dr. William Ray

Dr. James J. Sluss, Jr.

Dr. Joseph P. Havlicek

Dr. Samuel Cheng

# Acknowledgments

I would first like to thank my advisor, Dr. Hazem Refai, whose vision, support and technical guidance throughout this work have been invaluable. His advising style with always positive attitude gave me just enough freedom to endeavor into a new research area, thought me the fundamental elements of engineering research, and most importantly helped me build confidence. It has been a true privilege working with him.

I would like to thank the members of my doctoral committee, Dr. William Ray, Dr. James Sluss, Dr. Joseph Havlicek, and Dr. Samuel Cheng for their advice and suggestions in the early stages of this work.

I would like to thank Dr. Pramode Verma for all his advice and support over the years of my study at the University of Oklahoma.

I am also grateful to have the opportunity to work and interact with incredibly smart, talented and supportive students. Many thanks to Hassan and Maryam Moradi, William Barnes, Fadi Basma, Issam El-Arbaoui, and Ling Wang.

I have many fellow students to thank for their great friendship: Farnaz Zamani, Basel Salahieh, Samer Rajab, Mouhammed Akkoumi, Samer Shammaa, Moustafa Chmieseh Ussama Naal. They were always around for help and support during my difficult times.

Thanks to all members of OU Super-Computing Center for Education and Research (OSCER). Thanks for all their help and support. Especial thanks to Dr. Henry Neeman for all his technical advices, even outside his office working hours.

Finally, and most importantly, I would like to express my gratitude to my parents Madhat and Samar, and my beloved brothers Eyad and Basel, and my dear sister Dania.

They have been present throughout every step in my life, in spite of the thousands of miles that separate us. Their endless love and belief in me have been a great source of inspiration. For all their support in difficult times, and joy for every one of my accomplishments, this dissertation is dedicated to them.

# Table of Contents

# List of Tables

# List of Figures

# Abstract

Current wireless systems suffer severe radio spectrum underutilization due to a number of problematic issues, including wasteful static spectrum allocations; fixed radio functionalities and architectures; and limited cooperation between network nodes. A significant number of research efforts aim to find alternative solutions to improve spectrum utilization. Cognitive radio based on software radio technology is one such novel approach, and the impending IEEE 802.22 air interface standard is the first based on such an approach. This standard aims to provide wireless services in wireless regional area network using TV spectrum white spaces. The cognitive radio devices employed feature two fundamental capabilities, namely supporting multiple modulations and data-rates based on wireless channel conditions and sensing a wireless spectrum. Spectrum sensing is a critical functionality with high computational complexity. Although the standard does not specify a spectrum sensing method, the sensing operation has inherent timing and accuracy constraints.

This work proposes a framework for developing a cognitive radio system based on a small form factor software radio platform with limited memory resources and processing capabilities. The cognitive radio systems feature adaptive behavior based on wireless channel conditions and are compliant with the IEEE 802.22 sensing constraints. The resource limitations on implementation platforms post a variety of challenges to transceiver configurability and spectrum sensing. Overcoming these fundamental features on small form factors paves the way for portable cognitive radio devices and extends the range of cognitive radio applications.

Several techniques are proposed to overcome resource limitation on a small form factor software radio platform based on a hybrid processing architecture comprised of a digital signal processor and a field programmable gate array. Hardware reuse and task partitioning over a number of processing devices are among the techniques used to realize a configurable radio transceiver that supports several communication modes, including modulations and data rates. In particular, these techniques are applied to build configurable modulation architecture and a configurable synchronization. A mode-switching architecture based on circular buffers is proposed to facilitate a reliable transitioning between different communication modes.

The feasibility of efficient spectrum sensing based on a compressive sampling technique called “Fast Fourier Sampling” is examined. The configuration parameters are analyzed mathematically, and performance is evaluated using computer simulations for local spectrum sensing applications. The work proposed herein features a cooperative Fast Fourier sampling scheme to extend the narrowband and wideband sensing performance of this compressive sensing technique.

The précis of this dissertation establishes the foundation of efficient cognitive radio implementation on small form factor software radio of hybrid processing architecture.

# 1. Introduction

This introductory chapter identifies the motivation for this dissertation by describing the problem addressed and offering potential improvements to wireless communication. The scope is defined, achieved objectives are listed, and contributions to the research community are summarized. An outline of the dissertation is provided at the end of the chapter.

## 1.1. Motivation

The severe underutilization of the radio frequency spectrum reported by the Federal Communications Commissions (FCC) and the rapid advancements of wireless applications in recent years are synchronous events that have motivated the wireless research community to explore alternative solutions for improving spectrum utilization and satisfying bandwidth demand. Next-generation wireless communication devices will benefit from better wireless spectrum resource management and increased wireless transceiver flexibility that features the necessary frequency agility for efficient wireless spectrum utilization.

Cognitive radio based on software-defined radio platforms is considered a novel approach to improving the utilization of wireless resources. The configurability of software radio platforms and their spectrum sensing capability are essential features of next-generation wireless devices. These appealing attributes, however, require rigorous computational demands. As such, the realization of cognitive radio on small form factor platforms with limited computational and memory resources presents a number of design and implementation challenges, including high cost and considerable power consumption.

Cognitive radio transceiver configurability is necessary for the radio to operate in different communication modes based on available spectrum bandwidth and wireless channel condition, hence, leading to better spectrum utilization.

The speed and accuracy of spectrum sensing techniques are essential factors in cognitive radio network performance. The limitations imposed by computational complexity and shortened monitoring times impede the success of cognitive radio node spectrum sensing.

The forthcoming IEEE 802.22 air interface standard will enable wireless services in regional area networks by utilizing TV spectrum white spaces. The standard is recognized as the first based on a cognitive radio approach. Frequency agility, multimode transmission and spectrum sensing are critical functionalities in 802.22 compliant devices. Although the standard does not specify a particular spectrum sensing method, it does require the sensing operation to be performed within timing and accuracy constraints.

The successful implementation of cognitive radio on small form factor software radio platforms extends the range of cognitive radio networks applications and leads to increased availability and reliability of wireless services.

## 1.2. Scope

A number of software radio implementation platforms with a variety of processing architectures are available. The scope of this study includes design solutions for small form factor platforms with limited processing capabilities and memory resources. While the proposed design and implementation methods for a configurable cognitive radio transceiver have been tailored for a small form factor software radio provided by Lyrtech®, the compressive sensing technique (Fast Fourier Sampling FFS) adopted for spectrum sensing in this research is not based on a specific processing architecture. Rather, the compressive sensing technique features low processing complexity and a number of sample requirements, which are desirable for limited processing and memory resource platforms.

## 1.3. Objectives

This dissertation offers a foundation for developing fundamental cognitive radio capabilities, namely configurability and spectrum sensing, using a hybrid processing architecture in small form factor software radio. In this context, the objectives of this dissertation are presented in two parts:

- Configurable transceiver design, including
    - A demonstration of hardware reuse and task portioning techniques for developing wireless transceivers through case study implementation.
    - The realization of configurability in the main signal processing components of the cognitive radio transceiver, specifically the modulation and synchronization components.
    - The design of a switching mechanism between different communication modes
    - A framework for integrating cognitive radio components, e.g. digital communication system, switching mechanism, configurable synchronizer and spectrum sensing subsystems, in the targeted software radio platform.
- Compressive spectrum sensing, including

- The simulation and construct of conventional spectrum sensing techniques in targeted software radio platform used as a benchmark for evaluating spectrum sampling technique.
- The mathematical analysis of the impact of Fast Fourier sampling configuration parameters on its performance in spectrum sensing applications.
- An examination of the performance of Fast Fourier sampling in spectrum sensing through computer simulations.
- An improved compressed-sensing technique performance, using a proposed cooperative sensing scheme for narrowband and wideband sensing applications that better fits standard and resource constrains requirements.

## 1.4. Method

This study entails a mathematical analysis, computer simulations and implementation verification using hardware in loop simulation techniques. The development of a configurable cognitive radio transceiver uses system-level design tools to explore signal-processing requirements for each transceiver stage and to identify appropriate partitions for the digital signal processor and field-programmable gate arrays. The system-level design tools used in this work includes: Matlab® and System Generator for DSP by Xilinx ®.

A processor-specific development environment is then used for design implementation and verification on targeted processors. Texas Instrument® Code Composer and Matlab® Real-Time Workshop toolbox are used for digital signal processor code development, and Xilinx® Embedded Development Kit and System Generator for DSP are used for filed programmable gate array logic development.

The signal processing stages of the compressive spectrum sensing are analyzed mathematically to identify optimum algorithm configuration parameters. Computer simulations using OU Super-Computing Center resources are conducted to test the compressive sensing technique performance under a variety of wireless channel conditions.

## 1.5. Research Contributions

The contributions of this work are summarized as follows:

1- A proposed framework for developing a configurable transceiver on a small form factor platform using a combination of digital signal processor and field programmable gate arrays.

2- The design and implementation of a configurable synchronizer to support different symbol rates.

3- The design of a mode-switching mechanism.

4- The application of Fast Fourier Sampling technique for wireless microphone detections.

5- A simulation study of the Fast Fourier Sampling technique configuration parameters for wireless microphone detection under additive white Gaussian noise and Rayleigh fading channel.

6- A proposed wideband and narrowband cooperative sensing scheme based on Fast Fourier Sampling.

7- The mathematical analysis of frequency identification stage performance in FFS algorithm.

8- The mathematical analysis of optimal configuration parameters for frequency identification stage in FFS algorithm.

9- The mathematical analysis of optimal configuration parameters for a coefficient estimation stage in FFS algorithm.

## 1.6. Dissertation Organization

This dissertation is comprised of five chapters and is summarized as follows:

Chapter 2 provides a necessary foundational background. First it defines cognitive radio and software radio, and then explains the relation between the two. An overview of IEEE 802.22 wireless rural area network is presented and followed by a review of fundamental spectrum sensing techniques. Next a description of the compressed sensing technique—Fast Fourier Sampling—is provided.

In Chapter 3, the design of a configurable transceiver on small form factor software radio is presented. First, a demonstration of the design and implementation process of a digital transceiver system on the targeted platform is provided. Then, the implementation of a digital transceiver with configurable modulation capability is reported. The design and implementation of configurable symbol synchronizer to support various symbol rates follows. Subsequent to demonstrating the design and implementation of several configurable components in the cognitive radio transceiver, a switching mechanism for communication modes is proposed. The chapter concludes with a design proposal and implementation framework for a configurable cognitive radio transceiver.

Chapter 4 explores the use of Fast Fourier Sampling for spectrum sensing applications. First, a performance analysis for detecting a wireless microphone signal for IEEE 802.22 wireless network is given followed by a study of the impact of several configuration parameters on the wireless microphone detection performance. The chapter then extends the capabilities of this technique through two cooperative sensing schemes: one is tailored for narrowband spectrum sensing applications and another for wideband spectrum sensing applications. Next, a mathematical analysis of the performance of its signal processing stages, namely frequency identification and coefficient estimation, is reported. The analysis outlines performance boundaries in terms of the probability of success and estimation accuracy. The optimum choice of several configuration parameters for Fast Fourier Sampling technique is identified.

The research concludes in Chapter 5, and future research is proposed.

# 2. Background

This chapter details basic background material. Its goal is to provide the average reader with the necessary knowledge about the technologies and signal processing techniques later employed in this dissertation.

## 2.1. Software Radio and Cognitive Radio Technologies

Cognitive radio (CR) refers to an adaptive radio capable of changing operation parameters—frequency, modulation, power, and bandwidth, depending on the radio environment, user communication experience, and geo-location. The terms software radio (SR) and software defined radio (SDR) are used interchangeably throughout this work and refer to the implementation platform for cognitive radio. These platforms are based on highly configurable architectures to support flexible radio functionality needed for cognitive radio. In essence, SDR is considered the enabling technology for CR. This section provides formal definitions of CR and SDR and explains the relation between the two.

### *2.1.1. Software Radio Technology*

The term software radio was first introduced by Joseph Mitola in 1991 to describe a class of reconfigurable radios, i.e. "*radios that are substantially defined in software and whose physical layer behavior can be significantly altered through changes to its software"* [1] The Federal Communications Commission (FCC) defines software radio as "*a radio that includes a transmitter in which the operating parameters of the transmitter, including the frequency range, modulation type or maximum radiated or conducted output power can be altered by making a change in software without making any hardware changes that affect the radio frequency emissions*" [2]. Software radio allows different levels of reconfigurability, including flexible bandwidth and center frequencies, modulation format, protocols, and user functions. The highest level of reconfigurability is achieved by using an open software architecture for expansion and modifications.

The canonical software radio architecture is defined by Mitola [3] in two dimensions: the hardware architecture and functional architecture. The hardware architecture is comprised of a power supply, an antenna, a multi-band RF converter, data converter, processing engine(s), and memory that perform the radio functions and required interfaces. The functional architecture includes the channel processing stream, the environment management stream, and associate software tools. This architecture is presented as a framework for software radio development. The Joint Tactical Radio System program has taken software radio architecture a step further by defining an internationally endorsed open Software Communications Architecture (SCA) [4]. This open architecture defines the interaction between hardware and software elements in SDR.

Software defined radio provides many economical and practical advantages, specifically, the use of fewer expensive customized silicon and the benefit of declining prices in computing components. The computation power of software radio processing engines reduces the number of required RF components allowing digital signal processing algorithms to compensate RF imperfections. Although there are many advantages of SDR, the technology also presents a number of challenges and drawbacks. Significant power consumption is caused by high sampling rate data converters and massive signal processing demands. However, the continuous development and improvements of several enabling technologies [5-8] can reduce the drawbacks and increase the feasibility of software radio implementations. These enabling technologies appear in multiband antennas and RF conversion, wideband data converters, and the implementation of IF, baseband, and bitstream processing functions in different processing engines, such as general processors, digital signal processors, and field programmable gate arrays.

A number of silicon solutions are available for applying various digital signal-processing functions in software radio. Programmable, embedded signal processing systems generally use three types of advanced processing devices to execute digital signal processing: 1.) field programmable gate arrays (FPGAs); 2.) general purpose processors (GPP); and 3.) digital signal processors (DSP) [7, 8]. Table 2.1 summarizes the features of the major computing devices used in software defined radio implementations:

Table 2.1 Computing devices summary

| **DSP** | **FPGA** | **GPP** |
|---|---|---|
| □ Application specific, yet programmable<br>□ Parallel/sequential operations<br>□ Multiprocessing configurations<br>□ Arithmetic capabilities<br>□ Faster MAC (in a cycle)<br>□ Wider accumulator<br>□ Multi memory access in a cycle<br>□ multi address generator<br>□ less power than GPP | □ building block are connected by a cross bar connections<br>□ more parallelism and pipelining than DSP<br>□ mainly for prototyping and not for mass production<br>□ longer to program<br>□ match custom numerical precision<br>□ more energy efficient than DSP | □ faster than DSP<br>□ better software development tools<br>□ portable<br>□ backward compatibility<br>□ Application:<br>-user interface<br>-user application<br>-handling protocol stack |

An early SDR implementation was the US military SPEAKeasy project [9]. Several SDR implementations appeared later, namely the Wireless Information Transfer System (WITS) by Motorola [10, 11] the Digital Transceiver Subsystem SDR-3000 by Spectrum Signal Processing Inc [1], and SpectrumWare by MIT [12]. Aside from these SDR implementations, a number of SDR development kits to support the further advancements are currently available. These include the Universal Software Radio Peripheral USRP developed for the GNU Radio project [13] and the Small form factor SDR development kit by Lyrtech [14]. The latter is chosen during the course of cognitive radio development in this work. Appendix A briefly describes the architecture of this kit. The choice was chiefly based on three reasons. First, the kit supports the operation spectrum of IEEE 802.22. Second, the kit features heterogeneous processing architecture, which facilitates a balance in cost, power, performance, flexibility, and reliability. Third, the kit eases rapid prototyping by supporting several software development environments, including Matlab® [15], Code Composer Studio®[16], and Xilinx ISE® [17].

Many studies have reported the implementation of wireless waveforms using a variety of SDR development kits. Some focused on porting the implementation of existing wireless waveforms such as IEEE 802.11 [18], CDMA [19], GSM [20] or a combination of these in a single SDR solution [21]. The implementation in these studies tends to be a migration from waveform dependant architecture to generic hybrid SDR architecture. Other studies examined waveform implementation on a single processing engine (FPGA [22, 23], DSP [24] or GPP [20]) and highlight the unique features and benefits of utilizing this processing engine. However, due to limitations poised by each processing device, there is a trend to use hybrid SDR platforms based on a combination of different processing engines, e.g. GPPs, FPGAs, DSPs and application specific integrated circuits (ASICs). For example, in [25-27] combinations of FPGA and GPP have been used, whereas, in [28-31] FPGA and DSP have been used.

With extensive effort to build software radio solutions, researchers have extended software radio characteristics by incorporating intelligence and learning capabilities in proposed platforms. Such features are the characterizing elements of cognitive radio.

### *2.1.2. Cognitive Radio Technology*

The interdisciplinary study of cognition is addressed by [32]. This study is concerned with exploring general principles of intelligence through a synthetic methodology called learning by understanding. In this context, Simon Haykin [33] defines cognitive radio as " *Cognitive radio is an intelligent wireless communication system that is aware of its surrounding environment (i.e., outside world), and uses the methodology of understanding-by-building to learn from the environment and adapt its internal states to statistical variations in the incoming RF stimuli by making corresponding changes in certain operating parameters (e.g., transmit-power, carrier-frequency, and modulation strategy) in real-time, with two primary objectives in mind:*

- *highly reliable communications whenever and wherever needed;*
- *efficient utilization of the radio spectrum".*

Mitola introduced the term cognitive radio in 1999, describing an upgraded version of SDR where new waveforms are self-created. In his doctoral dissertation [3], Mitola describes how a cognitive radio enhances the flexibility of wireless services through a new language called the "Radio Knowledge Representation Language (RKRL)". This language is a set of signs and symbols that allows different internal elements of the radio to communicate with each other during the cognitive process. The cognitive process is illustrated by [33] as a cycle of three states:

- Radio-Scene analysis state:

In the receiver, the radio-scene analysis state performs the estimation of interference level and the detection of spectrum holes. The "*Spectrum hole is a band of frequencies assigned to a primary user, but, at a particular time and geographic location, the band is not used by that user*" [33]

- Channel Identification state:

  The receiver performs an estimation of channel state information and channel capacity.

- Transmit-power control and dynamic spectrum management state:

  This state is managed by the transmitter.

  Figure 2.1 [33] below shows the basic components of the cogitation cycle.

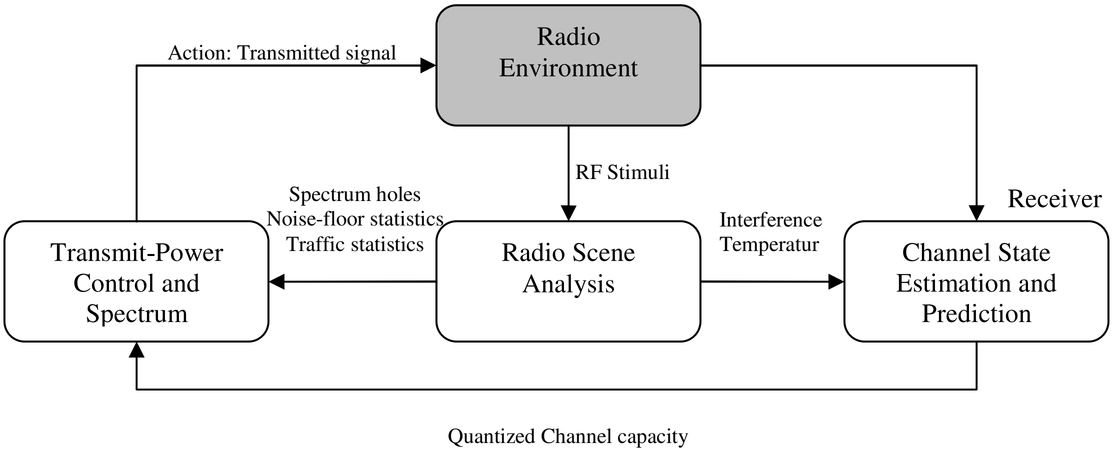

Figure 2.1 The basic cognitive cycle: Radio-scene analysis state, Channel identification state, and Transmit-power control and Dynamic spectrum management state

The basic cognitive cycle shown in Figure 2.1 is an abstraction of different levels of radio capability at which the cognitive radio can operate. Table 2.2 summarizes these levels. [3]

Table 2.2 Summary of cognitive radio capabilities

| Level | Capability | Comments |
|---|---|---|
| **0** | Pre-programmed | A software radio |
| **1** | Goal Driven | Choosing waveforms suitable to achieve a given goal based on the surrounding environment |
| **2** | Context Awareness | Knowledge of user activity |
| **3** | Radio Aware | Knowledge of network components, environment models |
| **4** | Capable of Planning | Analyze (level 2& 3) to determine goals (qos, power) following a prescribed plan |
| **5** | Conducts Negotiations | Settle on a plan with another radio |
| **6** | Learns Environment | Autonomously determines structure of environment |
| **7** | Adapts Plans | Generates new goals |
| **8** | Adapts Protocols | Proposes and negotiates new protocols |

Cognitive radio technology has attracted the attention of governmental, industrial, and institutional research. The Defense Advanced Research Project Agency (DARPA) sponsored the next Generation program (XG) to develop enabling technologies and system concepts along with novel waveforms to dynamically redistribute allocated spectrum in order to provide dramatic improvements in assured military communications in support of a full range of worldwide deployments [34]. The End-to-End Reconfigurability E2R Research group in Europe [35] aims to develop architectural design of reconfigurable devices and supporting system functions to offer an extensive set of operational choices in the context of heterogeneous systems for users, application and service providers, operators, and regulators. Also, the IEEE has established the development of the first wireless standard over the regional area network (WRAN) based on cognitive radio. The standard is named 802.22 and is described in section 2.2.

The research community has also addressed the development of cognitive engines for cognitive radio. The cognitive engine is an intelligent agent that manages cognition tasks in cognitive radio. The engine approaches can be based on different machine learning techniques, including genetic algorithms, markov models, neural network, fuzzy logic or based on game theory. A survey of the implementations artificial intelligence techniques in cognitive radio can be found in [36]. A reliable feedback is required in order to maintain a successful operation of cognitive engines and to convey information of available resources and performance metrics to cognitive engines. An essential form of feedback is achieved through spectrum sensing. Additional details are found in Section 2.3.

In summary, cognitive radio enjoys the advantages of software defined radio while improving spectrum utilization and wireless link performance. In fact, SDR paves the way for promising industry advancements such as high speed internet in rural areas. In contrast, cognitive radio suffers from typical software radio drawbacks and additional challenges. Presently, there are regulatory concerns regarding ways to avoid loss of control of wireless node behavior adaptation. Likewise, there are concerns relating to hardware selection, the design and implementation of decision, and learning.

### *2.1.3. The Relation Between Cognitive Radio and Software Radio*

While cognitive radio and software radio share common characteristics such as flexible operation and the ability to support various air interfaces, neither can replace the other. Historically, several wireless communication standards have evolved with no global planning. SDR has been considered an inherent solution for interoperability and multi-mode support problems exposed by incompatible wireless communication standards. However, the utilization of SDR in CR is not [37]. The configurability of SDR architecture also supports CR specific functionalities such as spectrum sensing. Several signal processing techniques have been proposed for spectrum analysis. However, they vary in their accuracy and complexity. Thus, the configurability in SDR allows the implementation of various spectrum-sensing techniques to support a number of sensing speed and accuracy requirements for CR sensing operation.

Table 2.3 summarizes a comparison among software radio, cognitive radio and traditional radio [38].

Table 2.3 Comparison among software radio, cognitive radio and conventional radio

| **Conventional Radio** | **Software Radio** | **Cognitive Radio** |
|---|---|---|
| - supports a fixed number of air interfaces<br>- configurability is decided at the design stage<br>- supported services are chosen at the design process | -supports multiple air interfaces dynamically<br>-interface with diverse systems<br>-provide a wide range of services. | -creates a waveform automatically<br>-performs session negotiation<br>-adapt to wireless channel conditions |

## 2.2. Introduction to IEEE 802.22

The IEEE 802.22 air interface (i.e., MAC and PHY) [39] was organized in November 2004 to provide the first wireless services based on cognitive radio technology in a wireless regional area network using unused TV spectrums. The standard specifies a fixed point to multipoint air interface. The topology of an 802.22-based network consists of a Base Station (BS) and a number of associated Consumer Premise Equipments (CPEs). The BS manages the control access and is based on a unique feature of distributed sensing. The CPEs reports the results of their local spectrum sensing, e.g. wide band spectrum sensing, real time spectrum allocation and acquisition and real time measurement dissemination. to the BS. The BS manages spectrum access and allocation to protect primary users (sometimes called incumbents) from interference, of which there are two types: Analog/Digital TV services and wireless microphones. Analog and digital television broadcasts depend on the format in the region of operation. In North America, for example, analog TV is based on NTSC, and digital television is based on ATSC. In addition to adaptation of the radio’s new characteristics, CPEs should also perform a real-time change of frequency.

WRAN applications raise unique challenges due to the wide area of coverage. One is the long delay spread, i.e. 20-50 us. Others include exploiting available frequency efficiently and flexibility in number of users with possible variable throughput, which impact the standard's physical and MAC layer design. Specifically in the case of physical layer, multi-carrier modulation using OFDMA is chosen, as it enables signal control in both time and frequency domains. The long delay spread requires the use of a cyclic prefix on the order of 40 us. Although the cyclic prefix is overhead, it is reduced with a relatively large number (2K) of carriers per channel. Also, strict frequency allocation requirements and link budget have encouraged standard use of channel bonding of contiguous and non-contiguous channels. However, having variable inter-carrier spacing may post additional implementation difficulties by having a variable system clock based on the number of bonded channels. Also, when a device begins synchronization, a super-frame structure (known as 6, not 7 or 8, MHz mode) provides then necessary frame information following the supper frame header. The header contains the preamble used for time synchronization, AGC setting and channel estimation. Table 2.4 is a summary of major IEEE 802.22 physical layer parameters.

The MAC layer defines the frame structures, network initialization, accessing, and spectrum management. Frames are categorized into upstream and downstream frames, and the base station manages them by including data payload, measurement activities, and a coexistence procedure. In network initialization, pre-defined channel (in time, frequency, or code) for CPE are not used to detect a BS. Therefore, when a CPE starts up, it scans available TV channels and builds a local spectrum occupancy map. The map aids the CPE in looking for base stations and conveys the map to available ones.

Spectrum management is governed by the BS and instructs the associated CPEs to perform a periodic in-band or out-of-band spectrum sensing. The CPEs are required to detect signals at very low SNR at an accuracy level controlled by the BS. Detecting low SNR signals is performed in non-coherent manner [40, 41]. CPE terminals in IEEE 802.22 standard may either experience or cause interference due to various types of possible coexistence scenarios, including TV and Wireless microphone, PLMR/CMRS, and self-coexistence, among others.

Table 2.4 Summary of major IEEE 802.22 physical layer parameters

| Parameter | Value |
|---|---|
| Frequency Range [MHz] | 54 – 862 (41-910 international use) |
| Channel Bandwidth [MHz] | 6, 7, 8 , allow fractional use and channel bonding up to 3 contiguous TV channels |
| Rates | 18 to 24 Mbps (Another source: Max: 72.6 Mbps-23 for 6 – 4.8Mbps |
| CPE TX power | 4W |
| Range [km] | 33-100 (33 at 4 watt CPE EIRP) |
| Propagation delay | 300 us |
| Primary user types | Microphones |
| Spectral efficiency [bit/s/Hz ] | 0.5-5 (average 3 corresponds to 18 Mbps in 6 MHzChannel) (min 0.81, max 4.03) bit average→18 Mbps |
| Throughput [bps] | (for 12 simultaneous users) 1.5 Mbps minimum downstream and 384kbps upstream |
| TPC dynamic range | 30 dB with 1 dB step |
| Modulations | QPSK, 16QAM, 64QAM |
| TX power | 4W EIRP for 33Km |
| MAC | OFDMA with 40 us cyclic prefix |
| FFT mode | 2K mandatory , 1 K/4K optional, 2/4/6 for channel bonding |
| Cyclic prefix mode | ¼, 1/8, 1/16, 1/32 |
| Duplex | TDD mandatory/ FDD supported |
| Network topology | Point to Multipoint Network |
| Number of subchannels | 48 |
| Coding rate | ½ ¾ 2/3 |
| Antennas | Omni directional for sensing and directional for BS communications |

To protect TV and wireless microphone users, i.e. primary users, the IEEE 802.22 sets sensing performance metrics, such as accuracy (the probability of detection and false alarm) and time constraints required by CPE spectrum sensing. The standard defines the reporting structure and employs a quiet period between transmissions to perform spectrum sensing at both fast- and fine-sensing time scales.

Fast spectrum sensing in the order of 1 ms is used to determine fine sensing range. In the event that the fast sensing method determines a channel range remains below a defined threshold, the BS may cancel a scheduled fine-sensing period. However, it should be noted that fine sensing operates in the order of milliseconds—25 ms in case of field-sync detection for ATSC—and scans each frequency channel to detect particular signatures or features of primary user transmitted signal. The 802.22 standard does not specify a spectrum sensing methodology. Additional details about proposed spectrum sensing methodologies are discussed in section 2.3.

The required total detection time for all primary user types, e.g. Analog/Digital TV and wireless microphones, is 2 seconds. The required sensing performance is defined by at least 0.9 probability of detection and at most 0.1 probability of false alarm. Table 2.5 lists timing requirements, and Table 2.6 lists the sensitivity requirements for 802.22 CPE spectrum sensors. Parameters listed in Table 2.5 and Table 2.6 are sometimes called Dynamic Frequency Selection Model [42].

The sensing framework is defined by the inputs and outputs, as well as the behavior of spectrum sensor [43]. Spectrum sensor inputs include Channel Number, Channel Bandwidth, Signal Type, Sensing Mode, and Maximum probability of false alarm. The outputs are Sensing Mode, Signal Type, Signal Present Decision, Confidence Metric, Field Strength Estimate, and Field Strength Accuracy. There are three sensing modes, the first of which is mandatory and requires only a present decision output; the others are optional and provide additional outputs.

The BS may perform distributed sensing so that all associated CPEs are not necessary to perform the same spectrum sensing procedure. IEEE 802.22 incorporates an efficient algorithm to synchronize overlapping cells by arranging the quit period for spectrum sensing between the overlapped cells. Protecting PLMR/CMRS users is an easier task, as they are geographically based and, thus, 802.22 BSs can maintain a spectrum usage table that classifies channels based on availability. The table is updated by the PLMR/CMRS system operator.

The wide coverage range of 802.22 causes the self coexistence to have larger impacts when compared to shorter wireless standard coverage ranges. The standard resolves this problem by having the BSs and CPEs transmitting coexisting beacons, i.e. beacons with time stamps. Upon detecting a beacon from a neighboring cell, the BS adjusts the start time of its superframe accordingly based on certain rules in a distributed manner. These rules ensure the convergence of synchronization between coexisted cells. The convergence time is a function of the cell size (number of users).

Table 2.5 IEEE 802.22 spectrum sensing timing requirements

| **Parameter** | **Value for all primary users** |
|---|---|
| Channel Availability Check Time | 30 sec |
| Non-Occupancy Period | 10 minutes |
| Channel Detection Time | ≤ 2 sec |
| Channel Setup Time | 2 sec |
| Channel Opening (Aggregate transmission time) | 100 msec |
| Channel Move Time (In-service monitoring) | 2 sec |
| Channel Closing (Aggregate transmission time) | 100 msec |

Table 2.6 IEEE 802.22 spectrum sensing sensitivity requirements

| | Analog TV | Digital TV | Wireless Mics |
|---|---|---|---|
| Sensitivity [dBm] | -94 | -116<br>(-11.3 for pilot carrier) | -107 dBm (200 kHz bandwidth)<br>-94 dBm (peak synchronization) |
| SNR [dB] | 1 | -21 | -12 |

## 2.3. Spectrum Sensing Techniques

Spectrum sensing is a fundamental task for cognitive radio operation and is defined as "*finding spectrum holes by sensing the radio spectrum in the local neighborhood of the cognitive radio receiver in an unsupervised manner*" [44]. Spectrum holes are unutilized sub-bands in the radio spectrum. The utilization of radio spectrum sub-bands is categorized as follows:

1) White spaces: sub-bands free from RF interference except from natural (thermal with Gaussian distribution) or artificial (impulsive generated by motor vehicles in urban areas [44]) noise.
2) Gray spaces: sub-bands partially occupied by interference and noise.
3) Black spaces: sub-bands occupied by communication signal combined with interference and noise signals.

A comprehensive spectrum sensing task detects spectrum holes, estimates the available spectrum bandwidth in each spectrum hole, estimates the spatial directions of interfering signals, and classifies detected signals.

Many spectrum sensing techniques have been proposed in literature. Comprehensive reviews of recent spectrum sensing techniques and dynamic spectrum access can be found in [45-48]. Spectrum sensing techniques applied in IEEE 802.22 can generally be categorized into two main groups: blind sensing techniques and signal specific signal techniques [43]. The sensitivity measurement indicated in this section refers to the lowest SNR value under which the sensing technique is able to achieve a false alarm of 0.1 and detection probability of 0.9.

### *2.3.1. Blind Sensing Techniques*

Blind sensing techniques are not reliant on signal features. Instead, they are generally fast but suffer severe performance deterioration in low SNR signals. This section assesses blind sensing techniques, including the energy detector and Eigenvalue Sensing Technique, as well as tunable wavelet filters.

- <u>Energy Detector</u>

The energy detector is a well-known blind sensing technique that has been investigated thoroughly in literature [49-61]. The energy detector estimates received signal power as defined in equation 2.1:

$$P = \frac{1}{N}\sum_{n=1}^{N} y(n)y^{*}(n) \qquad 2.1$$

Thanks to Parseval's equality, power can be estimated in frequency domain using periodograms. Figure 2.2 shows an energy detector based on FFT.

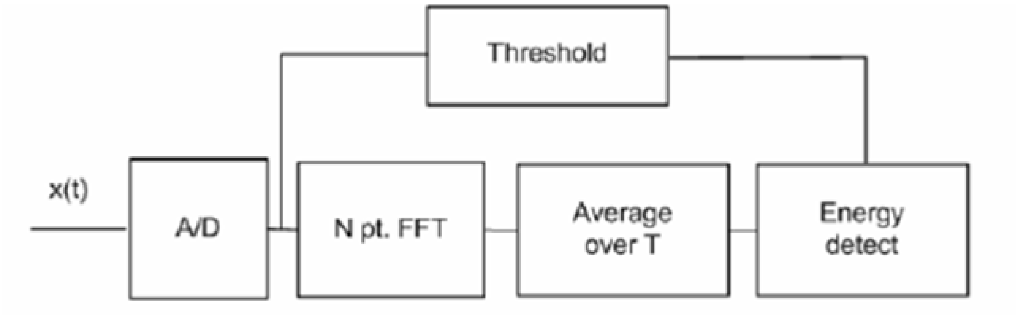

Figure 2.2 Energy detector block diagram

For large *N*, the central limit theory can be applied in equation 2.1, and the estimated power *P* will have a normal distribution $\mathrm{N}\left(P_S + P_N, \frac{(P_S + P_N)^2}{N}\right)$, where $P_S$ is the received signal power, and $P_N$ is the noise power. The energy detector compares the estimated power *P* with a predefined threshold $\gamma$ previously selected to meet the false alarm probability target (Neyman-Pearson method) as shown in equation 2.2.

$$\gamma = P_N\left[1 + \frac{Q^{-1}(P_{FA})}{\sqrt{N}}\right] \qquad 2.2$$

where *Q(x)* is the tail probability of a normalized zero-mean Gaussin random variable and is given by equation 2.3.

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-\frac{x^2}{2}} dx \qquad 2.3$$

Note that noise power needs to be estimated to calculate the detection threshold. The probability of detection can be calculated as shown in equation 2.4 subsequent to choosing a threshold based on acceptable false rate.

$$P_D = 1 - Q\left(\frac{\sqrt{N}}{P_s + P_N}\left(P_S + P_N - \gamma\right)\right) \qquad 2.4$$

The quality of energy detector performance is chiefly governed by two factors: the number of samples *N* and the accuracy of noise estimation. The larger the number of samples, the better the performance of the energy detector. The uncertainty in the power noise estimation limits the detector to an SNR little less than zero and makes it an unreliable detector at lower SNR values. It has been shown that the energy detector is not significantly impacted by multipath [53] when used for relatively wide channel bandwidth (6 MHz) as is the case in 802.22. Table 2.7 [62] shows a list of the minimum SNR required for energy detection to properly operate under various noise uncertainties of $\Delta$.

Table 2.7 Minimum SNR required for energy detection operation under different noise uncertainty

| Sensing Time | $\Delta$=0dB | $\Delta$=0.5dB | $\Delta$=1dB |
|---|---|---|---|
| | Required SNR (dB) | | |
| 0.2 ms | -11 | -5 | -2.5 |
| 1 ms | -15 | -6 | -3 |
| 5 ms | -18 | -6 | -3 |

- Eigenvalue-based detector

The foremost limitation of energy detectors is required accurate estimation of noise power. One blind spectrum sensing technique proposed to solve this problem is Eigenvalue-base sensing [43], which calculates the autocorrelation function of the received signal. The correlation matrix is then transformed by a whitening matrix to correct the frequency response of the receiver filter.

The Eigenvalues of the transformation matrix output are used to formulate different decision parameters. The detection algorithm can be described mathematically as follows:

1- Define the impulse response of a filter *f(k),k=0,1,…,K,*

2- The output of the filter is found by convolving the filter impulse response with the received signal *x(n)*, as shown in equation 2.5

$$\tilde{x}(n) = \sum_{k=0}^{K} f(k)x(n-k),\ n = 0,1,... \qquad 2.5$$

3- Formulate a signal ***x(n)*** based on $\tilde{x}(n)$ and a predetermined smoothing factor *L* (with suggested value for *L* being 10 [62]) as shown in equation 2.6

$$\boldsymbol{x}(n)=[\ \tilde{x}(n)\ \ \tilde{x}(n-1)\ \dots\ \tilde{x}(n-L+1)\ ]^T \qquad 2.6$$

4- Compute the covariance matrix using equation 2.7

$$R(N_s) = \frac{1}{N_s}\sum_{n=0}^{N_s-1}\mathbf{x}(n)\mathbf{x}^{\mathrm{H}}(n) \qquad 2.7$$

5- Calculate the whitening matrix *Q* as follows:

- formulate *H* matrix *L*×*(K+1+(L-1)M)* as shown in equation 2.8:

$$H = \begin{bmatrix} f(0) & \dots & \dots & f(K) & 0 & \dots & 0 \\ 0 & \dots & f(0) & \dots & f(K) & \dots & 0 \\ & & \dots & & \dots & & \\ 0 & \dots & \dots & \dots & f(0) & \dots & f(K) \end{bmatrix} \qquad 2.8$$

- Let $G=HH^H$ then decompose the matrix into $G=QQ^H$ where *Q* is the *LxL* Hermitian matrix.

6- Transform the sample covariance matrix using equation 2.8

$$\tilde{R}(N_s) = Q^{-1}R(N_s)Q^{-H} \quad 2.9$$

7- Calculate the Eigenvalues of $\tilde{R}(N_s)$

Based on Eigenvalues calculated in step 7, a number of decision parameters can be obtained, including the ratio between the largest and smallest Eigenvalues, referred to as Maximum Minimum Eigenvalue (MME) detection. Another parameter is the ratio of the average signal power to the smallest Eigenvalue, referred to as Energy with Minimum Eigenvalue (EME) detection. Both parameters are measurements of "non-whiteness" of the sensed spectrum. A similar technique based on the covariance matrix is used to detect wireless microphones signals [43]. The pure noise received signal is expected to have a diagonal matrix, although two different parameters are used for this technique. The first is the ratio of the sum of all covariance matrix elements magnitudes to the sum of the diagonal elements magnitudes. The second is the ratio of the sum of the magnitude squared of all elements to the magnitude squared of the diagonal elements.

The performance of the Eigenvalue-based sensing method is governed by the choice of smoothing factor L and the bandwidth of the baseband filter [62]. The larger the value of *L*, the longer the sensing time and the better the detection performance achieved.

Table 2.8 and Table 2.9 show the minimum SNR value required for the received signal to be detected using an Eigenvalue based on the method with maximum probability of false alarm of 0.1 and minimum detection probability of 0.9 for a single channel and three consecutive channels, respectively [62]:

Table 2.8 Required SNR for DTV signal detection (single channel)

| Method | 4 ms | 8 ms | 16 ms | 32 ms |
|---|---|---|---|---|
| MME | -11.6 dB | -13.2 dB | -15 dB | -16.9 dB |
| EME | -10.5 dB | -12.1 dB | -14 dB | -15.8 dB |

Table 2.9 Required SNR for DTV signal detection (three consecutive channel)

| Method | 4 ms | 16 ms |
|---|---|---|
| MME | -17.5 dB | -20.9 dB |
| EME | -15.6 dB | -19.1 dB |

Similarly, the Eigenvalue-based detection method is proposed to detect the wireless microphone signals in 802.22 spectrum sensing application [43]. The sensing time based on covariance matrix technique is found to be approximately 10 ms with sensitivity roughly around -23 dB SNR [43].

- Tunable wavelet filters

Tunable wavelet filters are also used as a blind sensing technique by changing their center frequency and bandwidth. This tuning feature allows the estimation of a multi-resolution power spectral density, which is used to test the "non-whiteness" of the sensed spectrum [63-65].

### *2.3.2. Signal Specific Sensing Techniques*

The signal specific sensing technique is another sensing category which relies on prior knowledge of specific features of the received signals. The focus of this section is to provide a brief description of such techniques that could be implemented as part of the IEE802.22 standard.

- Match filtering and correlations

Match filtering is known as the optimum signal detection method when the transmitted signal is known [66]. The main advantage of match filtering is its simplicity and the small amount of time required to achieve a certain probability of positive detection. The required number of samples used for match filtering is in the order of (1/SNR) for a given probability of false alarm at low SNR values [67]. However, the main disadvantage of match filtering is the overhead of preprocessing the received signal prior to signal detection; including down-converting and pulse shaping.

A practical way to realize match filtering is to use correlators. Correlation-based detectors can be used in IEEE 802.22 to detect different features of the 802.22 signal, one such being the Data Field Sync in ATSC Data Field based on well known PN sequences (511-PN and PN63) [68]. This detector operates on the received baseband signal by correlating it with a known sequence over 24.2 ms—the duration of an ATSC Data Field. This sensing time corresponds to over $10^5$ samples at 6 MHz sampling. The sensitivity of this technique can be improved by extending the sensing time over an increased number of samples so that combining results from multiple Data Field Sync signals is possible. However, this combination can be challenging over a multipath fading channel since the radio channel can change its characteristics if the sensing time exceeds 24.2 ms, which is the time required for a single Data Field Sync signal. To avoid this one must combine the correlation output peaks of multiple data fields. The sensing time for 16 Data Fields can achieve a sensitivity of -14 dB SNR [43].

The correlation can also be performed in the frequency domain. Power spectral density (PSD) of the received signal can be used to detect ATSC and NTSC signals. PSD estimates on non-uniform sampling may emphasize specific spectral features. This sensing technique compares the output of PSD to a pre-stored PSD of the signal of interest.

- <u>Detecting Pilot Signals</u>

Another signal feature detected in 802.22 primary user signals is the ATSC signal pilot, where frequencies can be either 309440.6 Hz or 328843.6 Hz in the baseband and are measured from the lower frequency edge of the TV channel. Pilot frequencies are accurate to within ± 10Hz. It is important to mention that the pilot is a narrowband signal and therefore more prone to fading as described by Rayleigh distribution rather than AWGN channel.

One technique for detecting pilot signals is using the periodogram after converting the midpoint between two pilot frequencies to DC and low-pass filtering with cutoff frequency of 25 kHz. The output is passed through a periodogram, and the maximum output value is used as a detection parameter. Studies have shown that this technique achieves a sensitivity of -25 dB SNR when averaging four sequential values of the periodogram output [43]. The technique performance can be improved by averaging a larger sequence of periodograms.

Tracking pilot frequencies using digital PLLs in a version of the Costas loop is yet another technique that relies on the existence of pilot signals in ATSC signals [43]. Two PLL trackers are initialized 30 kHz above and below the nominal pilot signal value, and the difference of the final frequency estimates of both PLLs is used as a decision parameter. This difference should be above a given threshold in order to decide the presence of ATSC signal. This technique requires 50 to 75 ms and achieves a sensitivity of -12 to -14 dB [43].

- Gaussianity test using higher order statistics

Another technique shifts the pilot frequency carrier to base band and then passes it through a low-pass filter, finding its frequency domain representation using 2048 point FFT. This allows the use of the same FFT module as for OFDM in the transceiver to examine Gaussianity, which is tested by high order statistics, namely second and third moments of the received signal power. This technique relies on the assumption that the received noise power follows the normal distribution. Hence, a pure noise signal will have zero high order cumulants [69].

- Cyclostationary sensing technique

Another signal specific spectrum sensing technique that gains wide interest for 802.22 spectrum sensing applications is the cyclostationary sensing technique [70-72]. This technique relies on the built-in periodicity in transmitted signal to distinguish them at random noise signals. In other words, the statistics of cyclostationary signals, in particular the mean and autocorrelation, exhibit periodicity. Similar to techniques previously discussed, the received signal is down converted with the pilot signal converted to DC and then filtered by low pass filter. The spectral correlation function (SCF) is then calculated from the sampled signal, also termed as cyclic spectrum density (CSD). The SCF is a function of two parameters, namely the digital frequency $k$ and the cyclic frequency $\alpha$ as seen in the SCD function definition in equation 2.10:

$$CSD(k, \alpha) = E\left[Y\left(k + \frac{\alpha}{2}\right)Y^{*}\left(k - \frac{\alpha}{2}\right)\right] \qquad 2.10$$

where $\alpha$ is the cyclic frequency. The CSD of AWGN is zero for $\alpha \neq 0$, and the SCD function outputs peak values when the cyclic frequency is equal to the fundamental frequency of a transmitted signal. The cyclic frequency can be assumed to be known [73, 74] or can be extracted as an identification feature of the transmitted signal [75]. This technique can be used to detect certain features in received signals, such as modulation type, symbol rate, and presence of interferers. The cyclostationary-based technique performs well in low SNR; however, it is a relatively complex technique since it is based on two-dimensional transformations. Its implementation complexity increases by N$^{2}$

complex multiplications due to the need to compute cross correlation of the *N* point FFT outputs. Figure 2.3 shows a block diagram of SCD based detector.

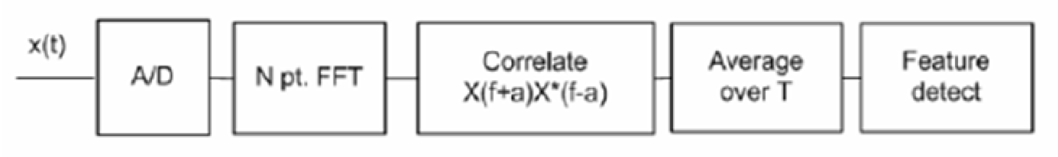

Figure 2.3 Cyclostationary detector block diagram

A typical method for validating and evaluating spectrum sensing techniques is through simulation. Thus, defining simulation methodologies is important to perform a fair comparison between sensing techniques. Many spectrum sensing techniques have been evaluated by way of simulation models by 802.22 working group [76]. The simulations use fifty captured ATSC signals in New York and Washington DC to incorporate real world multi-path in simulations. The signals are collected at an IF frequency of 5.38 MHz and are sampled at a rate of 21.5 MHz. Noises are added to the captured signals to simulate lower SNR signals. The simulations pre-determine the detection threshold in order to achieve a 0.1 probability of false detection. The probability of detection is then averaged over all captured signals. The simulation results of spectrum sensing techniques performance are generally presented as probability of misdetection (1- probability of detection) versus SNR at different tolerances of noise power estimate accuracy.

### *2.3.3. Spectrum Sensing Implementations*

There are a limited number of cognitive radio implementations presented in literature. A key reason for the limited number of implementations is the complexity of the current spectrum sensing methods and their strict requirements in terms of sampling frequency and processing power.

Some implementations perform a multi-resolution spectrum sensing technique originally introduced in [77, 78]. Recent work presented in Globecom07 by Motorola labs has shown a cognitive radio prototype capable of performing channel allocation within 20 MHz bandwidth—fine sensing—using spectrum correlation density technique [79]. Georgia Tech has suggested and implemented a multi-resolution spectrum sensing technique based on wavelet tunable filters [64, 65, 78]. Researchers at Berkeley Wireless Research Center have studied spectrum sensing implementation consideration for cognitive radios in [80, 81].

## 2.4. Fast Fourier Sampling Overview

In digital systems, signal spectrum can be decomposed into an equi-spaced collection of frequencies (orthogonal functions), as shown in equation 2.11. The coefficients $a_k$ of these frequencies are retrieved using Discrete Fourier Transform (DFT), as shown in equation 2.11.

$$x(t) = \frac{1}{\sqrt{N}} \sum_{k=1}^{m} a_k e^{2\pi i \omega_k t / N}, \qquad t = 0,1,2,..., N-1$$

$$a_k = X(\omega_k) = \frac{1}{\sqrt{N}} \sum_{t=0}^{N-1} x(t).e^{-2\pi i \omega_k t / N}, \qquad \omega_k \in (0,1,2,..., N-1) \tag{2.11}$$

where N is the number of samples.

The complexity of this transform is O(N2). The computational complexity is reduced O(NlogN) using FFT algorithm, and the number of samples is a power of two. While FFT provides a dramatic complexity reduction, the complexity of calculating a large number of N remains a challenge in current computing systems that operate under severe power and chip real-estate limitations, including wireless mobile devices. Nevertheless, FFT appears to be inefficient when detecting only m << N frequencies in the received signal spectrum. In this case, it is possible to use fewer sample numbers to detect the most m "energetic" frequencies in the spectrum, and then approximate the spectrum using a non-uniform IFFT, as shown in equation 2.12.

$$y(t) = \frac{1}{\sqrt{N}} \sum_{k=1}^{m} a_k e^{2\pi i \omega_k t / N} \qquad 2.12$$

One computational algorithm proposed to calculate this approximation is called Fast Fourier Sampling (FFS). The algorithm was first presented in [82] and then enhanced by [83]. FFS produces spectral approximation using time and storage of: m.poly(ε-1,log(δ-1),log(N)) where ε>0 is approximation quality factor, δ>0 is probability of failure. This section provides a brief overview of the algorithm based on the tutorial published in [84].

The algorithm is performed in three stages iterated for a fixed number of times:

1- Identification stage

2- Estimation stage

3- Updating the set of K frequencies and coefficients

The Identification stage constructs a list of up to K<N frequencies that are likely to carry a significant amount of received spectrum energy based on K number of samples that are randomly accessed. K is an integer multiple of m. The larger the value of K, the better the spectrum approximation at the expense of more computational complexity. The Identification is achieved in two stages: first by "Sample Shattering" and second by "Bit Testing." Sample shattering aims to separate significant frequencies and pass them through a sub-band decomposition filter bank. The separation is performed by time dilation of signal samples by a factor $\sigma$, which results in frequency permutation by

factor $\sigma^{-1}$ where $\sigma^{-1}$ is the multiplicative inverse of $\sigma$ that satisfies $\sigma . \sigma^{-1}$ mod(N) =1. This argument can be expressed mathematically in equation 2.13:

$$y(t) = x(\sigma t) \qquad \text{for all t} \qquad \Leftrightarrow \qquad Y(\omega) = X\left(\sigma^{-1}\omega\right) \qquad \text{for all}\, \omega \qquad 2.13$$

The bit test examines the output of filter banks to determine the major K frequencies (or less) at the filter bank output and detects the dominant frequencies using two frequency filters; $g_{\text{even}}$ and $g_{\text{odd}}$.

The Estimation stage receives a list of up to K frequencies and estimates the corresponding coefficients for these frequencies. The estimation of $c_\omega$ is performed by demodulating the signal by $\omega$ so that we can instead estimate the zero frequency. The signal is then randomly dilated to separate frequencies at approximately zero, i.e. the time dilation fixes zero frequency and shuffles other frequencies. After separation, a low pass filter is used to pass the zero frequency, and the output is considered an estimation of the frequency coefficient cw. This operation can be expressed mathematically using equation 2.14.

$$c_\omega = \sqrt{N} e^{2\pi i\omega / N} \sum_{j=0}^{K-1} h(j) x(t - \sigma j) e^{2\pi i(\omega\sigma K / N) j / K} \qquad 2.14$$

The third stage adds the new K (or less) findings of $\omega, c_\omega$ to the previous findings to obtain a total of 2K pairs (or less) of $\omega$ and $c_\omega$ and then shrink the figure back to K or (less) by retaining the frequencies with the largest coefficients. The algorithm then repeats the three stages for a predetermined number of times. Figure 2.4 shows a block diagram of the overall Fourier Sampling algorithm.

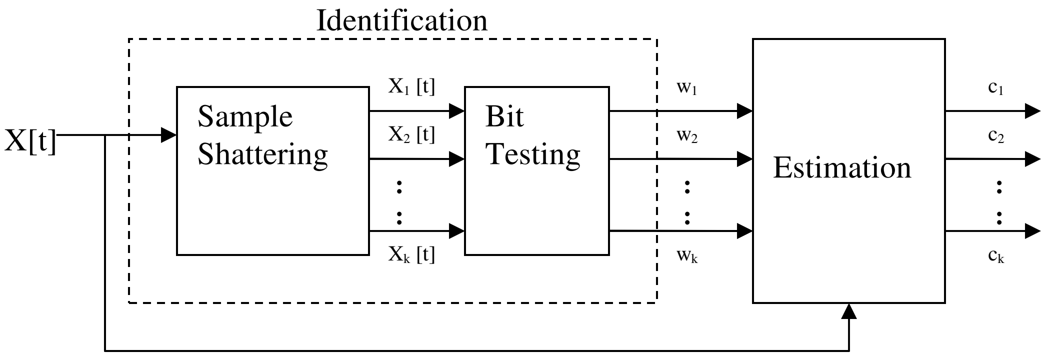

Figure 2.4 Fourier Sampling algorithm block diagram

The performance of this algorithm is evaluated by two parameters. The first is the probability of algorithm success in finding the most energetic frequency components in the received spectrum. The second is the quality of approximation when compared with the optimum approximation that can be achieved. The performance of this algorithm can be controlled through various parameters, including the number of random samples collected and number of iterations performed at each stage [84].

The algorithm guarantees the number of samples, running time, and success rate [80]. However, it is important to investigate the capabilities and limitations of the Fourier Sampling algorithm in spectrum approximation. Advantages of Fourier Sampling algorithm in Spectrum Sensing applications include:

1- Collected samples depend on random choices and not the signal or algorithm progress; therefore, it is possible to decide in advance (prior to algorithm execution) the samples to be collected.

2- The computational resources required by the algorithm are approximately proportional to the number of frequencies used in approximation m and not by the signal length N; therefore, the Fast Fourier Sampling can be potentially exponentially faster than FFT.

3- The output approximation is comparable to amount of noise; however, the algorithm can tolerate a relatively low SNR below which the algorithm fails to provide valid approximation. For example, the algorithm can recover a single tone in signal of length N=222 (around 4 million) 90% of the time at an SNR of -15 dB using only 1% of N samples. More samples, such as 4% of N, can achieve the same performance just lower than -18 dB [83]. See Figure 2.5 [84].

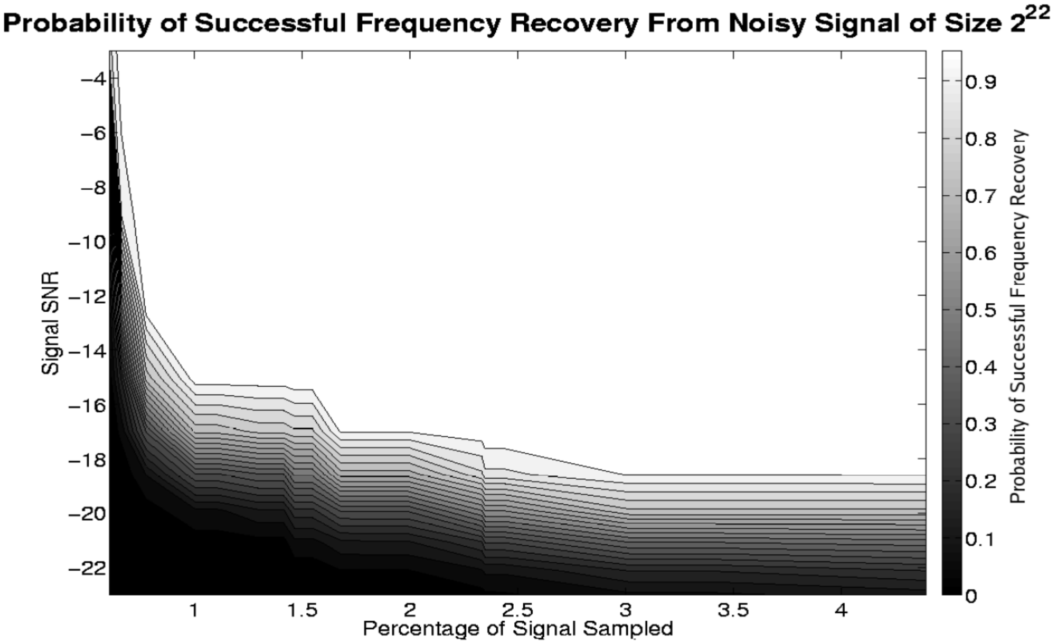

Figure 2.5 Probability of a single frequency recovery using FFS Technique

Alternately, one algorithm limitation is the appropriate choice of *m*. Using too many frequencies to approximate the signal may result in poor approximation.

# 3. Configurable Transceiver Design on Small Form Factor Software Radio Platform

An important feature of cognitive radio is the ability to change wireless communication parameters to satisfy certain operational constraints, including interference, power consumption, and communication throughput. The configurability of the cognitive radio transceiver determines the level of adaptability. The inherited flexibility in software makes software radios a viable enabling technology for cognitive radio. However, this flexibility comes at the expense of additional signal processing complexity.

Current wireless communication standards rely on different physical layer architectures with various digital radio blocks. The design of the baseband transceiver architecture depends completely on the services supported by the underlying wireless standards. Table 3.1 [85] shows various wireless standards with their required computational complexities.

Table 3.1 Symbol Rates for Some Wireless Standards

| Wireless standard | Approximate computational complexity (MIPS) |
|---|---|
| 802.11 a&b | 9000 |
| WCDMA | 6000 |
| GPRS | 300 |
| GSM | 100 |

Building a configurable transceiver that supports these various physical layer architectures can be achieved by instantiating several configurable digital radio blocks and a mechanism for switching between different communication modes.

A basic digital radio is comprised of the following radio blocks: encoding, modulation, pulse shaping, sample rate conversion, carrier frequency conversion, synchronization, channel estimation, and channel equalization. A simplified digital radio functional block diagram is shown in Figure 3.1. In it, the following radio blocks are listed:

- Encoding includes source encoding and channel encoding. Source coding is the process of efficient information conversion into a sequence of binary digits. Efficiency is achieved by removing undesired redundancy in information to be transmitted. Channel coding introduces intelligent redundancy in order to detect and/or correct possible data corruption that occurs during wireless transmission.
- Modulation maps the binary bits into symbols. Mapping determines the level of tradeoff between power and spectral efficiency of the wireless transmission.
- Pulse shaping is a filtering process that changes the waveform of transmitted pulses by limiting the effective bandwidth of the transmission to better suit the communication channel and reduce the inter-symbol interference caused by the wireless channel.
- Carrier frequency conversion is required to transfer the signal between baseband frequency level and IF frequency levels. Currently, most SDR platforms implement the digital radio at the IF level.
- Sample rate conversion is necessary upon the transition from one frequency level to the other. Because the sampling rate requirements for IF signal and baseband signal are different, the sampling rate should be adjusted accordingly.
- Synchronization occurs in three types, namely frame, symbol and carrier frequency.
- Mode Switching Control is required upon building configurable radio blocks to facilitate error-free transition between configurations.

Additional standard specific components are required, including Fast Fourier Transform FFT for orthogonal frequency division multiplexing; antenna combining and beamforming for multi-input, multi-out (MIMO) wireless devices; encryption; and frequency spreading, among others. This chapter focuses on commonly used digital radio blocks appearing in Figure 3.1. In particular, the design and implementation of configurable modulation, configurable symbol synchronization, and mode switching mechanism are provided in the following sections.

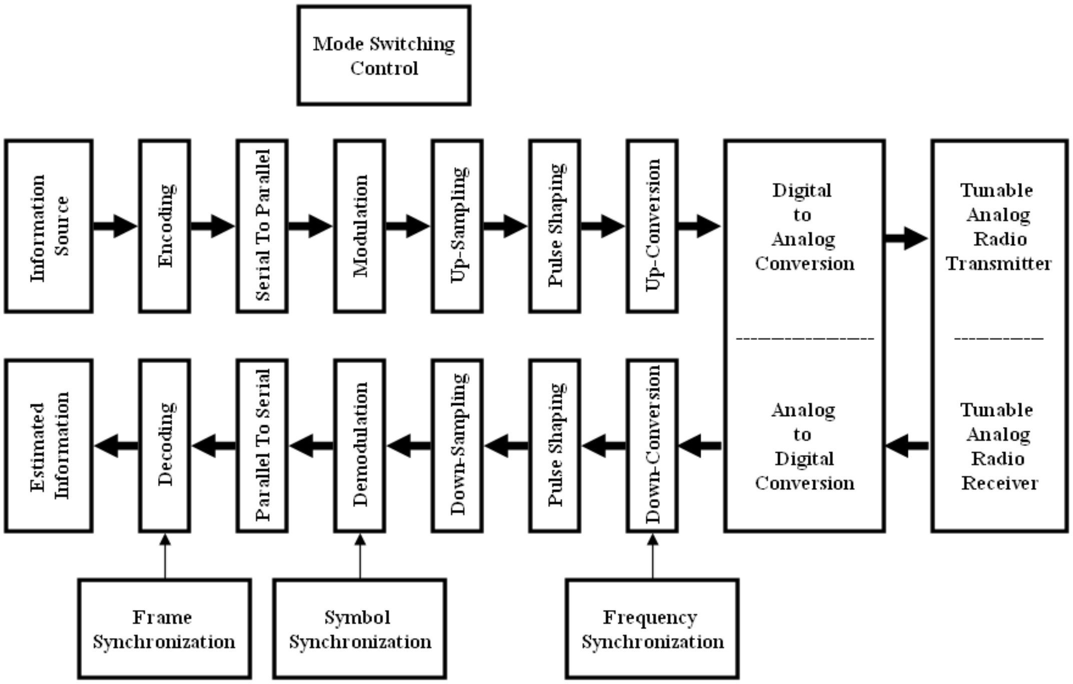

Figure 3.1 Simplified configurable digital radio system

The limited memory and signal processing resources in SFF-SDR platforms with hybrid processing architecture pose additional challenges in configurable transceiver design. To achieve efficient implementation, the design process must take SFF-SDR resources into consideration. In this chapter, two fundamental design techniques are used. The first is task partitioning, which maps signal processing stages into the appropriate processing devices in the SFF-SDR platform. The second is hardware reuse, which maximizes the common signal-processing path and minimizes configurable blocks in the configurable transceiver architecture. Both techniques are explained in further details throughout the sections of this chapter.

This chapter begins with a simple description of transceiver architecture based on single modulation scheme to demonstrate the design and implementation process on the SFF-SDR platform. The transceiver chosen for this purpose is a binary phase shift keying (BPSK) digital transceiver system. The implementation of a digital transceiver with configurable modulation capability is then reported, followed by a discussion of the design and implementation of a configurable symbol synchronizer to support various symbol rates. After demonstrating the design and implementation of several configurable components within the cognitive radio transceiver, the switching mechanism between communication modes is proposed. The chapter concludes with a proposal of a design and implementation framework for configurable cognitive radio transceiver, which is supported by an implementation case study.

The targeted SFF-SDR platform is provided by Lyrtech®. An overview of the platform architecture is provided in Appendix A.

## 3.1. Implementation of a BPSK Transceiver on Hybrid Software Defined Radio Platforms

This section demonstrates a practical design and implementation procedure for a BPSK modem on a Lyrtech® SFF-SDR platform and reports a detailed description of the baseband signal processing logic design in the FPGA portion. Design verification is performed through hardware in loop testing methodology. A framework for designing wireless digital modems on hybrid software radio platforms is discussed. The digital transceiver is implemented using a system generator for DSP® [17] for implementation of FPGA blocks and SIMULINK® and Real Time Workshop® to develop DSP signal processing modules [15].

### *3.1.1. Hardware Design Partitioning*

The BPSK transceiver is decomposed into signal processing functions and distributed between FPGA and DSP. Partitioning is based on the inherent characteristics of each processing engine. While the distributed arithmetic principles associated with digital filter designs enable the parallel processing scheme in FPGA, algorithms that require sequential analysis and decision making, such as synchronization and AGC, are implemented on the DSP.

Communication between FPGA and DSP is maintained through two connection methods; for high speed data flow, the VPSS connection has been chosen. Shared custom registers were used to exchange data at low sampling frequency for AGC or synchronization task use. A block diagram of the complete BPSK transceiver is illustrated in Figure 3.2 and shows the signal processing task partitioning between FPGA and DSP.

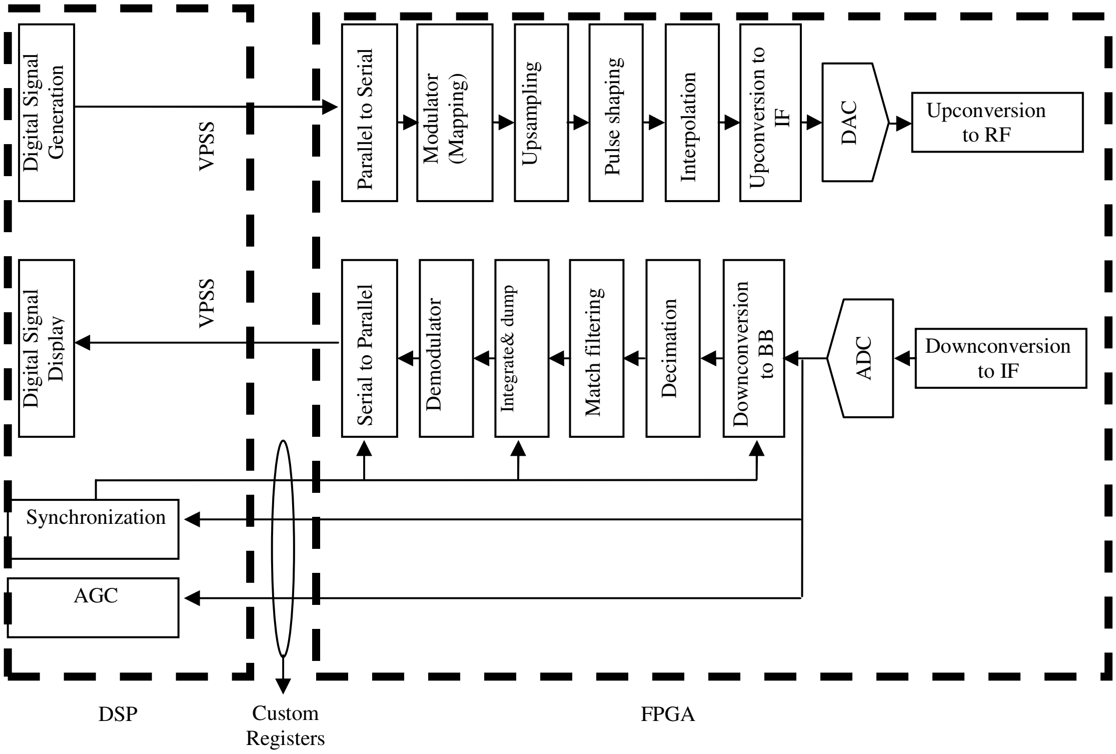

Figure 3.2 BPSK transceiver block diagram

### *3.1.2. FPGA Based Implementation*

The digital transceiver is comprised of transmitter and receiver stages that operate in duplex mode. While the receiver stage represents the inverse data path of the transmitter, it operates additional functionalities to maintain synchronization. Synchronization in a digital receiver is essential. As such, its design procedure is addressed in section 3.3.

The transceiver is designed to transmit and receive signals in a mobile communication environment. The signal is sampled at 24.414 kHz using a PCM3008 Codec chip, one which is often used for voice signals. The design requirements must therefore comply with the operation environment and available hardware resources. Presently, the mobile communication environment faces challenges of rapidly changing wireless channel characteristics primarily caused by mobility. The signal to noise ratio of the wireless signal is expected to be low. Binary Phase Shift Modulation (BPSK) is chosen for the proposed prototype because of its ability to tolerate low SNR values, in addition to the simplicity and low cost implementation. Also, the available codec in the platform defines the required data rate. For many digital communications, 10 bits appear to be sufficient for voice applications. As a result, the digital transceiver data rate should be 244.14 kbps using BPSK modulation.

In light of this discussion, the design procedure of a BPSK digital transceiver is provided below.

### *3.1.3. BPSK Transmitter Design in FPGA*

The design procedure for each transmitter block is discussed below. Figure 3.3 shows the transmitter logic design of BPSK transceiver.

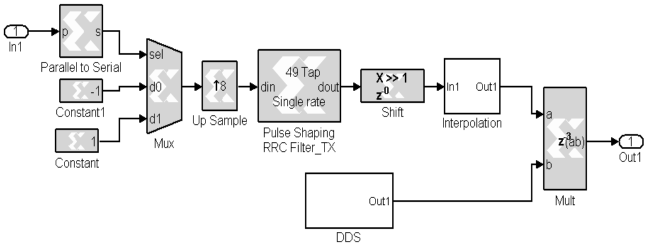

Figure 3.3 Transmitter logic design

- Parallel to serial. Voice signal samples are represented by 10-bit words. During this stage the input word is split into 10 time-multiplexed output words. The order of output bits (MSB or LSB first) should be considered for serial to parallel and parallel to serial stages.
- Modulator: For BPSK, the modulator maps binary zeros and ones to –1 and 1, i.e. the normalized values of the maximum input range of the digital to analog converter.
- Upsampling: The number of samples per symbol is a design choice subject to a tradeoff between hardware cost and an elevated error rate. A relatively low

number of samples per symbol lessen the requirement of pulse shaping filter design at the expense of higher symbol error rate, which results from a fewer number of samples used to determine the symbol value. Alternately, a high number of samples per symbol diminishes design specifications for a synchronization circuit, but increases the order of the pulse-shaping filter to achieve the same frequency response.

An upsampling ratio of 4 has been shown as the minimum ratio to enable PLL loops for frequency tracking [86]. However, in the proposed prototype, a ratio of 8 is chosen as the balancing point between the aforementioned upsampling tradeoff.

- Pulse shaping: A square root raised cosine filter is used for pulse shaping. The excess bandwidth factor is 0.2, and the order of the pulse shaping filter is an integer multiple of the upsampling factor (48 in this design), which allows the impulse response of the filter to span an integer number of symbols. An odd number of taps allows the middle point of the impulse response to be at the peak position; this is beneficial for synchronization.

The pulse-shaping filters coefficients are scaled to avoid overflow in hardware. A generally accepted scale factor is given by equation 3.1 [87]

$$SF = \frac{1}{\sqrt{\sum\left(h^2(k)\right)}} \qquad 3.1$$

Another practical method employed to avoid hardware overflow is scaling the coefficients so that the highest coefficient value will be as close as possible to one using a scaling factor ($2^k$) and applying a left shift operation of the filtering result.

- Interpolation: The sampling frequency at which the pulse shaping filter is operating is 1.9531 MHz (244.14 symbols/sec x 8 samples/symbol). The output is forwarded to the DAC, which runs at 125 MHz. Therefore, an upsampling stage of 64 is required. Although possible to achieve through the use of one interpolation stage, splitting the interpolation operation into multiple stages reduces the order of interpolating filters [87]. In this study, two 8x interpolating stages are chosen, and the order of each is 26 taps.
- Upconversion to IF: It is theoretically possible to convert the signal to analog at the baseband domain. However, due to hardware limitations, an upconversion stage to IF level is required to avoid loss caused by the capacitive coupling between the data conversion and RF boards. In order to adapt, upconverting to 30 MHz was chosen to reuse the 30 MHz DDS available in the receiver downconversion stage.

### *3.1.4. BPSK Receiver Design in FPGA*

This section discusses the design procedure for each receiver block. Figure 3.4 shows the receiver logic design of BPSK transceiver.

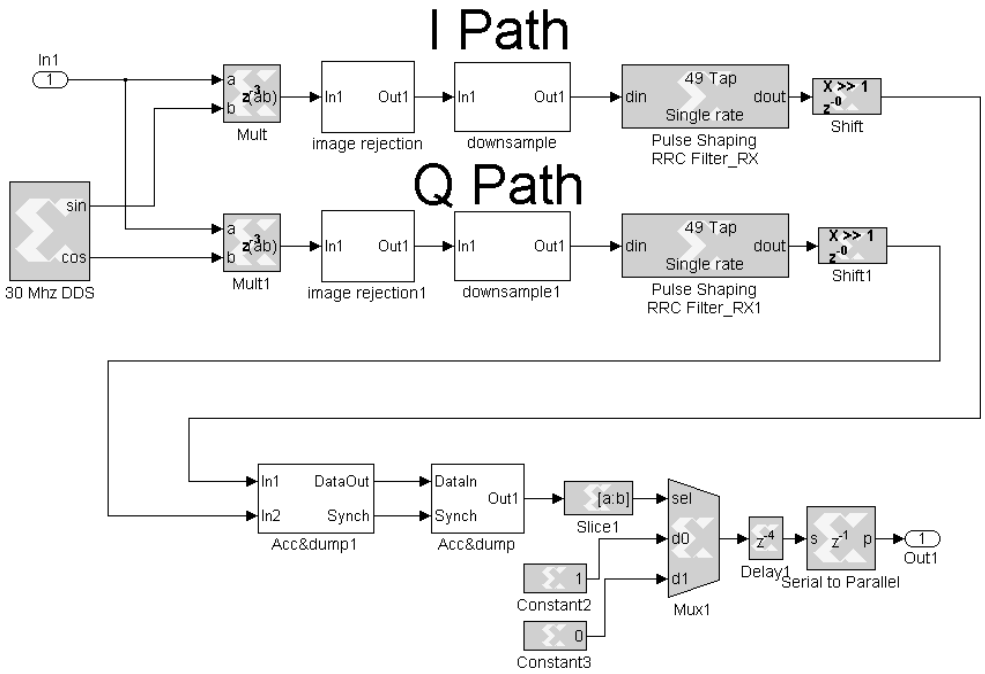

Figure 3.4 Receiver logic design

- Downconversion to Baseband: The Super-Heterodyne receiver transfers the receiver RF signal to IF=30 MHz for sampling at a frequency of 125 MHz. At the FPGA side, a downconversion stage to baseband is performed using a 30 MHz DDS. Phase matching between the IF and DDS is achieved by DPLL circuitry in the synchronization block.

- Decimation: After downconversion, a downsampling stage by 64 is required to transfer the signal sampling frequency to 1.9531 MHz, a level identical to the sampling frequency of the pulse shaping filter at the transmission stage.

- Match filtering. This square-root-raised-cosine filter is identical to the pulse-shaping filter at the transmitter.

- Integrate and dump: Symbol extraction is performed by an integrate and dump circuit, and the timing alignment of this stage is regulated by the synchronization block, which is implemented by an accumulator and 3 bit free-running counter. The sampling frequency is reduced by a factor of 8 at the output of this stage.

- Demodulator: The inverse of the modulation stage occurs at this stage and is implemented by slicing the sign bit of the symbol, then inverting it.
- Serial to Parallel: The 10-bit word is reproduced at that stage, and frame synchronization is required to ensure the extracting of an accurate 10-bit sequence each time.

### *3.1.5. Implementation Results*

A hardware-in-the-loop simulation technique is implemented wherein a host station running Simulink is used to generate data to the SDR platform through the Ethernet interface with the DSP. Initially the transceiver logic is performed in the hardware, and then the receiver signal is returned to the host station. The FPGA clock is 125 MHz, and the signal is transmitted at 400 MHz. A channel filter of 5 MHz bandwidth is chosen due to the nature of the narrow band-type signal.

Implementation results are shown in Figure 3.5. A portion of a sinusoidal signal to be transmitted is shown in Figure 3.5 (a), and the 10-bit binary representation is shown in (b). Figure (c) shows output results of the binary signal being forwarded to the modulator for conversion to 1/-1 signal and subsequently applied to a pulse-shape filter. A conversion of the signal to 30 MHz is represented in (d). While BPSK is considered a constant envelope modulation type, (d) shows the effect of pulse shaping on modifying its amplitude. The signal is down converted from IF to baseband at the receiver. Figure (e) shows the baseband signal following image filtering. In (f), the output of match filter (square root raised cosine pulse shaping filter) is presented. The synchronized integrate and dump circuit results, shown in (g), are forwarded to a sign bit slicer in order to extract the binary signal back in (h). Figure 3.5 demonstrates that a 7,500 clock cycle (equivalent to 60 $\mu$ sec) is required to perform the signal processing operation in FPGA.

FPGA resource utilization is reported in Table 3.2. The results can be further reduced by implementing polyphase filters, as most resources were used by two 50-taps filters for pulse shaping, and decimation/interpolation filters.

Table 3.2 FPGA resources usage

| Slices | FlipFlops | LUTs | IOBs |
|---|---|---|---|
| 2237 | 3471 | 3310 | 181 |

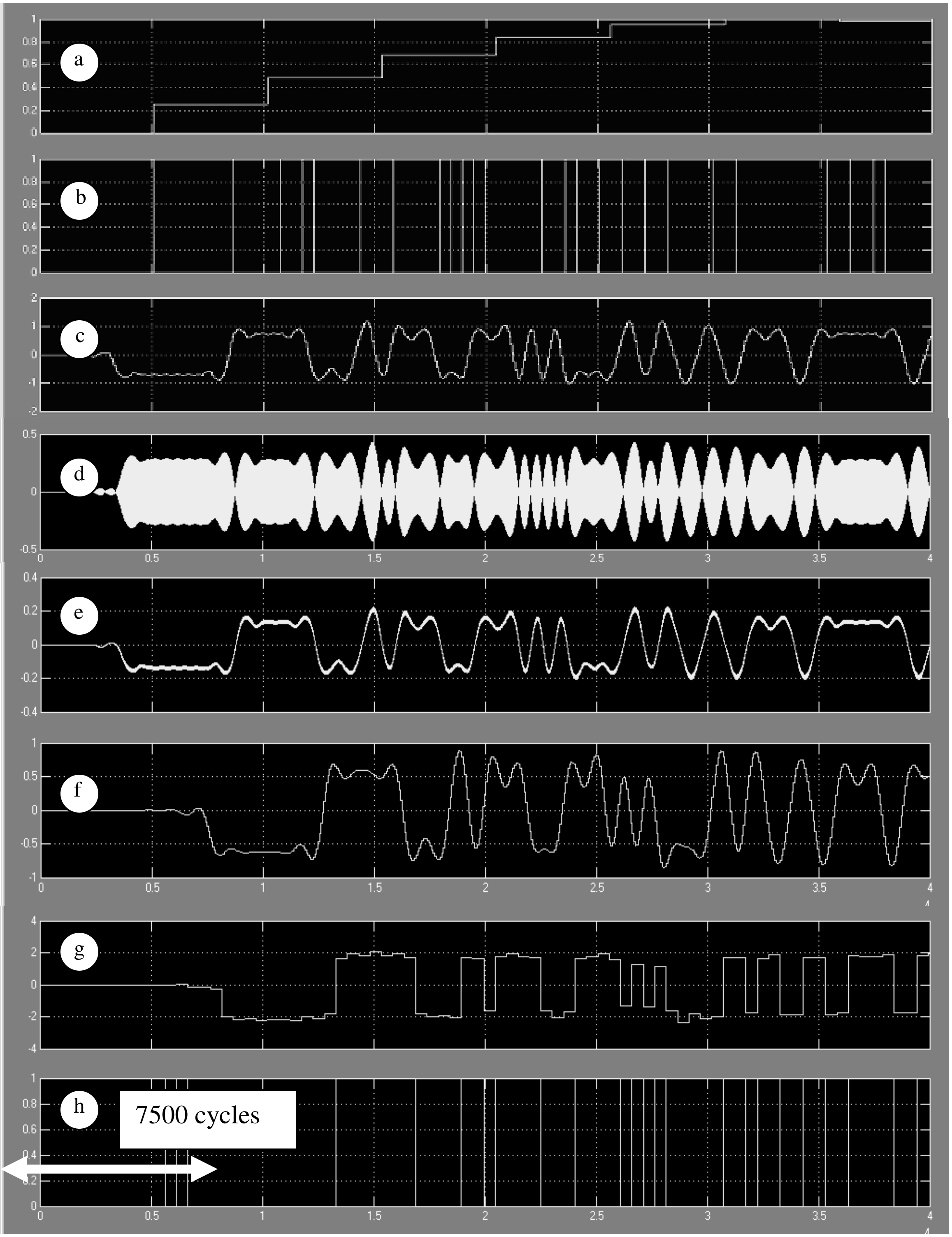

Figure 3.5 Signal conditions along different stages of the FPGA transceiver. a. original signal b. binary representation of samples c. pulse shaped signal d. upconverted signal to IF e. downconverted signal to BB after image rejection f. signal after match filtering

## 3.2. The Design and Implementation of a Configurable Baseband Transceiver on DSP/FPGA based Software Defined Radio Platforms

This section describes the design and implementation of a configurable baseband transceiver on a Lyrtech SFF-SDR platform using rapid prototyping and development tools. The software radio platform is based on a hybrid processing architecture comprised of a Digital Signal Processor (DSP) and a Field Programmable Gate Array (FPGA). The transceiver is able to switch between different digital modulation techniques on the fly. This section also explains the design and task partitioning procedures for specific design specifications and hardware limitations in each processing device. Implementation and verifications results are also reported.

### *3.2.1. Hardware Design Partitioning*

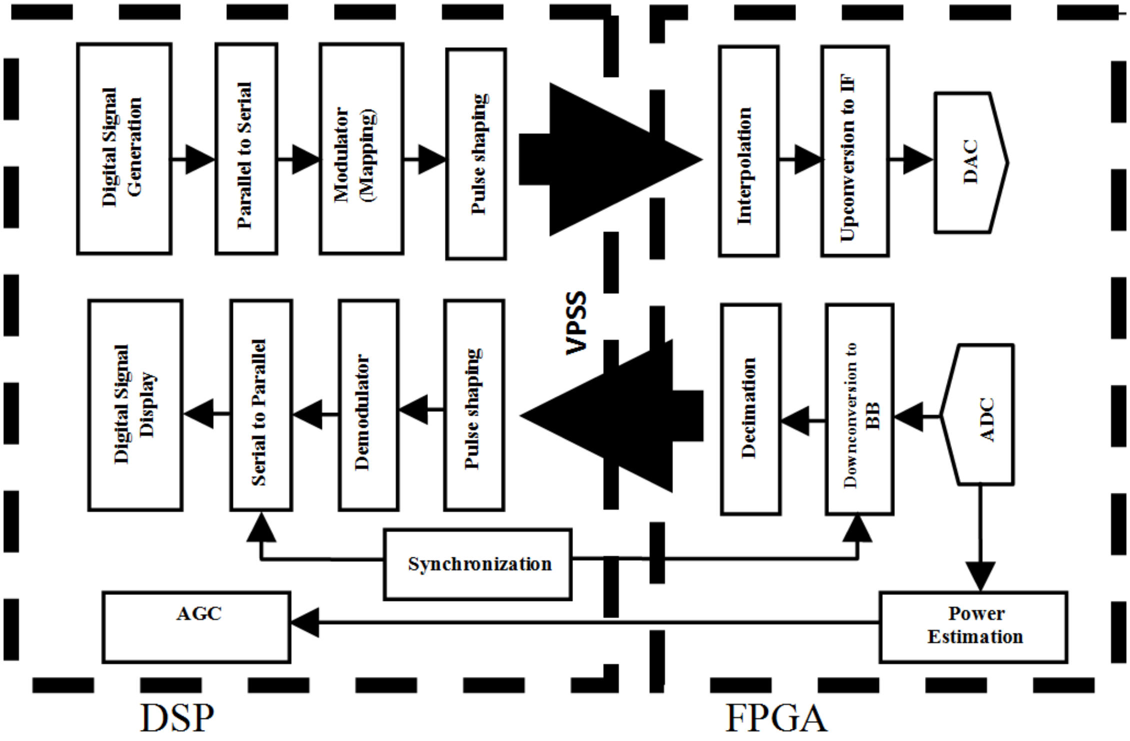

Figure 3.6 Configurable transceiver block diagram

The configurable transceiver design is partitioned between the DSP and FPGA processing devices. Figure 3.6 demonstrates that most high speed digital filtering is implemented in FPGA, while packet management bit mappings and control tasks—such as synchronization and automatic gain control—are implemented in DSP. This section describes the design procedure and implementation details for the signal processing modules in each processing device.

Table 3.3 Design specification summary

| | Parameter | Value |
|---|---|---|
| RF | Channel Bandwidth | 5 MHz |
| | Carrier frequency | 450 MHz |
| IF | Input signal quantization* | 14 bits |
| | Output Signal Quantization* | 16 bits |
| | IF Sampling Rate | 125 MHz |
| | IF Frequency* | 30 MHz |
| Baseband | Transmit Spectral Mask | 60 dB |
| | Symbol Rate | 488.28 ksps |
| | Roll off factor | Max 0.5 |
| | Total sample rate conversion | 64 |
| | Supported Modulations | BPSK,QPSK, 16,32,64QAM |
| | Maximum bit rate | ~ 3Mbps |

* Influenced by the platform hardware specifications.

Table 3.3 summarizes the chief communication system specifications and performance constraints used in the design process. Some values are influenced by software radio platform specifications and can be altered according to the capabilities and limitations of the targeted implementation platform. Before exploring FPGA and DSP design details, a number of configuration parameters for the RF and Data conversion boards are discussed, as is the communication mechanism between DSP and FPGA, i.e. the VPSS bus.

The flexibility of the RF parameters appears in two dimensions: channel bandwidth (a choice between 5 and 20 MHz) and carrier frequency. A channel bandwidth of 5 MHz was chosen to achieve an enhanced signal to noise ratio at the receiver ADC. The antennas used in the development platform are tuned to the frequency range between

450 and 490 MHz. Hence, the carrier frequency was set to a value within the range, e.g. 450 MHz.

Digital sampling is conducted in the IF stage at a sampling frequency of 125MHz. This rate is chosen to force the IF signal harmonics to fall away from the original IF signal. Although harmonics are attenuated through image rejection filtering after mixing, choosing fs so that fs=(N/M)IF (collect N samples for M periods of IF) completely safeguards the original IF signal. This process is termed non-IQ sampling [88]. It reduces errors introduced by DC offsets, clock jitter and ADC quantization. Of note is that this process comes at the expense of more latency during sampling M IF periods. Also it is important to mention that while the ADC resolution is 14 bits, the Effective Number of Bits (ENOB) available for processing is less. Hence, the receiving signal processing modules, especially the FPGA section, should be designed accordingly. The ENOB of an ADC can be calculated from equation 3.2 [1]:

$$ENOB = (SINAD - 1.763)/6.02 \qquad 3.2$$

The SINAD value of the software radio platform analog to digital converter ADS5500 chipset is 72.3 dBFS at IF = 30 MHz. Hence, ENOB is approximately 11 bits. A 10-bit resolution at the receiving side was chosen to account for additional SINAD deterioration caused by various distortion sources prior to the ADC stage.

Before determining the splitting point between DSP and FPGA for the transceiver design VPSS connection, it should be noted that the maximum clock rate supported by this connection is 37.5 MHz from DSP to FPGA and 70 MHz in the opposite direction. The higher clock rate comes at the expense of additional energy consumption and increased processing load on the DSP side.

Communication between DSP and FPGA is carried out by two ways: 1) via a VPSS connection that passes both a processed data stream, and 2) via a shared memory to control words and status flags exchange between the two devices. The communication between DSP and FPGA in either ways is clocked at four times the symbol rate (~ 2 MHz). The data transfer over VPSS is frame based and sized between 32 and 512 samples. The size choice is a trade-off between latency and throughput. In the proposed implementation, a frame size of 128 samples was selected as a balancing point. Upon analyzing the sampling rate conversion along the configurable transceiver design for different modulation and bit rate setups, it was found that placing the VPSS connection between pulse shaping and sample rate conversion attained an acceptable balance for processing loads between the two devices. Also, the VPSS location falls within the clock rate capabilities of the VPSS connection.

### *3.2.2. FPGA Based Implementation*

Signal processing modules implemented in FPGA include the sample rate conversion, complex IF mixing and FIFO buffering. The complete FPGA processing chain design is illustrated in Figure 3.10.

- The Sample Rate Conversion module converts the sampling rate between and the VPSS rate and the DAC sampling rate. (Of note is that part of the sample rate conversion is carried out by the pulse-shaping filter in DSP.) The total sampling rate conversion ratio is 4096 or 125,000,000/488,250. The rate conversion is typically implemented in multiple sample rate conversion stages. Table 3.4 summarizes the implementation complexity in terms of total number of filter tabs required for various implementation configurations. The table shows in descending order all possibilities for maintaining the sampling rate ratio change.

The most efficient sampling rate conversion configuration is 4x4x2x2, with 107 total filtering taps. The first rate conversion ratio (x4) is implemented in the pulse shaping stage, which was discussed in the DSP based implementation section. The total rate conversion ratio implemented in FPGA is 64.

The filter coefficients are quantized into Q16.15 format. Using the FDA tool in Matlab® and the Kaiser windowing method, the coefficients for each of the four filtering stages in FPGA are calculated. The order of each filter is 60, 14, 20 and 17 for 4x, 4x, 2x, 2x respectively for sample rate conversion ratios.

All filters, except the final filtering stage nearest the data converters, are designed with dual channel architecture to increase hardware reuse of the design, as shown in Figure 3.7.

- The Complex IF Mixing stage is comprised of two embedded high-speed multipliers and a direct digital synthesizer (DDS). The resolution of the transmitter IF mixing is 16 bits to utilize the full dynamic range of the DAC. However, since the ENOB is 10 bits at the receiving side, the receiving IF mixing stage resolution is 10 bits, also. The DDS is controlled by the phase synchronization circuit implemented in DSP. The control word (frequency drift) is passed through a 32-bit shared memory register.

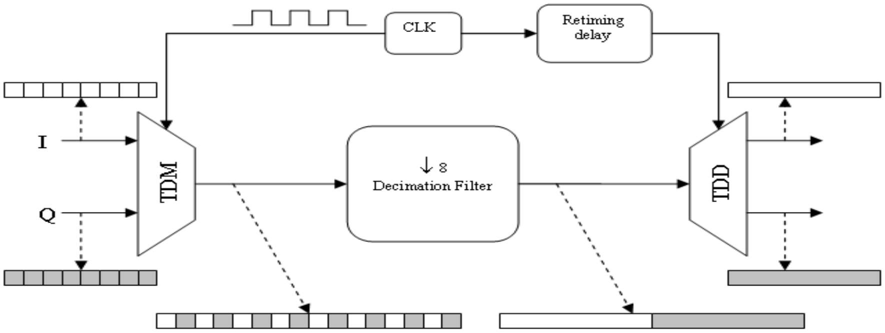

Figure 3.7 Implementation of dual channel decimation filter

Table 3.4 Decimation/interpolation filter orders for different implementation configurations

| | Configuration 1 | Configuration 2 | Configuration 3 |
|---|---|---|---|
| 1 stage | 64x | | |
| | 1857 | | |
| 2 stages | 32x 2x | 16x 4x | 8x 8x |
| | 266 | 192 | 266 |
| 3 stages | 16x 2x 2x | 8x 4x 2x | 4x 4x 4x |
| | 147 | 118 | 152 |
| 4 stages | 8x 2x 2x 2x | 4x 4x 2x 2x | |
| | 115 | 107 | |
| 5 stages | 4x 2x 2x 2x 2x | | |
| | 107 | | |
| 6 stages | 2x,2x,2x,2x,2x,2x | | |
| | 107 | | |

The DDS has two outputs, namely a sin and a cosine. Both are used for upconverting and for quadrature mixing at the receiver. Many approaches are used to attain direct digital synthesizers. The DDS approach used in the proposed prototype is the ROM Look-up Table approach, shown in Figure 3.8.a.

The phase accumulator formulates the address utilized by ROM, storing samples of a sinusoidal signal at each cycle and outputting one such sample. A number of clock cycles is required to revisit the entire ROM LUT; however, phase truncation reduces the amount of samples required to be stored in the ROM. A detailed functionality description of the LUT-based structure can be found in [1].

Because the output waveform has finite precision and the signal is periodic, spurious harmonic components and noise will be unavoidably evident in the spectrum. Spurs Free Dynamic Range (SFDR) is a measure of desired component power relative to spurious components in the spectrum; this number should be large for spectrally pure sinusoids. By choosing a 10-bit output size to match the dynamic range of the received signal, the SFDR becomes -60 dB. Figure 3.8.b confirms the SFDR value.

- FIFO Buffering. The communication between the DSP and FPGA is frame based. In order to stream samples into data conversion filters, a FIFO buffer is implemented in the transmitter side. The buffer collects received frames from DSP and streams them entirely through the interpolation filters to the DAC interface. On the receiver side, the FIFO buffer collects the decimated signal and forwards it through VPSS on a per frame basis. The size of the buffer is equal to 2 frames (2x128 samples). The word length is 32 bit; the lower 16 bit contains the "I" sample, and the higher 16 bit contains the "Q" sample. The "I–Q" combination and separation stages both concatenate and separate the samples.

### *3.2.3. DSP Based Implementation*

The chief signal processing modules implemented in DSP constitute the bit rate management executed by channelization module, as well as the modulation and pulse shaping achieved in the modulation/demodulation module. The entire DSP processing chain design is provided in Figure 3.11.

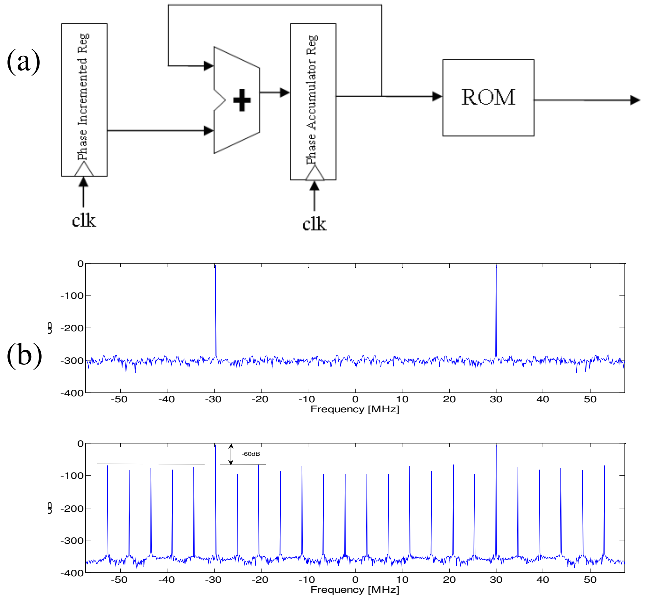

Figure 3.8 . (a) Direct digital synthesizer using ROM look-up table approach. (b) The spectrum of analog if frequency and quantized if frequency generated by DDS

- Bit rate management. Without loss of generality, various bit rates are produced by changing the signal source-sampling rate. A research goal is to design a configurable transceiver that supports a variety of bit rates using different modulation types. Hence, a mechanism to manage variable bit rate streams is necessary and accomplished using a time division multiplexing (TDM) system. The TDM frame consists of N time slots, each slot being B bits. N is determined by the maximum data rate multiplier, which is 6 in the suggested design and corresponds to the use of 64QAM. The modulation module maps time slot bits into complex numbers (I + j Q). The time slot is 32 bit influenced by the DSP architecture. Using a BPSK modulation scheme, a single time slot per TDM frame stream is employed. For QPSK, 2 time slots are used. 16QAM uses 4 time slots, and 64QAM uses the entire TDM frame. To simplify the TDM frame synchronizer design at the receiver side, unused time slots at lower modulation schemes contain copies of used time-slot values, thus relaxing the need for synchronizing the beginning of the frame at the receiver side. In this way, the value of the time slot is replicated at the adjacent time slots.

- Modulation. The modulation stage maps symbols into I and Q 16 bit values. Although the maximum bit size needed for either I or Q is 3 (for 64 QAM), the values will be passed to the pulse shaping function in 16-bit variables. The output is also represented by 16-bit variable; therefore, a 32-bit width is used for I and Q combined. At the demodulation side, detection thresholds for various modulations are calibrated for a fixed received power signal, which should be maintained by the AGC module.

- Pulse shaping. The pulse shaping stage uses a square root raised cosine filter. The excess bandwidth factor is 0.2. The order of the pulse-shaping filter is an integer multiple of the upsampling factor, e.g. 49 in the design presented herein, allowing the filter impulse response to span an integer number of symbols. An odd number of taps allows the impulse middle point response to reach peak position, which is beneficial for symbol timing recovery.

The coefficients of the pulse-shaping filters are scaled to avoid hardware overflow. A generally accepted scale factor is given by equation 3.1 [87].

The scaling factor results in fewer quantization errors at the expense of rare overflow events. Another practical method for preventing hardware overflow is scaling the coefficients so that the highest coefficient value is as close as possible to 1, using a scaling factor (2^k) and applying a left shift operation of the filtering result (<<k). This technique is used to scale coefficients in FPGA-implemented filters, as well.

The pulse-shaping filter applies a sampling rate conversion of 4. The number of samples per symbol is a design choice subject to a tradeoff between hardware cost and elevated error rate. A relatively low number of samples per symbol relax the pulse-shaping filter design requirements, but only at the expense of higher symbol error rate, resulting from fewer sample numbers to determine symbol value. Alternately, a high number of samples per symbol relax design specifications for a synchronization circuit while increasing the pulse-shaping filter order to achieve the same frequency response. It has been shown that a sampling conversion ratio of 4 is the minimum ratio to enable PLL loops for frequency tracking [86].

### *3.2.4. Implementation Results*

Several rapid prototyping and software development tools based on Matlab® and Lyrtech® SFF SDR software development packages are used in this work. The FPGA signal processing modules development uses Xilinx® System Generator for DSP, and the DSP signal processing modules are developed using the Real Time Workshop® from Mathworks®.

Results presented in this section include the hardware resources used by FPGA and DSP; the design verification by computer simulation; and BER testing implementation results using hardware in the loop simulation method for a supported modulation technique.

<u>Hardware resource utilization</u>

Table 3.5 shows a consumed hardware resource summary for the FPGA section design. Results can be further enhanced implementing polyphase filters for interpolation and decimation stages.

Table 3.5 FPGA resources usage

| Slices | DSP48 | Embedded Multipliers | RAMB 16 | DCM | BUFG |
|---|---|---|---|---|---|
| 6455 out of 15360 (42%) | 73 out of 192 (38%) | 4 out of 192 ( 2%) | 10 out of 192 (5%) | 3 out of 8 (37%) | 6 out of 32 (18%) |

On the DSP side, a stack size of 16kbyte in L1DRAM was allocated. The code, initialization parameters and constants are placed in SDRAM, and dynamic variables and switch jump tables are stored in L2RAM.

Although resource utilization could be optimized, results proved reasonably acceptable given the pace of rapid prototyping.

Design verification

Design verification is by way of computer simulation. Figure 3.12 shows the signal processing operation for DBPSK and 64QAM modulations over AWGN wireless channel with Eb/n ratio of 10 dB.

For DBPSK modulation, a single value (13859) is produced (Figure 3.12_1.1). The binary signal is then generated using 16 bit representation (Figure 3.12_1.2). The DBPSK modulation is performed in two stages. First differential coding and then binary signal mapping generates I and Q values with all Q values equal to zero for BPSK modulation; I values are shown in Figure 3.12_1.3. The pulse-shaping stage output is shown in Figure 3.12_1.4. The signal then passes through several upsampling stages until reaching the rate of 125 Msps, as shown in Figure 3.12_1.5. Note the delay caused by buffering an entire VPSS frame, and likewise the small delay caused by interpolation filters. An AWGN channel is simulated in Figure 3.12_1.6, and then the received signal is down sampled by a ratio of 64 (shown in Figure 3.12_1.7), before passing through the pulse-shaping filter at the receiver (shown in Figure 3.12_1.8. The signal is shown demodulated, in Figure 3.12_1.9 and mapped to its corresponding 16 bit value, as shown in Figure 3.12_1.10.

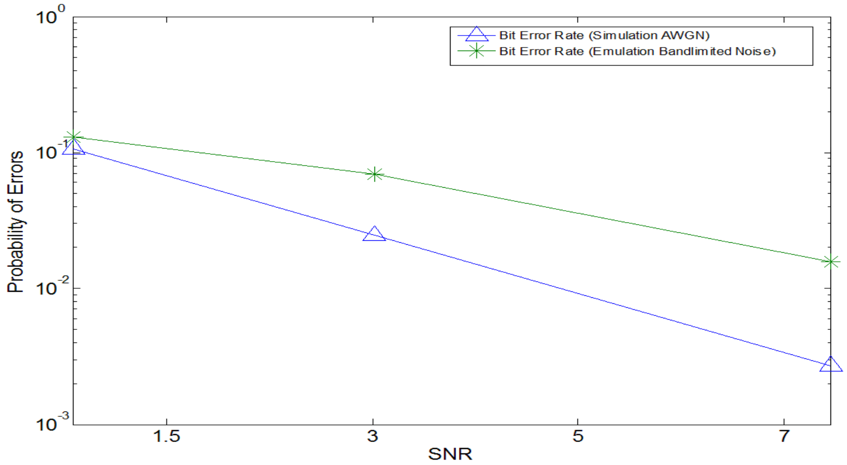

Figure 3.9 DBPSK transceiver bit error rate

Similarly, in the 64QAM simulation scenario, all the symbol values from 0 to 63 were examined, as shown in Figure 3.12_2.1. The I value (in blue) and Q value (in green) are produced in the modulation stage (shown in Figure 3.12_2.2) and passed through the pulse-shaping filter, as shown in Figure 3.12_2.3. The end of the upsampling stages are shown in Figure 3.12_2.4, and a white Gaussian noise is added to the signal in Figure 3.12_2.5, and the received signal is then down-sampled in Figure 3.12_2.6 and passed through the receiver pulse-shaping filter in Figure 3.12_2.7 before being completely decoded in Figure 3.12_2.8. Results show occasional sample errors in 64QAM, which is more sensitive to noise when compared with BPSK modulation.

Hardware in the loop simulation

Hardware in the loop simulation technique is performed to verify design implementation, and a host station running Simulink® is used to generate testing data for the SDR platform through an Ethernet connection with the DSP. Transmitter signal processing is performed on the SDR platform and then fed back to the host station to simulate an AWGN channel with variable SNR values; the signal then returns distorted to the SDR platform to perform the receiver signal processing operation. The transceiver is configured for DBPSK modulation. The FPGA clock is 125 MHz, and the signal is transmitted at 450 MHz. Because the signal is a narrow-band type, a channel filter of 5 MHz bandwidth is chosen. Figure 3.9 lists the BER achieved by the platform and compares is to the theoretical BER performance of DBPSK. Results show an implementation loss of approximately 3 dB.

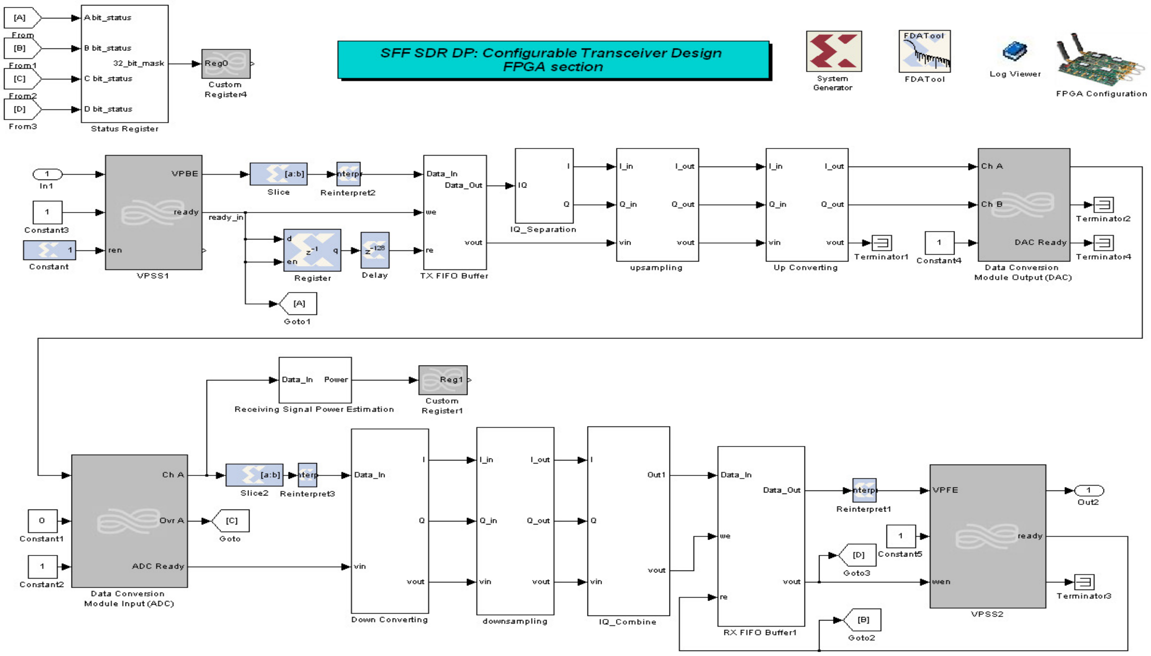

Figure 3.10 FPGA section of the baseband transceiver design

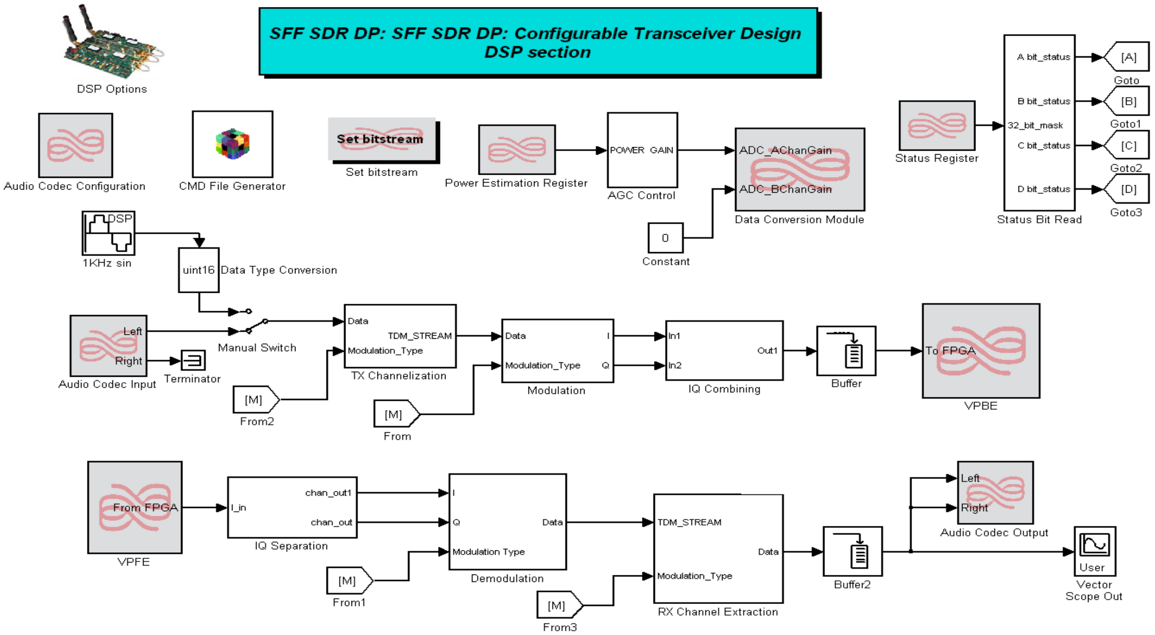

Figure 3.11 DSP section of the baseband transceiver design

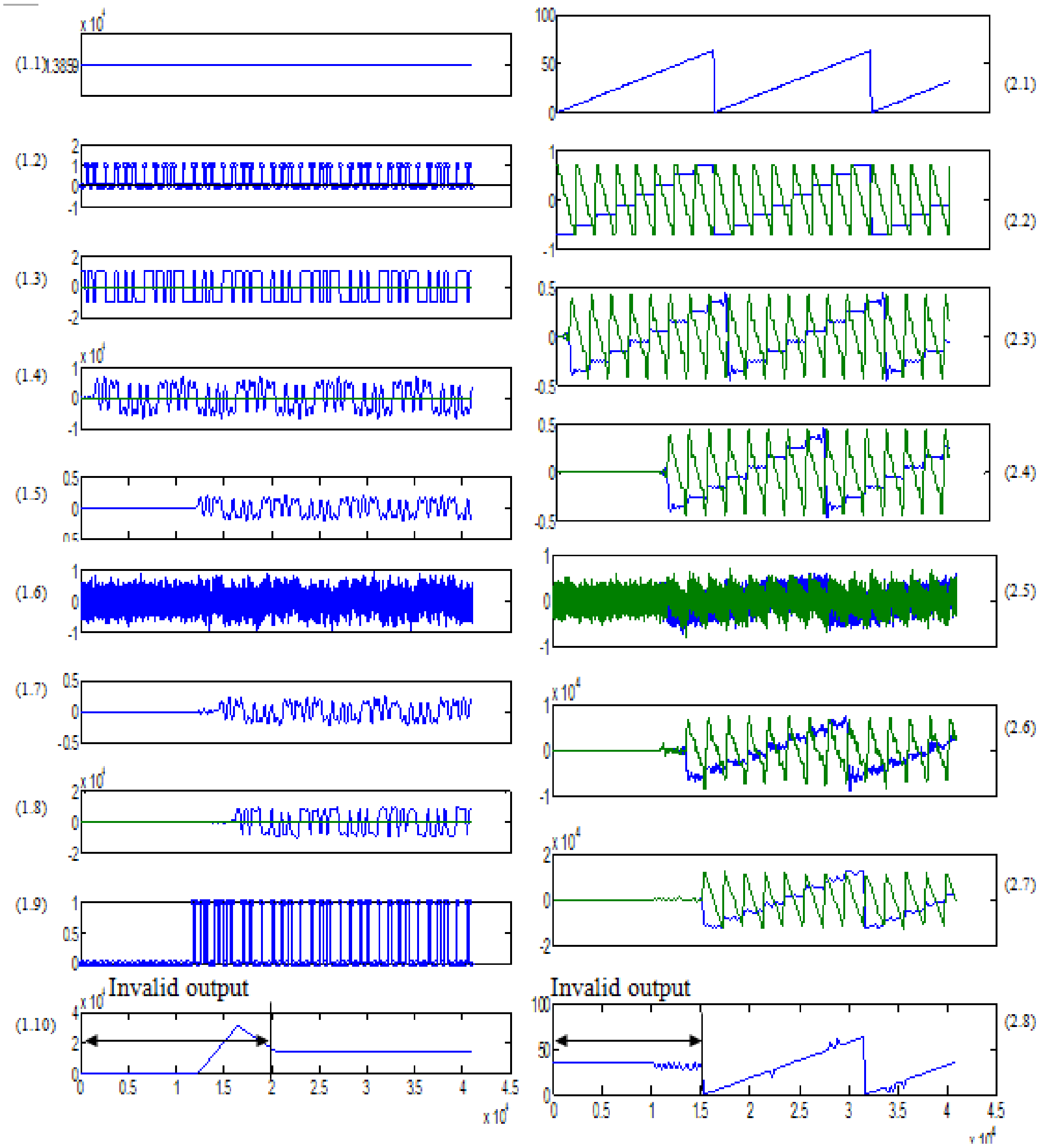

Figure 3.12 The simulation results of DBPSK and QAM modulations

## 3.3. Configurable Symbol Synchronizers for Software Radio Applications

Symbol synchronization is a critical SDR signal processing function required by the receiver to recover timing information and identify the point at which the symbols begin and end. By recovering the symbol timing, it is possible to determine the optimum symbol sampling points at which symbol decisions are made.

Several symbol synchronization techniques and architectures have been proposed [89]. Symbol synchronization techniques are classified as either data aided (DA) or non-data aided (NDA). DA techniques extract timing information based on known received signal information and are achieved by transmitting a known training or sounding sequence or by using symbol decision feedback. NDA techniques assume no knowledge about the received signal, therefore, tends to be more complex. The architectures of symbol synchronizers are also classified into two groups. The first is based on open loop architectures that perform certain signal processing operations on the received signal to extract the timing information. Such open loop synchronizers suffer from unavoidable non-zero mean tracking error. The second is founded on closed loop architecture in which the timing of a controlled local reference is compared with incoming signal transitions. Closed loop synchronizers solve the problem of non-zero mean tracking error at the cost of additional complexity.

In software radio platforms, symbol synchronization is implemented in digital domain. Many all-digital symbol synchronizers have been proposed by literature. Some examples include [90-95]. However, these are designed to operate under a single symbol rate.

Building a symbol synchronizer for software radio applications requires additional considerations. To support a variety of wireless standards, the synchronizer should support various symbols rates. (See Table 3.6) H. Harada has proposed an algorithm based on state patterns to build a flexible symbol timing synchronizer. However, the configurability posts hardware design challenges, as the symbol rate is an important factor in determining the down-conversion stage parameters at the receiver. It is impractical to have different down conversion stages to support dissimilar symbol rates, as down conversion operations have costly hardware. For example, a down conversion solution with a sample rate change of 192 used for GSM receivers can occupy 99% of a Virtex-II Pro 2vp2 -7 fg256 (Xilinx®).

The primary advantage when building a multi-symbol rate synchronizer operating at a fixed system clock is the ability to reuse the down conversion stage. Thus, if the multi-symbol rate synchronizer tracks a symbol rate of R, it can later switch to either a 2R rate and half the oversampling factor or R/2 and twice the oversampling factor. It should be noted that the purpose for building a multi-symbol rate synchronizer is not to support different data rates; rather the purpose is to support a variety of wireless standards on the same receiver hardware. Different data rates are typically achieved using a variety of modulation techniques. The multi-rate support is featured in some wireless standards, including 802.11.

In this section, configurable symbol rate synchronizer architecture for employment in software defined radio (SDR) is proposed. To the best of the author's knowledge, little has been done to address digital symbol synchronizer configurability. The work detailed herein can serve as a reference in the design and implementation of a configurable symbol rate synchronizer for software radio applications. Simulation results were obtained using the Lyrtech SFF SDR Development Platform, which employes the Virtex 4 Pro FPGA made by Xilinx. Controllable PLLs for down-conversion and FPGA symbol synchronization are utilized to obtain the SDR. For lower data rates, a simple technique is employed for a quick estimate of the phase difference between the received and reference signals and is completed before the signal is fed to the symbol synchronizer. This improves convergence rate while maintaining a constant sampling period at the ADC. The down conversion stage is also maintained at the receiver.

Table 3.6 Symbol rates for some wireless standards

| Wireless Standard | Symbol Rate |
|---|---|
| GSM/GPRS | 270.833 ksps |
| CDMA | 3.84 Mcps |
| WCDMA | 3.84 MSPS |
| EVDO | 1.2288 Mcps |
| EDGE | 270.833 ksps |
| HSDPA | 3.84 Msps |
| Bluetooth | 1 Msps |
| 802.11b | 11Mchips/sec |

### *3.3.1. Digital Symbol Synchronization Process*

After passing through the ADC, the sampled training sequence is then passed through a matched filter. In most cases, the training sequence is characterized as a train of rectangular pulses at constant frequency. For example, in BPSK the training sequence alternates between +1 and -1 every Ns samples. The corresponding matched filter is given by equation 3.3:

$$h(n) = u(n) - u(n-N_s) \qquad 3.3$$

The output of the match filter is a triangular wave reaching a maximum or minimum value every Ns samples. If the reference signal can be driven toward local maxima or minima of the matched filter output, the reference signal will be locked.

A digital synchronization process replaces the voltage-controlled oscillator with a counter that has an initial value equal to Ns. When the counter reaches three, two and one, data signal is sampled to obtain early, current, and late samples, respectively. The late sample is then subtracted from the early symbol and multiplied by the most current symbol to determine the decision parameter. This parameter approaches zero when the early and late (straddle a maxima or minima) samples approach one another. Current symbol multiplication indicates whether or not to approach minima or maxima. Based on operation results, a new initial value can be loaded into the counter when zero is reached, thus moving the current sample closer to the desired sampling time. When the current value aligns with the maxima/minima, symbols are then ready to be decoded. An operation similar to the integrate-and-dump performed in the analog scenario can be replicated by an accumulator reset every Ns samples. The general structure of a digital symbol synchronization process is outlined in Figure 3.13. In this figure, ES is early sample, LS is late sample, and CS is the current sample.

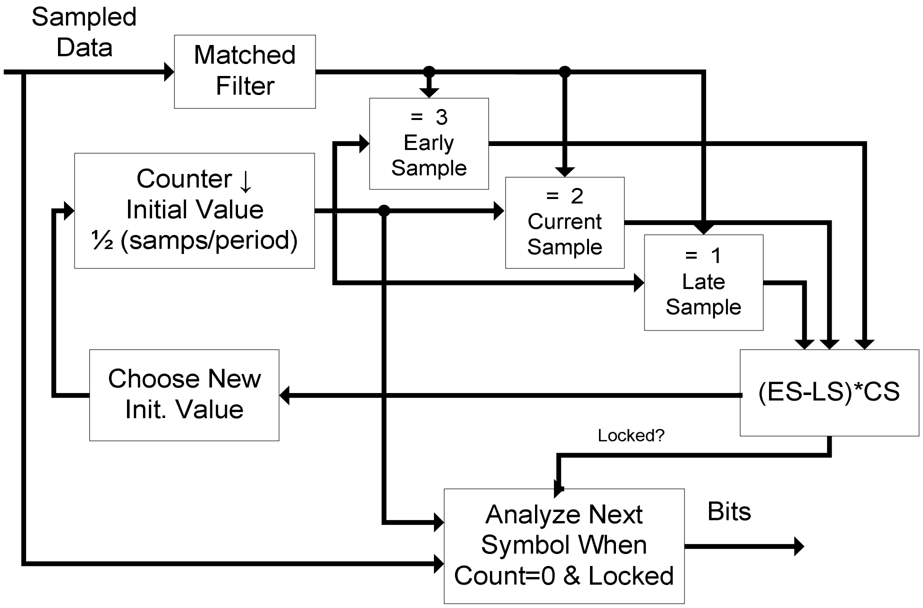

Figure 3.13 General structure of digital symbol synchronizer

### *3.3.2. The Proposed Configurable Synchronization System Overview*

The proposed configurable synchronizer is placed in a software radio receiver architecture subsequent to the generic down conversion stage, as shown in Figure 3.15.

The proposed synchronizer is capable of running at multiple predetermined symbol rates. Each is associated with pre-configured parameters that determine the performance of the synchronizer. Hence, upon switching from one symbol rate to another, the synchronizer parameters is updated accordingly to adapt to a new symbol rate at the same synchronizer driving clock frequency. Synchronizer parameters are accessible through an external memory. The design of the synchronizer is based on the following assumptions:

1- Sampling rate is fixed

2- The configurable symbol synchronizer follows a phase recovery stage

3- The supported symbol rates are known

4- Symbol rate switching can be asynchronous, which means it may or may not occur at the end of a symbol period

5- The synchronizer does not support wideband signals

The proposed architecture can support two symbol rate-switching scenarios:

- The first is planned switching and is defined by two communicating nodes that agree to change the symbol rate to a specific symbol rate value indicated in exchanged packets. In this case, configuration parameter memory is updated by higher layers, which have access to the agreed upon symbol rate.
- The second is automatic switching in which the receiver does not know the incoming symbol rate and accordingly is required to estimate and then update the configuration parameter memory. Symbol rate estimation can either be performed

by an optional symbol rate estimator subsystem in the proposed configurable synchronizer or by an external symbol rate estimation connected to configuration memory.

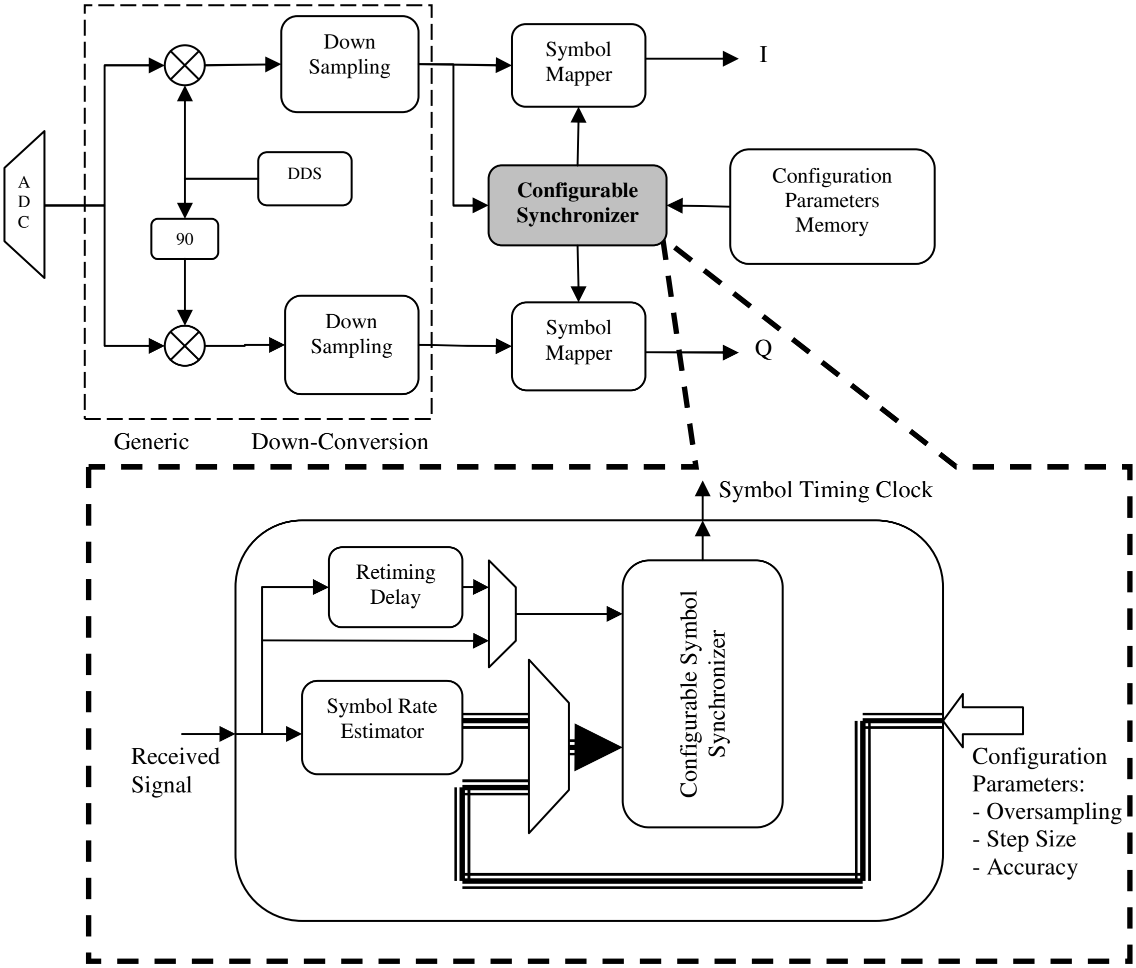

Figure 3.14 Configurable synchronizer in software radio architecture

If the optional estimator is enabled, configuration parameters are obtained directly from the estimator and not from the external memory. Hence, a predefined retiming delay is used to compensate for time duration consumed by the symbol rate estimator, as shown in Figure 3.14.

The configurable synchronizer is based on early-late architecture. Configuration parameters are listed in Table 3.7.

### *3.3.3. Proposed Configurable Synchronizer Architecture*

The proposed configurable synchronizer is comprised of four sub-systems, as shown in Figure 3.15. They are listed as follows:

- Matched filter. A configurable, averaging filter is used to generate the triangle waveform used for synchronization.
- The symbol sampler captures early, late and current samples of the incoming symbol. Samples of each are adjusted by the offset update subsystem.
- During the offset update, the timing error is calculated, and the symbol sampler obtains three samples and their appropriate timing is offset for the symbol sampler.
- Triggering system: generates the symbol sampling clock based on the current clock used by the symbol sampler. (See Figure 3.15):

Table 3.7 Configuration parameters

| Parameter | Description |
|---|---|
| Number of Samples $N_s$ (oversampling factor) | As the driving clock frequency is stationary and the down conversion sample rate change is fixed, the number of samples per symbol period is inversely proportional to the coming symbol rate. This parameter is essential in determining the accuracy that can be achieved for a certain symbol rate. |
| Step Size | This parameter determines the convergence rate to locked state. It defines the size of the phase shift toward the optimum sampling instants. The step size value can be fixed or a function of the measured timing error that can be calculated by external loop filter. |
| Accuracy | The time span between the late and early samples is used to find the appropriate phase shift. As the number of sample per symbol increases, a higher accuracy can be achieved. |
| Threshold | Defines the maximum timing error below which the synchronizer is considered locked. The parameter is expressed by the maximum difference between early and late symbol samples. |

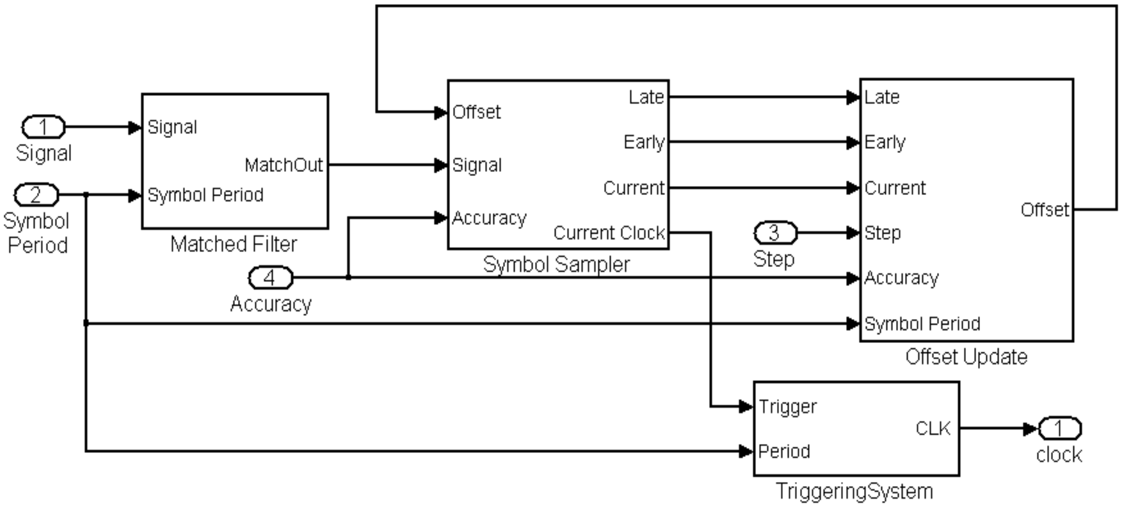

Figure 3.15 The proposed configurable synchronizer block diagram

In the following sections, the functionality and architecture of each subsystems is discussed in detail.

Match Filter

The received data signal at the output of the generic down conversion stage is passed to the match filter, where the goal is to boost the received signal SNR to match the pulse shape for the received signal and rejecting base-band noise. Many types of pulse shaping filters are used for various wireless standards. To make the configurable synchronizer a wireless-standard-independent, a flat averaging filter that is configurable online for different symbol rates was chosen. The filter's flexibility and configurability come at the expense of a reduced SNR improvement as compared to the improvements achieved via RRC or Gaussian filters. The filter configurability is illustrated by the transfer function of the configurable matched filter in equation 3.4:

$$H_{matched}(z) = \frac{1}{1 - z^{-1}} + \frac{z^{-n}}{1 - z^{-1}} \qquad 3.4$$

where n is the number of samples per symbol reflected during the symbol period given the sampling rate is fixed.

The programmable delay $z^{-n}$ is realized by a shift register with maximum length N. The sampling rate fs determines the upper bound of supported symbol rates, which is fs/2 while the maximum shift register size determines the lower bound of supported symbol rates, e.g. fs/(2N). Figure 3.16 shows the match filter as part of the Lock System.

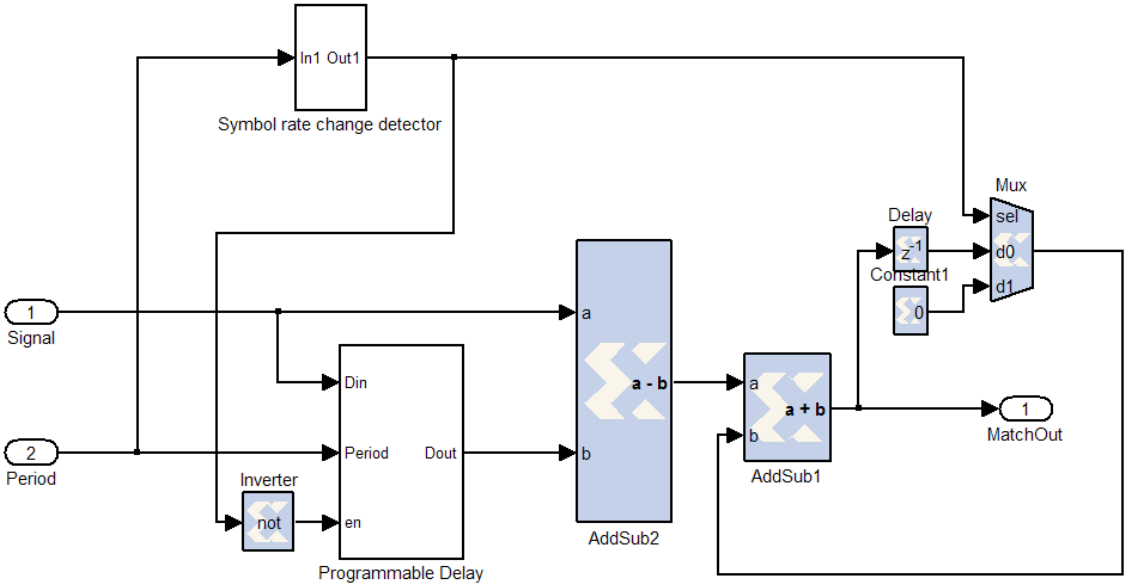

Figure 3.16 Configurable match filter model

Upon asynchronous switching to an alternate symbol rate, the match filter resets to avoid DC accumulation. Asynchronous switching indicates that changing the symbol rate may not occur at the end of the symbol. Therefore the match filter integration operation will carry an initial value from previous symbol rates that cause a DC shift in its operation. To avoid this problem, the symbol rate change detector—basically an edge detector—generates a reset signal to the shift register and accumulators.

Symbol Sampler

The symbol sampler captures early, current, and late samples. The difference between the late and early sample is used to estimate the timing error. Subsequently the timing is adjusted using the offset-decision circuit toward the optimum sampling instants. The accuracy parameter defines the span between the early and late signal. The sampler is implemented using comparators, registers and a down counter, which is auto-loaded with the offset value. The comparators enable the capturing registers at the counts 1, 1+accuracy, 1+2*accuracy. Figure 3.17 shows the model of symbol sampler.

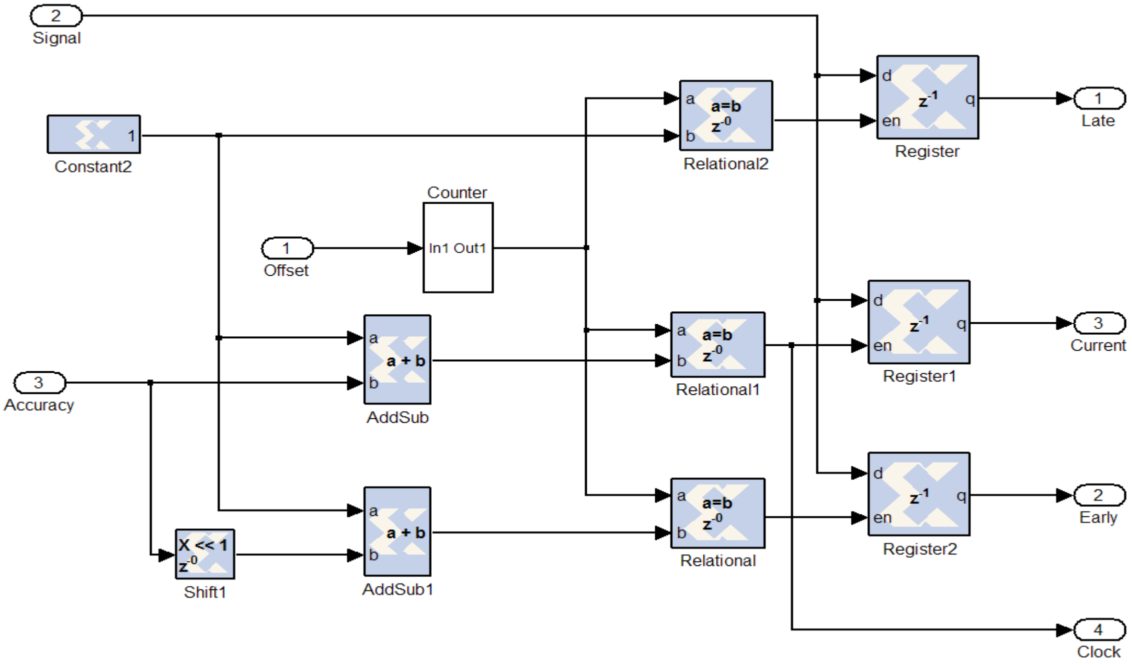

Figure 3.17 Configurable symbol sampler model

## Offset Updates

The offset decision uses the difference between early and late symbol samples to estimate timing error. The most current sample sign is used to determine the direction of the offset—forward or backward—along with the phase error. The offset value is determined using the step size parameter. Results can be one of the following values:

Table 3.8 Offset values

| Offset value | Condition | Effect |
|---|---|---|
| Number of samples + step | The timing error exceeds the maximum threshold (accuracy parameter value) | Shift the sampling forward |
| Number of samples – step | The timing error is below the minimum threshold (accuracy parameter value) | Shift the sampling backward |
| Number of samples | The timing error is within the threshold margin (accuracy parameter value) | Lock signal is generated |

The locked signal (not shown in the model) is asserted if the timing error is below the threshold parameter (see Table 3.8). The locked signal can be used as a ‘‘valid’’ signal by the symbol mapper. It is important at this stage to highlight the resolution (number samples per symbol) effect on determining the bit size of the accumulator. This is defined by equation 3.5:

$$\textit{Number of accumulator bits} = \textit{ceil}(\ \log_2(\ N+2^{res}\ )\ ) \tag{3.5}$$

where N is the maximum number of samples/symbol and is equal to the size of the shift register of the match filter and res is the bit resolution of the ADC. Figure 3.18 shows the model of offset updating system.

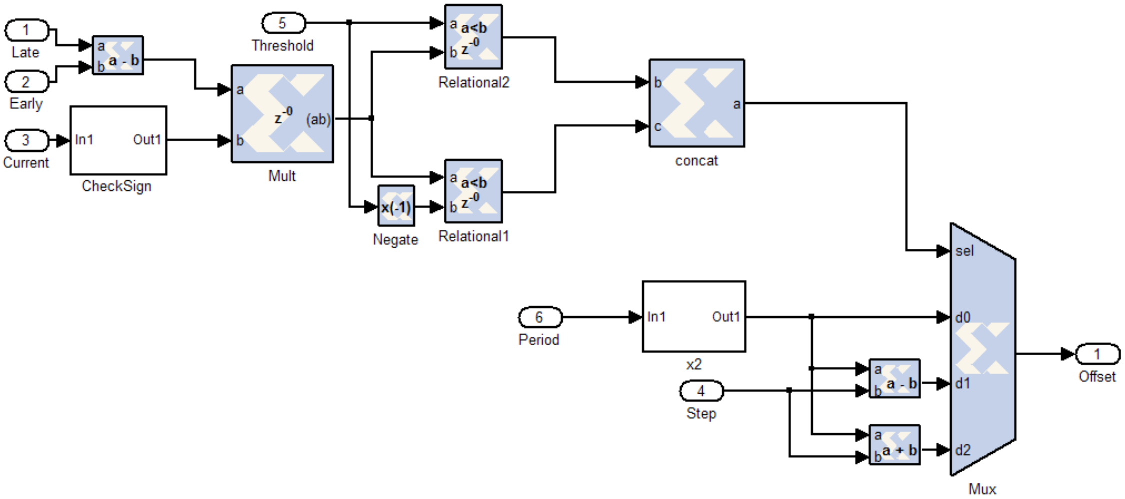

Figure 3.18 Offset update system model

## Triggering system

The triggering system provides the triggering clock to symbol mapper, e.g. an integrate and dump circuit. The clock defines the start of each symbol and is aligned with the current sample time instants, which should converge to the symbol transition time instants. These instants occur at half the symbol rates; therefore, the triggering system doubles the clock rate while maintaining timing alignment with the current sample time instants. The block diagram of the triggering system is shown in Figure 3.19.

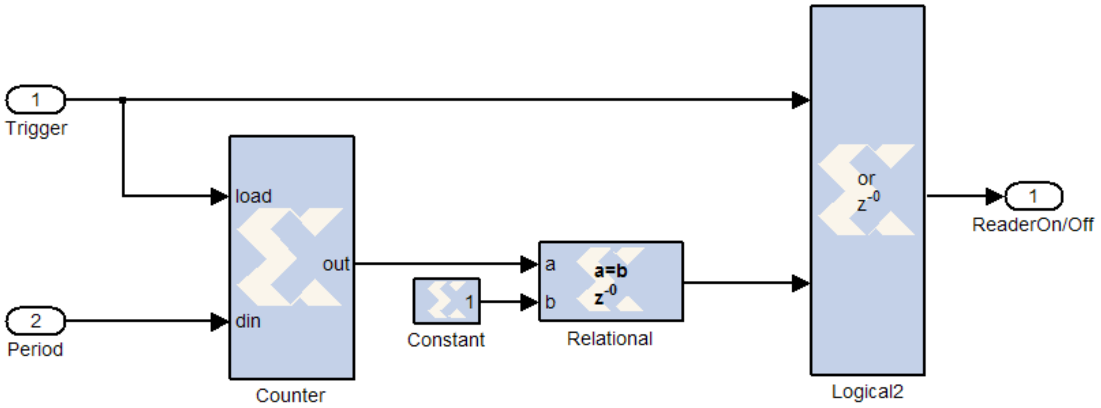

Figure 3.19 Triggering system model

The Period Estimator monitors the zero crossing of the training sequence and provides an estimate of data rate. The estimated value is then used to determine the associated parameters.

### *3.3.4. Implementation Results*

Implementation and computer simulation results are listed of which implementation reports the FPGA resource utilization. A computer simulation verifies operation of the design over various symbol rates.

<u>FPGA implementation results.</u>

The proposed configurable synchronizer is implemented using Xilinx Virtex4 XC4VSX35-10ff668. The FPGA is running at 12 MHz and support data rates of 10 ksps, 100 ksps, 1 Msps, 2 Msps, and 3 Msps. A Xilinx System Generator for DSP is used to synthesize the synchronizer model.

The implemented synchronizer is input by a 16-bits sampled data, which is anticipated to come from the 16-bits ADC at the end of the RF front end stage. Table 3.9 illustrates the FPGA resources used for implementing the configurable synchronizer.

Table 3.9 FPGA Resources used for the configurable synchronizer

| Resource Utilization | Symbol rates 1-3 Msps | Symbol Rates 10 ksps-3 Msps |
|---|---|---|
| Number of Slice Flip Flops | 245 (0.8%) | 980 (3.2%) |
| Number of 4 input LUTs | 821 (2.7%) | 3834 (12.5%) |
| Number of occupied Slices | 509 (3.3%) | 2016 (13.1%) |

Several FPGA solutions are known for symbol synchronizer implementation given a minimum utilization area utilization or maximum data rate support [96, 97]. However, supporting a wide dynamic range of symbol rates under a single FPGA solution is the focus of this work. Table 3.9 shows the wider the dynamic range of supported symbol rates the more required FPGA resources. The wide range of supported symbol rate requires many memory resources, including look up tables, among others. It is possible to fine tune the proposed system to minimize the area utilization or increase the supported data rate by decreasing both the number of samples taken per symbol and the resolution of the synchronization input, e.g.10-bit rather than 16-bit.

### Simulation Results

The performance of the proposed system is examined under two types of simulations. The first is intended to validate synchronizer functionality and is therefore conducted under noise free conditions. The second examines the synchronizer performance through noisy channel.  The simulation assumptions and parameters are listed below:

- Supported symbol rates:10 ksps, 100 ksps, 2 Msps, 4 Msps and 6 Msps
- System clock:12 MHz
- FPGA platform: Xilinx Virtex4 XC4VSX35-10ff668
- Simulation tools: Simulink®, Xilinx System Generator for DSP® and Lyrtech SDR toolkit ®
- Symbol rate switching: asynchronous, indicating the symbol rate does not necessarily change at the end of the symbol
- The simulation explores all symbol rate switching permutations

Table 3.10 lists the parameters associated with the predefined symbol rates:

- Simulation 1: Functionality Validation

This simulation is performed under noise free conditions and is decomposed into two phases to provide improved visualization of simulation results. The first phase examines the fine symbol rate changes between 1, 2 and 3 Msps. The second examines the large symbol rate changes between 10 ksps, 100 ksps and 1 Msps.

Figure 3.20 shows the configurable synchronizer operation at symbol rates 1, 2 and 3 Msps. The first plot shows the arriving symbol sequence at different rates; the second shows the generated saw signal emerging from the coming symbol sequence; the third demonstrates the convergence of the current symbol sample to the peak or valley of the saw signal at which an optimum sampling clock sequence can be generated; the fourth plot represents the locked signal asserted when the sampling clock aligns with the symbol beginning; and the fifth plot shows the symbol rate estimator output and represents the symbol rate index where 1 is 1 Msps, 2 is 2 Msps and 3 is 3 Msps. From this plot, it is apparent that the synchronizer loses the locked state upon switching from one rate to another. Recovery time to the locked state varies based on previous and current data rate.

Table 3.10 Supported data rates and their associated parameters

| Symbol Rate | Number of Samples | Step Size | Accuracy |
|---|---|---|---|
| 10 ksps | 600 | 100 | 50 |
| 100 ksps | 60 | 10 | 5 |
| 1  Msps | 6 | 1 | 1 |
| 2 Msps | 4 | 1 | 1 |
| 3 Msps | 2 | 1 | 1 |

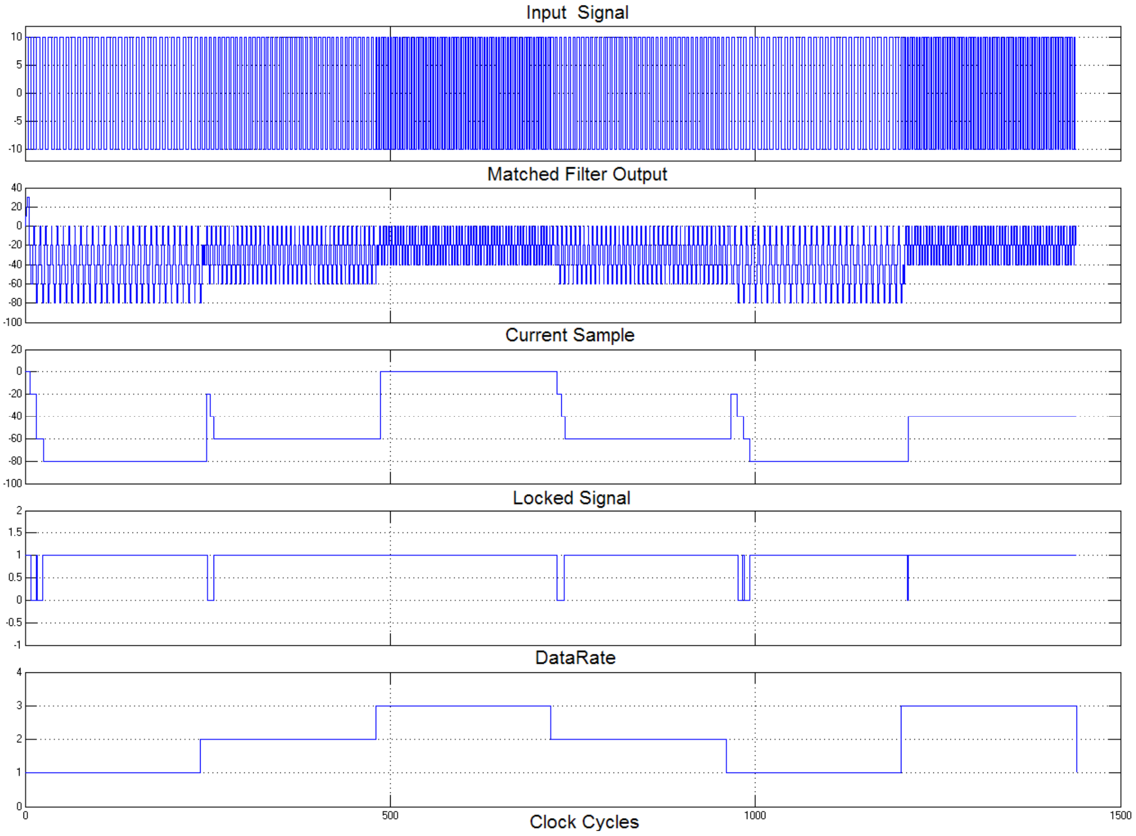

Figure 3.20  The Configurable synchronizer simulation results for symbol rates 1,2 and 3 Msps

Figure 3.21 shows the configurable synchronizer operation at symbol rates 10 ksps, 100 ksps and 1 Msps and illustrates the synchronizer functionality at coarse symbol rate changes.

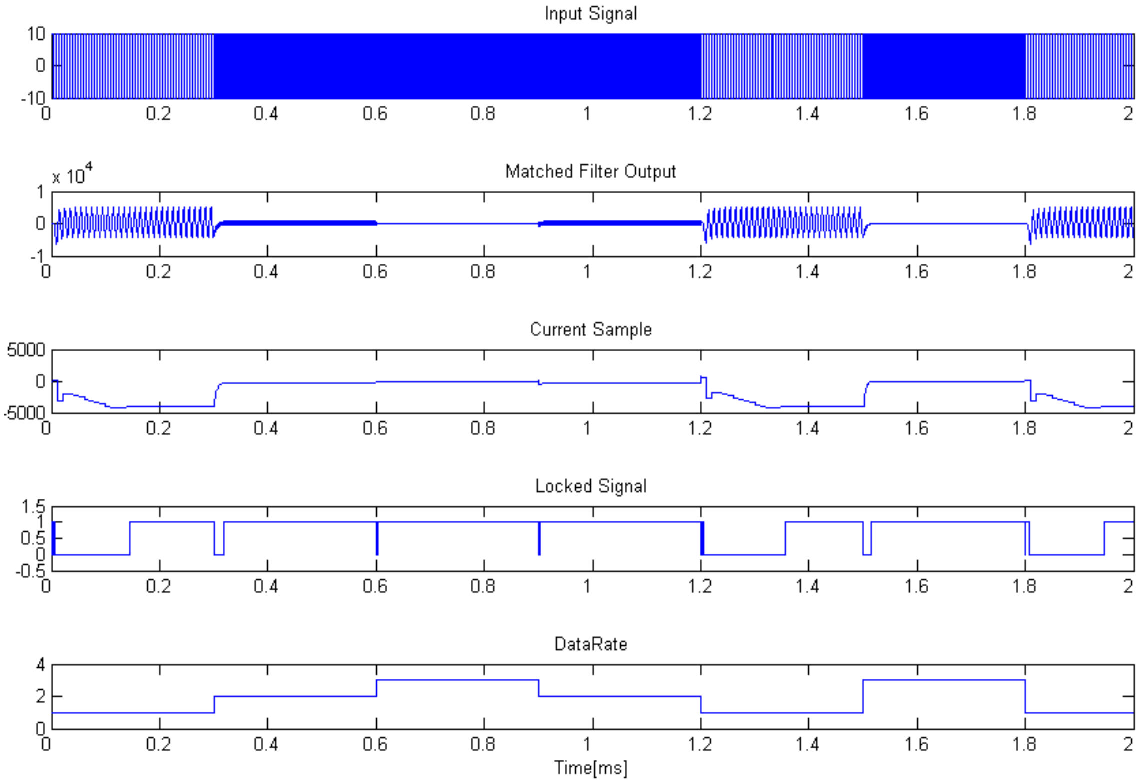

Figure 3.21 The configurable synchronizer simulation results for symbol rates 10 ksps, 100 ksps and 1 Msps

- Simulation 2: Synchronizer performance under noisy channel

A band limited noisy channel is simulated by adding normally distributed band limited noise. The synchronizer performance is evaluated by the symbol error rate at three different noise powers. However, it should be noted that errors include those generated by symbol rate switching. The simulation is conducted until 10,000 symbol errors are captured to ensure statistically valid error rates. The supported symbol rates chosen for this simulation are 1, 2 and 3 Msps. The switching event occurs every 100 symbols. Table 3.11 lists error rates obtained at various PSD noise powers:

Table 3.11 Symbol error rates at different noise powers

| Eb/N [dB] | Error Rates |
|---|---|
| 10 | 0.035 |
| 0 | 0.05 |
| -6 | 0.279 |
| -9 | 0.4 |

### *3.3.5. Performance Analysis*

Several factors determine performance of the configurable synchronizer. Following is a summary of such simulation effects:

- Resolution of the ADC

Higher bit resolution causes wider area utilization and lower supported speed rates. However, higher bit resolution also relaxes the range of critical thresholds available for the symbol reader, allowing for higher noise immunity.

- Dynamic range of the data rates

The smaller the difference between supported data rates, the higher the accuracy required by the Period Estimator. However, this phenomenon allows for smaller area utilization and faster convergence to the locked status. The larger the dynamic range, the smaller number of samples in the fastest data rate supported because the design maintains a fixed sampling rate for all supported data rates.

- Convergence time

The convergence time is affected by the step size and the moment at which the symbol rate is changed. The fastest convergence rate occurs when the symbol rate changes at the end of a symbol because in this case alignment with a symbol beginning is not lost.

- The step size

While a large step size makes for a faster convergence, it also causes instability under high accuracy synchronizer requirements. Hence, the step size is affected by the choice of the ADC resolution. The higher the ADC resolution, the wider the dynamic range of possible step sizes.

- Effect of noise

The effect of noise is shown when adding a band limited noise to the received signal. While noise power has a direct effect on the latency of the convergence, the relation is not linear, meaning that doubling the noise power does not double the latency to converge to the locked state.

- Effect of the data rates on convergence time

Simulation results indicate there is no direct relation between data rates and convergence time. In fact, convergence time is highly dependent on the moment of switching between data rates and not the data rate values because convergence time depends on the phase shift at which the clock is away from its optimum phase.

## 3.4. Design and Implementation of Online Mode Switching on Hybrid Software Radio Platforms

Software radio is a promising solution for bridging the interoperability gap between incompatible wireless standards. However, its success is dependant upon the advancement of a good number of enabling technologies [5-8]. Configurable computing platforms are considered the core enabling technology for SDR since it provides the flexibility required to maneuver between different wireless communication bands and operation modes and the computational power required to support a wide range of bandwidths.

In order to design and implement a configurable transceiver, two important design decisions are required. First is a configurable transceiver architecture featuring sufficient flexibility and hardware efficiency. Second is an appropriate online mode switching mechanisms to obtain seamless and error-free mode transitions.

In SFF-SDR platform with hybrid DSP/FPGA processing architecture, the design of configurability and switching mechanisms in FPGAs are especially challenging. While the unique capability of field programmable gate arrays to perform parallel computations [4] have attracted the attention of software radio researchers, the technology suffers from a number of limitations addressed in studies, including design portability [98, 99]; dynamic configurability [31]; and interoperability [100] with processing engines in hybrid SDR platforms. These limitations, however, have not prevented FPGA design from being an integral element of software radios used mainly to perform computationally intensive signal processing functions such as up/down conversions, coding/decoding and error corrections algorithms.

The dynamic configurability in FPGA transceiver design is achieved through the selection of an appropriate flexible architecture that requires reasonable hardware resources and reconfiguration time [31]. Based on the work of [31], a partial dynamic reconfiguration architecture is suggested. This architecture exploits the signal processing commonalities between supported air interfaces to formulate a common digital processing backbone. The foremost common digital radio functionalities are sample rate conversion and channelization [101]. The architectural benefit is hardware reuse gained through sharing a common signal processing backbone between supported modes. Given that the architecture was adopted for the proposed configurable multi-mode radio, details regarding the design considerations for developing a common signal processing backbone and building a partial configurable signal processing chain are explained. In addition, a switching-control architecture was added to ensure seam-less transition.

This section details the design and implementation of a configurable transceiver in which the communication modulation and frequency carrier can be altered while maintaining operation. The architecture and mode switching mechanism are discussed in terms of design process. The proposed transceiver switches between amplitude modulation (analog) and differential binary phase shift keying (digital) upon user request. Also the transceiver can automatically switch frequency carriers upon the event of a bit error rate increase due to wireless channel degradation. While BER is not as sufficient as the wireless channel condition indicator upon which the channel switching decision is made, using this parameter as a switching trigger source is sufficient to build a proof-of-concept configurable wireless transceiver model. The transceiver combines signal processing modules to perform a configurable multimode radio. The design of such is implemented using a small form factor software defined radio (SDR).

### *3.4.1. Configurable Transceiver Design*

The configurable transceiver is based on DSP/FPGA software radio architecture and comprised of four subsystems:

- Digital modulation in terms of differential binary phase shift keying (DBPSK).

The digital modulation transceiver is based on differential binary phase shift modulation. It supports a data rate of 244.14 kbps with signals sampled at 24.414 kHz and 10 bits for each sample.

- Analog modulation. Amplitude modulation with double/single sideband, e.g. AM-DSB/SSB. The analog modulation transceiver is based on double-side, band-amplitude modulation with optional single side-band mode. The analog modulation mode is also capable of transmitting a signal sampled at 24.414 kHz, where the modulation mode is chosen manually by the user.

- Manual modulation mode switching mechanism.

To achieve efficient hardware utilization, a common structure between the two modulation modes is highlighted and used to implement an efficient multimode wireless transceiver with online mode switching capability. Switching between modulation types is accomplished by configuring modulation-specific blocks. The switching is manual and triggered by the user.

- The Automatic frequency switching mechanism is effective in digital modulation mode, enabling the transceiver to automatically switch between two communication channels on-the-fly upon detecting wireless channel quality degradation.

The design and implementation of the first and second subsystems are discussed in sections 3.2 and section 3.3. The focus of this section is building the third and fourth subsystems.

### *3.4.2. Building the Common Signal Processing Structure*

The exploitation of commonalities among different communications modes proves a challenge to the design of a configurable digital transceiver for a software radio terminal. This is especially true if modes are dissimilar, i.e. analog and digital modulation modes. The proposed software radio design aims to maximize the reuse of digital signal processing blocks. This section details the design approach for determining the common structure between AM and DBPSK transceivers, and then suggesting how to avoid implementing two independent transceivers in the software radio platform. The approach is performed in two stages.

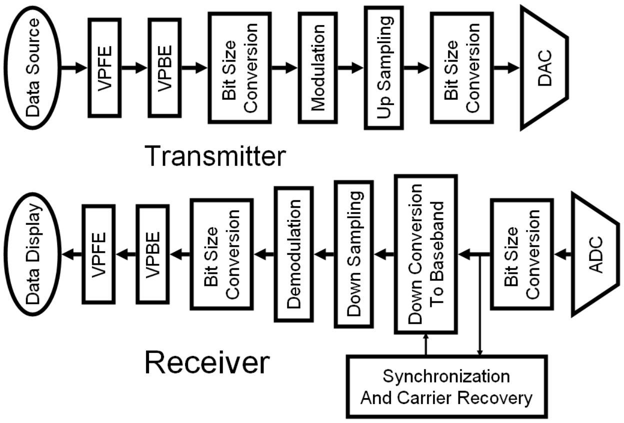

Figure 3.22 The common structure of the digital portion of the wireless transceiver

The first stage determines the high-level view of the data path through different SDR platform components. The second determines the bit-level design of the common structure based on design parameters, including signal characteristics, sampling rate, and sample bit resolution.

Figure 3.22 illustrates a high-level view of the common wireless transceivers digital structure using a Lyrtech® software radio platform. The figure is independent of a particular modulation mode and is used instead to offer an idea of possible signal processing blocks reuse between different supported modulation modes. At transmission, the signal is generated in a host machine and then passed to the DSP via an Ethernet connection. The signal is then passed to FPGA through a VPSS 16-bit bus, represented by the VPFE (Video Processor Front End) at the DSP side and VPBE (Video Processor Back End) at the FPGA side. The output of VPBE buffer is normalized and truncated to 10 bit format using slicing and reinterpretation operations in the bit conversion stage. The signal is then modulated, and finally the modulation stage output is up-sampled to the DAC 125 MHz sampling rate of 125 MHz. It should be noted that the sampling rate of the modulation stage output might differ based on modulation technique used.

Hence, the up-sampling stage cannot be completely reused. A bit conversion stage is required before the DAC in order to map 10 bit representation to 16 bit. The receiver digitizes the analog signal using 14 bit ADC. A bit conversion block is used again to map the signal to 10 bits. A down-conversion stage is required at the digital domain, since the superheterodyne receiver used at the radio module brings the signal to the IF level (at 30 MHz) at the ADC stage.

After down-conversion, the signal is down-sampled with the same sampling rate conversion ratio used at the transmission. A demodulation stage is then performed to extract the original signal and pass it to DSP through VPSS bus. Note the possible need for synchronization and carrier recovery stages at the receiver when using coherent receiver structures. Further details on building configurable symbol synchronizers are found in section 3.3.

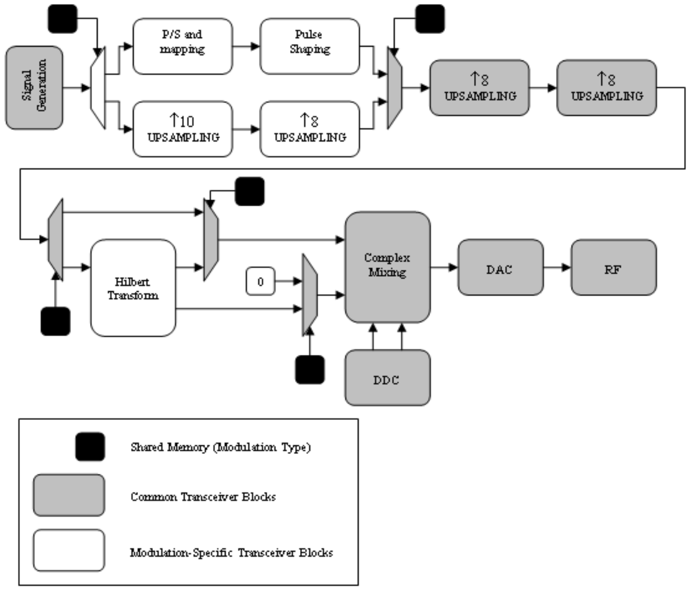

Figure 3.23 Hybrid AM/DBPSK Transmitter Structure

When inspecting the high-level view of the signal processing chain, we realize that many stages can be reused. The modulation stage and sampling rate conversion should, however, be carefully designed to improve hardware reuse and support both modulation techniques. This concept is addressed in the second stage, namely the bit-level AM/DBPSK transceiver structure design.

The following design elements are considered:

- Transmitted signal characteristics or bandwidth is one

One of the most important design considerations is the transmitted signal bandwidth. With a sampling rate of 24.414 kHz, it is possible to have two signals: 1.) the analog modulated signal, which is limited to 12 kHz bandwidth approximately; and 2.) the digital modulated signal, which is limited to 24.414

$\frac{ksample}{\sec ond}$ x 10 $\frac{bits}{sample}$=244.14 kbps. When using an excess bandwidth factor of 0.5 in the pulse shaping stage, the occupied bandwidth used by the digital modulated signal is 366.21 kHz—the largest among all possible baseband signals that may pass through the signal processing chain. It therefore identifies the Nyquist boundaries required for sampling rate conversion stages.

- Sample rate conversions.

Software radio design utilizes multi-rate signal processing techniques to manipulate the design trade-off between transceiver supported data rates and energy consumption and to achieve a balance between performance and energy consumption. Regarding the design of a multi-rate system, it is important to comply with anti-aliasing constraints. The common structure is conservatively designed for the widest possible bandwidth of 366.21 kHz. Designing with this in mind facilitates the reuse of  a portion of the signal processing blocks for alternate communication modes that run at less bandwidth, namely the AM signal.

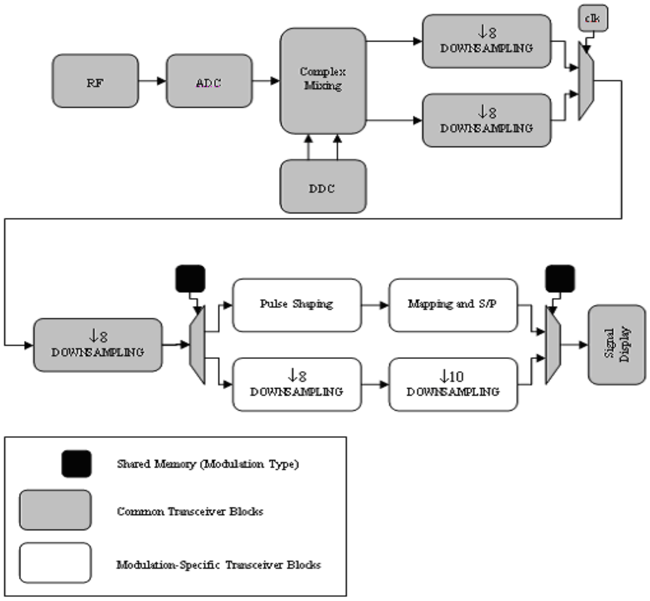

Figure 3.24 Hybrid AM/DBPSK Receiver Structure

- The resolution (bit size) used at different stages of the transceiver digital portion.

  The digital portion of the wireless transceiver is comprised of various digital components, including DSP, FPGA, and ADC/DAC, each with different capacities and resolutions. DSP is a 32bit digital component and is connected to FPGA through a 16 bits VPSS bus. (An interested reader can find information about the buffering and communication mechanism of VPSS bus in [14].) Due to its flexible structure, FPGA can manage a variety of bit-sizes. DAC has a 16 resolution, while ADC has only 14 bits. Most wireless communication waveforms utilize 10- to 12-bit resolution in their baseband processing level. The configurable transceiver in this work uses 10-bit resolution based on ADC SNR limitations.

The hybrid transmitter design is illustrated in Figure 3.23, and the hybrid receiver design is shown in Figure 3.24. Common transceiver signal-processing blocks include the signal generation and display; up/down conversion; DAC; ADC; and RF board. Bit conversion stages are omitted for simplification.

The signal generation stage samples the signal to be transmitted, where the sampling rate is the main constraint and must fulfill Nyquist requirements for all communication modes.

The up/down conversion stage is comprised of up/down-sampling and mixing subsystems. To share the up/down-sampling subsystem the up/down-sampling ratio is required for each communication mode and is calculated by finding the maximum common divider of total up/down sampling ration for each communication mode. In our configurable transceiver, the AM transmitter requires a total up/down-sampling ratio of 5120, while the DBPSK transmitter requires 64. Therefore, the common up/down-sampling subsystem is 64, which is the maximum common divider of 5120 and 64. The mixing stage is defined by the IF frequency choice and is selected based on hardware limitations, namely the ADC and the RF board.

The DAC, ADC, and up/down conversion stages are mutually dependant. The sampling rate of the ADC and DAC determines one boundary of the total sampling rate conversion ratio. In the configurable transceiver, ADC and DAC sampling rates are identical and fixed (125 MHz).

An RF board can be designed to achieve certain selectivity and sensitivity for all communication modes and must support the maximum possible signal bandwidth of any communication mode.

### *3.4.3. Automatic Channel Switching Mechanism*

The automatic channel switching mechanism platform can change its communication channel upon degradation in the wireless channel quality (defined as the suitability of a wireless channel to carry the transmitted signals with an acceptable received error rate). Both direct and indirect techniques can be used to estimate the wireless channel condition.

Direct techniques, such as measuring error rate, identify channel quality degradation and require the availability of the original transmitted signal at the receiver—a phenomenon not possible in practice. However, a periodic known pattern (preamble) can be injected at the beginning of each frame and then used to perform a number of receiver adaptation tasks, including synchronization, channel equalization, AGC, and channel estimation. While this may appear useful for bursty traffic, it is certainly not efficient for continuous communication due to overhead occupied by the preamble.

Indirect techniques use statistical parameters to estimate the received signal SNR, which proves a good parameter for measuring channel quality. Other methods are based on time diversity techniques, i.e. sending each sample twice and comparing both received samples. This trivial technique is adopted for the proposed configurable transceiver because of its implementation simplicity and minor design changes. Although the technique suffers from considerable overhead, it is considered a simple circuit that both monitors wireless channel degradation and triggers the channel switching control system.

The decision to switch channels is dependent upon the detection of N1 corrupted samples in a row, e.g. when n bit errors are detected in a sample. Regardless, continuous switching between two channels may occur when after switching channels, the receiver experiences a number of bit errors depending on the circuit transient state behavior. To avoid limit cycle behavior in the control circuit, two switching thresholds (high and low) are used, where the high threshold is equal to N1 and activates the channel switching task. Switching is performed just once and does not allow a switch back to the original channel, except for two cases:

1.) The error rate is below N2.

2.) The manual switching button is reset.

The parameters N1, N2, and n are determined manually based on sensitivity required for monitoring channel quality. In the proposed configurable transceiver, n=2 bits, N1=6, and N2=3. The channel sensing system's inability to recommend a vacant channel proves a limitation of the channel switching controller. As such, the channel switching circuit is limited to two predefined channels: 1.) the extreme case when switching to a noisy channel, and 2.) manual switching to return to the previous channel. It should be noted that the simple circuit is implemented to validate the channel switching mechanism. Reliable spectrum sensing is necessary and discussed in chapter 4.

Figure 3.25 illustrates a simplified block diagram of the switching control circuit. Error detection is performed using an XOR gate between two copies of the received samples, which are extracted from the serial-to-parallel converter. The number of bit errors is counted by passing XOR output serially to a counter. The bit error counter resets every 10 cycles (or 1 sample) and is synchronized with the serial-to-parallel stage. If the output is larger than n, the sample error counter stage increases. The symbol error counter output is captured every N1 samples, or 10xN1 cycles. The symbol counter resets upon receiving a correct sample. The output of the sample error counter is forwarded to a hysteresis relay with two thresholds. The relay is set when the number of sample errors is above N1 and is reset when the number is lower than N2 or upon pressing the reset button. The relay output is passed to a toggle flip flop T-FF, which switches between 1 and 0 on the rising edge of the relay output. T-FF controls a multiplexer which passes one of two carrier frequency values to the radio board controller.

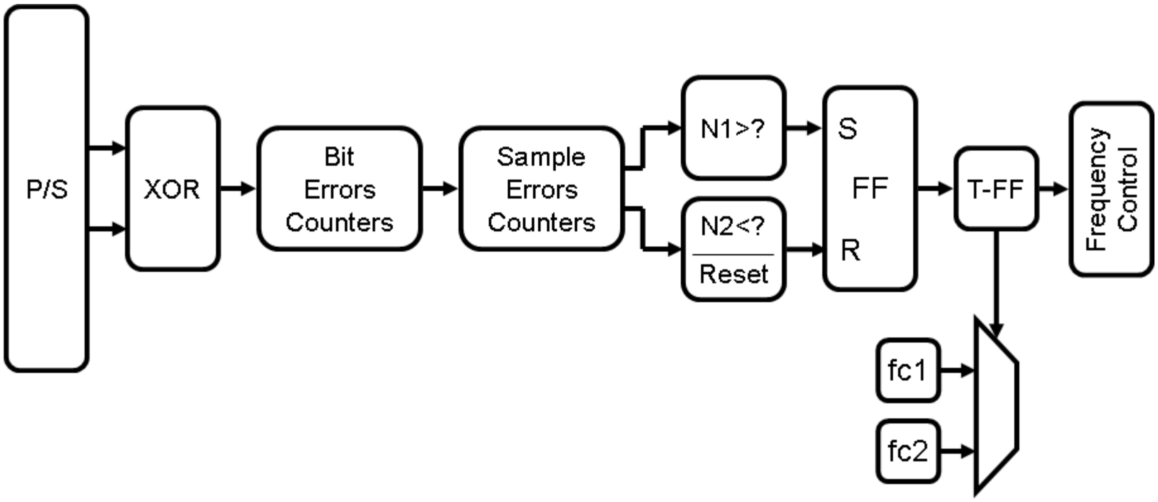

Figure 3.25 Switching control circuit block diagram

### 3.4.4. Switching Control Design

Online switching presents a challenge for software radio. In the proposed configurable transceiver, two modes of switching are presented, namely manual switching between analog and digital modulation and automatic switching between different communication channels. Challenges to these modes are listed as follows:

Data integrity during switching operation

It is important to design a switching mechanism that protects the transmitted signal from loss or corruption during switching. This is achieved by buffering the data during the switching interval. Transmitter and receiver circular buffers are used to maintain data integrity during mode switching. The design of data buffering is shown in Figure 3.26.

At the transmitter side, the circular buffer is placed after the VPSS port which provides a “ready” control signal subsequent to passing a new data word from DSP. The signal is connected to a write-enabled buffer port. The switching control circuit provides a read-enable control signal when the switching operation is idle and the buffer is not empty. If the buffer is full, a “full” control signal is passed from the buffer to shared memories (absent in Figure 3.26) to notify the DSP that the transmitter cannot transmit additional data.

At the receiver side, a circular buffer is placed after the ADC. Similarly, the data converter provides a “ready’ control signal subsequent to passing a new digitized sample (14 bits) from the analog domain. The signal is connected to a write-enable port of the buffer. The switching control circuit provides a read-enable control signal when the switching operation is idle and the buffer is not empty. If the buffer is full, the received sample is dropped.

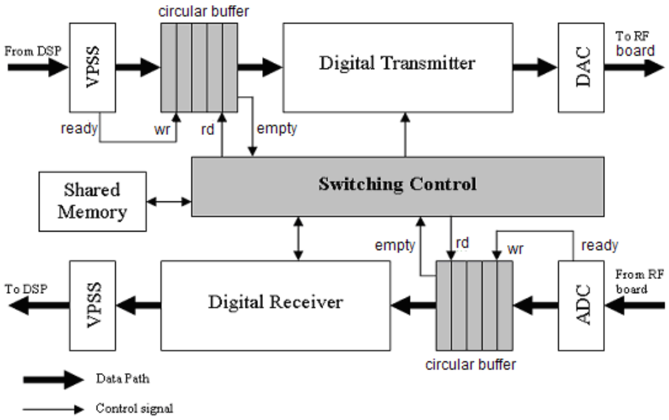

Figure 3.26 Data buffering for switching operation

The switching control circuit regulates the switching mechanisms, whether switching between digital and analog or switching between different wireless RF frequencies. For the former, the command is received manually through a push button connected to a shared memory. Upon receiving the switching command, the switching control block passes inter-connection signals to the transmitter and receiver so as to connect the appropriate filters and signal processing stage to perform digital or analog modulation. For the latter switching mechanism, the circular buffer reading operation is ceased for a sufficient period of time to perform the switching operation. The buffer size is determined by the switching settling time.

### Synchronization

Another challenge caused by switching is the loss of synchronization between the two communicating wireless nodes. This justifies choosing differential coding for digital modulators and single side band AM analog modulators. The SSB-AM demodulator is less sensitive to phase shifts in the analog modulation receiver than the coherent AM demodulator.

For coherent digital demodulators, there are two types of synchronization: 1.) phase/frequency synchronization and 2.) symbol synchronization. The differential coding in DBPSK eliminates the need for phase synchronization at the expense of SNR degradation of 3 dB.

Symbol synchronization can be replaced by a high oversampling ratio at the pulse shaping stage (8 samples). Hence, the averaging mechanism for the "integrate and dump circuit" compensates a portion of performance degradation at the benefit of reducing the hardware complexity.

### Wireless Channel and Antenna Characteristics

Automatic channel frequency switching, i.e. the frequency response of the wireless channel and antennas, serves as an additional design complexity. Wireless channels are band-limited, and their response varies among different RF frequencies. Propagation loss and multi-path characteristics are wireless channels properties dependant on carrier frequency.

Operating over a wide range of frequencies requires the utilization of wide range antennas. The SDR platform used is equipped with antennas optimized at the range of 450-490 MHz, and the operating channels used in the proposed design fall within this range. To perform wider range operation, static gain scheduling or AGC is needed to control the signal level at the input of the analog-to-digital converter and compensate the degradation in the antenna sensitivity outside its optimized operation range.

## 3.5. Adaptability and Configurability in Cognitive Radio Design on Small Form Factor Software Radio Platform

The previous sections of this chapter discuss the design and implementation of configurable radio block, including the modulation and symbol synchronization blocks, and propose a mode-switching architecture. This section develops a system-level design and implementation framework for building cognitive radios on a small form factor platform with heterogeneous processing architecture and shows how to fit these configurable building blocks in the design and implementation process of a cognitive radio system based on SFF-SDR platforms.

With respect to the system level, two important design concepts for cognitive radio systems should be featured in cognitive radio systems, namely configurability and adaptability. Configurability is defined as the ability to change the communication mode, as  inferred from the definition of cognitive radio. Communication mode is defined by operation parameters, among them modulation type, carrier frequency and transmission power. Adaptability enables an automatic communication mode selection based on wireless channel conditions, thus maintaining predefined operation constraints, including bit error rate, power consumption, and interference temperature, among others.

The current literature presents a variety of software radio computing architecture used to develop cognitive radios [25-27, 30, 31, 102-105]. Most utilize configurability and adaptability concepts in their implementation efforts. Even though a variety of hardware platform designs are suggested, most SDR merely report overall system description and performance results without providing sufficient details regarding implementation challenges and design techniques used in building cognitive radios.

This work focuses on design and implementation issues of a cognitive radio physical layer on small form factor platforms, as well as the projection of design concepts (configurability and adaptability) into a practical implementation procedure. A case study in section 3.6 illustrates the practical execution of these concepts.

### *3.5.1. Software Defined Radio Architecture*

A chief determinant in choosing software defined radio as an implementation platform for cognitive radio is its superior configurability. This is achieved by using software to implement all signal processing functions. Software defined radio (SDR) allows radio configuration to be carried out on the fly through software download or parameters update. A number of technical and practical limitations, however, impede the implementation of this idyllic version of SDR, especially in small form factor. One limitation is the sizable computational power needed to perform complex baseband signal processing algorithms on configurable computing architecture in real time. Another is the excessive time required to configure the radio operation mode online, thus preventing service continuity for time-sensitive applications, such as voice and video calls.

Such limitations give cause to move the execution of time sensitive functionality, e.g. most physical and data link layer operations, closer to hardware. This is achieved by exploiting hardware capabilities in the software, e.g. DSP programs, or using parameterizable hardware designs, e.g. FPGA designs.

SDR was originally proposed to perform the entire wireless communication stack in software targeting general-purpose processor-based platforms [20, 102]. In such platforms, flexibility and interoperability between different radio systems is managed by different software architectures i.e. Software Communication Architecture (SCA) [4].

Although this architecture facilitates modular paging of radio systems using object-oriented design principles, it suffers from computational overhead caused mainly by the hardware abstraction managed by Common Object Request Broker Architecture (CORBA) [106]. This computational overhead, in addition to the physical size of general-purpose processors machines—due to cooling requirement, prevent them from being a valid option for small factor software radio platforms.

Currently several computing architectures are suitable for SDR. These are characterized by tradeoffs between computational performance and programmability. Some researchers have attempted to build cognitive radios in field programmable gate arrays (FPGA) [103]. Cognitive Radio implementations based on digital signal processors have been presented [104, 105], and architectures based on various processing engine combinations, such as FPGA and GPP [25-27] or FPGA and DSP [28, 29, 31], have been proposed.

The author believes future SDR architecture will feature heterogeneous processing engines on a small form factor platform, including FPGA and DSP. General-purpose processor systems offer flexibility. Thus, the portability of software has great appeal, as does the maturity of software development tools. When coupled with multi-core and multi-processor techniques, the sizeable signal processing demand of current wireless air interface can be satisfied using a general-purpose processor solution. However, when compared with other computing devices, including FPGAs with similar processing power capability, the scalability of the GPP solution is limited due to excessive size and power consumption. In fact, a combination of heterogeneous computing devices can achieve a balance in cost, power, performance, flexibility, and reliability. This balance is achieved by appropriate task partitioning of the entire communication stack and signal processing functions for CR implementation. GPPs are tailored to perform control-oriented tasks, as well as user interface related tasks. The architecture of DSPs is designed to better perform some arithmetic and signal processing operations. DSPs also introduce a limited level of parallelisms in their instruction sets. These can be exploited by algorithm designers. Massive parallelism in signal processing capabilities is found in FPGAs. Based on the diverse nature of these processing devices, they can be placed in software-defined architecture, as shown in Figure 3.27. In this architecture, the FPGA acts as a coprocessor and hardware accelerator for DSP. The connection between the two processing devices is dependant on the interface specifications of the DSP chip. Most DSPs support a variety of communication interfaces. Some are proprietary, such as EMIF, VPSS (TI), MPX (Motorola), and Link-Port (ADI). Others are standard communication interfaces, such as PCI, RapidIO, and Hypertrnasport [107]. This heterogeneous processing architecture changes the software implementation from an algorithm sequence description to a data exchange and function control description [107].

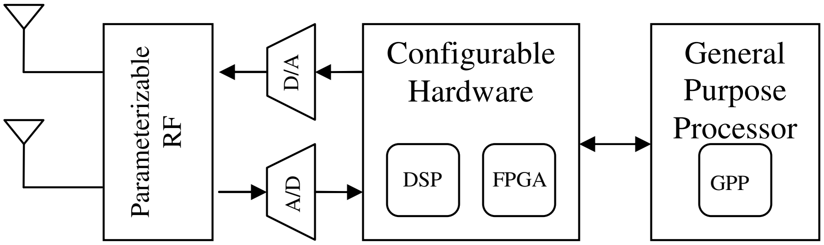

Figure 3.27 Hybrid software defined radio architecture

Several studies have addressed the task portioning of signal processing functions for wireless receivers [108, 109]. However, these studies focus on projecting a single communication mode into the heterogeneous processing architecture. Building a configurable multimode communication system is accomplished by extracting the commonality—sample rate conversion and channelization—between wireless air interfaces to build a common signal processing backbone. This backbone is easily used and can be configured by all supported wireless standards [101].

In this work, a systematic design procedure is introduced to incorporate configurability and adaptability elements into these architectures. A direct application of this procedure is presented as a practical case study in section 3.6.

### *3.5.2. Configurability*

Configurability in cognitive radio reflects the degree of freedom of its behavior. The key design technique in building configurable transceiver systems is maximizing the reuse of memory and computational resources, which differ based on the target hardware architecture. The main advantage of this technique is the reduced reconfiguration time when compared to that of the complete signal processing chain of the wireless communication mode. However, the challenge is the mechanism of online mode transitioning when building configurable systems. Figure 3.28 illustrates the building of a configurable multimode transceiver. In the figure, the wireless transceiver is comprised of a chain of function blocks described by physical layer specifications of their wireless standard. Examples of function blocks include sampling rate conversion, modulation, coding, and accessing method (TDMA, FDMA or CDMA). Each is broken into a combination of digital signal processing blocks, such as filters, signal transforms, and logical and arithmetic operations.

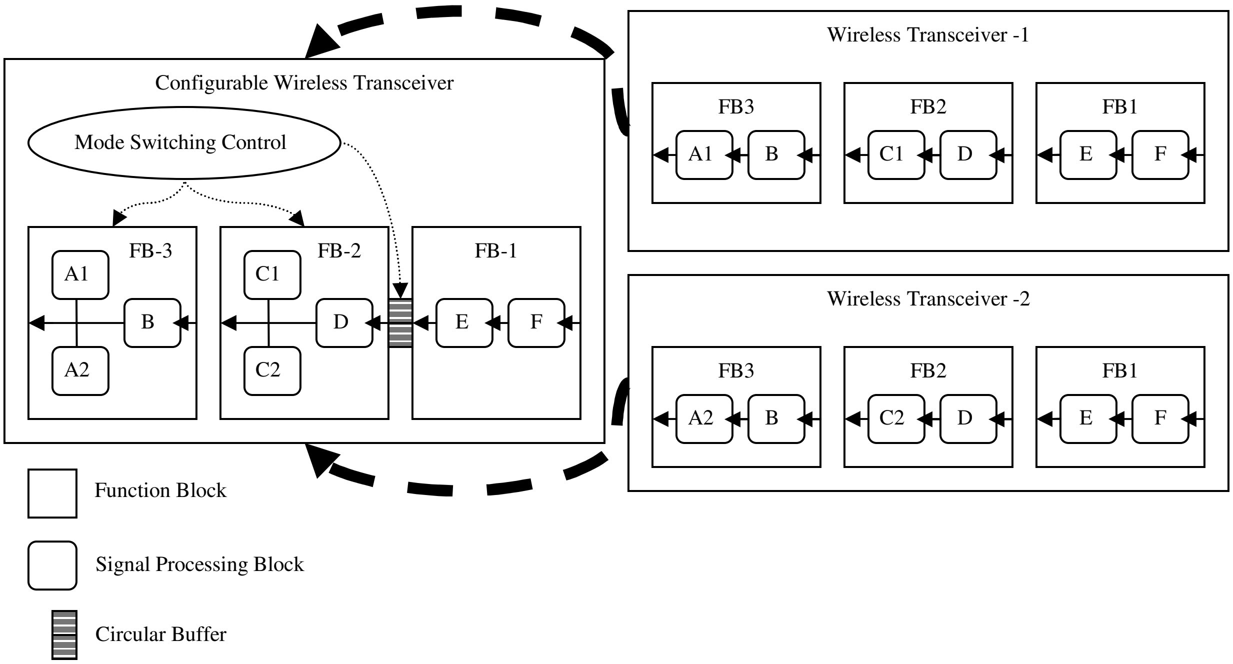

Figure 3.28 Building configurable transceiver

The following steps outline the general design approach on DSP/FPGA platforms used to build a configurable multi-mode transceiver:

- Extract the common signal processing functions between supported wireless communication modes.

The goal of exploiting commonalities of different communications modes is to maximize the reuse of digital signal processing blocks. However, this is a challenge when designing a configurable digital transceiver, especially if the modes are dissimilar, e.g. analog and digital modulation modes. To simplify the process, it is suggested that a system level analysis is followed by a bit level analysis.

In the system level analysis, the transceiver is observed as a chain of general function blocks. In Figure 3.28, wireless transceivers 1 and 2 share the same function block 1, but differ in function blocks 2 and 3. Therefore, these blocks are carried out to the next level of analysis—the bit level—to determine the common signal processing blocks based on function block parameters, including signal bandwidth, sampling rate and sample bit resolution. As shown in Figure 3.28, function block 2 in both transceivers shares signal-processing block D but differ in signal processing C1 and C2. A similar argument is applied to function block 3. This two level analysis results in a configurable transceiver with a common signal processing backbone comprised of blocks B, D, E, and F.

The implementation of this step in a heterogeneous processing engine platform involves task partitioning of the signal processing chain among different processing engines. Therefore, a thorough analysis of the hardware architecture and communication methods between the processing engines is necessary. In DSP/FPGA architectures, the signal processing functions are mapped to the DSP instruction sets and FPGA hardware resources. Task partitioning is then carried out based on two constrains: 1.) performance constrains, such as computational speed or energy consumption and 2.) communication overhead between processing engines.

- Design a mode switching mechanism.

Maintaining data integrity and service continuity during mode transitioning is critical. Data integrity is governed by holding the processed data in circular buffers at the first stage of the configurable section of the wireless transceiver. This prevents data from being processed in part both before and after mode transition. The buffer size is determined by the longest possible mode transitioning operation. Service continuity constraints define the maximum allowable buffer size.

Timing synchronization becomes a challenge during mode switching, as it is affected by seizing data processing sequence during mode transitioning and by data as it moves between modes with different timing characteristics, i.e. different symbol rates. An example for building a configurable symbol synchronizer is provided in section 3.3.

It is important to note that the configurations of some stages in the configurable transceiver may have a direct effect on the operation of other stages. For example, changing the carrier frequency will change the characteristics of the wireless channel. Propagation loss and multi-path characteristics are wireless channel properties that depend on carrier frequency. Therefore, the behavior of other stages, e.g. channel equalizer and antennas, of the configurable radio can be affected by changing this configuration parameter.

- Design the radio operation finite state machine and then define the mode transitioning triggering sources and the interlock system to avoid unstable mode transitioning.

Mode transitioning can be triggered manually by the user or automatically through the cognitive engine of the radio. In either case, it is important to recognize system stability concerns when radio communication modes share the same triggering sources. If two modes share the same triggering condition, it is possible to experience a continuous mode transitioning between the two modes as long as the triggering source is active.

### *3.5.3. Adaptability*

Adaptive radio adjusts its communication modes either to optimize a given performance parameter or to comply with certain performance constraints. The design approach of adaptive radio is summarized by the implementation of the following capabilities:

- Capability 1: Measuring performance parameter and accessing associated control knobs.

  The first step in building an adaptive radio is to determine the set of measurement parameters and control knobs governing its adaptive behavior. The radio should monitor the status of the performance parameter to determine whether to keep the current communication mode or switch to a different one to achieve an improved performance score. The reliability of the adaptive radio is dependant on the accuracy of the performance parameter measurement system. Changing communication mode requires access to the appropriate control knobs that perform the switching operation.

- Capability 2: Mode selection.

  When switching the communication mode, the radio should have the capability to find appropriate alternatives.

  One example of adaptive behavior in cognitive radio is the dynamic spectrum allocation. Assuming the performance parameter is bit error rate (BER), the cognitive radio aims to occupy a vacant radio channel so that wireless communication with limited bit error rate can be maintained. When the signal to noise ratio (SNR) of the wireless channel deteriorates as a result of excessive noise or interference, the BER exceeds its nominal range. This divergence from normal BER value is detected by cognitive radio (Capability 1), and an alternative radio channel is allocated using spectrum sensing techniques (Capability 2).

- Capability 3: Co-existence management.

  The coexistence of cognitive radios in the same wireless network presents additional difficulties when selecting an appropriate communication mode, e.g. the radio channel and transmission power. Therefore, co-existence management capability is required to avoid conflicting behavior between cognitive radios. This is achieved by using base-stations located in a centralized cognitive network, such as IEEE 802.22 standard, or through a dedicated broadcasting channel in a distributed cognitive network.

Figure 3.29 summarizes the key design techniques introduced in this section.

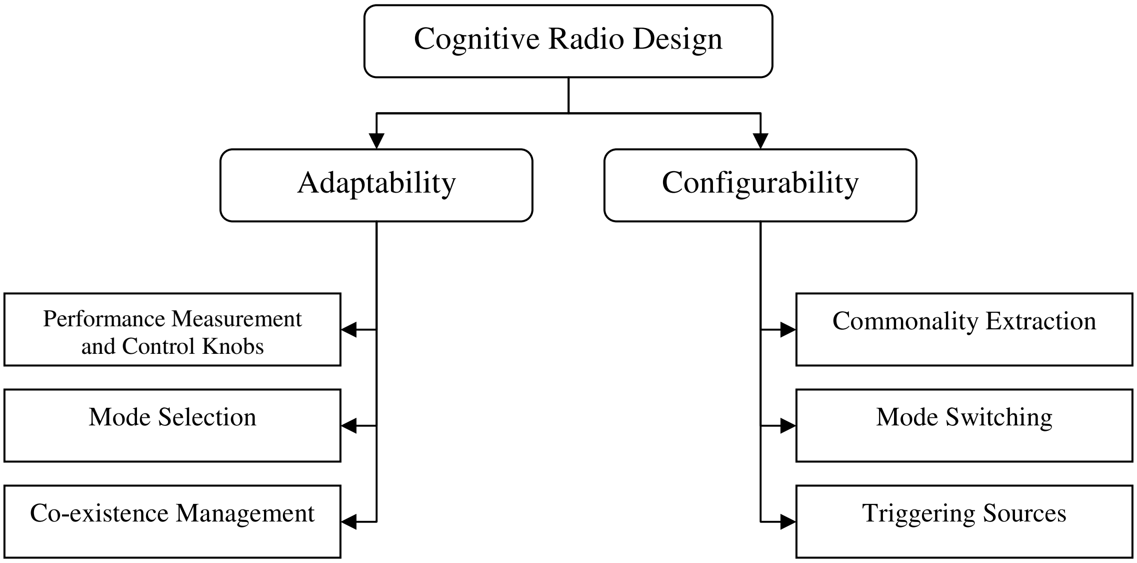

Figure 3.29 General cognitive radio design steps

## 3.6. A Case Study: Rapidly Deployable Heterogeneous Wireless Network

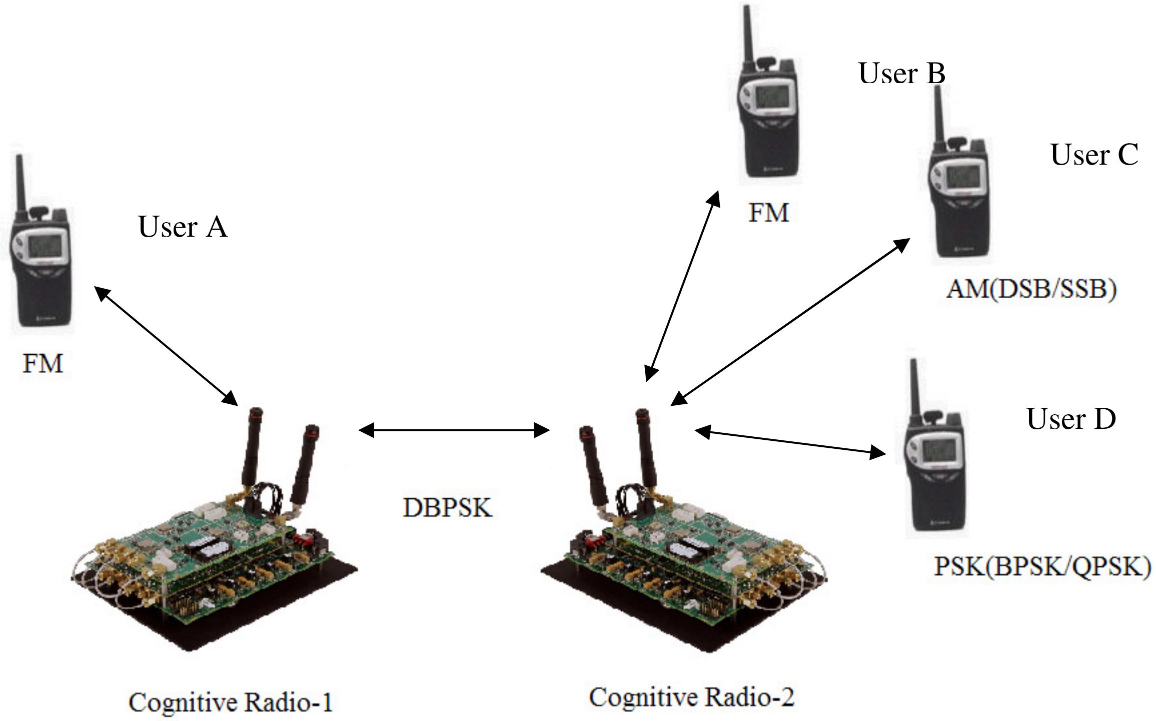

Figure 3.30 Case study system overview

Reusing licensed spectrum is the most widely utilized cognitive radio application employed to solve spectrum scarcity. The case study depicted in Figure 3.30 shows a practical implementation of this application based on the systematic design approach discussed earlier in section 3.5. The goal of this study is to rapidly establish a wireless network between a heterogeneous set of radio terminals to establish a communication infrastructure following either a natural disaster or in a geographic area with limited communication infrastructure. This network is realized by using cognitive radio terminals. In Figure 3.30, user A needs to communicate with either user B: FM modulation, user C: AM modulation, or user D: PSK modulation.

The communication service in Figure 3.30 is voice messaging. The utilized spectrum is assumed to be licensed to a set of primary users. Therefore, all radio terminal users are considered secondary and characterized with a lower priority than that of the primary users to occupy the wireless channel. It is assumed that user A is outside the range of other users and that the connection to one of the users is established through a set of two cognitive radio terminals 1 and 2, where cognitive radio 1 is connected wirelessly to user A and cognitive radio 2 is connected to the remainder of the users. The set of cognitive radios are connected using DBPSK modulation, which is chosen for its immunity to noise. Also, differential encoding is used to relax the carrier synchronization requirements at the receiving cognitive radio.

User A sees each user as an individual FM channel. For example, user B is on channel 2; user C is on channel 3; and user D is on channel 4. Therefore, if user A chose to communicate with user C, user A would transmit over channel 3. Cognitive radio-1 would then receive user A data (voice samples) and would forward it along with additional information about the destination address (channel 3 in this case) through a dynamically allocated wireless channel that has been assigned dynamically. Cognitive radio-2, which is located at the other side, would receive the data from cognitive radio-1, thus recognizing the destination address (channel 3) as user C. As a result, the cognitive radio reconfigures its communication mode to comply with user C communication mode, and then forward the data received accordingly. If the cognitive radios are required to change their communication channel due to interference, noise or channel occupancy by the primary user, they would terminate the current communication channel, switch to an alternative channel, and then resume the communication accordingly.

This example carries a considerable amount of practical detail that necessitates the implementation of complete communication stack layers. However, the author has chosen to focus on the physical layer and limit the data link layer functionality discussion to emphasize the concept of implementing adaptability and configurability features in the physical layer of cognitive radios. These features are illustrated as follows:

- Configurability.

User A sees the remaining users as FM users at different radio channels. Heterogeneous modulation techniques are hidden under an abstraction layer created by cognitive radio-2. The latter has a configurable modulation transmitter and receiver that communicate with radio sets of different modulation techniques. The configurable transceiver architecture is built each of the three modulations, namely AM, FM and PSK, is analyzed to find the commonality between the modulation techniques, and then to build the common signal processing backbone of the configurable transceiver. The ability to switch between different communication modes, e.g. modulation types, demonstrates the configurability feature of cognitive radio.

- Adaptability.

Cognitive radio monitors the performance parameter, e.g. bit error rate, during voice conversation between user A and other users (B, C or D). Cognitive radio maintains the allocated channel connection as long as the performance parameter is within defined constraints. Once the bit error rate exceeds a certain threshold, both cognitive radios will switch to an alternate channel. the agreement of which is achieved through periodic spectrum sensing and exchanging sensing reports through a dedicated logical channel. The alternative channel is the closest common channel available to the current channel between both cognitive radios. This ability to adjust the radio operation parameter (i.e. carrier frequency) based on a given performance metric (i.e. bit error rate) demonstrates the adaptability feature of cognitive radio.

The following sections highlight hardware considerations carried out through the design process. Next, the implementation of configurability built upon a configurable AM/PSK transceiver is illustrated. The FM configuration is omitted to simplify the illustration. Subsequently, the adaptability implementation is discussed by building two subsystems: error detection and spectrum sensing. Implementation results are reported at the end of the case study.

### *3.6.1. System Design Overview*

Using the Lyrtech® SFF-SDR platform described in appendix, the case study illustrates the implementation of a cognitive radio transceiver in which its communication modulation and frequency carrier can be altered while maintaining operation. The cognitive radio is comprised of three subsystems: 1.) configurable digital transceiver; 2.) spectrum sensing; and 3.) channel switching subsystems, as shown in Figure 3.31:

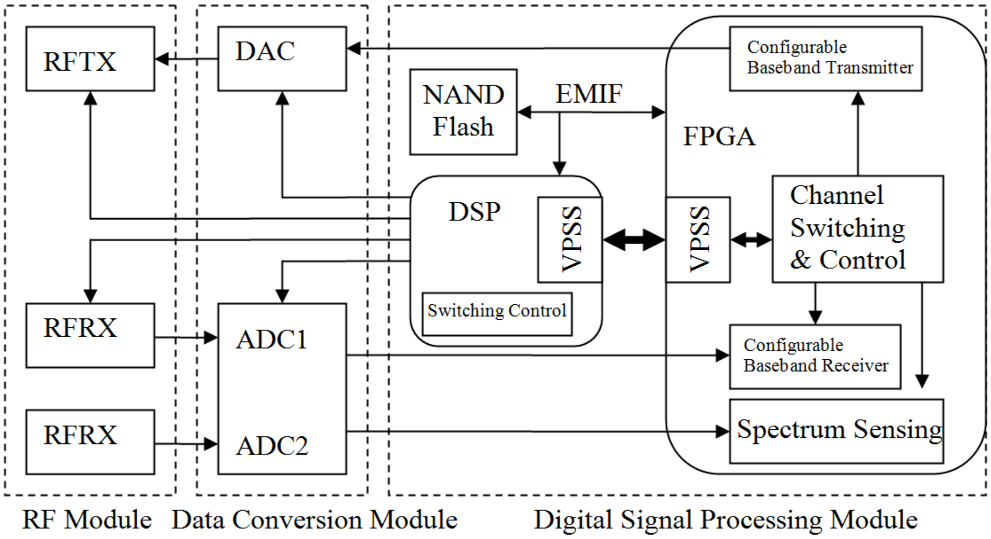
RFTX
DAC
Configurable Baseband Transmitter
NAND Flash
EMIF
FPGA
DSP
VPSS
VPSS
Channel Switching & Control
Switching Control
RFRX
ADC1
Configurable Baseband Receiver
RFRX
ADC2
Spectrum Sensing
RF Module
Data Conversion Module
Digital Signal Processing Module

Figure 3.31 Function block diagram of the cognitive radio system using a software defined radio platform

Generally, the DSP is dedicated for communication, control and packet handling tasks. Digital signal processing intensive operations, such as filtering and up/down conversion, are performed in FPGA. Therefore, part of the digital transceiver is implemented in the DSP, while most of the digital transceiver design, as well as the complete spectrum sensing, is implemented in FPGA. The channel switching and control subsystems are task partitioned between DSP and FPGA. In the following section, a detailed description of the spectrum sensing, channel switching and control subsystems are listed.

### *3.6.2. Design Constraints*

It is important to detail the performance specifications of the wireless device at the beginning of the design process. Table 3.12 lists a summary of transmitter specifications.

Table 3.12 Performance specification for configurable transceiver

| Parameter | Value |
|---|---|
| Transmit Spectral Mask | 40 dB for adjacent frequencies |
| Channel Bandwidth | <1 MHz |
| Baseband Rate | 15.625 ksps or 156.25 kbps |
| Input signal quantization | 10 bits |
| Output Signal Quantization | 16 bits |
| IF sample Rate | 80 MHz |

The choice validity of listed parameter values is discussed below. The transmit mask is dependant on the wireless standard. The range of 40 dB attenuation for outband signals is generally acceptable to transmit a good-quality wireless signal in a wireless channel condition. However, some wireless standards, including WCDMA, require more attenuation (up to 70 dB) for far adjacent signals. The author chose a simple spectral mask (see Figure 3.32) as a guideline for the design process.

The bandwidth of the channel shall be less than 1 MHz, which is sufficient to transmit 156.25 Kbps using BPSK modulation scheme or an analog AM-SSB voice signal sampled at 15.625 KHz. This bandwidth can be utilized without the need for equalization in an urban environment with rms delay spread < 8us (coherent bandwidth > 700kHz).

Also, quantizing baseband signals with 10 bits word resolution is sufficient for most wireless applications [17]. Further analysis of the 10 bits selection is discussed in section 5.3.1. The output signal quantization is the full resolution of the digital-to-analog converter.

To simplify the design, the transmitter IF sampling rate is chosen to be identical to the receiver IF sampling rate, thus eliminating the need to produce different clock rates for data converters. The decision of the IF sampling rate is dependant on the value of the IF frequency. To avoid overlap between the IF harmonics due to the nonlinearity of the mixer, the sampling frequency is chosen to be a non-integer multiple of the IF frequency.

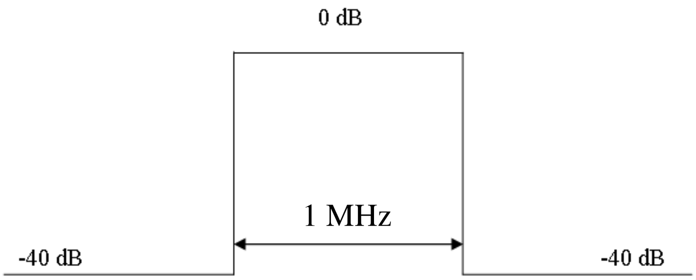

Figure 3.32 Transmit spectral mask

### *3.6.3. Configurability Implementation Procedure*

Two communication modes (or control knobs) can be configured in the example. The first is the carrier frequency, which is configured by adjusting the frequency synthesizer of the RF module using control signals from the digital module. The second is the modulation type. This paper reports the building of a digital baseband transceiver with configurable modulation types. However, designing a flexible RF module to demonstrate the first adjustable communication mode is beyond the scope of this dissertation.

A configurable modulation transceiver can be built by finding commonality between modulation techniques necessary to build the common signal processing backbone and the configurable signal processing blocks that hold the differences between the modulation techniques.

After building the configurable multimode architecture, a mode switching control mechanism is shown to preserve the data during mode transitioning.

### Building the configurable architecture

This section details the approach taken to determine the common structure between AM and PSK transceivers. It suggests how to avoid implementing two independent transceivers in the software radio platform. The approach is performed in two stages:

The first is to determine the high-level view of the data path for different modulation techniques. The second is to determine the bit-level design of the common structure based on design parameters, such as signal characteristics, sampling rate and sample bit resolution.

The purpose of the high level view is to gain an understanding of design constraints resultant of hardware limitations and also to determine the possible commonality between different communication modes. Figure 3.33 illustrates a high-level view of the common structure of narrowband digital wireless transceivers.

The transceiver signal processing functions inside the dashed box are implemented in the heterogeneous processing engines (DSP and FPGA) of the Lyrtech development platform and accomplished by task partitioning. The initial design constraint is that the raw base band signal resides in DSP because the DSP Ethernet port is used to interface the SDR platform with host machines that can be used for signal generation and display. Also, the IF signal is located in FPGA because FPGA is interfaced with digital converters that operate at the IF level. Both the IF and baseband signals are sampled with different rates; therefore, the communication chain encounters sampling conversion. According to design constraints in section 5.2, the sampling conversion ratio required to convert the sampling rate from 15.625 kHz (baseband signal sampling rate) to 80 MHz (digital converter sampling rate) is 5120. Base on this constraint, task partitioning is performed as follows:

- DSP Tasks: framing, modulation/demodulation and partial sampling conversion operations.
- FPGA tasks: sampling conversion and digital up/down conversion operations.

It should be noted that synchronization and carrier recovery stages are needed at the receiver when using coherent receiver structures. However, in the following implementation process, a perfect timing synchronization is assumed. The process of building configurable synchronizers can be found in section 3.3.

When inspecting the high-level view of the signal processing chain, it is found that many stages can be reused. However, the modulation stage and sampling rate conversion should be carefully designed to increase the hardware resource reuse and support both modulation techniques. This is addressed at the next stage of the design.

The second stage is the bit-level AM/PSK transceiver structure design. Similar analysis can be carried out for FM transceiver bit analysis to be included with the balance of modulations in the configurable transceiver. However, to simplify the analysis the author has limited the discussion for AM and PSK. The following design elements are considered:

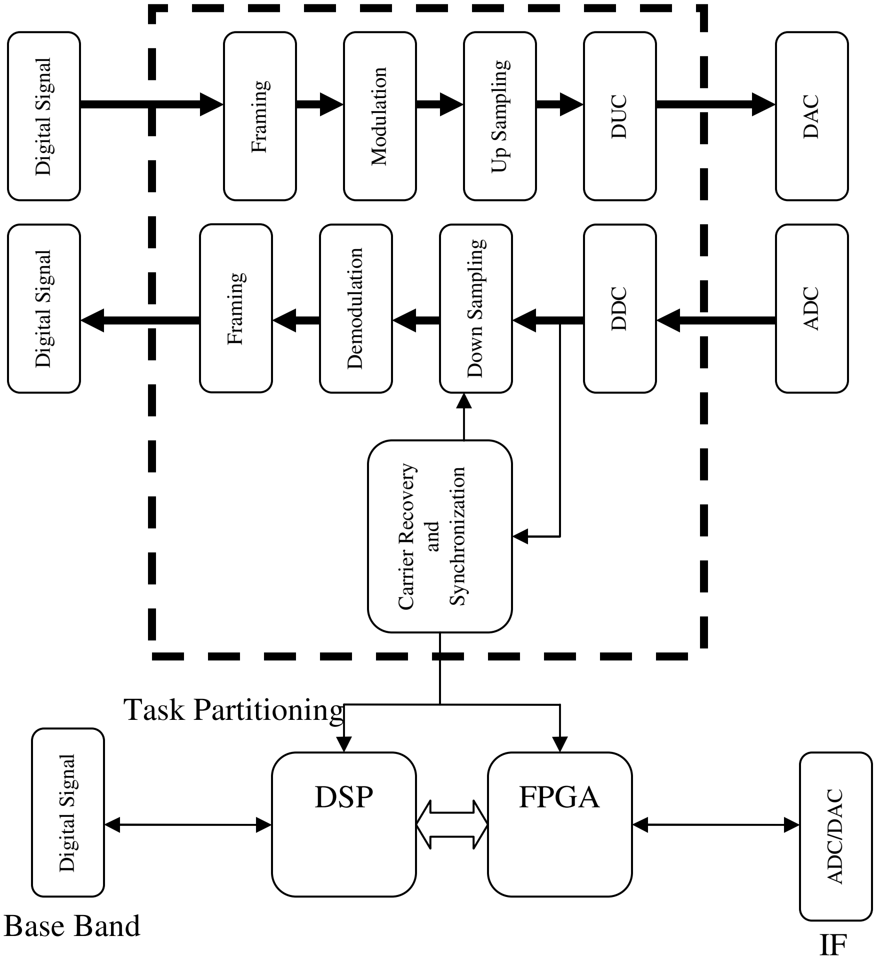

Figure 3.33 The common structure of the digital portion of the wireless transceiver

- Transmitted signal characteristics (bandwidth).

One of the most important design considerations is the transmitted signal bandwidth. By having a sampling rate of 15.625 kHz, it is possible to have two signals: 1.) the analog modulated signal, which is limited to 15 kHz bandwidth approximately; and 2.) the digital modulated signal, which is limited to 15.625 $\frac{ksample}{\sec ond}$ x 10 $\frac{bit}{sample}$ =156.25 Kbps. Upon using an excess bandwidth factor of 0.5 in the pulse shaping stage, the occupied bandwidth used by the digital modulated signal is 234.375 kHz. This baseband signal bandwidth is the largest among all possible baseband signals that may pass through the signal processing chain. Therefore, it identifies the Nyquist boundaries for sampling rate conversion stages.

- Sample rates conversions.

A design trade-off exists between transceiver supported data rates and energy consumption. Software radio design utilizes multi-rate signal processing techniques to manipulate the trade-off and achieve a balance between performance and energy consumption. Regarding the design of a multi-rate system, it is important to comply with anti-aliasing constraints. The common structure is conservatively designed for the widest possible bandwidth, i.e. 234.375 kHz. Designing for this scenario provides the opportunity to reuse a portion of the signal processing blocks for alternate communication modes, which run at less bandwidth, i.e. the AM signal.

- The resolution (bit size) used at different stages of the digital portion of the transceiver.

The digital portion of the wireless transceiver is comprised of various digital components, including DSP, FPGA, and ADC/DAC, each with different capacities and resolutions. DSP is a 32bit digital component and is connected to FPGA through a 16-bits VPSS bus. (Interested readers can find the buffering and communication mechanism of VPSS bus in [14]). Due to its flexible structure, FPGA can manage a variety of bit-sizes, e.g. DAC has a 16 resolution, and ADC has only 14 bits. Most wireless communication waveforms utilize 10- to 12-bit resolution in their baseband processing level. The configurable transceiver in this work uses 10-bit resolution based on ADC SNR limitations [14]. Assuming that n is the Effective Number of Bits (ENOB), the relation between n and SNR is given by equation 3.6 [110]:

$$SNR = 10\ log_{10}(1.5\ x\ 2^{2n}) \qquad (3.6)$$

The SNR of the ADC is approximately 60dB for many ADC operation modes specified in its datasheet. Hence, 10 bits becomes the effective number of bits of a 14-bit ADC with a signal to noise ratio (SNR) of 60 dB.

The hybrid transmitter design is illustrated in Figure 3.34. The hybrid receiver design is shown in Figure 3.35.

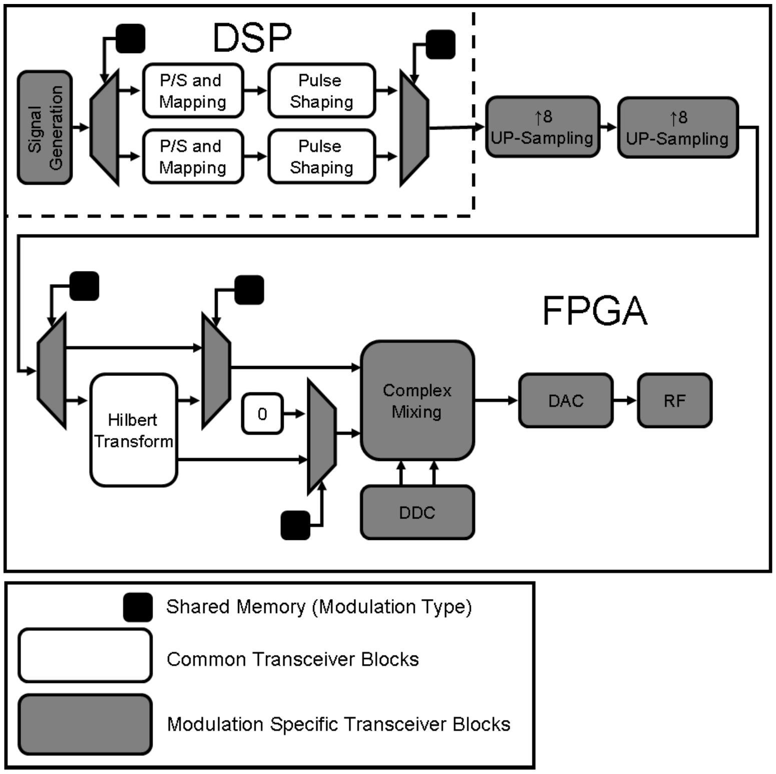

Figure 3.34 Hybrid AM/PSK transmitter structure

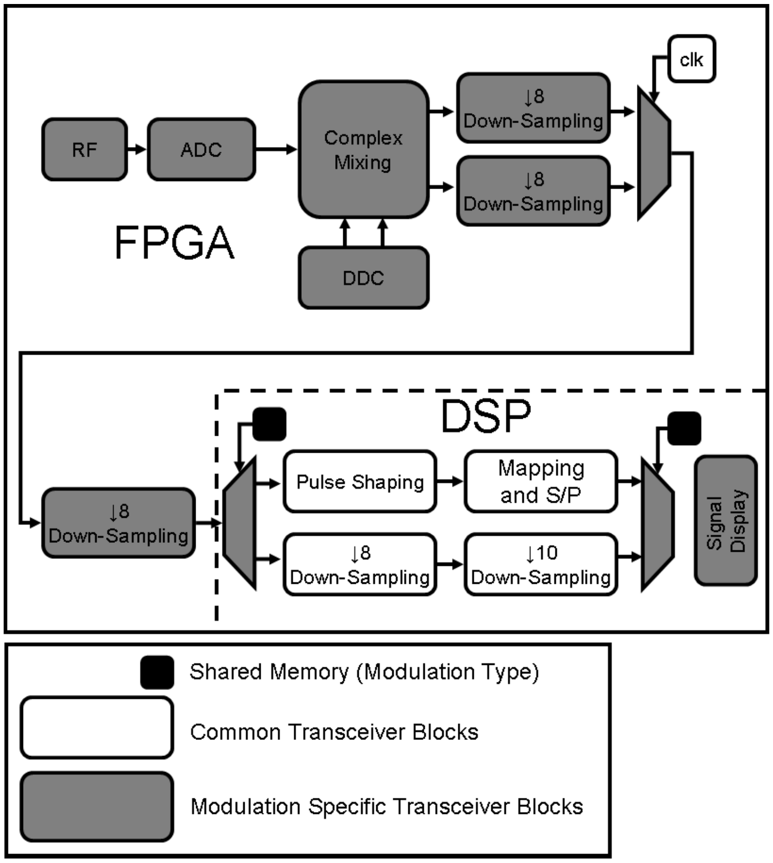

Figure 3.35 Hybrid AM/PSK receiver structure

Common transceiver signal-processing blocks include the signal generation and display, up/down conversion, DAC, ADC, and RF board. Bit conversion stages are omitted for simplification.

The up/down conversion stage is comprised of up/down-sampling and mixing subsystems. In order to share the up/down-sampling subsystem, it is necessary to determine the up/down-sampling ratio for each communication mode. This ratio is calculated by finding the maximum common divider of total up/down sampling ration for each communication mode. In the configurable transceiver shown in Figure 3.34 and Figure 3.35, the AM transmitter requires a total up/down-sampling ratio of 5120, while the QPSK/BPSK transmitter requires 64. This is so because the mapping and pulse shaping stage involves a sampling rate conversion of 80. Therefore, the common up/down-sampling subsystem is 64, which is the maximum common divider of 5120 and 64. This common sampling rate conversion is implemented in the FPGA in two 8-sampling rate conversion stages since this configuration consumes the minimum hardware resources available in the various down conversion configurations. Table 3.13 shows the total order of the interpolation filters for different configurations:

Table 3.13 The interpolation filter order for different up-sampling configurations

| | Configuration<br>Stage1xStage2 | 64 | 8x8 | 16x4 | 4x16 | 32x2 | 2x32 |
|---|---|---|---|---|---|---|---|
| Stage 1 | Fs [MHz] | 80 | 10 | 20 | 5 | 40 | 2.5 |
| | Fstop [MHz] | 1.25 | 1.25 | 1.25 | 1.25 | 1.25 | 1.25 |
| | Filter order | 259 | 32 | 65 | 14 | 130 | 4 |
| Stage 2 | Fs [MHz] | | 80 | 80 | 80 | 80 | 80 |
| | Fstop [MHz] | | 9.9531 | 19.9531 | 4.9531 | 39.9531 | 2.4531 |
| | Filter order | | 31 | 14 | 63 | 3 | 130 |
| | Total Order | 259 | 63 | 79 | 77 | 134 | 134 |

Based on task partitioning discussed earlier, the signal processing functions are distributed between DSP and FPGA, as shown in Figure 3.34 and Figure 3.35.

Note that symbol synchronization can be replaced by a high oversampling ratio at the pulse shaping stage—8 samples. Hence, the averaging mechanism using the “integrate and dump circuit” compensates for a portion of performance degradation at the benefit of reducing hardware complexity. Also equalization is unnecessary since the bandwidth of the channel (< 250 KHz) is small enough to assume that coherence bandwidth is larger than channel bandwidth.

The DAC, ADC, and up/down conversion stages are mutually dependant, and the sampling rate of the ADC and DAC determines one boundary of the total sampling rate conversion ratio. In the configurable transceiver, the sampling rates of the ADC and DAC are identical and fixed—80 MHz.

An RF board can be designed to achieve a certain selectivity and sensitivity for all communication modes. It supports the maximum possible signal bandwidth of any communication mode.

### Switching control design

Online mode switching presents challenges for cognitive radio design. In the proposed configurable transceiver two modes of switching are presented, namely switching between modulations and switching between different radio channels. Data integrity is the foremost challenge during mode switching.

It is important to design a switching mechanism that protects the transmitted data from loss or corruption during switching. This is achieved by buffering the data during the switching interval.

Since the configurable transceiver is implemented in DSP and FPGA, the mode switching occurs in the signal processing functions in both processing engines. Although mode switching inside DSP is easily managed by software, it is more challenging in FPGA. Therefore, transmitter and receiver circular buffers are used in FPGA to maintain data integrity during mode switching. The design of data buffering inside FPGA is shown in Figure 3.26.

For generic switching mechanism design, circular buffers are placed outside the transceiver design. One is placed after the VPSS port at the transmitter side and the other after the interface with ADC at the receiver side.

At the transmitter side, the VPSS port provides a “ready” control signal subsequent to passing a new data word from DSP. This signal is connected to a write-enable port of the buffer. The switching control circuit provides a read-enabled control signal when the switching operation is idle and the buffer is not empty. If the buffer is full, a “full” control signal is passed from the buffer to shared memories (absent in Figure 3.26) to notify the DSP that the transmitter is unable to transmit more data.

At the receiver side, the circular buffer is placed after the ADC. Similarly, the data converter provides a “ready” control signal subsequent to passing a new digitized sample (14 bits) from the analog domain. This signal is connected to a write-enabled port of the buffer. The switching control circuit provides a read-enabled control signal when the switching operation is idle and the buffer is not empty. If the buffer is full, the received sample is dropped.

The switching control circuit regulates the switching mechanisms, whether switching between modulations or different wireless RF frequencies. The switching command is provided by the DSP through shared memory, as shown in Figure 3.26. In all occurrences of mode switching, the switching control block holds the data in the circular buffer until the switching operation is accomplished. However, there are additional details pertaining to mode switching type.

In the instance of modulation switching, the switching control box holds data in circular buffers until the processing in last stage of the old modulation mode is accomplished. This is followed by switching to the new modulation mode. The switching control block passes inter-connection signals to the transmitter and receiver to connect the appropriate filters and signal processing stage. Thus, the digital or analog modulation is performed.

In the instance of radio channel switching, the switching control regulates the spectrum sensing operation, as discussed in section 5.4.2, and data are held in circular buffer until spectrum sensing is accomplished and carrier frequency switching is performed.

The buffer size is determined by the longest mode switching duration, i.e. the radio channel switching, as it performs a lengthy spectrum sensing operation.

### *3.6.4. Adaptability Implementation Procedure*

The adaptive behavior of cognitive radio in this case study is the selection of the radio channel. Therefore, the cognitive radios require: 1.) the capability to detect the event that necessitates channel switching, and 2.) the ability to find a vacant radio channel. These are discussed in the following sections.

<u>Interference detection capability</u>

The necessary condition to change radio channels is the degradation of wireless channel quality due to the existence of interference, noise or the transmission of a spectrum primary user. Wireless channel quality is defined as the suitability of a wireless channel to carry the transmitted signals with an acceptable received error rate. Direct and indirect techniques can be used to estimate the wireless channel condition. See section 3.4.3. In this case study, a repetitive coding technique for error detection is implemented, as it is simple with only minor changes to the design of the configurable transceiver architecture. The cyclic redundancy coding method, however, is more efficient in detecting errors caused by noise in transmission channels [111]. The functionality and switching control circuit are detailed in section 3.4.3.

Spectrum sensing capability

Figure 3.36 shows a simplified block diagram of the spectrum sensing subsystem. The heart of this subsystem is the FFT core provided by Xilinx [17]. The FFT core used in this design computes an N-point forward DFT where N can be $2^m$, m = 3 to 16. The input data is a vector of N complex values represented as bx-bit two's-complement numbers – bx bits for each of the real and imaginary components of the data sample (bx = 8 to 24). Similarly, the phase factors bw can be 8 – 24 bits wide. All memory is on-chip using either Block RAM or Distributed RAM. The N element output vector is represented using by bits for each of the real and imaginary components of the output data. In addition, the core supports three architecture options:

1.) Pipelined, Streaming I/O, allowing continuous data processing.

2.) Radix-4, Burst I/O, offering a load/unload phase and a processing phase; is smaller in size but has a longer transform time.

3.) Radix-2, Minimum Resources, using a minimum of logic resources in a two-phase solution.

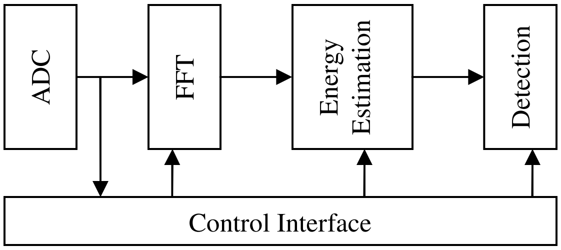

Figure 3.36 A simplified block diagram of the spectrum sensing subsystem

In the proposed design, the core is configured to perform pipelined streaming I/O, which has the ability to simultaneously perform transform calculations on the current frame of data; to load input data for the next frame of data; and to unload the results of the previous frame of data. The core has an online dynamic FFT length configurability, which enables the multi-resolution capability of the spectrum sensing subsystem in the proposed cognitive radio.

Figure 3.37 shows a detailed view of the spectrum sensing subsystem and the various control signals used in the model. In the following sections, the sensing modes of this subsystem are described.

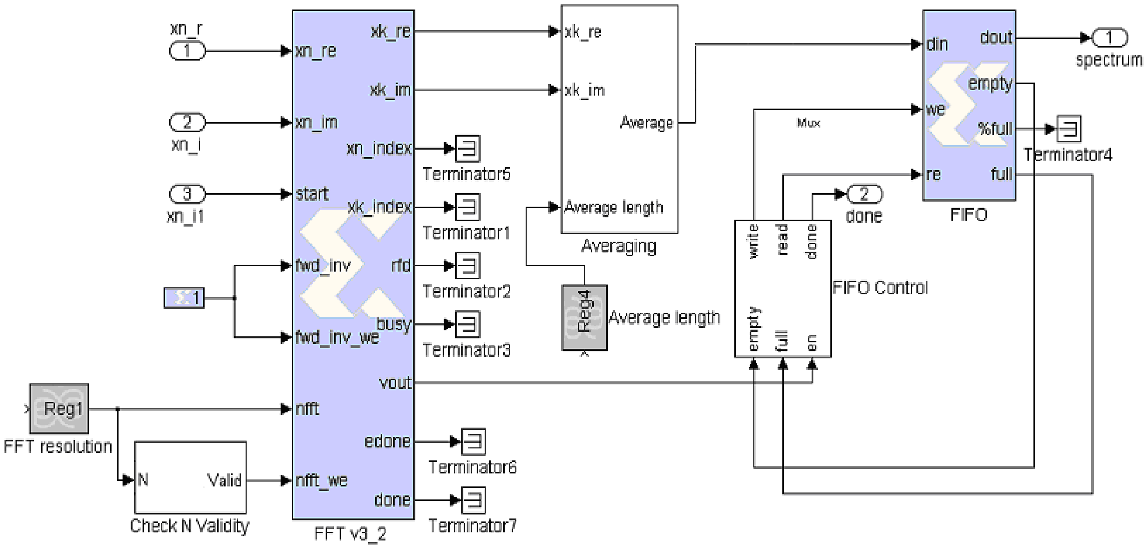

Figure 3.37 A detailed view of the spectrum sensing implementation using Simulink®, Xilinx System Generator® and Lyrtech® development tools

The spectrum sensing is based on energy detection by averaging the FFT output amplitude over different cycles. Figure 3.37 shows that the FFT output is directed to the averaging where a preconfigured averaging window length M can be specified. The output of the averaging is loaded into a first-input-first-out (FIFO) memory block. The FFT block provides a complete DFT output every N clock cycle, where N is the size of FFT. Hence, an averaged FFT output is calculated every N.M cycles. While it is possible to have a pipelined implementation of the sensing operation that allows an FFT output every N cycles, the decimation operation by M that is performed by the averaging filter is chosen due to the maximum VPSS bus speed limitation. The maximum speed from FPGA to DSP is 150 Mbps (16 bits, 75 MHz), while the maximum speed from DSP to FPGA is 75 Mbps (16 bits, 37.5 MHz).

### Channel Switching and Control Subsystems

The channel switching and control subsystem controls channel frequency switching as a result of adjusting the VCO in the analog RF board. During the spectrum sensing operation, the subsystem tunes the VCO so that the cognitive radio receiver scans the entire shared spectrum. The FFT "done" control signal is used to trigger the increment (or decrement) of the VCO frequency controller, taking into consideration the averaging length M stored in a shared memory register so a VCO frequency shifting of 20 MHz (the receiver bandwidth defined by the passband filter in the RF board) is performed every M done signals.

It is important to note that the capability of scanning the entire RF range requires the use of two SDR RF receivers so as to implement the complete cognitive radio terminal: one receiver is used for spectrum sensing and is connected to one channel of the dual channel ADC and the other RF receiver is part of the configurable RF transceiver in the cognitive radio and is connected to the other channel of the dual channel ADC. The control words for VCO are passed to the DSP through shared memory registers. Figure 3.31 highlights the control signal sent from DSP to the data conversion board and radio boards.

### *3.6.5. Implementation Results*

This section lists the hardware resource usage and cognitive radio implementation results. The implementation is performed in three stages: 1.) spectrum sensing module, 2.) configurable transceiver, and 3.) mode switching control implementation stages.

<u>Spectrum sensing module implementation</u>

In the first stage, a spectrum-sensing module is built completely in FPGA, while the control mechanism, including the bit error rate detection, is implemented in DSP. It should be noted that the author implemented the spectrum sensing operation within a range of 20 MHz due to the lack of a second RF board on the SDR platform. A second RF receiver is needed so that a complete spectrum scanning by one receiver and communication over the control channel using the other can occur simultaneously. In the case study, the author used a single receiver for each SDR platform. Therefore, the spectrum sensing is performed within a range of 20 MHz. To demonstrate the functionality of the spectrum sensing module, a complete module is implemented in a separate SDR platform where the ADC clock used is 125 MHz.

Figure 3.38 and Figure 3.39 show a snapshot of the spectrum sensing output using 1K FFT. Two signals were generated at three frequency bins, as shown in figures 16 and 17. The first is an analog narrow-band signal (pure sinusoid); the second is a wideband

DSSS generated at 5 and 7 MHz. These signals are generated in the frequency bin 440-460 MHz and 500-520 MHz. The frequency span on these plots range from 0 to 62.5 MHz with an increment of 61 KHz.

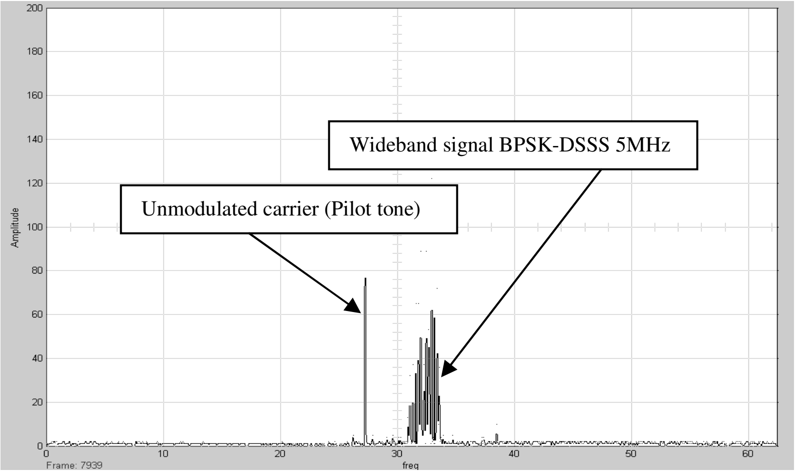

Figure 3.38 A snap shot of the spectrum sensing result and frequency bin of 440-460 MHz

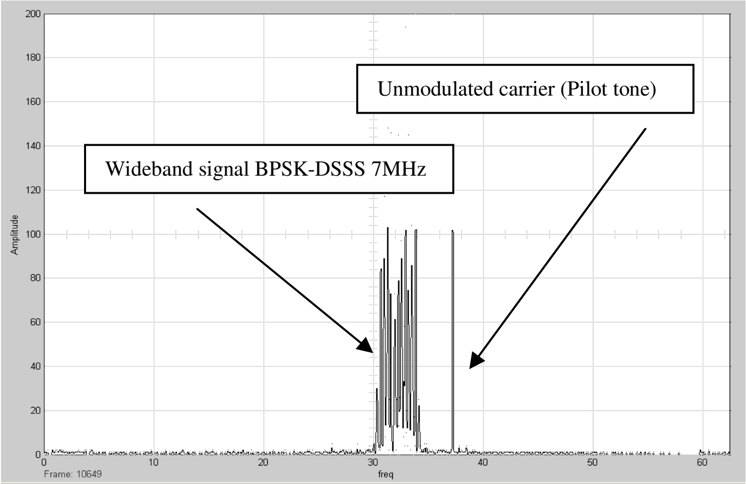

Figure 3.39 A snap shot of the spectrum sensing result and frequency bin of 500-520 MHz

Configurable transceiver implementation

Table 3.14 shows the hardware resources consumed by two implementation cases, exclusively in FPGA. The first is the implementation with task partitioning; the second is without task portioning.

Table 3.14 Configurable transceiver FPGA resources usage

| | Configurable Transceiver Without Task Partitioning | Configurable Transceiver With Task Partitioning |
|---|---|---|
| Number of Slices | 8212 out of 15360 (53%) | 6,205 out of 15,360 (40%) |
| Number of Slice Flip Flops | 14623 out of 30720 (47%) | 8,017 out of 30,720 (26%) |
| Number of 4 input LUTs<br>Number used as logic<br>Number used as Shift registers | 16487 out of 30720 (53%)<br>8201<br>8286 | 5,135 out of 30,720 (16%)<br>5,135<br>5,177 |
| Number of FIFO16/RAMB16s<br>Number used as RAMB16s | 1 out of 192 (0%)<br>1 | 5 out of 192 (2%)<br>5 |

Mode switching control implementation

The switching control implementation stage is tested by applying a known interference on the communication channel. Bit error rate and switch time are tested. Figure 3.40Figure 3.41 demonstrates the testing setup: Cognitive radio-1 sends voice traffic over a chosen channel (430MHz) and receives BER feedback over the control channel on 400 MHz. A computer is connected to cognitive radio-2 through the Ethernet port and logs the BER calculations and channel switching delay. In this test, it is assumed that a list of available channels has already been provided by the spectrum sensing module. Hence, upon interference detection, both cognitive radio modules switch to the closest common vacant channel to the current noise channel.

To test switching behavior, the computer triggers the signal generator, applying signal interference between cognitive radio-1 and cognitive radio-2 at the current wireless channel (430MHz). The cognitive radio sets detect interference if the BER measurements exceed 0.2 and subsequently switch to an alternative channel. After channel switching is performed, BER calculations are reset. Cognitive radio-2 notifies the computer when the interference is detected and subsequent to channel switching. Figure 3.41 shows simulated measurement results recorded in the logging computer.

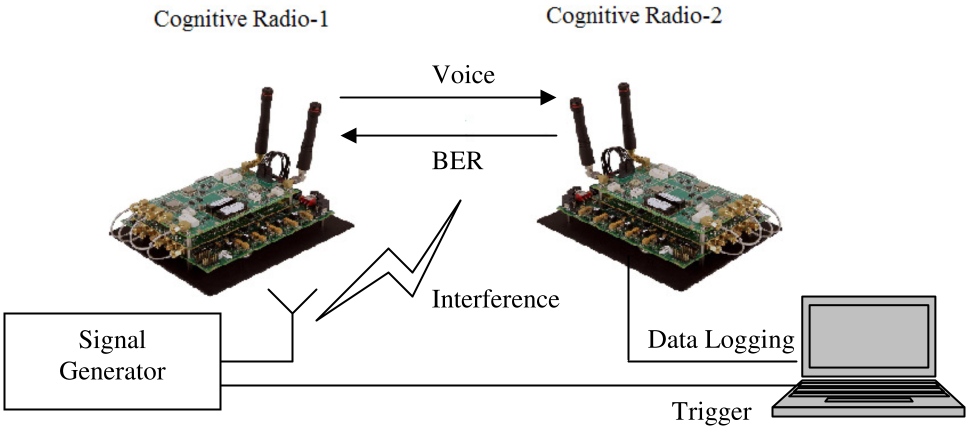

Figure 3.40 Channel switching test setup

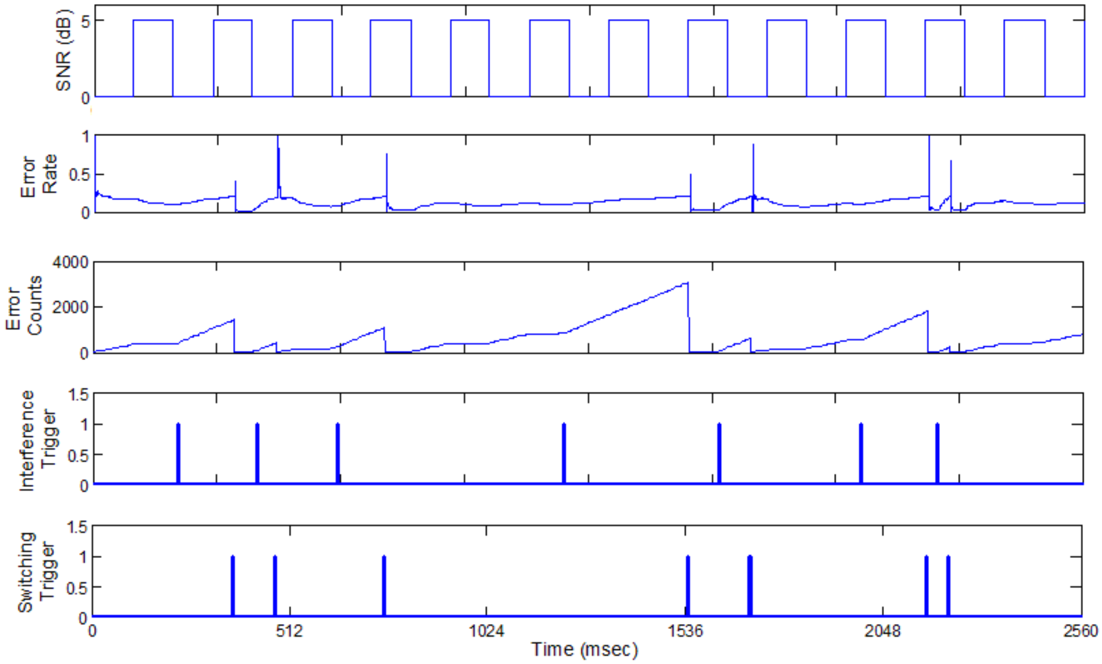

Figure 3.41 Cognitive radio switching behavior generated by simulation

The results shown in Figure 3.41 are obtained for an AWGN channel model with an SNR value alternating between 0 and 5 dB. The error count increment rate is less for 5 dB. (See the time interval around 1024 msec.) The interference trigger pulses are generated at the time an interfering signal is applied to the wireless channel. The switching trigger pulses are generated by cognitive radio-2 upon exceeding a BER of 0.2. The results in Figure 3.41 demonstrate the variable interference detection delay: the time between applying the interference (at the pulse locations of interference trigger signal) and the time of performing the switching operation (at the pulse locations of switching trigger signal). This delay varies based on the level of BER at the moment of interference. The lower the BER level, the longer the interference detection delay, since the error rate increment rate is almost fixed during interference. The instability of the BER calculation after switching operations is also shown. The spikes in the BER signal subsequent to switching are due to BER calculation reset. The BER calculation is based on the ratio of the error counts to received bits counts. BER resets both counts to zero. Therefore, in the event that the next bit is corrupted, the BER will shoot to 1, since only one bit is received and the number of errors is one. Thus, the BER decays rapidly as the channel switching stabilizes. This observation indicates the need to delay BER monitoring after the switching operation in such a sufficient time as to stabilize the switching operation.

## 3.7. Conclusion

This chapter presents the feasibility of building cognitive radios using software defined radio platforms based on hybrid processing engine architecture. Upgrading conventional radios to cognitive radios implies the incorporation of configurability and adaptability design features. A systematic design approach to implement these features on a hybrid software radio platform is discussed in this chapter.

The challenges of building cognitive radios, with configurability and adaptability among its operations, also calls for a more complex system that is beyond the scope of pure physical-layer implementation. An important finding is the need for utilization of cross-layer design techniques to address problems that physical layers cannot overcome, including spectrum sensing and communication session negotiations, as well as channel condition evaluation.

# 4. Compressive Spectrum Sensing

The performance of spectrum sensing is characterized by two essential parameters, namely; selectivity and complexity. The selectivity is described by vacant channel detection success rate that has to be maximized in order to improve cognitive radio throughput, and by false alarm rate that has to be minimized to protect the spectrum primary users from harmful interferences. Improving selectivity of spectrum sensing technique comes at the cost of its complexity, which is the second important parameter in spectrum sensing performance. However, the scalability of spectrum sensing complexity is bounded by several limitations. One is the maximum sensing period in the operation of cognitive radio networks e.g. 2 seconds for IEEE 802.22 WRAN networks. Another limitation is the bounded processing capabilities of wireless devices, especially small form factor devices with limited silicon real for data storage and signal processing.

Compressive sensing techniques are viewed as novel approaches to solve scalability problems in several signal processing applications where the signal of interest is sparse i.e. the amount of valuable information is small compared to the size of the acquired signal [112]. Motivated by the fact that the wireless spectrum in rural areas is highly underutilized i.e. sparse, recent studies have examined the application of these compressive techniques in spectrum sensing [113-115].

Fast Fourier sampling is a compressive sensing technique that is based on the theory of streaming algorithms [84]. This technique does not depend on solving norm minimization or optimization problems. Instead, it is based on iterative greedy algorithm that uses minimum computational resources and number of samples to estimate the most energetic frequency coefficients in the spectrum. The successful implementation of fast

Fourier sampling technique for spectrum sensing in wireless cognitive networks can improve the network performance in many ways. Using fewer samples to sense the spectrum may lead to faster spectrum sensing allowing wider spectrum range to be scanned within the same sensing time window. Moreover, compressive sensing may reduce the computational complexity of spectrum sensing, hence reducing the cost of computing devices.

This chapter studies the feasibility of Fast Fourier sampling for spectrum sensing applications. In particular, this chapter focuses on the detection of wireless microphone signals as one of the spectrum sensing challenges for IEEE 802.22. WM signals are mostly analog frequency modulated signal with no preambles or outstanding carriers typical to digital transmissions that can be recognized as a signature of the particular signals [116]. Additionally, the frequency separation between WM signals is not clearly defined. Therefore, multiple WM may appear in one or more TV channels [117]. These challenges in obtaining any priori information about WM signals have forced the researchers to use blind spectrum sensing techniques (i.e. techniques that do not rely on specific signal statistical features) such as energy detectors.

The chapter starts by describing the simulation setup used for evaluating the performance of FFS algorithm including the wireless channel models and wireless microphone signal models. Then, a performance analysis of FFS technique in detecting wireless microphones signal for IEEE 802.22 wireless network is provided, followed by a study of the impact of several configuration parameters on the wireless microphone detection performance. Next, a proposal for extending the FFS sensing capabilities through two cooperative sensing schemes; one is tailored for narrowband spectrum sensing applications while the second is for wideband spectrum sensing applications. Finally, the chapter concludes with a mathematical analysis of the performance of FFS signal processing stages. Namely; the frequency identification stage and coefficient estimation stage. This analysis outlines the performance boundaries in terms of probability of success and estimation accuracy and identifies the optimum choice of several configuration parameters for Fast Fourier Sampling technique.

## 4.1. Simulation Study

The following simulation studies in sections 4.2, 4.3 and 4.4 examine the performance of FFS algorithm in detecting different wireless microphone signals under various wireless channel conditions. The simulation models for wireless microphone signals and wireless channels are based on the IEEE 802.22 simulation procedure for wireless microphone signal [118]. The FFS algorithm is simulated using the AAFFT_0.9 code developed and made publically available by Mark Iwen [119]. The algorithm has several configuration parameters, the most important ones are summarized in Table 4.1. However, the complete configuration parameters are reported in each simulation study in sections 4.2, 4.3 and 4.4:

Table 4.1. Main FFS configuration parameters

| Parameter | Description |
|---|---|
| Signal Size | The size of the input signal for FFS algorithm. |
| Num_FreqID_CoefEst_Iterations | The number of iterations for the whole FFS algorithm including the frequency identification (FI) and coefficient estimation (CE) stages. |
| Num_Rep_Terms | The number of frequency terms for the approximated signal produces by FFS |
| Working_Rep_Terms | The maximum number of frequency terms that FFS carries from one iteration of FI/CE stages to another. |
| Num_KShattering_Sample_Points | The number of samples FFS use to isolate energetic frequencies. |
| Num_FCE_Sample_Points | The number of sample points FFS use for coefficient estimation. |
| Norm_Estimation_Num | The number of odd and even filter output norm estimates from which the median is taken during FI stage |
| Num_FCE_Medians | The number of coefficient estimates from which the median is taken during CE stage |
| FFCE_Iterations | The number of iterations for CE stage |

The followings are brief descriptions of for the wireless microphones and wireless channel simulation models:

### *4.1.1. Wireless Microphone Signal Simulation Model*

A wireless microphone signal is typically frequency modulated that can be modeled as shown in equation 4.1:

$$s(t) = A_c \cos\left[2\pi f_c t + 2\pi\Delta\int_0^t m(\tau)d\tau\right] \qquad 4.1$$

where :

$A_c$ is the carrier amplitude , $f_c$ is the carrier frequency

$\Delta$ is the frequency sensitivity (Deviation factor), and

$m(\tau)$ is the modulating signal.

Three models for simulating wireless microphone signals are suggested in [118]. The first one is a silent microphone signal, second is a microphone signal of a soft speaker, and the third is for a loud speaker. For each model, the modulating signal *m(t)* is modeled as a sinusoidal signal of different frequency. The simulation results presented later for the FFS detection performance are the average of the detection performance for these three WM models. Table 4.2 summarizes the simulation model parameters for the three wireless microphone signal models.

Table 4.2: The simulation parameters of WM signal models

| | Silent | Soft Speaker | Loud Speaker |
|---|---|---|---|
| M(t) frequency [kHz] | 32 | 3.9 | 13.4 |
| FM deviation factor [kHz] | ± 5 | ± 15 | ± 32.6 |

Figure 4.1 shows the power spectral density of the three WM signals with identical power level used generated in AWGN channel with signal to noise ratio (SNR)

of 20 dB. The carrier frequency is 3 MHz and sampling frequency is 18 MHz. The sampling frequency allows sensing 3 channels (6 MHz channels) at a time as per IEEE 802.22 requirements [39, 43]. The bandwidth of WM signals does not exceed 200 kHz.

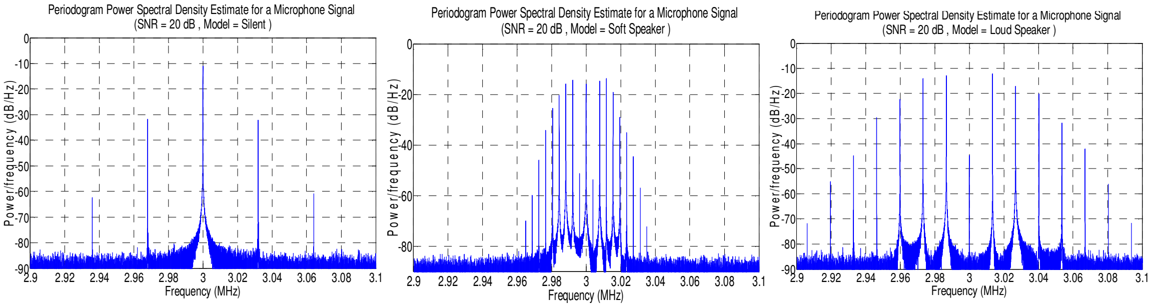

Figure 4.1: The power spectral density of three WM signal models (Silent, Soft Speaker, Loud Speaker)

### *4.1.2. Wireless Channel Model*

The following wireless channel models are used:

- Additive White Gaussian Noise Channel (AWGN)

This model is used to represent outdoor environment with line of sight transmission path between the microphone transmitter and the sensing receiver. The SNR values of AWGN channel model range from 8 dB to -24 dB.

- Rayleigh Fading Channel

The Rayleigh fading channel model is used to simulate the indoor environment [118]. The model implementation is based on IIR filter as described in [120]. The Rayleigh channel model introduces a small Doppler frequency shift (around 1.6 Hz) to the received signal frequency spectrum.

- Shadowing effect

This effect is used for the simulation of cooperative schemes for FFS algorithm in order to evaluate the impact of shadowing on a number of cooperative sensors. A simple model is used to simulate the shadowing effect inspired by the shadowing model presented in [121]. The purpose of the model in [121] is to relate shadowing effect to the distance dependence path loss. However, the following model aims to examine the effect of extreme signal attenuation caused by the shadowing phenomena rather than simulating signal power variation caused by shadowing which may lead to a gain in sensing performance in some cases. The shadowing effect is modeled as a probabilistic signal power blockage. This can be realized by a random received signal gain that is equal to zero for 80% of the time as shown the equation 4.2 [121]:

$$\|g(t)\| = \begin{cases} 1 & w.p.0.2 \\ 0 & w.p.0.8 \end{cases} \qquad 4.2$$

In summary, the overall wireless channel model is comprised of a multiplicative element caused by shadowing and an additive element which is the white Gaussian noise in the AWGN channel.

## 4.2. The Feasibility of a Fast Fourier Sampling Technique for Wireless Microphone Detection in IEEE 802.22 Air Interface

Several studies have investigated the detection of WM signals [117, 122, 123]. Most of those studies are based on energy detection method have shown better detection performance when FM deviation factor is smaller due to higher spectral power density [124]. A study by Rutgers University and Thomson Corporate Research has examined the energy detection method to detect WM signals [122]. They have used two different voice source models; the colored noise and tone signals. Philips research labs have examined the performance of energy detectors based on 2048-point FFT [117]. Their lab experiments have shown a probability of detection that exceed 90% for signals as low as -116 dBm. Table 4.3 summarizes the performance of WM detectors in [117, 122].

Table 4.3. Wireless microphone detectors performance summary

| Detector Type | PMD | PFA | Sensing time [ms] | SNR [dB] | Voice source |
|---|---|---|---|---|---|
| Energy Detector [122] | 0.1 | 0.1 | 10 ms | -24.8 | Tone |
| | 0.1 | 0.1 | 10 ms | -27 | Colored noise |
| | 0.1 | 0.01 | 10 ms | -23.8 | Tone |
| | 0.1 | 0.01 | 10 ms | -26 | Colored noise |
| Welch Periodogram [117] | 0.1 | 0.001 | 5 ms | >-22 | Tone |
| | 0.1 | 0.1 | 5 ms | -23.4 | Tone |
| | 0.1 | 0.1 | 10 ms | -25 | Tone |

The size and sampling rate of energy detectors are constrained by the computing power and sensing time window. A novel method to reduce the required number of samples for signal sensing is compressive sensing [10]. There are several methods for signal reconstruction and some have been investigated for spectrum sensing applications. This section presents the results of using Fast Fourier Sampling technique to detect wireless microphone signals.

### *4.2.1. Simulation Results*

The FFS simulation is done using the AAFFT_0.9 code developed by Mark Iwen [119]. The simulation parameters are listed in Table 4.4. More details about the simulation parameters can be found in [82-84, 119].

In the simulation, the FFS is used to detect a single WM signal at 3 MHz under different environment conditions (AWGN and Rayleigh Fading) with SNR values from 10 to -24 dB. The simulation scenarios have been repeated for 1000 times. At each iteration, the FFS provides the most energetic single frequency and its associated estimated coefficient.

Table 4.4. Fast Fourier Sampling simulation parameters

| AAFFT Parameter | Value |
|---|---|
| Signal Size | $2^{22}$ |
| Num_FreqID_CoefEst_Iterations | 5 |
| Num_Rep_Terms | 1 |
| Working_Rep_Terms | 1 |
| Max_KShattering_Sample_Points | 128 |

| | |
|---|---|
| Num_KShattering_Sample_Points | 128 |
| Exhaustive_Most_Sig_Bits | 0 |
| Max_FCE_Sample_Points | 512 |
| Num_FCE_Sample_Points | 512 |
| Norm_Estimation_Max | 9 |
| Norm_Estimation_Num | 9 |
| Max_FCE_Medians | 9 |
| Num_FCE_Medians | 9 |
| Roots_Coef | 8 |
| Naïve_Bulk_Cutoff | 1 |
| Naïve_Coef_Est_Cutoff | 1 |
| Num_Fast_Bulk_Samp_Taylor_Terms | 8 |
| FFCE_Roots_Coef | 8 |
| Num_Fast_Freq_Coefnt_Est_Taylor_Terms | 8 |
| FFCE_Iterations | 5 |

The histogram of identified frequencies for the 1000 iterations is shown in Figure 4.2. Figure 4.3 lists the histogram of the estimated coefficients. The histogram plots are useful tools to examine the algorithm stability and accuracy across different SNR values. The frequency identification stage is considered successful if the identified energetic frequency is within 200 kHz range (The maximum bandwidth of WM signal) of the carrier frequency (3 MHz). The success rate of the identification stage in AWGN and Rayleigh channel is provided in Figure 4.4. Figure 4.5 shows the success rate of the coefficient estimation stage of the successful identified frequencies. The coefficient estimation for false frequency estimation is not counted.

This algorithm has used 68256 out of $2^{22}$ samples (1.63%) and found to be 170% faster than FFTW (highly optimized C subroutine library for computing discrete Fourier transform).

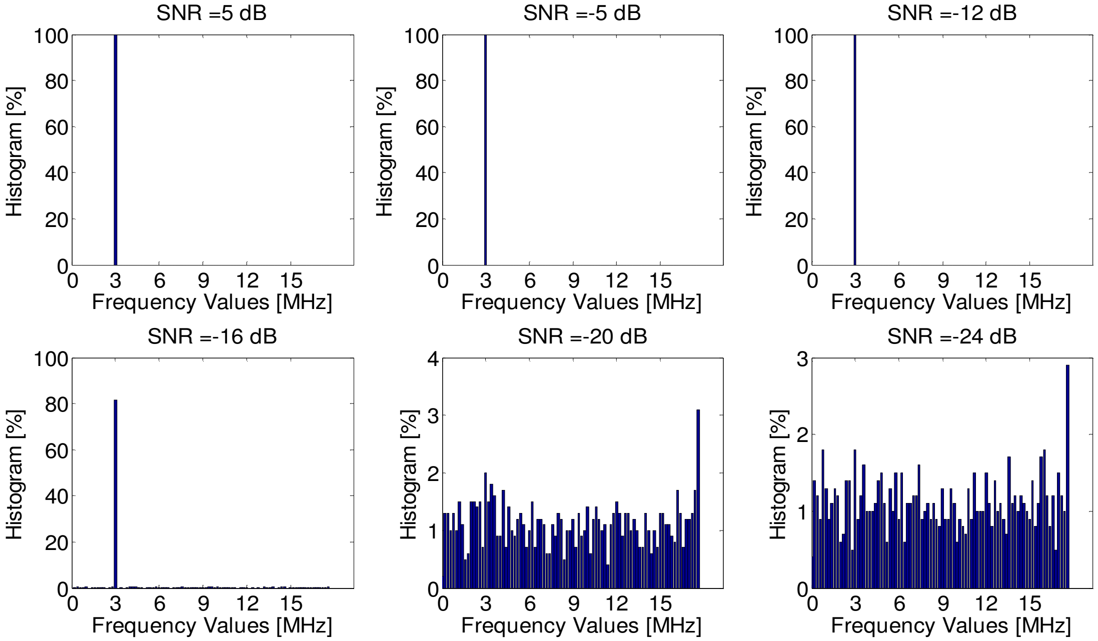

Figure 4.2. Histogram of the frequency identification output for 1000 tests (The energetic frequency is 3 MHz)

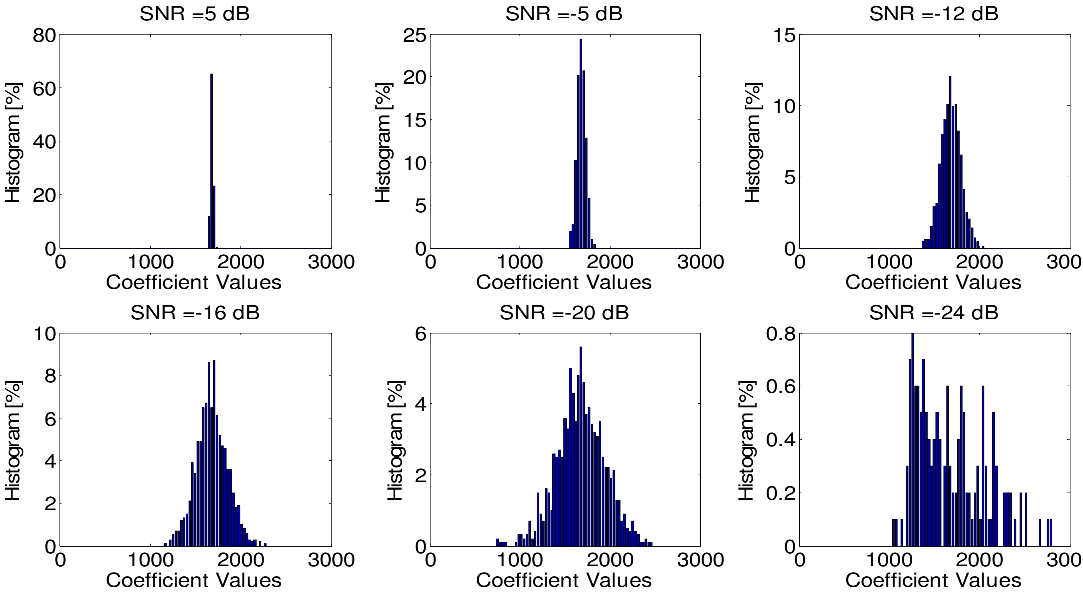

Figure 4.3. Histogram of coefficient estimation output for 1000 tests (The coefficient value is 1683.4)

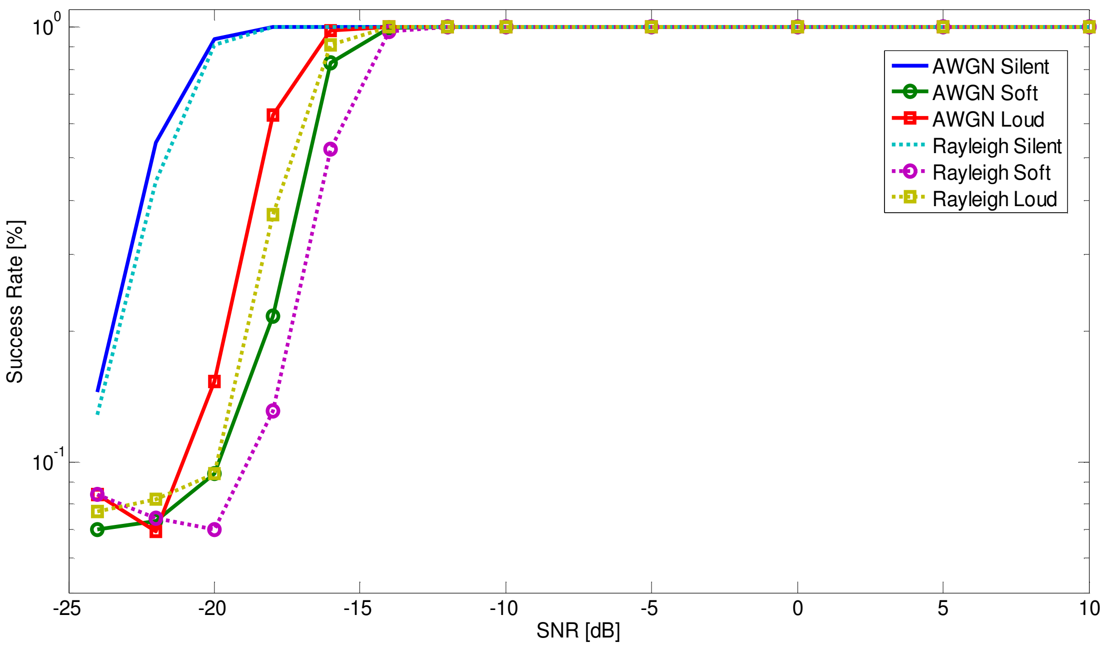

Figure 4.4. Success rate of frequency identification stage

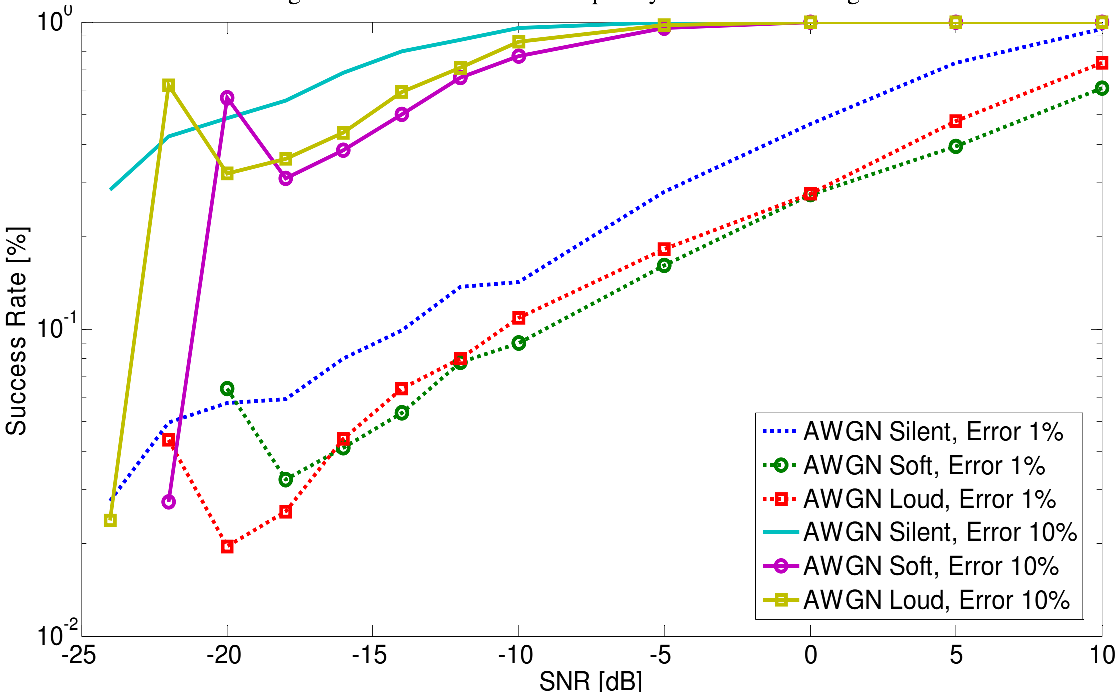

Figure 4.5. Success rate of coefficient estimation of a correctly identified frequency

### *4.2.2. Discussion*

The success rate plots in Figures 4.4, 4.5 uncover interesting heterogeneity in the algorithm performance at different stages. While the frequency identification performance deteriorates rapidly, the coefficient estimation stage performance degrades gradually as the SNR value decreases.  The histogram of identified frequencies and estimated coefficients explains this behavior. In Figure 4.2, the frequency identification stage shows a good stability in identifying the carrier frequency. The performance degrades partially at SNR= -16 dB and then it fails completely at -20 dB where the identification stage act almost as a uniform random number generator. In Figure 4.3, the coefficient estimated value follows a Gaussian distribution with increasing variance as the noise in the wireless channel increases (SNR value decreases). This distribution is expected as a result of the additive white Gaussian noise. This distribution is reflected in the performance behavior as a gradual decay in coefficient estimation success rate.

Another observation found in the result is the algorithm is the identification stage sustainability against Rayleigh fading since the success rate show a slight degradation in the performance (around 1 dB). However, the coefficient estimation performs poorly under Rayleigh fading (not shown in figures). This poor performance in coefficient estimation is expected from FFS technique which acts as an energy detector in the frequency domain. Energy detectors have been found to act poorly in Rayleigh fading channel [125].

Typical energy detectors show better performance in detecting soft WM than loud WM signals [117, 123] because they perform better in detecting narrow band signals (higher spectral energy density) than wideband ones. However, the energy of the frequency “spikes” in the signal is the main performance factor for FFS algorithm and not the signal bandwidth. This explains the third observation found in the performance across different WM signal models. The loud microphone signals are better detected than soft microphone signals by the algorithm because of the higher frequency spikes in the loud

WM speaker as compared to the soft WM speaker. The stronger the frequency spikes the better the detection of the signal by the algorithm.

The initial simulation results lead to see potential in using this technique for spectrum sensing for cognitive radio applications in general and for IEEE 802.22 standard in particular. The frequency identification performance complies to the IEEE 802.22 spectrum sensing sensitivity requirement for WM signals which is -12 dB SNR for a conservative sensing device with a noise factor of 11 dB. The frequency identification of Silent WM signals tolerates up to -22 dB SNR, while the detection limit is around -17 dB for loud and soft speaker signals.

The simulation results have shown considerable spectrum sensing complexity reduction as compared to results [117], The work in [117] uses a sampling rate of 7.5 MHz to process one channel using 2048 FFT. For sensing time of 5 to 10 ms, around so 37.5 to 75 ksamples is needed to scan 1 channel. That means around 112.5 to 225 ksamples for three channels as compared to 68256 samples for FFS algorithm ~40% to 70% reduction in number of processed samples. However, it is important to mention that while the energy detector accuracy in [117] outperform FFS algorithm, the results presented in this work comply to IEEE 802.22 time constrains at far less processing complexity.

In summary, the successful implementation of Fast Fourier Sampling technique for spectrum sensing in wireless cognitive networks can improve the network performance in many ways. This simulation study has demonstrated the feasibility of detecting wireless microphone signals using this algorithm for a given configuration parameters. The impact of these configuration settings is investigated in the next simulation study.

## 4.3. The Impact of FFS Configuration Parameters on Wireless Microphone Signal Detection Performance

This simulation study investigates the impact of different configuration parameters of a compressive sensing technique called Fast Fourier Sampling to detect wireless microphone signals as dictated by the IEEE 802.22 air interface.

The number of samples, iterations and energetic frequencies are among the configuration parameters used by the Fast Fourier Sampling algorithm. An empirical study has examined these parameters under simplified operation conditions [119]. In this simulation study, we examine the impact of different configuration parameters on the performance of fast Fourier sampling. The goal is to detect wireless microphone signals in IEEE 802.22 under AWGN and Rayleigh fading wireless channel models. The results are presented in terms of success rate and false alarm.

### *4.3.1. Simulation Setup*

In this simulation study, we examine the performance of FFS to detect a single wireless microphone signal according to the IEEE 802.22 spectrum sensing specifications [3]. Let us assume that the sensed spectrum is divided into multiple frequency bands with a bandwidth 200 kHz which is the maximum bandwidth of wireless microphone signals. The sampling rate of the sensing node is 125 MHz. So there is around 625 frequency bands that could be scanned. However, due to the limitation of RF-front end bandwidth in practice. The bandwidth of the spectrum to be scanned is assumed to be limited to 20 MHz (By using band-pass filters). The detection performance results are presented in terms of:

- Probability of successful detection:

The probability of correctly detecting the occupied frequency band,

- Probability of detection false alarm.

These results are reported for different FFS configuration parameters.

The wireless channel models used in the simulation are AWGN and Rayleigh fading channel models. However, the small Doppler frequency shift introduced by the Rayleigh model based on the settings in [118] had negligible impact on the frequency identification performance of FFS algorithm. In this study, a Doppler shift of 1.25 kHz is used instead to magnify the impact of Doppler shift on the sensing performance.

Fast Fourier Sampling Simulation Parameters

The FFS algorithm is simulated using the AAFFT_0.9 code developed and made publically available by Mark Iwen [119]. The simulation parameters of the FFS algorithm are listed in Table 4.5. These parameters are fixed (except for those with values N, K or m) in all the simulation scenarios to eliminate their impact on the performance results. The configuration parameters considered in this study are the signal size N, the number of frequency terms m and the number of Kshattering points K. We have considered the following values for these parameters:

N= $2^{18}$ , $2^{20}$ , and $2^{22}$

K= 32, 64, 128, and 256

m= 1, 2, 4, and 10

Note that the value of K affects multiple FFS parameters as shown in Table 4.5. While this is not necessary, this setting simplifies tracking the number of samples used by FFS because the K samples used in Shattering can be reused for coefficient estimation.

Table 4.5: Fast Fourier Sampling parameters

| AAFFT Parameter | Value |
|---|---|
| Signal Size | N |
| Num_FreqID_CoefEst_Iterations | 5 |
| Num_Rep_Terms | 1 |
| Working_Rep_Terms | m |
| Max_KShattering_Sample_Points | K |
| Num_KShattering_Sample_Points | K |
| Exhaustive_Most_Sig_Bits | 0 |
| Max_FCE_Sample_Points | K |
| Num_FCE_Sample_Points | K |
| Norm_Estimation_Max | 5 |
| Norm_Estimation_Num | 5 |
| Max_FCE_Medians | 5 |
| Num_FCE_Medians | 5 |
| Roots_Coef | 8 |
| Naive_Bulk_Cutoff | 1 |
| Naive_Coef_Est_Cutoff | 1 |
| Num_Fast_Bulk_Samp_Taylor_Terms | 7 |
| FFCE_Roots_Coef | 8 |
| Num_Fast_Freq_Coefnt_Est_Taylor_Terms | 7 |
| FFCE_Iterations | 5 |

### *4.3.2. Simulation Result and Discussion*

In this section, the performance of FFS in detecting wireless microphone signals are reported and discussed. Different FFS configuration parameters are considered to evaluate their impact on the detection performance; mainly, the signal size (N), number of KShattering sample points K, and the number of representation terms (m). Two performance metrics are provided. First is the frequency identification success rate which reflects the probability of the identified frequencies to fall within 200 KHz (Maximum WM signal bandwidth) around the WM carrier at 3 MHz. The second performance metric is the detection success rate. The wireless microphone signal is considered detected at the frequency range of [3MHz ± 100KHz] if at least n out of m frequencies are identified within that range and their corresponding coefficients are above the detection threshold

$\gamma_\sigma$. The number of frequencies n depends on the detection technique. In this study, we have examined a simple OR decision fusion technique at which n = 1.

The performance is presented in terms of success rate for a probability of false alarm equals to 0.1.

Signal Size (N)

Three different signal sizes are examined ($2^{18}$, $2^{20}$, $2^{22}$). The number of KShattering samples is set to 128 and the number of frequency terms used in the detection is m =1. It should be noted that the FFS is capable of examining signal sizes N other than a power of 2 at the expense of higher computational complexity (particularly at the frequency identification stage). Figures 4.6, and 4.7 show the success rate of frequency identification, and WM signal detection respectively for different signal sizes. It can be noted that this parameter has negligible impact on the FFS performance with the exception of frequency identification performance at Rayleigh channel model. The convolution effect of Rayleigh channel frequency response on the received signal causes the signal spectrum to spread across wider range of frequencies. This can lead to false frequency identification if the signal spectrum spread across multiple frequency bins causing some of them to be labeled mistakenly occupied. Additionally, the probability of error in the frequency identification stage increases as the number of bits b=log2(N) increases because the identification operation is sequential and is being performed on a bit by bit basis. As the signal size N increases, the number of possibly faulty bits increase. For example, the Doppler shift of 1250 Hz can be represented by 1 bit for signal size $2^{18}$ , while it is represented by 5 bits for N=$2^{22}$.

While increasing the number of samples may result in performance drop under Rayleigh fading conditions, the spectral intensity (coefficient values) of the signal increases leading to better signal detection based on the coefficient values (see Figure 4.8). As shown in Figure 4.7, this positive effect has cancelled out the performance drop in the frequency identification so the detection success rate for signal sizes $2^{18}$ and $2^{20}$ is almost equivalent. Yet, it can be seen that the increment in spectral intensity in the case of $2^{22}$ is not high enough to compensate for the drop in detection performance caused by the frequency identification stage.

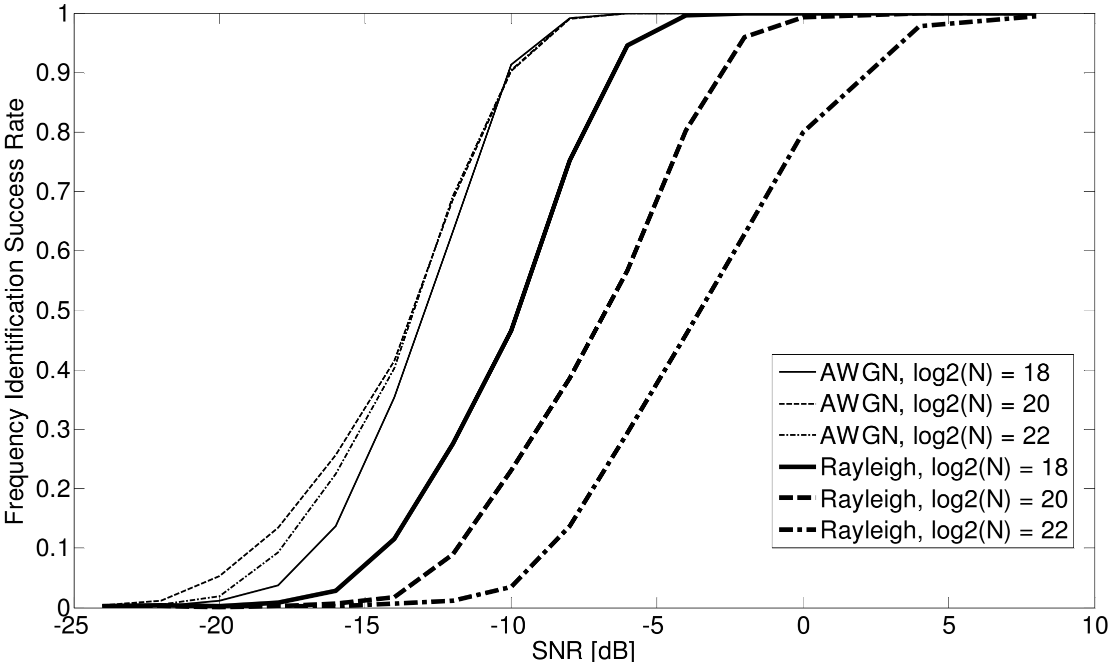

Figure 4.6. The probability of success of the frequency identification stage for different number of frequency terms and for K=128 and N=2^20.

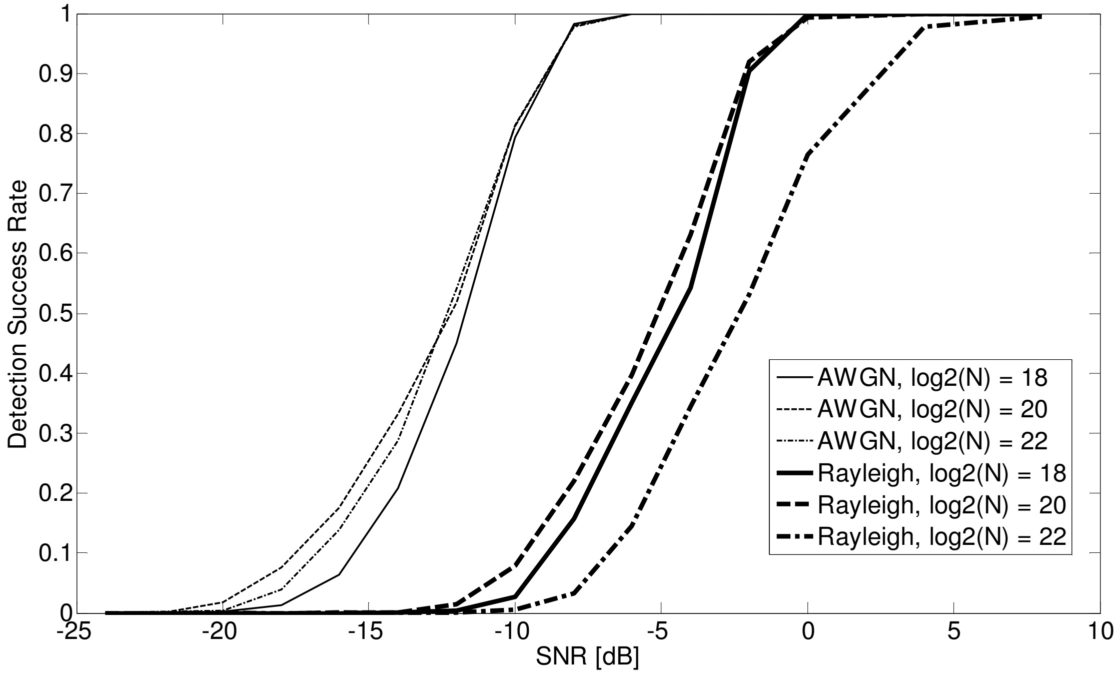

Figure 4.7. The detection probability of success for different number of frequency terms using the updated set of threshold values and for K=128 and N=$2^{22}$.

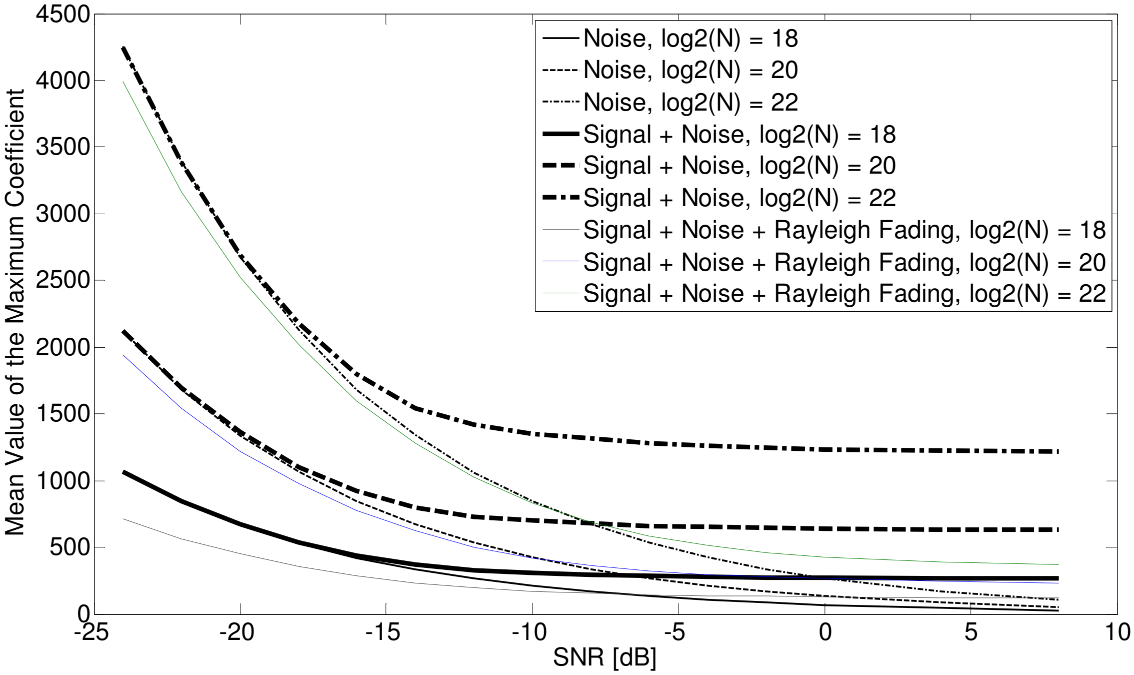

Figure 4.8. The average of the maximum coefficient values found by FFS in AWGN, Rayleigh and noise only environment.

Number KShattering Sample Points (K)

This critical parameter has a considerable impact on the runtime and probability of failure of different FFS stages. On the one hand, increasing K results in an increment in the number of samples required by FFS to estimate the spectrum, and on the other hand it improves the probability of success of the frequency isolation stage. Figures 4.9, 4.10 uncover the impact of this parameter on identifying WM signal frequencies as well as its impact on the detection success rate. As can be seen from these figures, a gain of approximately 3 dB in the detection success rate is obtained by doubling the size of K for AWGN or Rayleigh fading channel models. Figure 4.11 shows the Receiver Operating Characteristics curve ROC of the WM detector at SNR = - 10 dB.

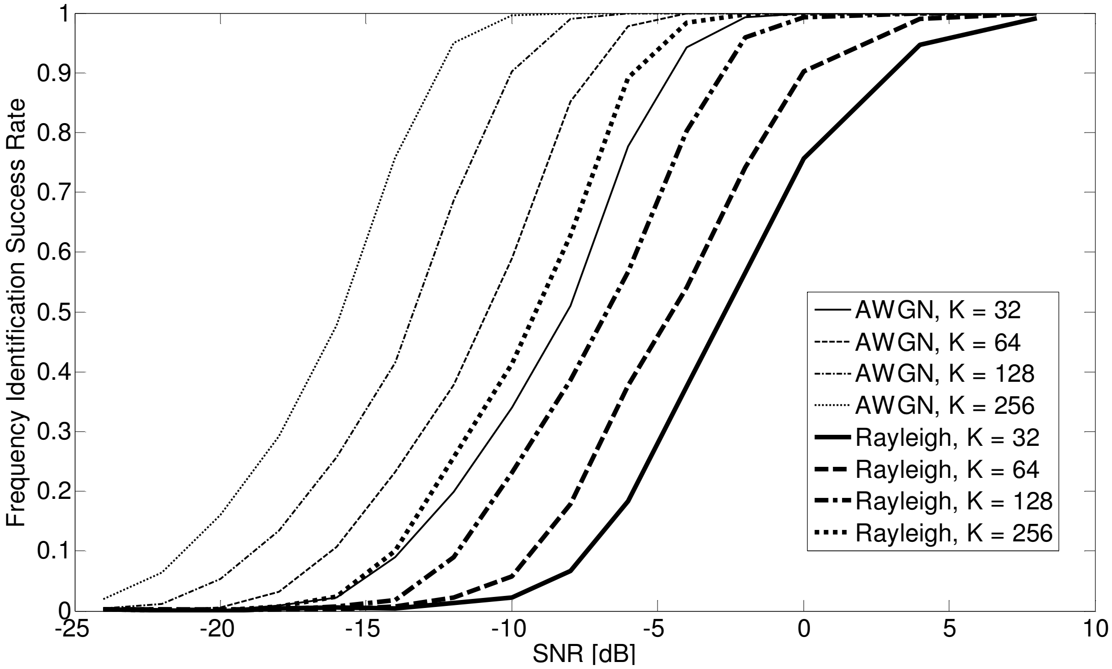

Figure 4.9. The probability of success of the frequency identification stage for different # of KShattering sample points, m=1 and N=$2^{20}$.

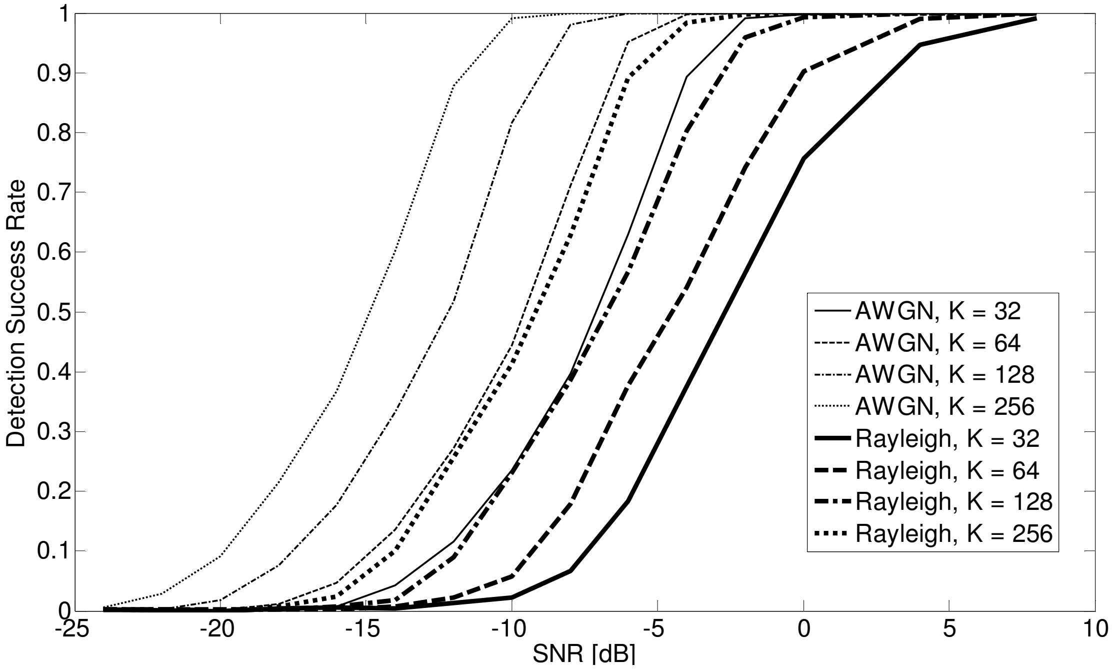

Figure 4.10. The detection probability of success for different number of Kshattering sample points for m=1 and N=$2^{20}$.

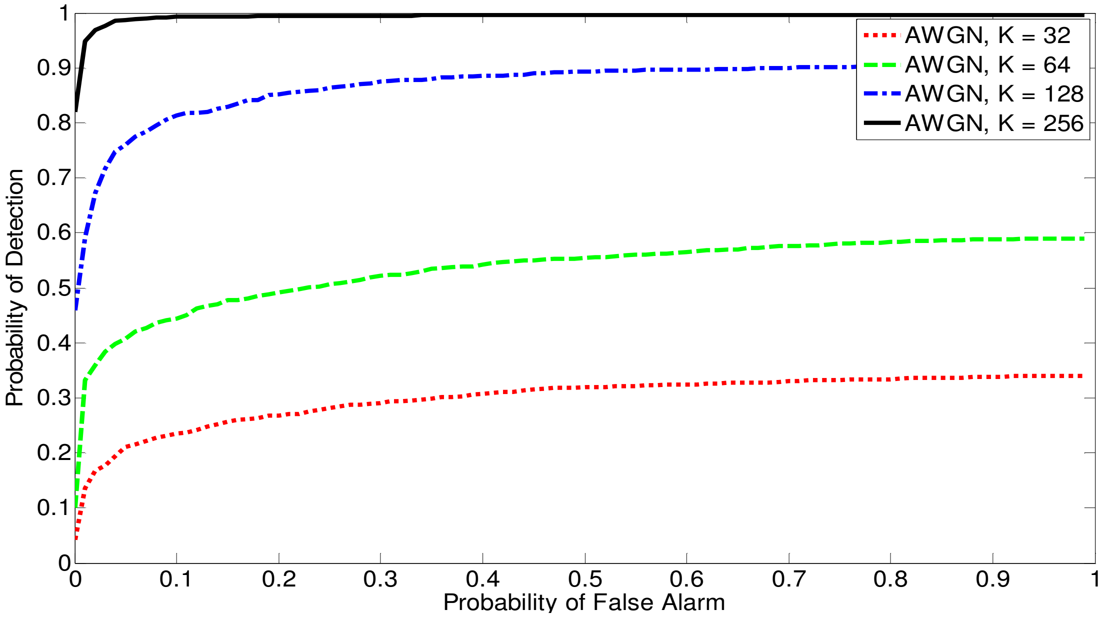

Figure 4.11. Receiver Operating Characteristics of WM detector for different K values and for SNR = -10 dB , m=1, N=$2^{20}$.

Number of frequency terms (m)

In the results presented earlier, the fast Fourier sampling was configured to return a single pair (frequency, coefficient) upon which the detection operation is performed.

However, it is tempting to use more than one (frequency, coefficient) pairs for WM detection. In other words, using more than one frequency term in the approximated spectrum found by the FFS algorithm may help in obtaining better detection performance. While this idea may be intuitive, it bears several design challenges.

The first challenge in using more than one frequency term in detecting WM signals is in obtaining a suitable decision fusion technique. Finding an optimum decision fusion method is beyond the scope of this study. Instead, in this simulation study, a simple decision fusion technique based on OR combining method is considered. In this fusion technique, the wireless channel is labeled occupied if at least the coefficient of one frequency (out of m frequencies) exceeds its corresponding threshold value.

The second challenge appears in setting the detection threshold values for the multiple frequency terms. Since FFS algorithm returns m (frequency, coefficient) pairs in a descending order based on the coefficient values, it is expected for the threshold values to follow the same trend i.e. decrease along multiple frequency terms. Let $\gamma_{\delta,m,j}$ be a set of thresholds, where $\delta$ is the noise variance, m is the number of frequency terms returned by FFS, and j is the term number (from 1 to m). For each noise variance value, the threshold value for each frequency term is set individually so that the probability of false alarm caused by this term alone does not exceed 0.1.

The following results are obtained for m values of 1, 2, 4, and 10. Figure 4.12 depicts the probability of one identified frequency (at least) to be within the 200 kHz bandwidth of the microphone signal. As the number of frequencies m increases, the probability of at least one of them fall within the frequency range of the WM signal increases. As a result, the WM detection success rate increases as shown in Figure 4.14.

However, the probability of false alarm of this detector designed based on OR decision fusion technique is more than 0.1 upon using multiple frequency terms (See Figure 4.13). Since, the coefficients of multiple frequency terms are correlated, doubling the number of terms used in the detection results in less than twice the probability of false alarm.

To correct this problem, the thresholds values should be increased upon using more frequency terms in WM detection. We propose the following heuristic approach to set the thresholds values so that the joint probability of false alarm for the detector is close to 0.1:

1. Set the threshold values $\gamma_{\delta,m,j}$ as described earlier so that $P_{FA}$ of the detector upon using each frequency term alone is 0.1.
2. We examine the joint $P_{FA}$ of the detector using the OR decision fusion method and the set $\gamma_{\delta,m,j}$ set in step one. (See Figure 4.13).
3. Calculate the mean values of the joint $P_{FA}$ across different SNR values for different m values $FA_m$. For example, the $FA_m$ values obtained from Figure 4.13 are $FA_1$= 0.1, $FA_2$=0.1588, $FA_4$=0.23, $FA_{10}$=0.313
4. Set the new threshold values $\gamma'_{\delta,m,j}$ so that $P_{FA}$ caused by using each frequency term alone is $\frac{0.1}{10.FA_m}$.

By using the procedure described above, the probability of false alarm of the WM detector based on the new set of thresholds for different m values become closer to 0.1 as shown in Figure 4.15. The $P_{FA}$ is improved significantly at the cost of a small drop in the probability of detection (shown in Figure 4.16). Yet, the gain obtained by using multiple frequency terms in the detection of a single WM signal (based on simple OR decision fusion technique) does not justify the increased FFS complexity. Designing better decision fusion technique is left as a future work.

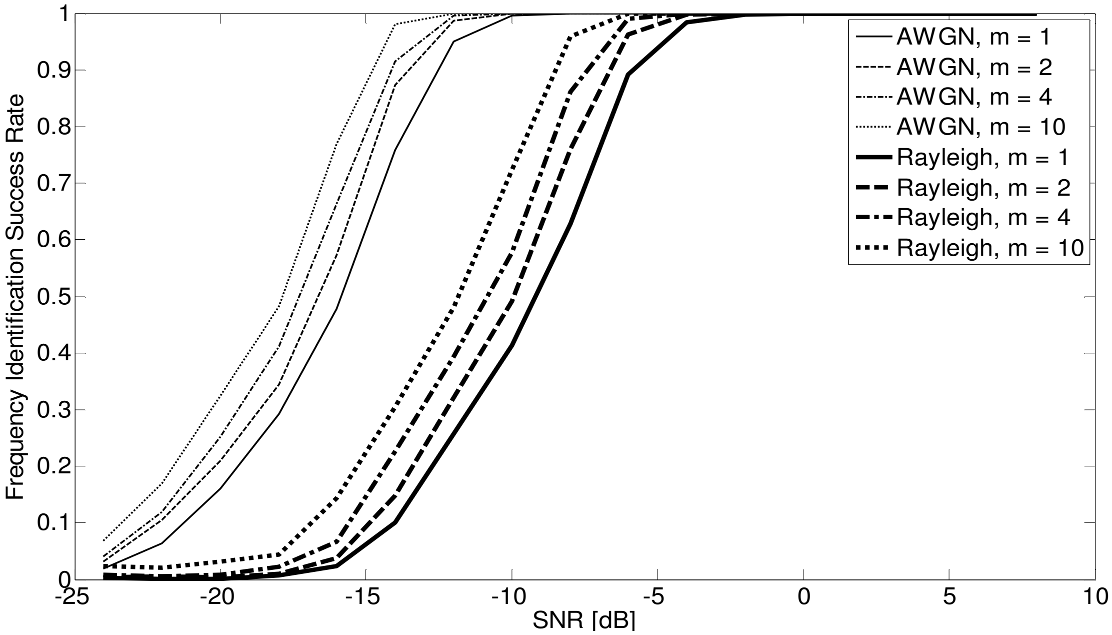

Figure 4.12. The probability of success of the frequency identification stage for different number of frequency terms and for K=256 and N=2^20.

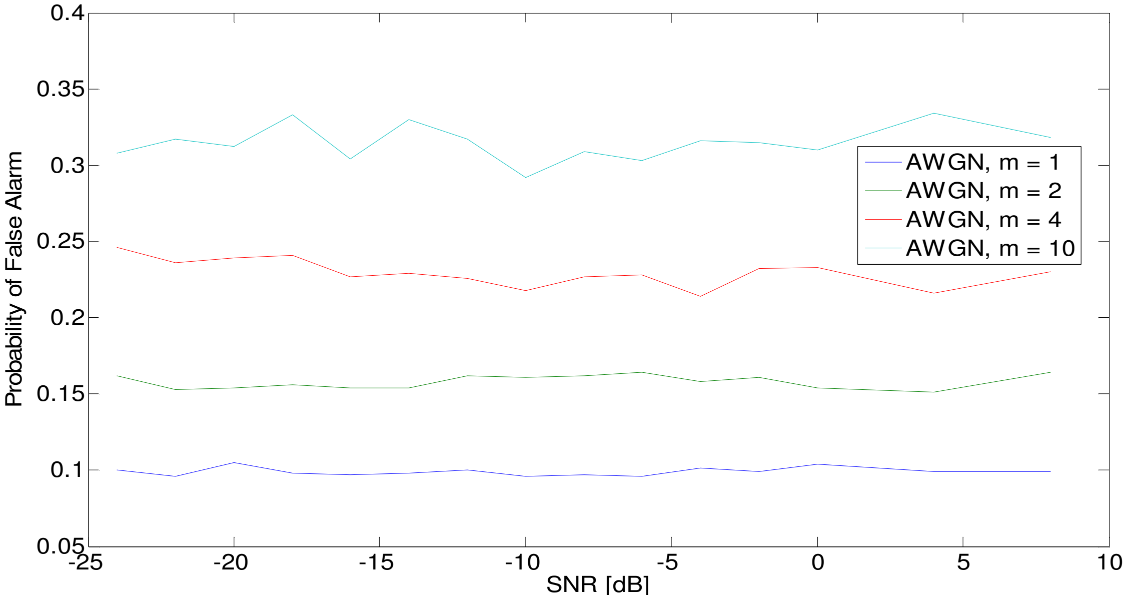

Figure 4.13. The probability of false alarm of the WM detector based on m frequency terms using preliminary threshold values and OR decision fusion technique. m=1,2,4, and 10

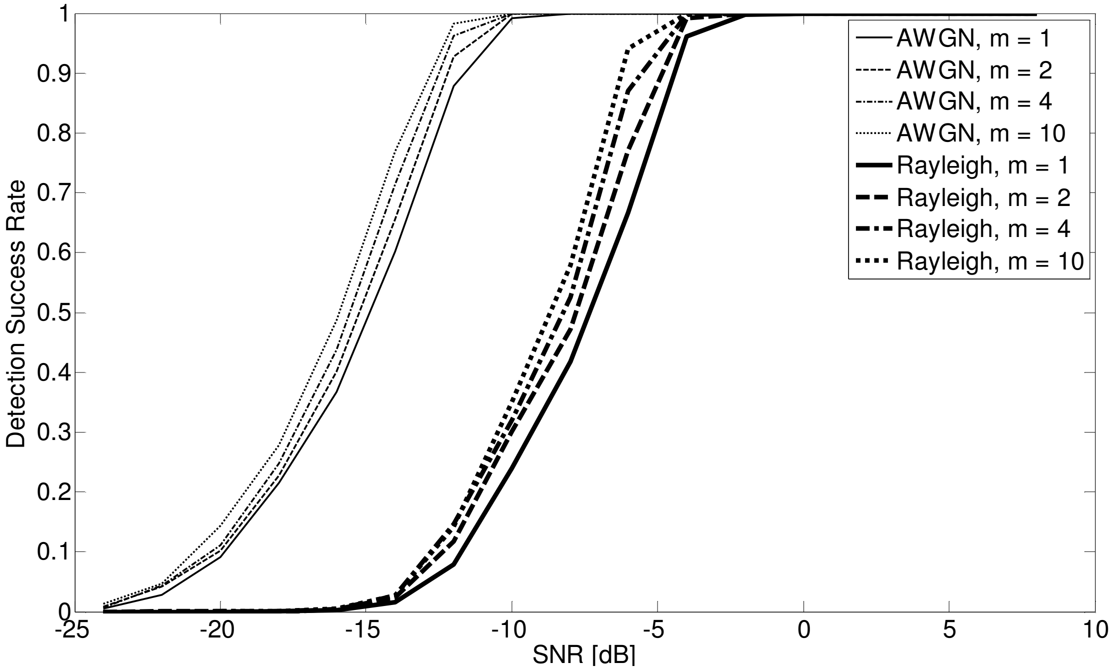

Figure 4.14. The detection probability of success for different number of frequency terms using the preliminary set of threshold values and for K=256 and N=2^20.

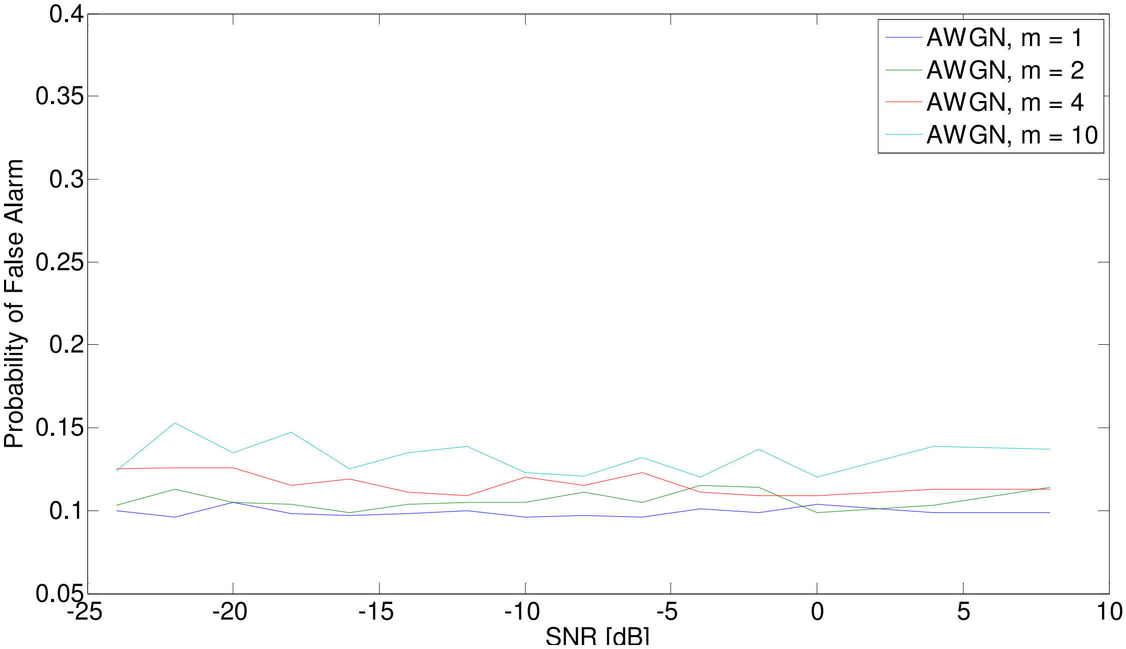

Figure 4.15. The probability of false alarm of the WM detector based on m frequency terms using the updated threshold values based on the proposed heuristic approach and OR decision fusion technique. m=1,2,4, and 10

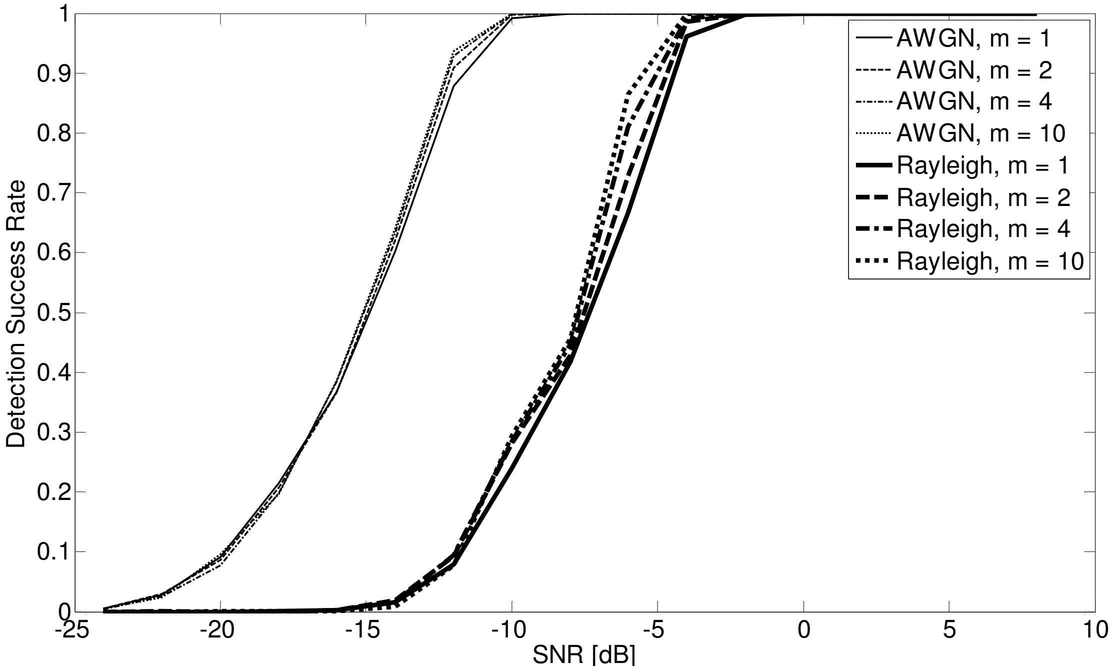

Figure 4.16. The detection probability of success for different number of frequency terms using the updated set of threshold values and for K=256 and N=2^20.

The FFS detector based on the configuration settings in Table 4.5 provides good detection performance around -10 dB approximately in AWGN and around 0 dB for Rayleigh fading distribution. Yet, it should be noted that the Rayleigh fading applied in this simulation study is more intense than the one described in [118]. The best FFS performance encountered in this simulation study is when K = 256, m=10 and N=$2^{22}$ in AWGN channel where the probability of detection exceeds 0.9 at SNR = -12 dB. These results can be further improved by adjusting the other configuration parameters to reach a probability of detection of 0.9 at SNR -18 dB as presented in section 4.2.

In summary, the results in this study show considerable impact of the kshattering number of samples on the detection performance in all wireless channel conditions. On the other hand, the signal size appeared to be sensitive to configure under Rayleigh fading conditions. In addition, this study found a negligible detection performance gain upon using more than one frequency term based on coefficient thresholding technique and OR decision fusion method.

## 4.4. Cooperative Fast Fourier Sampling for Spectrum Sensing in Cognitive radio Networks

The ideal spectrum sensing operation in cognitive radio network is capable of detecting very low SNR primary users over very wide range of spectrum in a short period of time. In this context, there are three optimization objectives: the sensitivity of sensing algorithm, the speed of sensing algorithm and the spectral range of sensing operation.

The simulation studies presented earlier (section 4.2 and 4.3) show a considerable complexity reduction, allowing for faster spectrum sensing operation. This section proposes the use of cooperative techniques combines with FFS algorithm to further improve the other optimization objectives for sensing operations; namely, sensitivity and spectral range.

Cooperative sensing techniques for cognitive radio network can be categorized into two groups; narrowband cooperative techniques and wideband cooperative techniques [126]. Such cooperation can be accomplished in centralized or distributed fashion. The cooperative sensing technique presented in this work is a centralized sensing technique which is the one adopted in IEEE 802.22.

Let us assume a wireless spectrum shared by a cognitive radio network, and is divided into multiple non-overlapping frequency bands of identical bandwidth.

In narrowband cooperative sensing, the cognitive network nods cooperate in order to sense the spectrum occupancy in a single frequency band. All cognitive nodes independently sense the same frequency band and report spectral analysis results to the base station. The base station then combines the collected data from all nodes to determine the band spectrum occupancy. In wideband cooperative sensing, the cognitive network nodes cooperate to scan multiple frequency bands. The sensing load is divided between the cognitive nodes so that each cognitive node is responsible for sensing just one frequency band at a time, all nodes report their measurements to the base station.

In both schemes, the base station performs the data fusion center to estimate the spectrum occupancy. However, the details of fusion implementation vary considerably between the two schemes. On one hand, the base station in narrowband cooperative schemes combines the spectral analysis results i.e. a collection of major frequencies and their associated Fourier coefficients. Hence, the cognitive radio nodes conduct the spectrum sensing locally before reporting the sensing results to the base station. On the other hand, the base station in wideband scheme combines time samples and the spectrum sensing is conducted at the base station.

Many studies have shown several benefits of using cooperative spectrum sensing in cognitive radio networks using conventional spectrum sensing techniques and compressed sensing ones [61, 126, 127]. Sensing cooperation combats the hidden terminal problem [61], and improves the overall primary user probability of detection in shadowing and deep fading due to the spatial diversity obtained by the geographically scattered sensing nodes. It can also lower the SNR wall of the sensing algorithm [67]. Compressed sensing techniques other than FFS have been examined in cooperative cognitive radio networks [114], where the spectrum recovery is performed by solving linear convex optimization problem using a Basis Pursuit (BP) technique. The results in [114] show that the lowest SNR at which a collaborative compressed sensing is approximately 5 dB with probability of detection around 90% and compression ratio of

50%. Another interesting observation in [114] is that the degradation of sensing due to compression is trivial when the number of cooperative nodes is large.

However, the benefits of the cooperative sensing technique comes at the price of additional processing complexity as well as additional spectrum bandwidth allocation for the control channel, which is used for exchanging the local spectrum sensing results.

In this study, we investigate two cooperative spectrum sensing schemes based on FFS algorithm, before describing the cooperative FFS schemes in details, it is necessary to define the following notations:

| | |
|---|---|
| N | The number of collaborative sensing nodes in a given scheme |
| m | The number of energetic frequencies to be estimated by FFS algorithm |
| K | The number of random samples and it is larger than m (typically $K=4m$ [84]) |
| $N_b$ | The number of frequency bands in the shared spectrum |
| $B_i$ | The spectrum band i where i =1,2 … N |
| $\{ F_{ij} , C_{ij} \}$ | The set of big frequencies in band i and their associated coefficients returned by FFS algorithm conducted at node j |
| $f_{k,i,j} , c_{k,i,j}$ | The frequency and coefficient elements in the set $F_{i,j} , C_{i,j}$ where $k = 1, 2 \dots m$ |
| $\{ R_{ij} \}$ | The set of random samples of band i collected by sensing node j |

### *4.4.1. Narrowband Cooperative Sensing Scheme*

The objective of cooperation in this scheme is to improve the sensitivity and reliability of FFS algorithm by exploiting the spatial diversity of sensing nodes.

In this scheme, several nodes use the FFS algorithm to perform local spectrum sensing of a single frequency band $B_i$. While local sensing is independent i.e. no information exchange occurs between the cognitive nodes, all nodes report their independent measurements $\{ F_{ij} , C_{ij} \}$ to the base station. The base station then performs the collaborative sensing technique in two stages: 1.) It detects energetic signals in band

$B_i$ using each node measurements. 2.) It combines the detection decisions into a global spectrum sensing decision.

The detection of energetic signals is performed using the test for each coefficient $c_k$ in equation 4.3:

$$\left\|c_{k,i,j}\right\| \quad \underset{0}{\overset{1}{\gtrless}} \quad \gamma_j \tag{4.3}$$

where $\gamma_j$ is the detection threshold for measurements made by node j. Obtaining an optimized set of threshold values for each sensing node can be an NP complete problem from the algorithmic point view if the measurement are not independent [128]. Instead, a sub-optimal performance can be rather obtained using a global threshold value $\gamma$ for all sensing nodes with a fixed probability of false alarm for the cooperative detection.

The binary decisions of each node measurement are then combined using one of the following combining methods:

- OR-logic combining: The frequency is labeled occupied if it is identified by one sensing node at least with a coefficient estimation that exceeds the global threshold $\gamma$.
- AND-logic combing: The frequency is labeled occupied if it is identified by all sensing nodes and all their coefficient estimations exceed the global threshold $\gamma$.
- Major vote (MV) combining: The frequency is labeled occupied if it is identified by the majority of sensing nodes (more than half of the sensing nodes) with coefficient estimations that exceed the global threshold $\gamma$.

Several parameters impact the performance of this cooperative sensing scheme such as the number of cooperative nodes, the decision threshold and the decision combining rule. Many of these parameters are investigated in the simulation study in next section. It should be noted that the performance of this scheme is limited by several factors such as shadowing, variance in channel modes and the correlation between node measurements.

### *4.4.2. Wideband Cooperative Sensing Scheme*

The objective of cooperation in this scheme is to speed up the spectrum scanning processing to cover a wide range of spectrum. In fact, the compressive nature of the FFS algorithm allows this scheme to scale to multiple nodes covering a wider range of spectrum, as long as the collective shared spectrum is still sparse. Hence, it can be efficiently processed using FFS algorithm.

In this scheme, each node monitors a specific frequency band $B_i$. However, the nodes do not perform the FFS technique to sense the local spectrum. Instead, each node $CR_i$ performs a random sampling of the received signals in band $B_i$ and forwards these samples to the base station. This can be viewed as a distributed sampling of $B_n$ non-overlapping frequency bands. The number of frequency bands is not required to be equal to the number of sensing nodes. Instead, it is possible for each node to scan more than one frequency band. However, due the RF front end bandwidth limitations, each node can monitor no more than a single band at a time assuming that the bandwidth of the frequency band BW is equal to the bandwidth of the cognitive node receiver.

In each sensing period, i.e. a pre-defined time window at which all cognitive nodes halt their wireless communication to conduct spectrum sensing, the base station sends two random sampling parameters (a and b) needed to generate the random sampling sequence to all cognitive nodes See equation 4.4. These are global, which means all sensing nodes use the same sequence generated by the base station. Once received, all nodes begin random sampling based on the acquired sequence at the same time and return samples to the base station. Again, cooperative sensing is conducted at the base station in two stages:

First, the base station normalizes the samples of different bands so that they all have equivalent power. This is done by estimating the spectral power of each band using its random time samples. The base station then combines the random samples from different bands and formulates the wideband signal ( R(n) ) in the time domain as shown in equation 4.4:

$$R(n) = \sum_{i=1}^{B_n} \sum_{n=1}^{K} r_i(n).e^{-2\pi j.BW.i} \qquad 4.4$$

Second, the base station conducts the FFS algorithm to estimate the collective spectrum comprised of all sensed bands $B_i$.

Figure 4.17 shows an illustration of the narrowband and wideband cooperative sensing schemes.

The choices of the number of bands and the number of random samples to be collected govern the performance of this scheme. It is also important to have reliable power estimation for each band in order to obtain a fair treatment for each band by the FFS algorithm. The simulation studies presented in next sections investigate part of these issues.

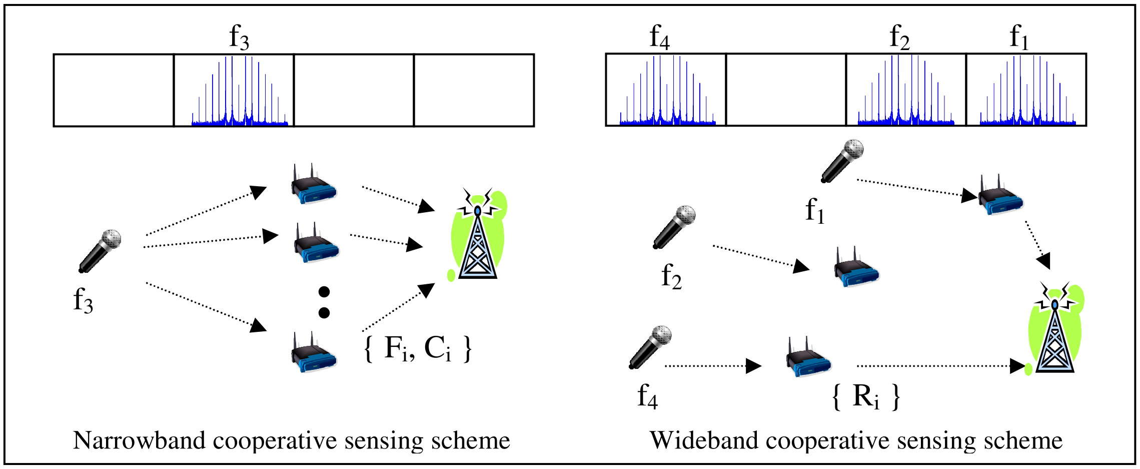
f3
f4
f2
f1
f1
f3
f2
{ F_i, C_i }
f4
{ R_i }
Narrowband cooperative sensing scheme
Wideband cooperative sensing scheme

Figure 4.17: narrowband and wideband cooperative sensing schemes

In every fusion technique, the combined data may be polluted by measurement noise caused by several sources of system irregularities. Reliable cooperative sensing technique should be able to tolerate a reasonable amount of disturbances in the fused data. We have examined the reliability of both proposed schemes in simulation. The disturbances in the fused measurement in the narrowband scheme are simulated by imposing shadowing on selected nodes while the disturbances in the wideband scheme are simulated by imposing a phase shift between the combined samples.

### *4.4.3. Simulation Results and Discussion*

In this simulation study, we examine the performance of the proposed cooperative FFS schemes when detecting wireless microphone signals in a single IEEE 802.22 cell. For each scheme, the performance is evaluated in terms of selectivity and reliability. Selectivity performance is measured by two metrics; the probability of detection and probability of false alarm. The reliability is measured by the drop in probability of detection upon introducing disturbances into the measurement fusion. In narrowband cooperative sensing scheme, shadowing effect of some local sensor measurements is introduced as one sort of disturbances that can appear on the fused data. In wideband cooperative sensing, random phase shift is introduced to the combined random samples.

For simplicity, it is assumed that the wireless channel is time invariant during sensing time. It is also assumed that the signals to noise ratios of the local node measurements are identical.

In the following, a detailed description of the simulation setup parameters is provided. Simulation results of several narrowband and wideband cooperative sensing scenarios are then reported and discussed. The results presented for all simulation scenarios are the average of 1000 test iterations.

### Simulation Setup

The cell is comprised of a base station and several sensing nodes distributed randomly in the coverage area of the base station. The number of sensing nodes varies between one and ten sensing nodes for each simulation scenario. The shared spectrum is divided into multiple frequency bands with a bandwidth 18 MHz i.e. three adjacent IEEE 802.22 channels of bandwidth 6 MHz.

The simulation parameters of the FFS algorithm are listed in Table 4.6. These parameters are fixed in all the simulation scenarios to remove their dependencies on the performance results. Instead, other cooperative sensing parameters such as number of cooperative nodes in narrowband sensing or number of bands in wideband sensing are used. These parameters can be adjusted to obtain a better sensing accuracy or a faster sensing operation.

Table 4.6: Fast Fourier Sampling parameters

| AAFFT Parameter | Value |
|---|---|
| Signal Size | 2^22 |
| Num_FreqID_CoefEst_Iterations | 5 |
| Num_Rep_Terms | 1 |
| Working_Rep_Terms | 1 |
| Max_KShattering_Sample_Points | 128 |
| Num_KShattering_Sample_Points | 128 |
| Exhaustive_Most_Sig_Bits | 0 |
| Max_FCE_Sample_Points | 512 |
| Num_FCE_Sample_Points | 512 |
| Norm_Estimation_Max | 9 |
| Norm_Estimation_Num | 9 |
| Max_FCE_Medians | 9 |
| Num_FCE_Medians | 9 |
| Roots_Coef | 8 |
| Naive_Bulk_Cutoff | 1 |
| Naive_Coef_Est_Cutoff | 1 |
| Num_Fast_Bulk_Samp_Taylor_Terms | 8 |
| FFCE_Roots_Coef | 8 |
| Num_Fast_Freq_Coefnt_Est_Taylor_Terms | 8 |
| FFCE_Iterations | 5 |

### Narrowband Collaborative Sensing Results

In this scheme, the performance of collaborative sensing is simulated at the base station to detect a WM signal randomly located in a single frequency band (18 MHz) using up to ten sensing nodes. Each sensing node performs FFS to estimate the most energetic frequency and its associated coefficient. The base station collects the frequency-coefficient pairs from all sensing nodes and determines the existence of a WM signal based on the coefficient values using one of the three combining rules: AND, OR, and MV. The performance of cooperative sensing has been tested at different SNR values. The decision threshold $\gamma$ is global for all sensing nodes and it is empirically determined so that the probability of false alarm of the cooperative sensing is less than 0.1.

In the following we discuss in further details the impact of the following design parameters on sensing selectivity: the global threshold, the number of cooperative nodes, the combining rule.  Then we examine the reliability of cooperative sensing upon subjecting some cooperative nodes to shadowing.

- Global threshold setting

Figure 4.18 shows the probability of false alarm and probability of detection at different $\gamma_n$ values. The threshold is normalized to the maximum coefficient value of the band spectrum estimated by FFT. However, determining the right threshold for all the nodes is based on the worst case scenarios for false alarms. This setting is achieved when the number of cooperative sensing nodes is 10 using OR combining rule.

It can be concluded from the figure that the choice of normalized threshold of 0.6 leads to a maximum probability of error of 0.1 for -20 dB SNR values. Lower SNR values will lead to higher probability of false alarm. However, the minimum SNR value used in our simulation is -20 dB. Reasonable probability of false alarm and detection can be achieved by adjusting the FFS parameters presented in Table 4.6.

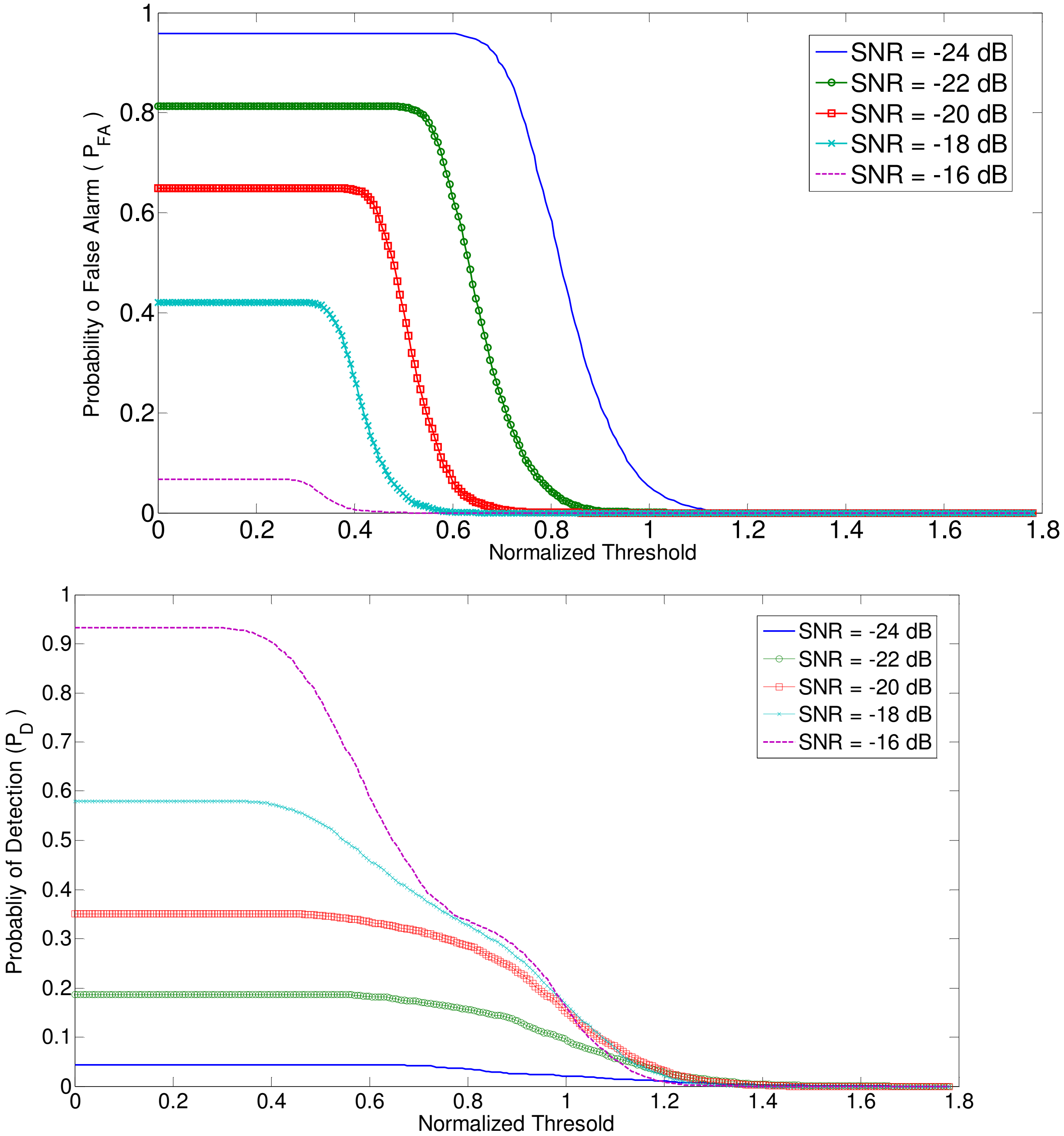

Figure 4.18: Probability of false alarm and the probability of detection at different threshold values for 10 sensing nodes

- The effect of the number of cooperative node

The narrowband cooperative sensing for 1, 2, 4 and 10 sensing nodes is simulated using the OR combining rule. Figure 4.19 illustrates the effect of the number of cooperative nodes on the probability of WM detection at different SNR values, while Figure 4.20 shows this effect on the probability of false alarm.

As can be noted for Figure 4.19 and Figure 4.20, increasing the number of cooperative nodes enhances the probability of detection at the expense of increasing the probability of false alarm. This effect on both probabilities diminishes as the number of cooperative nodes increases. This performance behavior is true for the OR combining rule, but it is different for other combining method as will be shown in the next section. Another observation is that detection performances of cooperative FFS sensing using different numbers of cooperative sensing converge to the same detection performance under very low SNR values. However, the probability of false alarm diverges at low SNR values under different numbers of cooperative nodes. This slight improvement in the detection performance is explained by the fact that the SNR values at all sensing nodes are equivalent. This is the worse case in cooperative detection because little information is introduced to the fusion center for another measurement at the same wireless channel condition. The cooperative sensing would provide much better performance gain upon having a portion of the fused data is collected at better ( larger ) SNR values.

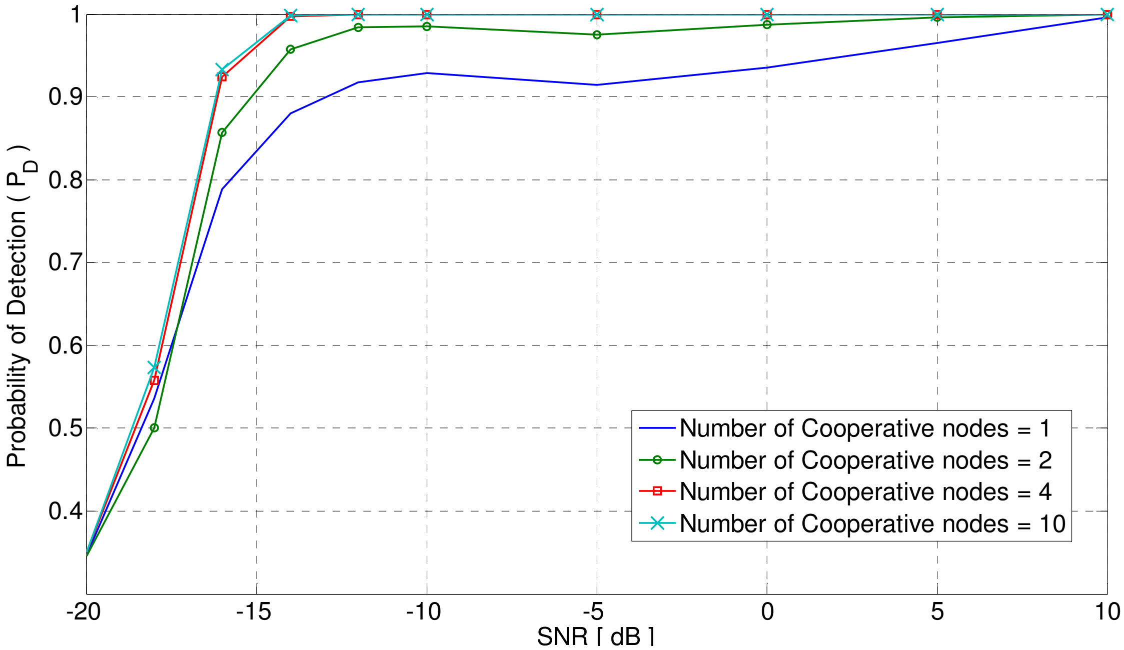

Figure 4.19: Probability of detection at different SNR values for different numbers of cooperating nodes and with a probability of false alarm of 10%

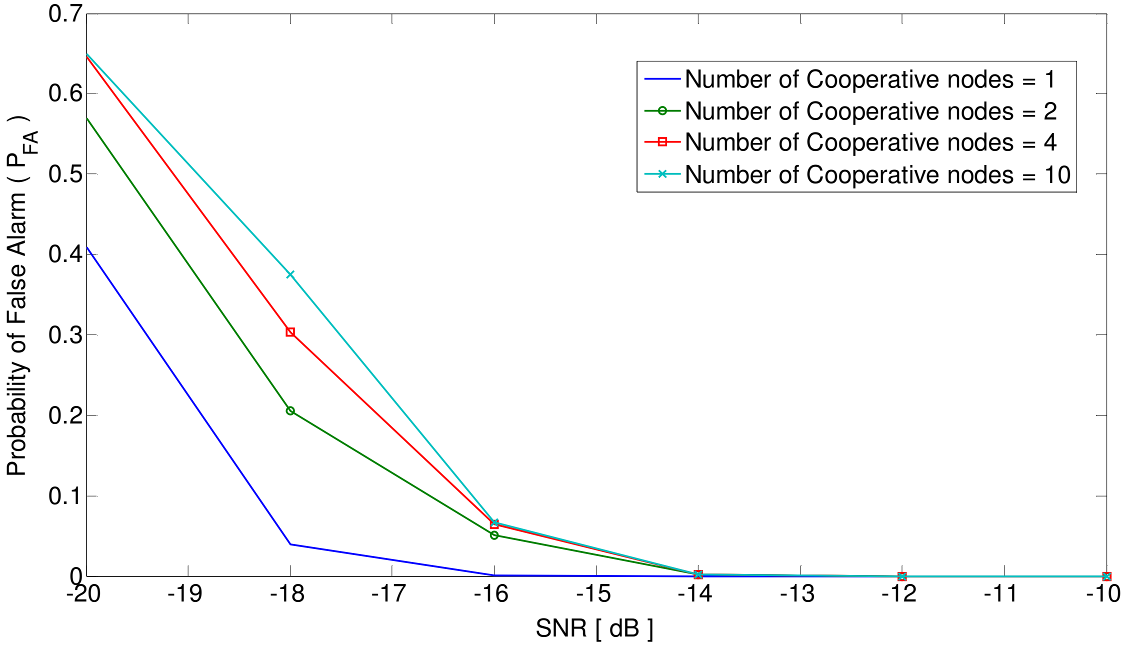

Figure 4.20: Probability of false alarm at different SNR values for different number of cooperative nodes

- The effect of the combining rule

The performance of narrowband cooperative sensing technique depends considerably on the combining rule. Figure 4.21 shows the probability of detecting WM signals for the three combining rules at different SNR values for a probability of false alarm of 0.1. The effect of the combining rule on the probability of false alarm is captured in Figure 4.22.

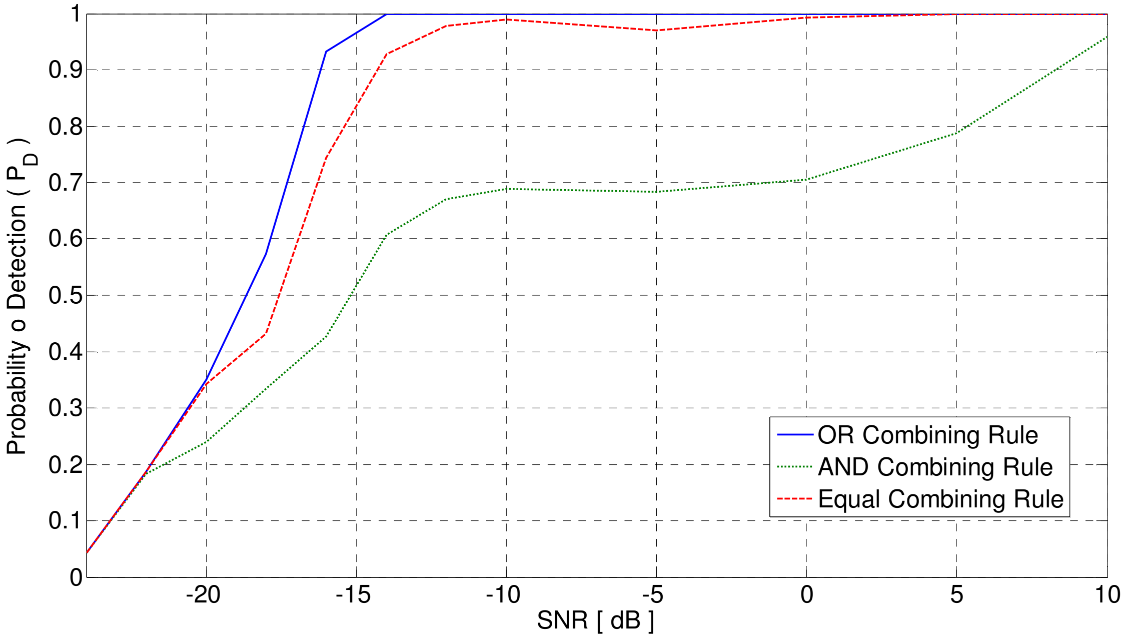

Figure 4.21: The effect of combining rule on probability of detection at different SNR values in a cooperative cognitive network of 10 nodes

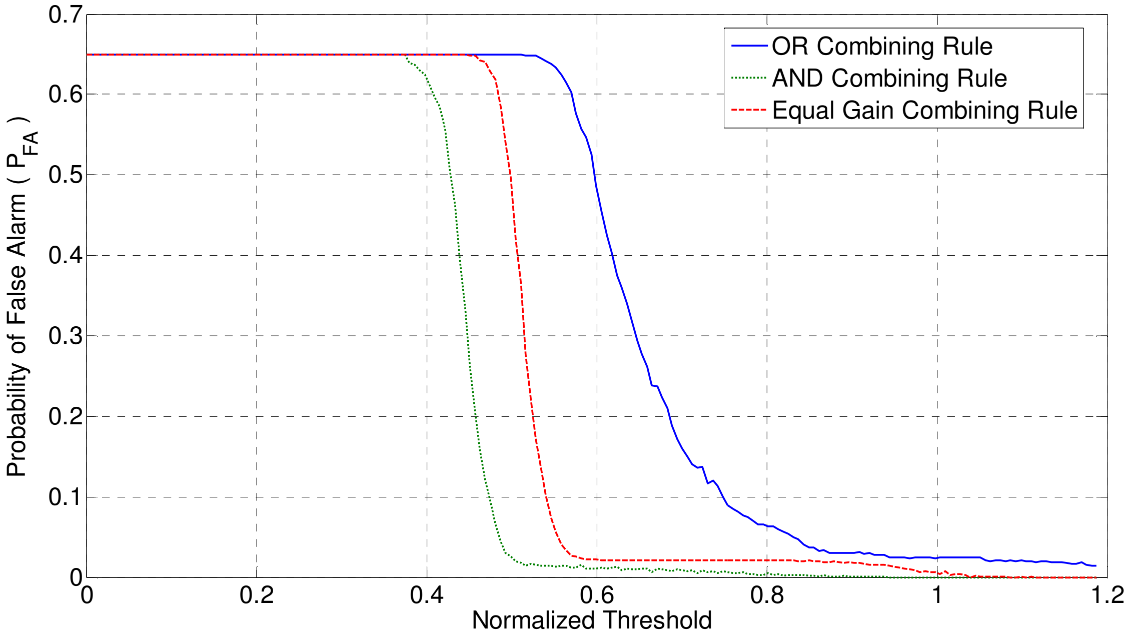

Figure 4.22: The effect of combining rule on probability of false alarm at different SNR values in a cooperative cognitive network of 10 nodes

- The performance of narrowband cooperative sensing under shadowing

This simulation aims to demonstrate the reliability of cooperate FFS sensing technique upon introducing faulty measurements to the fusion center. This faulty measurement is caused by shadowing. Assume that there is no shadowing correlation between adjacent nodes. This assumption is valid if the spatial distribution of the sensing nodes allows a sufficiently large distance between the nodes so that this correlation can be ignored. In a cooperative sensing network of 10 nodes, the performance of cooperative sensing performance in terms of probability of detection ad probability of false alarm (Figure 4.23) is shown when part of these nodes are under shadowing.

As can be seen in Figure 4.23, it appears that the relation between the loss in probability of detection and the number of shadowed cooperative nodes is approximately linear in medium to high SNR values (SNR values at which the probability of detection is more than 0.9 when no shadowing effect is introduced). However, we see that the probability of false alarm drops upon obtaining more shadowed measurements. This is because the false alarm occurs under large noise and interferences. Our shadow model aims to simulate total signal blockage which leads to WM signal misdetection. This fact is reflected in the probability of detection drop.

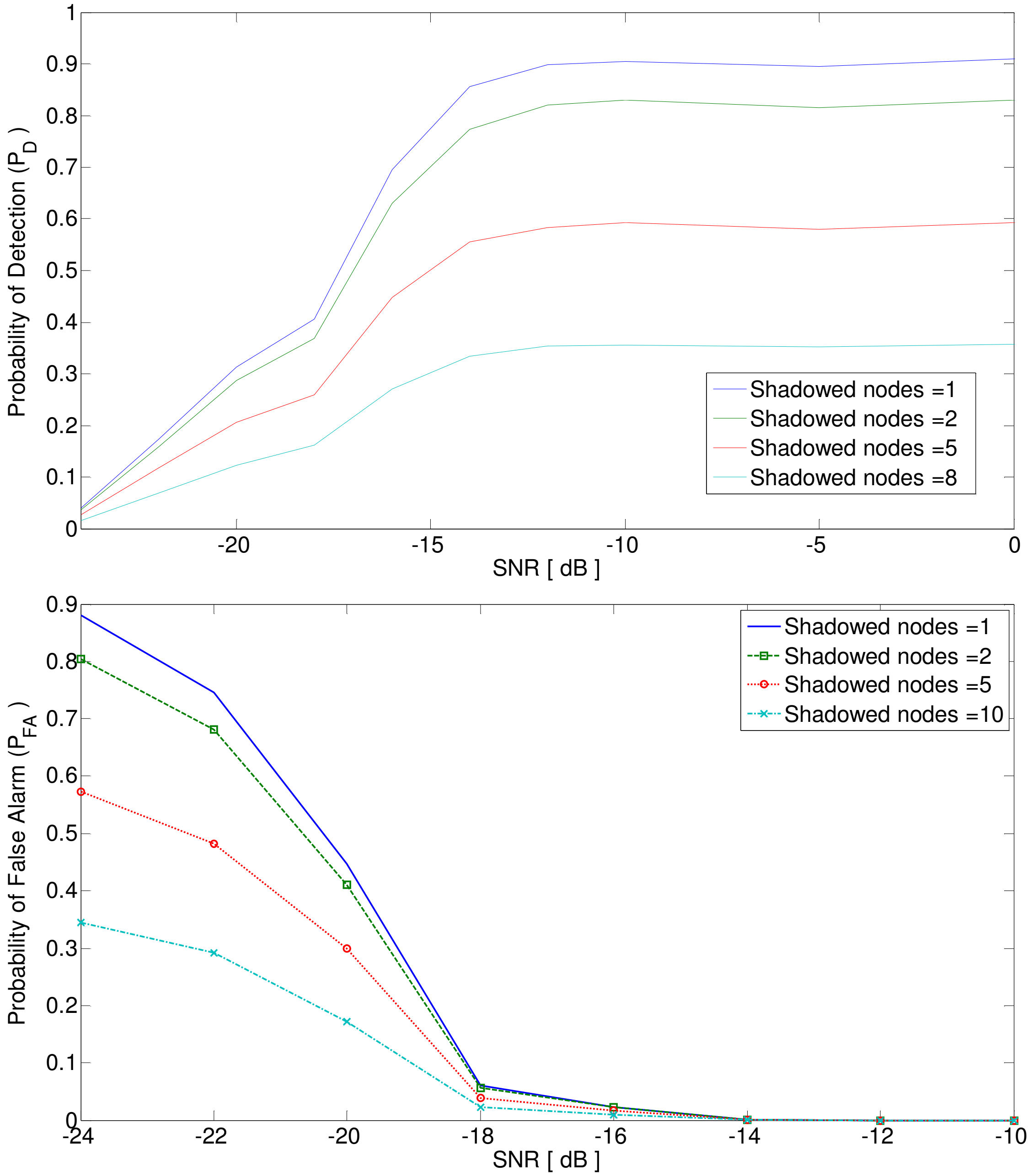

Figure 4.23: The effect of shadowing on probability of detection and probability of false alarm at different SNR values in a cooperative cognitive network of 10 nodes

Wideband collaborative sensing results

In a wideband cooperative sensing scheme, multiple adjacent frequency bands are monitored using several sensing nodes. Each is tasked to randomly sample a single band. Samples are then collected by the base station, where the FFS algorithm is conducted to estimate the collective shared spectrum comprised of these bands. In the following simulation results, we show the impact of number of bands on the selectivity of wideband cooperative sensing. The reliability of the algorithm is then tested by introducing a phase shift on samples collected by some cooperative nodes.

- Effect of number of bands

  In this simulation case study, a fixed number of energetic frequencies and their coefficients m = 100 is used to estimate the spectrum of a varying number of bands (2, 4 and 6 bands). For each band, a single WM signal is generated at different power level. The ratio between the maximum to the minimum WM signal power in the collective spectrum range is two. The probability of WM signal detection at each band is calculated and then the average probability of detection for all the bands is plotted in Figure 4.24. Since the probability of detection may vary from one band to another, the maximum and minimum probability of detection found in these bands are also included in the same figure to show performance consistency of FFS among the bands. The behavior of the probability of false alarm is also shown in Figure 4.25.

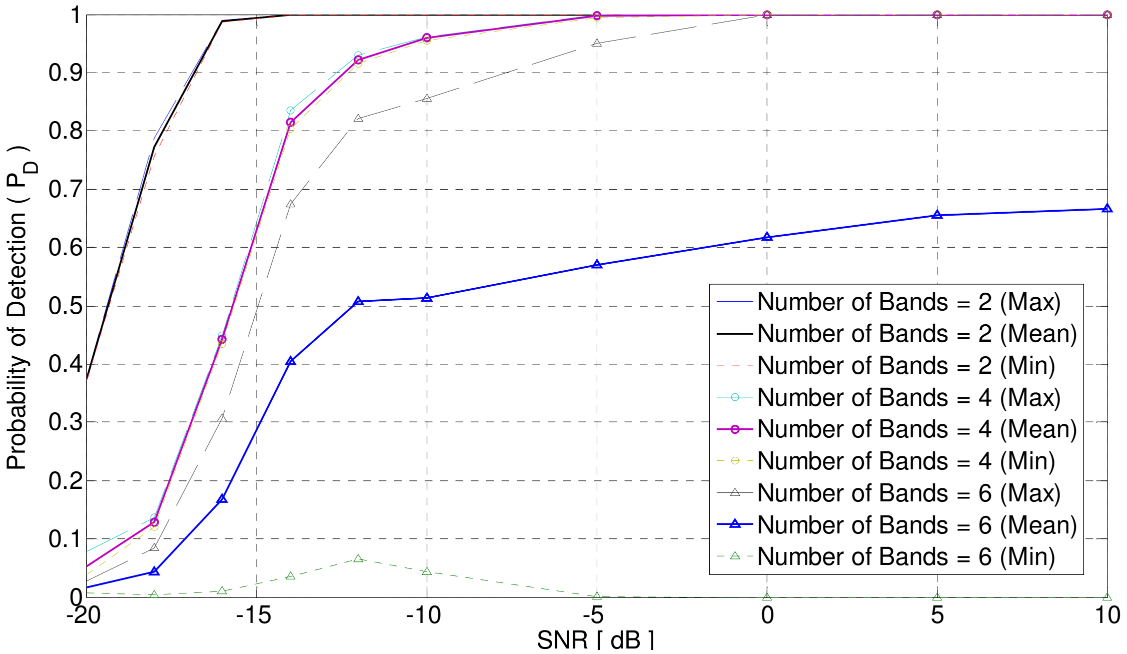

Figure 4.24: The probability of detection for wideband sensing technique at different SNR values

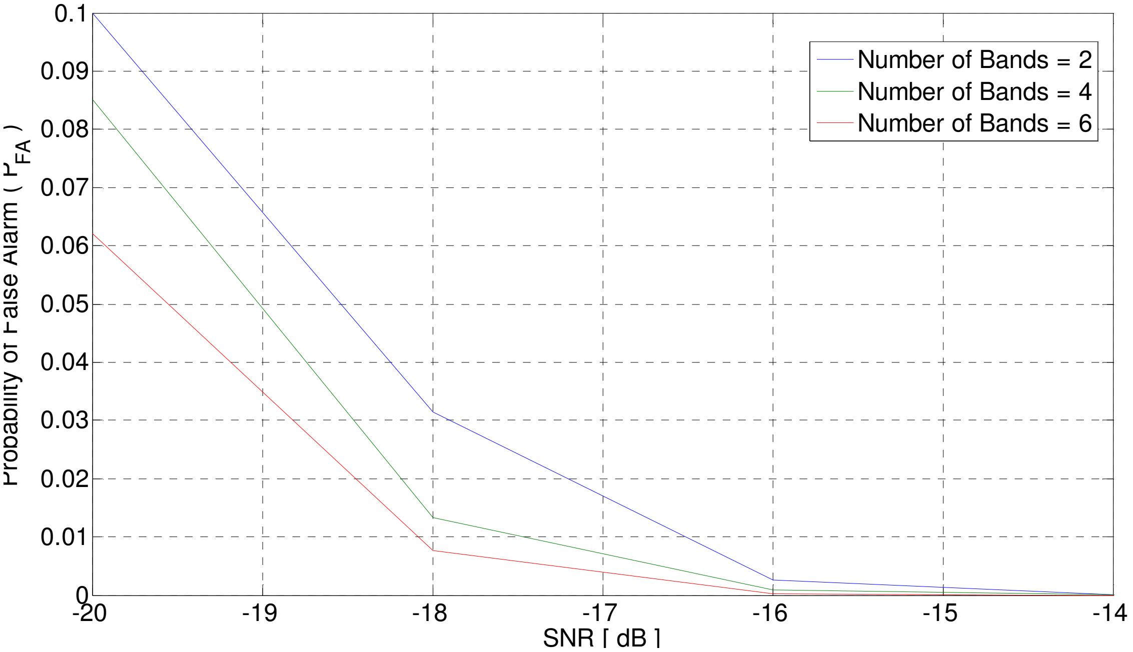

Figure 4.25: The probability of false alarm for wideband sensing technique at different SNR values

Upon doubling the number of bands from 2 to 4 adjacent bands, the detection sensitivity of the wideband sensing is dropped approximately by 5 dB but the variance in the probability of detection has slightly increased. This is because the average number of energetic frequency to be estimated per band has dropped from

50 to 25. However, when we expand the number of bands to 6 bands, the average probability of detection has dropped significantly and the variance in probability of detection among the bands has increased dramatically. This is because it is more likely for the signal power normalization to fail when more bands are added so that a band that contains a WM signal that is slightly more energetic than others (not perfectly normalized) will capture more of the energetic frequencies (m =100) budgeted to estimate the collective spectrum. The failure in normalizing the signal power in all the bands is attributed to the poor power estimation for each band using a set of random samples. Therefore, to improve the performance of the wideband cooperative sensing using FFS technique, we need to increase the number of energetic frequencies to be estimated by FFS algorithm. Also, improving the power estimation performance will improve the band power normalization and hence the collective detection performance of the wideband cooperative sensing technique.

- Effect of sampling phase delay

Although the cooperative nodes in the wideband sensing scheme are required to sample the frequency bands at the same time, it is possible that a small phase difference between the sampling at various nodes may occur due to imperfect wireless network synchronization. The effect of sampling phase error on the detection performance can be studied in two scenarios. First, the performance of wideband spectrum sensing for a shared spectrum of 10 bands (and 10 nodes) is tested upon having a phase error of sampling in eight cooperative nodes. The phase error is uniformly distributed between 0 and 5 sampling cycles. The sampling cycle in this context is defined as the smallest time interval between two consequent random samples which is reciprocal of the collective network bandwidth. In each band, a single WM signal exists. The base station estimates the most m=100 energetic frequencies in these 10 adjacent bands. Results in Figure 4.24 infer that this value of m is not sufficient to obtain an acceptable detection performance in very low SNR values. Therefore, the threshold was set up to limit the probability of false alarm to 0.1 for a minimum -14 dB SNR. In other words, the detection threshold was calibrated in this experiment to enable a reliable detection over the ten bands with -14 dB being the lowest SNR value that can be tolerated. Figure 4.26 shows the performance of detection with perfectly synchronized nodes and with eight phase shifted nodes.

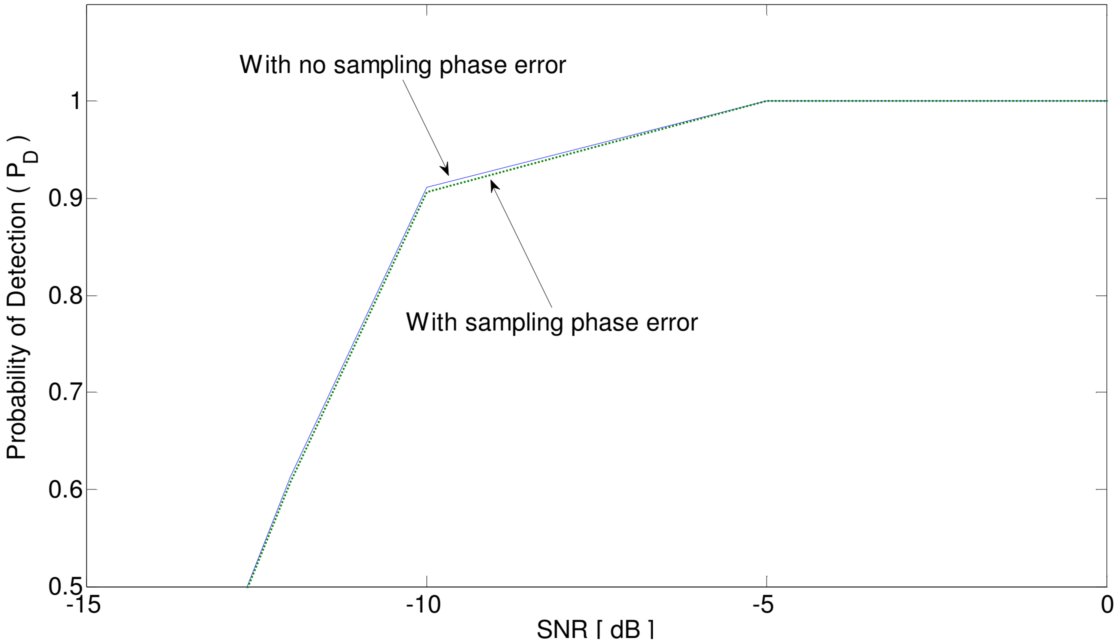

Figure 4.26: Detection performance loss in wideband cooperative sensing of 10 nodes where the sampling phase of 8 nodes is delayed by 5 sampling cycles with a proability of false alarm 0.1 at -14 dB

In the second scenario, the effect of 10 sampling cycle offset between two nodes sampling two different bands was examined. The base station estimates m=20 energetic frequencies in the spectrum composed of these two bands. Figure 4.27 shows the detection performance drop due to the sampling time offset.

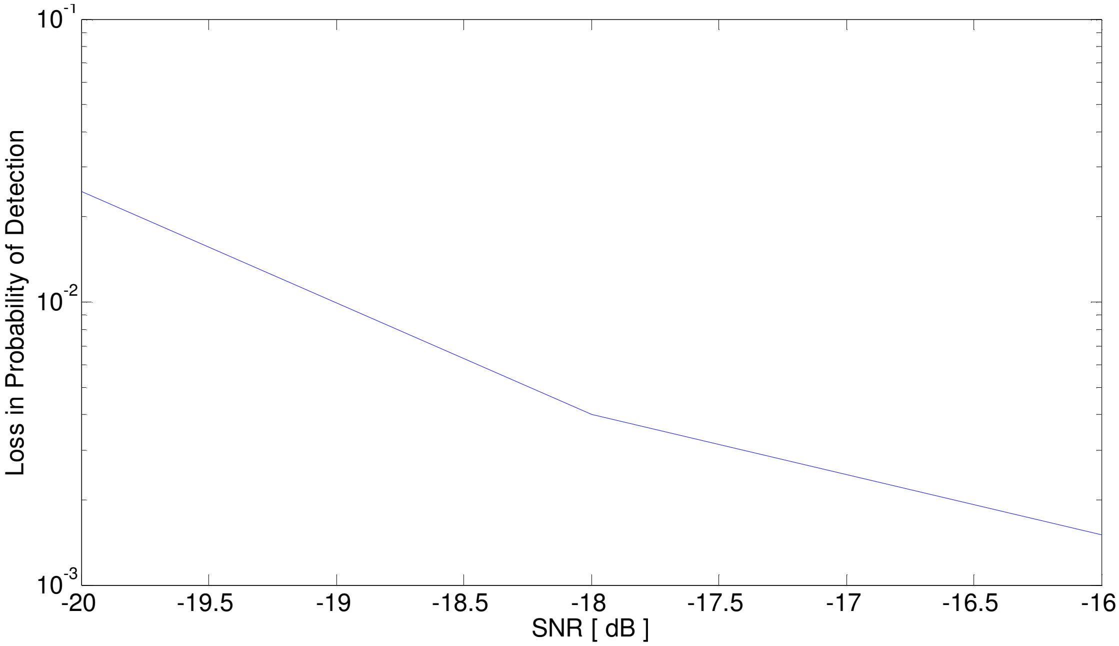

Figure 4.27: The effect of sampling phase delay of 10 sampling cycles for wideband cooperative sensing nodes of 2 nodes

It can be concluded from the from Figures 4.26 and Figure 4.27 that wideband cooperative sensing is insensitive to small phase errors between the samples collected from different cooperative nodes. This is because the FFS algorithm tries to fit the collected samples into a limited number of energetic frequencies m. Therefore, FFS has inherited de-noising effect to small samples disturbances. This result agrees with the effect of clock jitter on FFS estimation performance in [129].

## 4.5. Performance Analysis of Compressed Spectrum Sensing Based on Fast Fourier Sampling

Fast Fourier Sampling is a randomized compressive sampling technique used to approximate signal spectrum. Performance is evaluated by estimation outcome accuracy i.e. mean square error, algorithm complexity, and probability of success.

To understand the behavior of Fast Fourier Sampling (FFS), an in-depth study of the stability and complexity of its signal processing stages is required. The FFS architecture is described in Chapter 2, and in this chapter, a mathematical analysis for the performance of essential signal processing stages is provided. Of these, random sampling, norm estimation, Fourier coefficient estimation, frequency isolation, and bit testing are randomized stages based on a variety of configuration parameters, including the number of samples and iterations. Each stage is defined as an individual signal processing block with defined inputs, outputs and configuration parameters. Their performance bounds and optimum settings for their configuration parameters are explored.

This analysis is an extension of the theoretical analysis provided in [130]. It establishes a theoretical foundation for choosing optimum values for several key FFS configuration parameters. In addition, this study examines the collective FFS performance for spectrum sensing applications in cognitive radio networks.

### *4.5.1. Notation and Terminology*

The FFS algorithm is based on the “divide and conquer” algorithm methodology, and is broken into multiple simple stages that can be implemented with high computational efficiency. Iterating over these stages improve the quality of the results with low complexity addition.

## Definitions

| | |
|---|---|
| N | Signal size |
| $\psi_\omega$ | Fourier Transform Basis Function |
| $\hat{A}(\omega)$ | The vector representation in the frequency domain |
| $\langle A, \psi_\omega \rangle$ | The correlation between the vector *A* and a basis function $\psi_\omega$ |
| $\langle A, B \rangle$ | The correlation between vectors *A* and *B*. |
| $F * G$ | The convolution of F and G in time domain |
| $F \hat{*} G$ | The convolution of F and G in frequency domain |
| $\tilde{x}$ | The estimated value of random variable x |
| C | The set of complex numbers |
| Z | The set of integers numbers |
| $\|A\|_2$ | The second norm of vector *A* |
| $\omega$ | Radial frequency $\omega \in Z^N$ |

### *4.5.2. The Mathematical Formulation of FFS Algorithm Performance*

The FFS algorithm finds the most energetic frequencies in a *B* sparse signal $S \in C^N$ where *B*<<*N* with a probability that exceeds 1- $\delta$ and accuracy bounded by the following near-optimality bound:

$$\|S - R\|_2^2 \le (1 - \varepsilon).\|S - R_{opt}^B(S)\|_2^2 \qquad 4.5$$

where

*R* is the signal approximation composed of the *B* frequencies found in the algorithm

*R<sub></sub>*$R_{opt}$ is the best possible approximation of the signal *S* with *B* frequencies.

$\delta$ is the failure probability

$\varepsilon$ is the accuracy factor.

### *4.5.3. The Sampling Process*

Many stages in the Fast Fourier Sampling assume an ideal access to pairwise independent random signal samples. In particular, the coefficient variance reduction through averaging relies on the fact that samples are pairwise independent. The authors of [83] have introduced, however, the concept of *N*-arithmetic-progression-independence (AP-Independence). Without loss of generality, short arithmetic progressions with random initial points can serve as an example of random sampling indexes. The samples $X_k$ of a function $\phi$ are called AP-Independent if the joint distribution of the samples follow $X_k = \phi(n_k)$ where $n_K = a + bk$ , a is a random integer mod *N* and *b* is a random invertible integer mod *N*. These samples are viewed sufficiently random for estimation variance reduction purposed through averaging, and are adequately structured for reuse in several iterations. It should be noted that this random sampling process does not necessarily mean an overall reduction in sampling rate. Statistical characteristics such as the mean and variance of the average of AP-Independent samples are [83]:

$$E[X] = \frac{1}{\sqrt{N}} \hat{\varphi}(0) \qquad 4.6$$

$$\mathrm{var}[X] = \frac{1}{N}\left(\sum_n |\varphi(n)|^2 - |\hat{\varphi}(0)|^2\right) = \frac{1}{N}\sum_{\omega \neq 0} |\hat{\varphi}(\omega)|^2 \qquad 4.7$$

Figure 4.28 illustrates the impact of random sampling using the scheme on the signal frequency domain described earlier. The original signal has frequencies at locations 72, 94, 185 and 318; after random sampling, the frequencies are mapped to the locations 696, 994, 39, and 770 respectively.

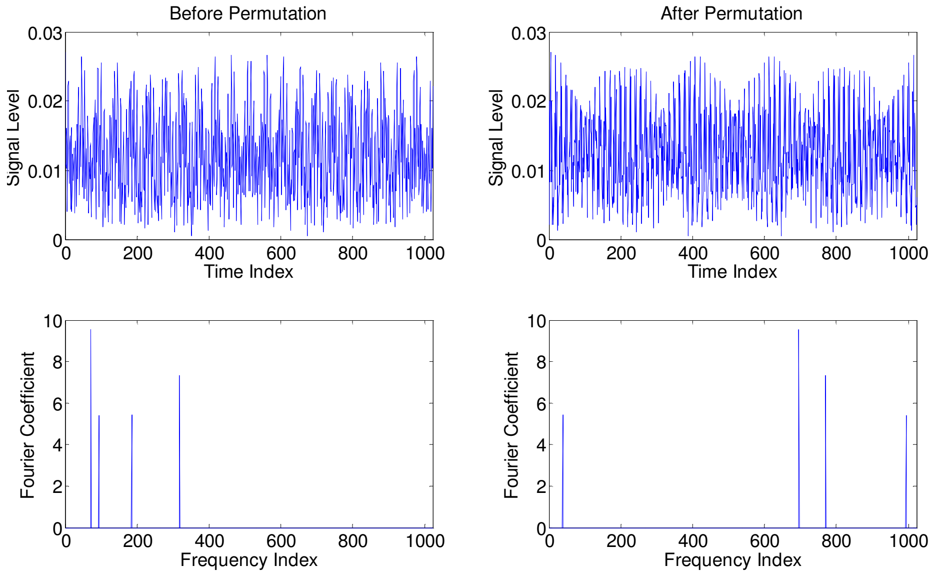

Figure 4.28 An example of random sampling a=5, b=31, $b^{-1}$=991

## *4.5.4. Frequency Identification Performance*

The purpose of frequency identification is to find the most energetic frequencies in the sampled signal. These are used in the coefficient estimation stage to calculate approximately the associate Fourier coefficients for these frequencies. Frequency identification is carried out in two stages: 1.) frequency isolation, where a bank of filters is used, and 2.) bit testing, where a sequential procedure of demodulating and simple filtering is performed for each bit of the frequency location in binary format. Section 4.2 details the implementations of each stage. Following is a mathematical analysis of the performance of each stage. Three performance metrics are evaluated in isolation stage:

- Probability of isolating *at least s* out of *m* major frequencies using a bank of *K* filters.
- Expected number of isolated frequencies using a bank of *K* filters
- Expected number of iterations to isolate all frequencies using a bank of *K* filters

For bit testing, the probability of success is calculated for a noisy sinusoidal signal.

### *4.5.4.1. Frequencies Isolation*

In the isolation stage, a signal comprised of *m* frequencies is processed by passing it through a bank of *K* filters, generating a collection of *K* signals, many of which have a dominant frequency. The filter bank divides the input total spectrum bandwidth *B* into *K* sub-bands, where each sub-band bandwidth is *B/K*. The chief goal of this process is to maximize the number of mono-frequency signals produced at each sub-band at the output of the isolation stage. In such a setup, the best case scenario is isolating all *m* frequencies (i.e., obtaining *m* mono-frequency signals) and the worst case scenario is failing to isolate a single frequency wherein all the *m* frequencies are gathered in one filter bank.

The problem of isolating frequencies using a bank of filters is not unlike the widely recognized “Balls in Bins” problem in the field of randomized algorithms where the balls resemble the frequencies and the bins are the bank of filters. The following analysis determines the appropriate size for the bank of filters *K* to achieve a desirable trade-off between isolation probability of success and isolation stage complexity. Intuitively, for a given number of frequencies *m*, the larger the size of bank of filters *K*, the higher the probability of isolating all frequencies and the higher the isolation process complexity.

The following analysis is based on the assumptions listed below:

- The location of the each of the *m* frequencies is a discrete random variable that is uniformly distributed between 1 and K.
- There is no frequency leakage between adjacent frequency sub-bands, indicating that the energy of a frequency located in a certain sub-band is concentrated in that single sub-band and not distributed across multiple sub-bands.

The probability of isolating at least *s* out of *m* frequencies in *K* frequency sub-bands

The event of isolating *at least s* out of *m* frequencies in *K* frequency sub-bands is identical to the event of having each of *s* sub-bands contains a mono-frequency signal while the remaining *K-s* sub-bands collectively have the remaining frequency components (*m-s*).

Consider a filter bank of size *K* and a signal comprised of *m* frequencies. Let us denote $p_{m,k}(s)$ (the probability of isolating at least *s* out of *m* frequencies in *K* sub-bands) is the probability of isolating *s* frequencies and the probability of having the remaining frequencies falling in any sub-band other than those containing isolated frequencies. This probability is calculated is follows:

$$p_{m,k}(1) = (1) \times \left(\frac{K-1}{K}\right)^{m-1}$$

$$p_{m,k}(2) = \left(1.\frac{K-1}{K}\right) \times \left(\frac{K-2}{K}\right)^{m-2}$$

:

$$p_{m,k}(s-1) = \left(1.\frac{K-1}{K}.\frac{K-2}{K}....\frac{K-s+2}{K}\right) \times \left(\frac{K-s+1}{K}\right)^{m-s+1}$$

$$p_{m,k}(s) = \left(1.\frac{K-1}{K}.\frac{K-2}{K}....\frac{K-s+1}{K}\right) \times \left(\frac{K-s}{K}\right)^{m-s} = \frac{\frac{K!}{(K-s)!}}{K^s}\left(\frac{K-s}{K}\right)^{m-s}$$

*so*,

$$p_{m,k}(s) = \frac{K!}{K^m(K-s)!}(K-s)^{m-s} \qquad 4.8$$

When *s*=*m*, which is the case of isolating all *m* frequencies, the probability of isolation becomes:

$$p_{m,k}(m) = \frac{K!}{K^m(K-m)!} \qquad 4.9$$

Calculating the factorial of *K* for large values of *K* (K > 170) may exceed the dynamic range of the floating point representation range. To facilitate the calculation of this probability, it is possible to use the Stirling approximation formula presented in equation 4.10:

$$n! > \sqrt{2\pi n}\left(\frac{n}{e}\right)^{n} \qquad 4.10$$

Hence, the approximated probability $p_{m,k(m)}$ becomes

$$p_{m,k}(m) = \frac{K!}{K^{m}(K-m)!} > \frac{\sqrt{2\pi K}\left(\frac{K}{e}\right)^{K}}{K^{m}.\sqrt{2\pi(K-m)}\left(\frac{(K-m)}{e}\right)^{(K-m)}} \qquad 4.11$$

$$\boxed{p_{m,k}(m) = \frac{\sqrt{K}}{e^{m}.\sqrt{(K-m)}}\left(\frac{K}{K-m}\right)^{K-m}} \qquad 4.12$$

Figure 4.29 shows the probability of isolating different numbers of frequencies *m* from 1 to 60 at different filter-bank sizes K. The figure below shows the results of analytical calculations and simulation for 1000 iterations.

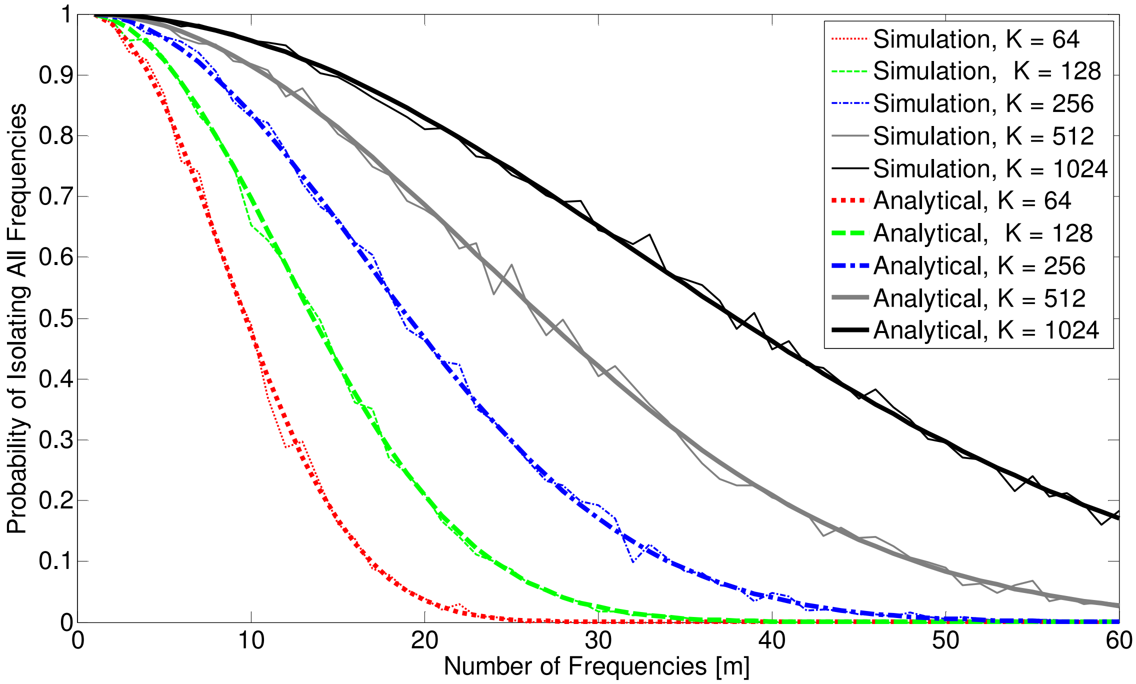

Figure 4.29 The probability of successful frequency isolation as a function of the number  of energetic frequencies for different filter-bank sizes

Calculating the minimum size $K_{min}$ required to achieve a probability p of isolating all the m frequencies can be done numerically. Figure 4.30 shows the minimum size $K_{min}$ to isolate $m$ = 1 to 100 frequencies with probabilities 0.9, 0.99, and 0.999. For example, in order to isolate 100 frequencies with a probability of 0.9, we need a filter-bank size of approximately $2^{16}$. Figure 4.30 shows the growth of the minimum size $K$ required to isolate frequencies at different probabilities of isolation 0.9, 0.99 and 0.999.

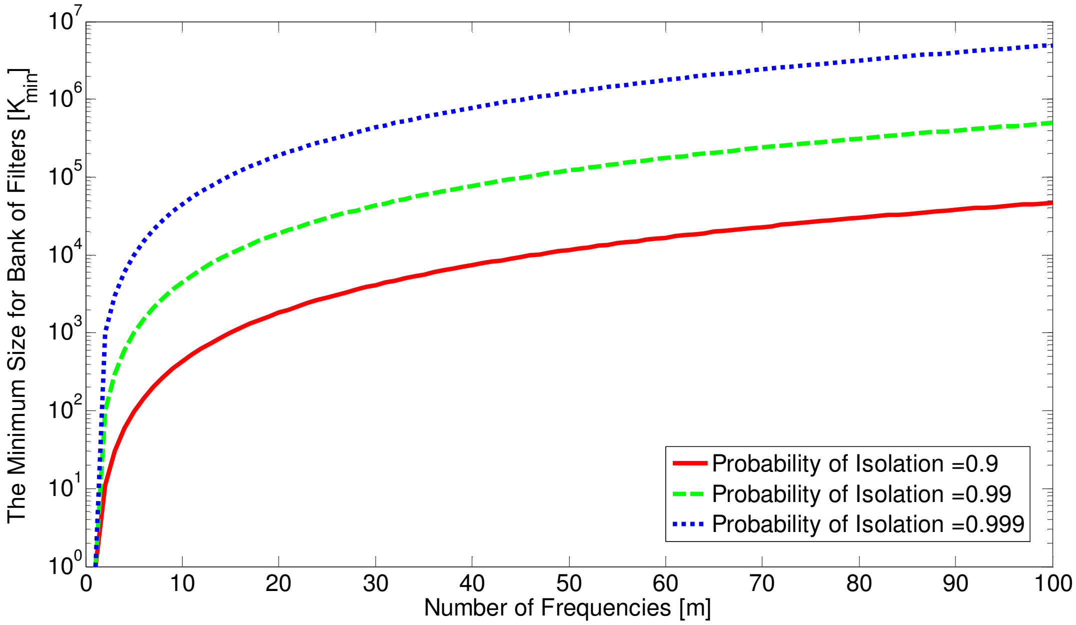

Figure 4.30 The minimum filter-bank size required to isolate m frequencies at different isolation success probabilities

In practice, relatively large filter bank size *K* may cause the FFS algorithm to run slower than FFT algorithm. Therefore, the scalability of the bank filter size is limited. It is desirable to maintain a low bank filter size during the FFS operation, and perform the isolation in multiple stages. In each stage, a certain percentage of the frequencies are isolated. Figure 4.31 shows the probability of isolating different percentages of 100 frequencies using different filter bank sizes. In the following analysis, the estimated number of iterations and complexity of iterative frequency isolation are examined.

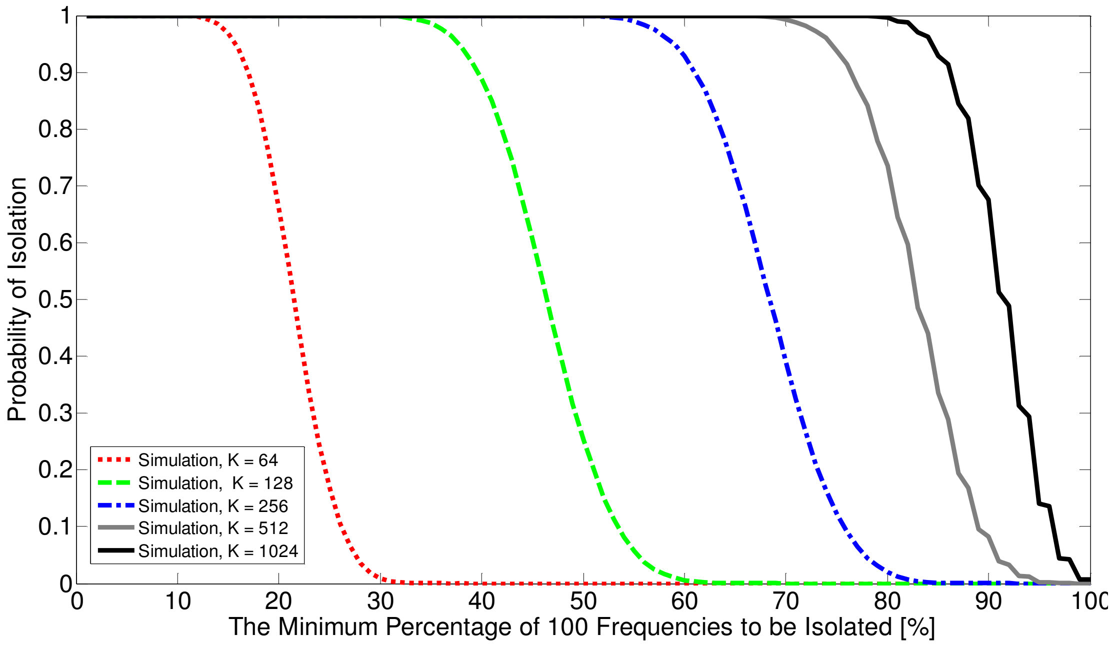

Figure 4.31 The probability of isolating different number of frequencies out of 100 frequencies at different filter-bank sizes

Expected number of isolated frequencies using a bank of *K* filters

Based on the results in figures 4.29, 4.30 and 4.31, it appears that the size of the bank of filters needed to isolate all sampled signal frequency components is too large, even for a relatively small number of frequency components. One way to reduce isolation complexity is by iterative methods. In each isolation iteration, a certain portion of received frequencies is isolated using a relatively small filter bank. These are suppressed in the received signal frequency spectrum if their coefficient estimations are sufficiently accurate. Hence, after only a few iterations, all frequency components are isolated. In order to calculate the estimated number of iterations needed during the isolation stage, the estimated number of frequencies to be isolated is required.

Let $E_K(s,m)$ denote the expected number of filters that will receive exactly $s$ frequencies when a signal of $m$ frequencies is applied to the filter-bank of size $K$. In the "balls and bins" problem setup, the expected number is equivalent to the expected number of bins receiving exactly $s$ balls after $m$ balls have been thrown. We are interested in finding $E_K(1,m)$ which is the expected number of filters that will receive a mono-frequency signal. However, in the following analysis, $E_K(s,m)$ is first calculated, then $s=1$ is substituted [131].

For simplicity, the process of applying $m$ frequencies signal to the bank of $K$ filters is described as a sequence of $m$ steps. At each step, one frequency is applied randomly to $K$ filters in the bank of filters.

Initially, the expected number of filters with no frequencies at their input $E(0,0)$ is $K$. After applying the first frequency to the bank of filters, the expected number of filters with no frequency at their input is decreased by 1, while the expected number of filters that have exactly one frequency is increased by one. Upon applying the $i^{th}$ frequency to the bank of filters, the probability that this frequency will land in a filter that has $s$ frequencies at its input is $E(s,i-1)/K$. If the event occurs, i.e., landing in a filter with $s$ frequencies at its input, the expected number of filters containing $s$ frequencies is decreased by one, and the expected number of filters with $s+1$ frequencies at their input is increased by one.

In other words, the expected number of filters containing $s$ frequencies after applying the $i^{th}$ frequency is equal to the expected number of filters containing $s$ balls in the previous step + 1x(the probability of the $i^{th}$ frequency to land in a filter that contains $s$-$1$ frequency) - 1.(the probability of the $i^{th}$ frequency to land in a filters that contains $s$ frequencies).

Mathematically,

$$E(s, i) = E(s, i - 1) + 1.\frac{E(s - 1, i - 1)}{K} - 1.\frac{E(s, i - 1)}{K} \tag{4.13}$$

In order to solve this recursive equation, the following initial conditions are defined:

- *E(0,0) = K* : the expected number of empty filters before applying any frequency
- *E(x,0) = 0* : the expected number of filters that contains *x*>0 frequencies before applying any frequency.

Let us define the normalized function $f(s, m) = \frac{E(s, m)}{K}$ : $f(0,0) = 1$ , $f(x,0) = 0$ and $f(-1, x) = 0 \qquad \forall x \geq 0$

Hence, equation 4.13 can be written as:

$$f(s, i) = A.f(s, i - 1) + B.f(s - 1, i - 1) : A = \left(1 - \frac{1}{K}\right), B = \left(\frac{1}{K}\right) \tag{4.14}$$

Table 4.7 shows the solution of this equation for *i=0,1,2,…,5* and *s=0,1,2,…,5*.

Table 4.7 Iterative equation solutions at different values of *s* and *i*

| i<br>s | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | *1* | *0* | *0* | *0* | *0* | *0* |
| 1 | *A* | *B* | *0* | *0* | *0* | *0* |
| 2 | $A^2$ | $2AB$ | $B^2$ | *0* | *0* | *0* |
| 3 | $A^3$ | $3A^2B$ | $3AB^2$ | $B^3$ | *0* | *0* |
| 4 | $A^4$ | $4A^3B$ | $6A^2B^2$ | $4B^3A$ | $B^4$ | *0* |
| 5 | $A^5$ | $5A^4B$ | $10A^3B^2$ | $10A^2B^3$ | $5AB^4$ | $B^5$ |

$$f(s, m) = \binom{m}{s} A^{m-s} B^{s} = \binom{m}{s}\left(1 - \frac{1}{K}\right)^{m-s}\left(\frac{1}{K}\right)^{s} \tag{4.15}$$

$$E(s, m) = K\,\frac{m!}{s!\,(m - s)!}\left(1 - \frac{1}{K}\right)^{m-s}\left(\frac{1}{K}\right)^{s} \tag{4.16}$$

In our case, the expected number of filters that receive a single frequency only is ( $s = 1$ ):

$$\boxed{E(1,\, m) \;=\; m.\left(1 - \frac{1}{K}\right)^{m-1}} \qquad 4.17$$

Figure 4.32 shows the expected number of isolated frequencies as a function of the number of frequency components in the samples signal for different filter bank sizes.

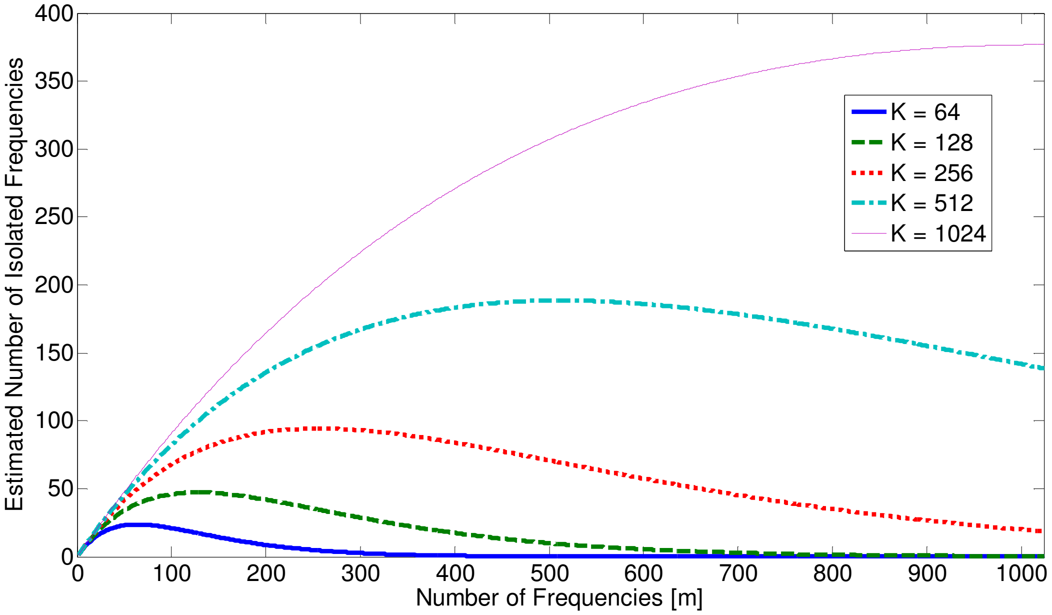

Figure 4.32 The estimated number of isolated frequencies as a function of the number of total energetic frequencies at different filter-bank sizes.

After few mathematical manipulations, the maximum number of frequencies expected to be isolated by a filter bank of size $K$ is:

$$m_{max} = \frac{1}{\ln\left(\frac{K}{K-1}\right)} \qquad 4.18$$

For $K>50$, $m_{max} \approx K$.

This means that the maximum expected number of isolated frequencies is obtained when the number of frequency components in the sampled signal is equal to the size of filter bank $K$.

When $m=K$:

$$E(1, K) = K.\left(1 - \frac{1}{K}\right)^{K-1} = \alpha K \text{ where } \boxed{\alpha \to 0.368 \qquad as \qquad K \to \infty} \qquad 4.19$$

Thus, for a given size of *K*, the estimated number of isolated balls does not exceed 0.36 of its size. Figure 4.33 shows the convergence of $\alpha$ to 0.368 as *K* increases.

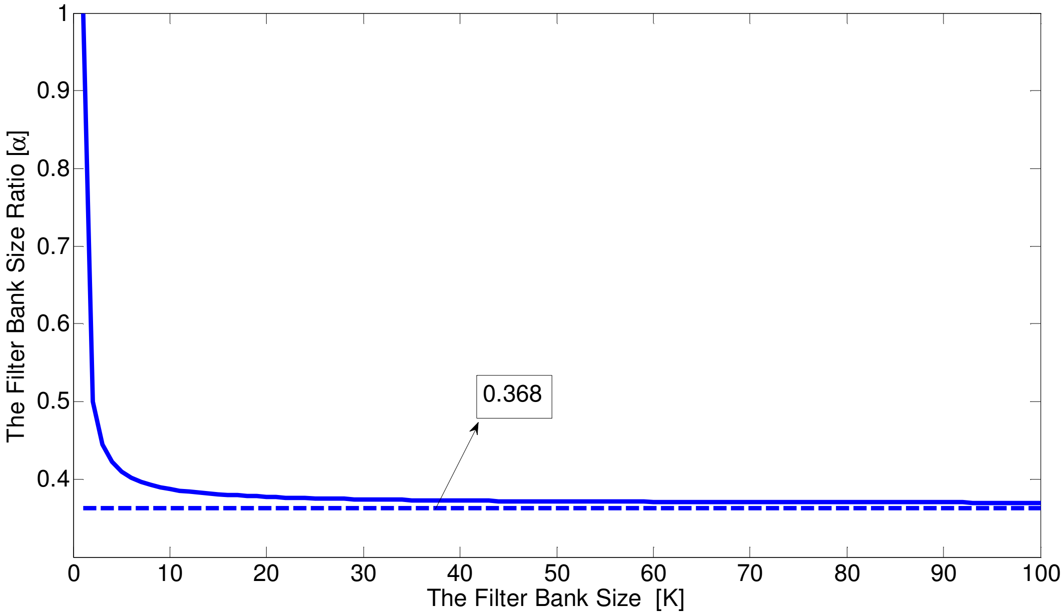

Figure 4.33 The filter bank size ratio as a function of the filter bank size K

The expected number of iterations to isolate all frequencies using a bank of *K* filters

The estimated number of isolated frequencies found in the previous section is useful for calculating the expected number of iterations required to isolate all frequencies in the received signal. This finding helps to determine the complexity of the iterative isolation process.

At each iteration, a certain proportion of frequencies is isolated, and is equal to the estimated number of isolated frequencies. In next iteration, the remaining frequencies (those not isolated) are subjected again to the isolation process. This iterative process continues until the remaining signal has no major frequency; hence, all frequencies are isolated.

$$E_1(1,\, m) \;=\; m.\left(1 - \frac{1}{K}\right)^{m-1}$$

$$E_2(1,\, m) \;=\; E_2(1,\, m - E_1(1,\, m)) \;=\; (m - E_1(1,\, m)).\left(1 - \frac{1}{K}\right)^{m - E_1(1,m) - 1}$$

$$= \left(m\left(1 - \left(1 - \frac{1}{K}\right)^{m-1}\right)\right).\left(1 - \frac{1}{K}\right)^{m\left(1 - \left(1 - \frac{1}{K}\right)^{m-1}\right) - 1}$$

So,

$$\boxed{E_i(1,\, m) \;=\; E_i\left(1,\, m - \sum_{j=1}^{i-1} E_j(1,\, m)\right) \;=\; \left(m - \sum_{j=1}^{i-1} E_j(1,\, m)\right).\left(1 - \frac{1}{K}\right)^{m - \sum_{j=1}^{i-1} E_j(1,m) - 1}} \qquad 4.20$$

Figure 4.34 shows the expected number of iterations to isolate all frequencies for different number of frequencies and filter bank sizes.

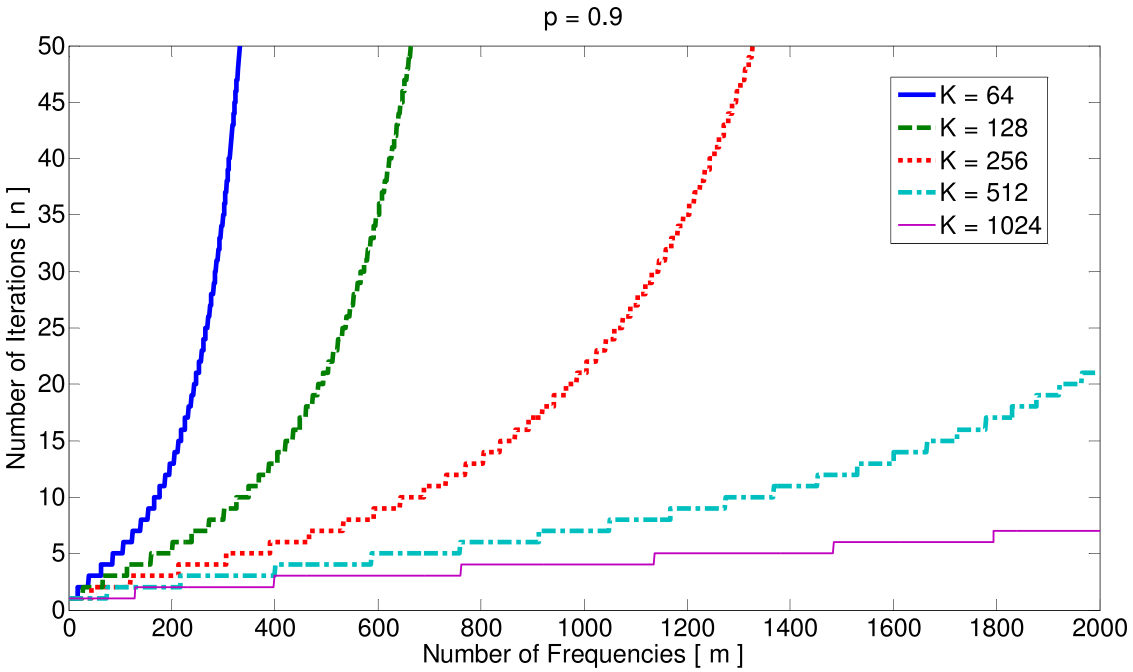

Figure 4.34. The total number iterations with probabilities 0.9 and 0.99 as a function of the total number of energetic frequencies at different filter-bank sizes

The complexity of the iterative isolation process

The complexity of the isolation process is determined by the number of iterations and the size of the filter bank. The filter bank is implemented using fast Fourier transform. Therefore, its complexity is in the order of *K.log(K)*. The total isolation complexity is :

$$\text{Complexity} = n\ .\ K.\ \log(\ K\ ) \qquad 4.21$$

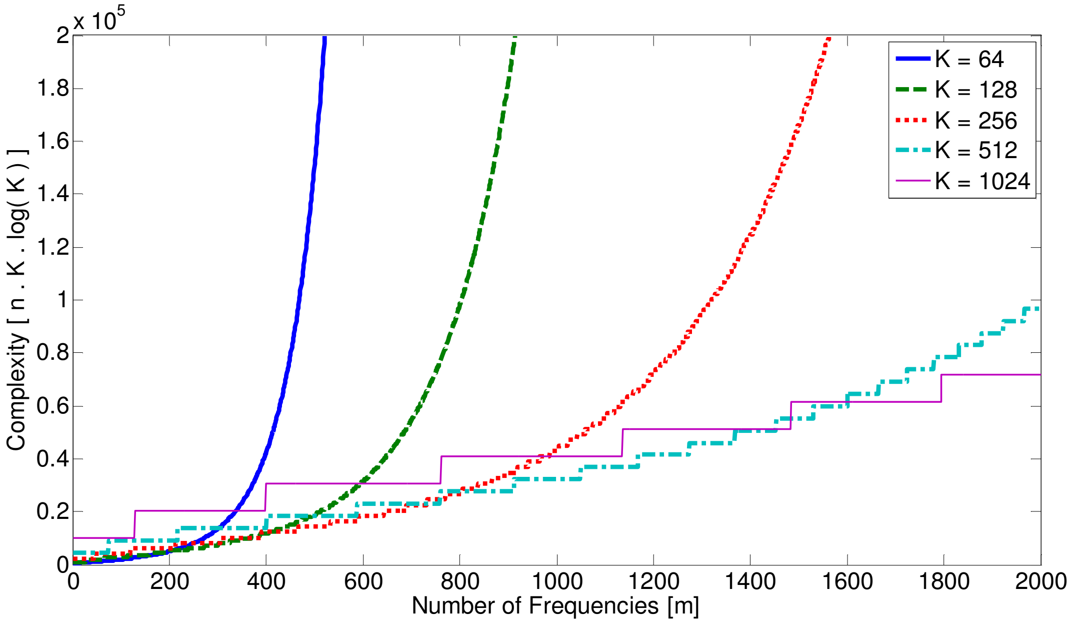

Figure 4.35 The complexity of isolation for different number of energetic frequencies and filter-bank sizes

Figure 4.35 shows the complexity growth of the isolation process as a function of the number of frequencies to be isolated with 90% probability of complete isolation at the last iteration.

Based on this figure, Table 4.8 lists the most efficient choice for *K* at various numbers of frequencies.

Table 4.8 Optimum filter bank sizes for different number of frequencies

| Number of Frequencies *m* | Filter Bank Size *K* |
|---|---|
| 1-200 | 64 |
| 201-400 | 128 |
| 401-800 | 256 |
| 801-1600 | 512 |
| 1600-3200 | 1024 |

### *4.5.4.2. Bit Testing*

Bit testing attempts to estimate the frequency of a given mono-frequency signal. Estimation ranges from *0* to *N-1*, where *N* is signal size. Hence, the frequency value can be represented using $log_2(N)$ bits. While it is possible to achieve this objective using a simple technique such as counting zero crossing events in the signal, the number of samples required by zero crossing counting is in the order of *N*. Bit testing attempts to identify the frequency using fewer samples in the order of $log_2(N)$. This technique is comprised of $log_2(N)$ stages. At each stage, a single bit of the binary frequency representation. The successful operation of this technique relies on the success of each bit testing stage. See [84] for more details on bit testing.

In the following analysis, the probability of success for each bit testing stage for a given noise level is examined. This result is useful to quantify the number of iterations needed at the frequency identification stage to obtain sufficiently reliable results.

Let $x_1$ and $x_2$ be the two random samples of a noisy sinusoidal signal of frequency *f*. These are used by the odd and even filters in the bit testing stage (see [84]):

$$x_1 = a.\cos\left(\frac{2\pi}{N} f.n + \phi\right) + \upsilon(n) \qquad 4.22$$

$$x_2 = a.\cos\left(\frac{2\pi}{N} f.\left(n + \frac{N}{2}\right) + \phi\right) + \upsilon\left(n + \frac{N}{2}\right) \qquad 4.23$$

where $\upsilon$is a Gaussian noise random variable with zero mean and variance $\sigma^2$. *N* is the signal size.

We define $x_+$ as the output of the even-frequency filter, and $x_-$ as the output of odd-frequency filter:

$$x_+ = |x_1 + x_2| = \left|2a.\cos\left(\frac{2\pi}{N} f.n + \phi + \frac{\pi}{2} f\right).\cos\left(\frac{\pi}{2} f\right) + \upsilon(n) + \upsilon\left(n + \frac{N}{2}\right)\right| \tag{4.24}$$

$$x_- = |x_1 - x_2| = \left|-2a.\sin\left(\frac{2\pi}{N} f.n + \phi + \frac{\pi f}{2}\right).\sin\left(\frac{\pi f}{2}\right) + \upsilon(n) - \upsilon\left(n + \frac{N}{2}\right)\right| \tag{4.25}$$

There are two possible scenarios for bit testing.

- Case I: When $f$ is even:

$$\begin{aligned} x_+ &= |x_1 + x_2| \\ &= \left|2a.\cos\left(\frac{2\pi}{N} f.n + \phi + \frac{\pi}{2} f\right).\cos\left(\frac{\pi}{2} f\right) + \upsilon(n) + \upsilon\left(n + \frac{N}{2}\right)\right| \\ &= \left|2a.\cos\left(\frac{2\pi}{N} f.n + \phi + \frac{\pi}{2} f\right) + \eta_+(n)\right| \end{aligned} \tag{4.26}$$

where $\eta_+(n) = \upsilon(n) + \upsilon\left(n + \frac{N}{2}\right)$

$$\begin{aligned} x_- &= |x_1 - x_2| \\ &= \left|\upsilon(n) - \upsilon\left(n + \frac{N}{2}\right)\right| = |\eta_-(n)| \end{aligned} \tag{4.27}$$

where $\eta_-(n) = \upsilon(n) - \upsilon\left(n + \frac{N}{2}\right)$

The bit testing is successful if $x_+ > x_-$. In other words, if:

$$\left|2a.\cos\left(\frac{2\pi}{N} f.n + \phi + \frac{\pi}{2} f\right) + \eta_+(n)\right| > |\eta_-(n)| \tag{4.28}$$

- Case II: When $f$ is odd:

$$\begin{aligned} x_+ &= |x_1 + x_2| \\ &= \left|\upsilon(n) + \upsilon\left(n + \frac{N}{2}\right)\right| = |\eta_+(n)| \end{aligned} \tag{4.29}$$

and

$$x_- = |x_1 - x_2| = \left| -2a.\sin\left(\frac{2\pi}{N} f.n + \phi + \frac{\pi}{2} f\right) + \eta_-(n)\right| \qquad 4.30$$

The bit testing is successful if $x_+ < x_-$, in other words, if:

$$\left| -2a.\sin\left(\frac{2\pi}{N} f.n + \phi + \frac{\pi}{2} f\right) + \eta_-(n)\right| > |\eta_+(n)| \qquad 4.31$$

Note that $\eta_-(n) = \upsilon(n) - \upsilon\left(n + \frac{N}{2}\right)$ and $\eta_+(n) = \upsilon(n) + \upsilon\left(n + \frac{N}{2}\right)$ are uncorrelated. Therefore, these can be treated as two independent random variables. Although the correlation is not necessarily a sufficient condition for independency, it is still used as an independency metric in practice [132]. The proof of this statement is given in lemma 4.1 listed below.

## Lemma 4.1

Assume that we have the following random variables *W* and *Z*:

$W = X + Y$

$Z = X - Y$

Where *X* and *Y* are two independent random variables.

$E(X.Y) = \rho_{X,Y}.var(X).var(Y) + E(X).E(Y)$

where $\rho_{X,Y}$ is the correlation coefficient of *X* and *Y*.

$$\begin{aligned} cov(W.Z) &= E(W.Z) - E(W).E(Z) \\ &= E((X+Y).(X-Y)) - E(X+Y).E(X-Y) \\ &= E(X^2) - E(Y^2) - (E^2(X) - E^2(Y)) \\ &= var(X) - var(Y) \end{aligned}$$

$$var(W) = E(W^2) - E^2(W) = E(X^2) + E(Y^2) + 2.E(X.Y) - E^2(X+Y)$$

$$= var(X) + var(Y) + 2E(X.Y)$$

$$= var(X) + var(Y) +2\,\rho_{X,Y}\,.var(X).var(Y)+2\,E(X).E(Y)$$

$$var(\,Z\,)= E(\,Z^2\,) - E^2(Z)= E(\,X^2\,) + E(\,Y^2\,) - 2.E(X.Y) - E^2(X-Y)$$

$$= E(\,X^2\,) + E(\,Y^2\,) - 2.\ \rho_{X,Y}\,.var(X).var(Y) - 2.E(X).E(Y) - E^2(X-Y)$$

$$corr(\,W, Z\,) \quad = \frac{\mathrm{cov}\,(\,W\,.\,Z\,)}{\mathrm{var}(W).\ \mathrm{var}(Z\,)}$$

For $X$ and $Y$ with equal variance, the covariance of $X$ and $Y$ is zero, hence the correlation between $W$ and $Z$ is zero.

In order to calculate the probability of success of the bit testing stage, the probability that inequalities 4.28 and 4.31 are true must be evaluated. The following mathematical treatment considers inequality 4.28. Similar analysis can be conducted for inequality 4.31.

Let $Y$ be a random variable that is comprised of random sample $X$ of a sinusoidal signal and a random Gaussian noise sample $\eta_-(n)$ with zero mean and variance $\sigma_2^2$.

$$Y \;=\; X \;+\; \eta_- \qquad 4.32$$

These components are defined as follows:

$$X \;=\; a.\,\sin(\omega n \;+\; \varphi) \qquad 4.33$$

where

$n$ is a uniformly distributed random variable in the range [0,2π],

$\omega$ is the radial frequency of the sinusoidal signal.

$a$ is the amplitude of the sinusoidal signal

$\eta_-(n)$ is a random variable with zero mean and variance $\sigma_2^2$. Since samples $\upsilon(n)$ and $\upsilon(n\,+\,N\,/\,2)$ are uncorrelated normally distributed random variables with equal variance $\sigma^2$, adding (or subtracting) them will result in a normally distributed random variable with variance $\sigma_2^2 = 2\sigma^2$.

The analysis aims to calculate the probability that $|Y| > |\eta_+(n)|$ where $\eta_+(n)$ is another random Gaussian noise sample with zero mean and variance $\sigma_2^2$, and is uncorrelated with $\eta_-(n)$ (based on lemma 4.1). First, the probability density distribution of $X$ is calculated ($p_X(x)$). Second, the probability density distribution of $Y$ ($p_Y(y)$) is derived from the convolution of $p_X(x)$ and $p_{\eta_-}(x)$. Third, the probability of success for bit testing stage is calculated by finding the probability that $|Y| > |\eta_+(n)|$.

Finding the probability density distribution of *X*

Calculating the probability density for a function of a random variable with known distribution can be performed using the cumulative density function. Since the distribution of *n* is known, i.e. uniform distribution, and $X = g(n) = a.\sin(\omega n + \varphi)$ is a function of *n*, the probability density function $p_X(x)$ can be calculated through the cumulative density function $F_X(x)$ as follows:

$$F_X(x) = P(X < x) = P(a.\sin(\omega n + \varphi) < x) = P\left(n < \frac{1}{\omega}\left(\sin^{-1}\left(\frac{x}{a}\right) - \varphi\right)\right) \qquad 4.34$$

$$p_X(x) = \sum_{i=1}^{m} \frac{p_n(n_i)}{g'(n_i)} \qquad 4.35$$

where

*m* is the number of roots for *X=g(n).* Assuming *n* takes the values *0, 1 , … , N-1,* the number of solutions for $a.\sin(\omega n + \varphi) = x$ is *m=Nω/π.*

*N* is the signal size,

$n_i$ are the roots for *X=g(t)* and they are functions of *X*. They are given by:

$$\begin{aligned} n_{2i-1} &= Z(X) + 2\pi i \\ n_{2i} &= \pi - Z(X) + 2\pi i \end{aligned} \quad \text{for } i=1,2,\ldots,m/2 \qquad 4.36$$

where $Z(X) = \dfrac{1}{\omega}\left(\arcsin\left(\dfrac{X}{a}\right) - \varphi\right)$

Therefore,

$$\begin{aligned} g'(n_{2i-1}) &= a\omega \cos(\omega(n_{2i-1}) + \varphi) = \\ &= a\omega \cos\left(\omega\left(\frac{1}{\omega}\left(\arcsin\left(\frac{X}{a}\right) - \varphi\right) + 2\pi i\right) + \varphi\right) \\ &= a\omega\sqrt{1 - \left(\frac{X}{a}\right)^2} = \omega\sqrt{a^2 - X^2} \end{aligned} \qquad 4.37$$

$$\begin{aligned} g'(n_{2i}) &= a\omega \cos(\omega(n_{2i-1}) + \varphi) \\ &= a\omega \cos\left(\omega\left(\pi - \frac{1}{\omega}\left(\arcsin\left(\frac{X}{a}\right) - \varphi\right) + 2\pi i\right) + \varphi\right) \\ &= a\omega\sqrt{1 - \left(\frac{X}{a}\right)^2} = \omega\sqrt{a^2 - X^2} \end{aligned} \qquad 4.38$$

Also,

$$p_n(n_i) = \frac{1}{N} \qquad \forall n \in [1, N] \qquad 4.39$$

$$p_X(x) = \frac{N\omega}{\pi} \frac{\left(\frac{1}{N}\right)}{\omega\sqrt{a^2 - x^2}} = \frac{1}{\pi . a\sqrt{1 - \left(\frac{x}{a}\right)^2}} \qquad 4.40$$

Figure 4.36 shows the probability density function of *X*.

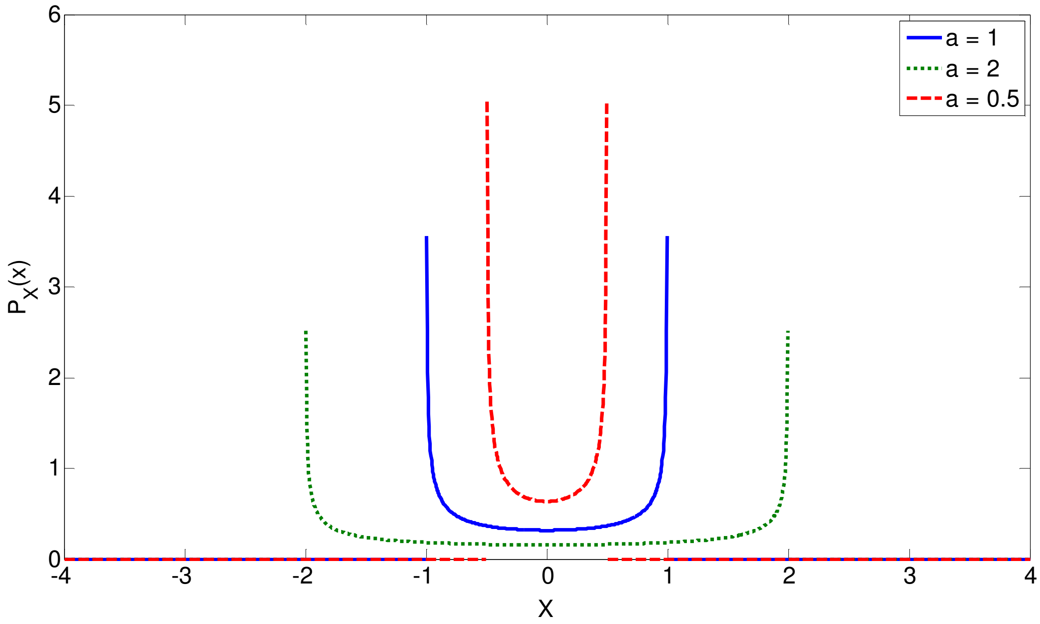

Figure 4.36. The probability density function of the sinusoidal signal sampled uniformly in the range [0, 2pi] for different sinusoidal amplitudes

<u>Finding the probability density distribution of *Y*</u>

$$Y = X + \eta_{-} \qquad 4.41$$

$$P_Y(x) = P_X(x) * P_{\eta_{-}}(x) = \int_{\tau=-\infty}^{\tau=+\infty} \frac{1}{\pi.a\sqrt{1-\left(\frac{\tau}{a}\right)^2}} \cdot \frac{1}{\sqrt{2\pi\sigma_2^2}}\, e^{\frac{-(x-.\tau)^2}{2\sigma_2^2}} .d\tau \qquad 4.42$$

Let us change the integration variable to $\tau' = \tau / a$. Then,

$$\boxed{P_Y(x) = \int_{\tau'=-1}^{\tau'=+1} \frac{1}{\pi.\sqrt{1-(\tau')^2}} \cdot \frac{1}{\sqrt{2\pi\sigma_2^2}}\, e^{\frac{-(x-a.\tau')^2}{2\sigma_2^2}} .d\tau'} \qquad 4.43$$

This integration can be calculated numerically using the Gaussian quadrature approximation formula below [133]:

$$\int_{-1}^{1} \frac{f(x)}{\sqrt{1-x^2}} .dx = \sum_{i=1}^{M} \omega_i f(x_i) + R_M \qquad 4.44$$

Abscissas: $x_i = \cos\frac{(2i-1)\pi}{2M}$, Weights: $\omega_i = \frac{\pi}{M}$, Reminder:

$$R_M = \frac{\pi}{2M!.2^{2M-1}} f^{(2M)}(\xi) \qquad -1 < \xi < 1 \mathrm{s}$$

M is the order of the approximation. In practice, the lower the noise variance, the higher the order of approximation needed to obtain sufficiently accurate pdf approximation. We have used *M=100* for $\sigma^2 = 0.01$. Figure 4.37 shows the probability density function of *Y*.

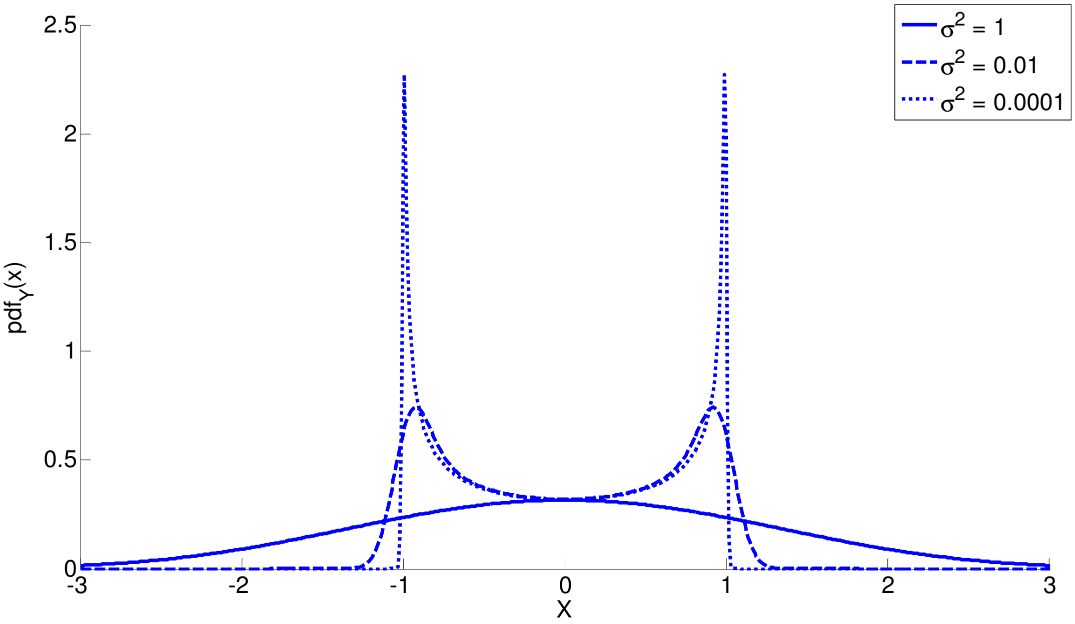

Figure 4.37 The probability density function of the noisy sinusoidal signal sampled uniformly in the range [0, 2pi] for different noise variance values

Note that the influence of the sinusoidal signal on the distribution of the samples diminishes as the noise variance increases until the point at which it nearly vanishes when the variance is equal to the amplitude of the sinusoidal signal.

Finding the probability that |*Y*|>|$\eta_{\pm}$*(n)*|

The symmetry of $p_Y(y)$ around *y=0* simplifies the calculation of $p_{|Y|}(y)$ which can be expressed as follows:

$$p_{|Y|}(y) = 2p_Y(y) \qquad y \in [0, \infty) \tag{4.45}$$

An example of $p_{|Y|}(x)$ is shown in Figure 4.38(a). Similarly,

$$p_{|\eta_+|}(x) = 2p_{\eta_+}(x) \qquad x \in [0, \infty) \tag{4.46}$$

An example of $P_{|\eta_+|}(x)$ is shown in Figure 4.38(b).

Let us define a random variable $Z$=$|Y|$-$|\eta_+(n)|$, the probability of success for bit testing is derived as follows:

$$P(|\ \mathrm{Y}\ |>|\eta_+(n)|) = P(|\ \mathrm{Y}\ |\ -|\eta_+(n)|\ >\ 0) = P(\mathrm{Z}\ >\ 0) = \int_0^\infty p_Z(x).dx \tag{4.47}$$

$$\boxed{\begin{aligned} p_Z(x) &= p_Y(x) * p_{\eta+}(-x) \\ &= \int_{\tau=-\infty}^{\tau=+\infty} \left(\sum_{i=1}^{M} \omega_i f(x_i)\right). \frac{1}{\sqrt{2\pi\sigma_2^2}}\, e^{\frac{-(x+a.\tau)^2}{2\sigma_2^2}} .d\tau \\ &= \int_{\tau=-\infty}^{\tau=+\infty} \left(\frac{\pi}{M} \sum_{i=1}^{M} \frac{1}{\sqrt{2\pi\sigma_2^2}}\, e^{\frac{-\tau^2}{2\sigma_2^2}}\right). \frac{1}{\sqrt{2\pi\sigma_2^2}}\, e^{\frac{-(x-.\tau)^2}{2\sigma_2^2}} .d\tau \end{aligned}} \tag{4.48}$$

This integration is calculated numerically as shown in Figure 4.38 (c). Note that the integration of the shaded area in Figure 4.38 (c) represents the probability of success for bit testing. Figure 4.39 shows the probability of success at different noise levels.

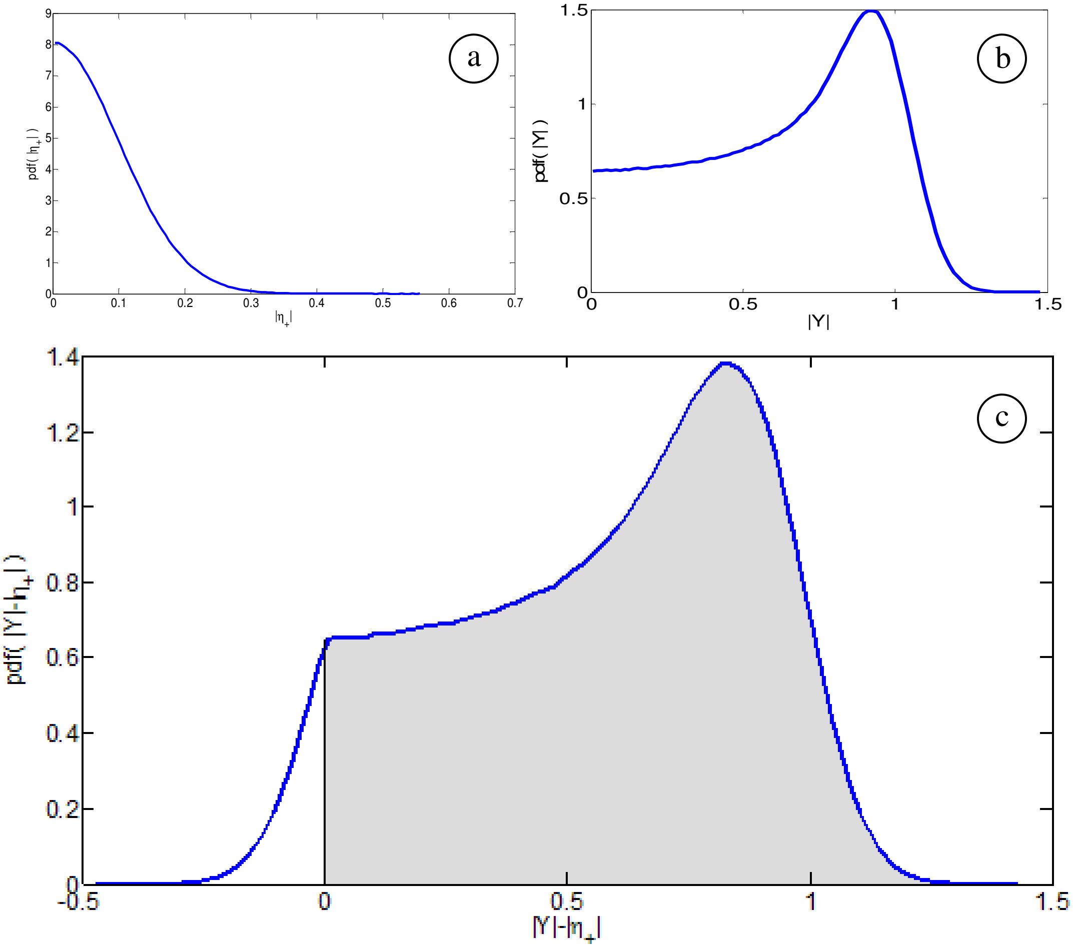

Figure 4.38. The probability density function of (a) the absolute value of white gaussian noise, (b) the absolute value of the random sinusoidal sample contaminated with Gaussian noise, (c) the convolution of (a) and (b)

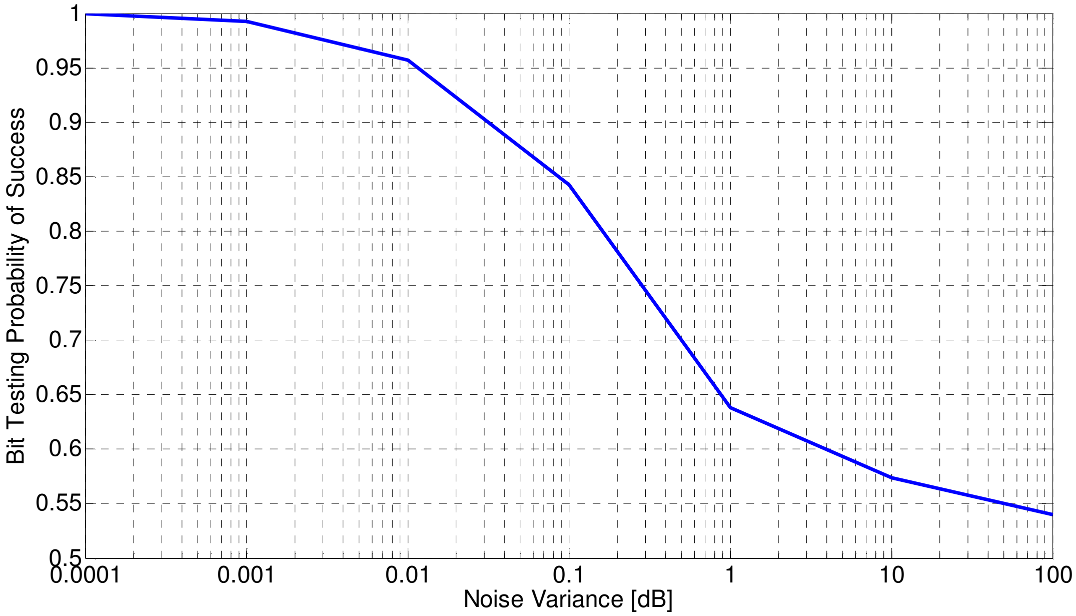

Figure 4.39. The bit testing probability of success at different white Gaussian noise level

### *4.5.5. Coefficient Estimation Performance*

The goal of the coefficient estimation stage is to obtain the Fourier coefficients of the frequencies identified during the frequency identification stage. However, only a few signal samples in the order of *m.poly(log(N))* are available for estimation. The following mathematical analysis investigates the performance of the estimation stage in terms of the probability of success in obtaining the coefficient values within a pre-defined accuracy limit. This accuracy limit is expressed as a fraction of the total received signal energy. The analysis is presented in three stages. First, the performance of single-step coefficient estimation is discussed. Second, a mathematical analysis is provided for finding the optimal values for the estimation algorithm parameters. Third, an iterative coefficient estimation performance is analyzed.

#### *4.5.5.1. Single-Step Estimation Method*

Assuming that we have an arbitrary signal $S$ , the following estimation algorithm can be used to estimate the Fourier coefficient $\hat{S}(\omega)$ by random sampling with an accuracy bounded by a fraction $\varepsilon$ of total signal energy ||*S*||, and a success probability 1-$\delta$ .

In mathematical context, the following algorithm produces a Fourier coefficient estimation $\tilde{\hat{S}}(\omega)$ that satisfies:

$$\Pr\left(\left|\tilde{\hat{S}}(\omega) - \hat{S}(\omega)\right|^2 \geq \varepsilon^2 \|S\|^2\right) \leq \delta \qquad 4.49$$

```
Algorithm 1.1 Estimating a single frequency coefficient

Input: Signal S, failure probability $\delta$, accuracy factor $\varepsilon$

Output: $\tilde{\hat{S}}(\omega)$

Initialization : $L = 4/\varepsilon^2$, $K = 7. \ln(1/\delta) +1$

1. For i = 1 to $K$
    For j = 1 to $L$
        Draw $n_{ij}$ ~ Uniform [ 0, 1, 2, … , $N$-1]
        Calculate $X_{\omega ij} = \langle S(n_{ij}), \phi_\omega(n_{ij})\rangle$

2. Take the mean of $L$ samples of $X_{\omega ij}$ and store it as $Y_{\omega i}$

3. Take the median of $K$ samples of $Y_{\omega i}$ and store it as $\tilde{\hat{S}}(\omega)$

4. Return $\tilde{\hat{S}}(\omega)$ as the estimate of $\hat{S}(\omega)$
```

Let *n* be a uniformly distributed random variable in the range *[1, N]*. Also, let *V(n)=NS(n)* be another random variable, which is a scaled signal sample *S(n)* at the time index *n*.

Assume that $X_\omega$ is the correlation between the random signal sample *V* and a Fourier basis function $\phi_\omega(n) = \frac{1}{\sqrt{N}} e^{\frac{-2\pi i\omega n}{N}}$. As a result, $X_\omega$ is a random variable such that:

$$X_\omega = \langle V, \phi_\omega \rangle \qquad 4.50$$

The mean and variance of the random variable $X_\omega$ is given as follows:

The mean:

$$E[X_\omega] = \sum_{n_i} V(n).\phi_\omega(n_i).\,\Pr(n = n_i) = \sum_{n_i} NS(n_i).\phi_\omega(n_i).\,\frac{1}{N} = \hat{S}(\omega) \qquad 4.51$$

The variance is bounded by the signal energy as shown:

$$\operatorname{var}[X_\omega] = E\left[\left|X_\omega - E[X_\omega]\right|^2\right] = E\left[\left|X_\omega - \hat{S}(\omega)\right|^2\right] = E\left[\left|X_\omega\right|^2\right] - \left|\hat{S}(\omega)\right|^2 \qquad 4.52$$

but,

$$\begin{aligned} E\left[\left|X_\omega\right|^2\right] &= \sum_{n_i} \left|NS(n_i).\phi_\omega(n_i)\right|^2 . \frac{1}{N} \\ &== \sum_{n_i} \left|NS(n_i).\frac{1}{\sqrt{N}} e^{\frac{-2\pi i\omega n}{N}}\right|^2 . \frac{1}{N} = \sum_{n_i} \left|S(n_i).e^{\frac{-2\pi i\omega n}{N}}\right|^2 \qquad 4.53 \\ &= \sum_{n_i} \left(S(n_i).e^{\frac{-2\pi i\omega n}{N}}\right)\left(S(n_i)^*.e^{\frac{2\pi i\omega n}{N}}\right) = \sum_{n_i} \left(S(n_i).S(n_i)^*\right) = \|S\|_2^2 \end{aligned}$$

So,

$$\operatorname{var}[X_\omega] = \|S\|_2^2 - \left|\hat{S}(\omega)\right|^2 \le \|S\|_2^2 \qquad 4.54$$

As noted, the variance of $X_\omega$ is relatively large. This is particularly true for signals of multiple frequencies and strong noise. For mono-frequency signals, this variance is equal to the noise energy. In order to improve the estimation accuracy, i.e., reducing the estimation variance, we take the average of *L* values of $X_\omega$ which is $Y_\omega$:

The mean: $E[Y_\omega] = E[X_\omega] = \hat{S}(\omega)$ 4.55

By assuming the samples are uncorrelated, the variance of the *L* time-averaged random variable is reduced by a factor of *L*. Hence, it is reasonable to assume that the variance upper bound is also reduced by the same factor:

$$\operatorname{var}[Y_\omega] = \frac{\operatorname{var}[X_\omega]}{L} \le \frac{\|S\|_2^2}{L} \qquad 4.56$$

To further improve the stability of the algorithm output, the median of $K$ samples of $Y_\omega$ is calculated. The median operation immunes the algorithm against noisy spikes. It is known that the median is a central point that minimizes the average of absolute deviation, and the mean minimizes the sum of squares. The output of the median operation is the estimated Fourier coefficient $\tilde{\hat{S}}(\omega)$. The following analysis calculates the probability that the output of the coefficient estimation algorithm $\tilde{\hat{S}}(\omega)$ fails to comply with the accuracy limits:

If $\left|\tilde{\hat{S}}(\omega) - \hat{S}(\omega)\right|^2 \geq \Gamma \Rightarrow$ at least half of $Y_\omega$ samples satisfies $\left|Y_\omega - \hat{S}(\omega)\right|^2 \geq \Gamma$

Based on Markov inequality, the probability that any $Y_\omega$ exceeds the limit $\Gamma$ is :

$$\Pr\left( \left|Y_\omega - \hat{S}(\omega)\right|^2 \geq \Gamma \right) \leq \frac{E\left(\left|Y_\omega - \hat{S}(\omega)\right|^2\right)}{\Gamma} = \frac{\operatorname{var}(Y_\omega)}{\Gamma} \qquad 4.57$$

Since this probability is always less than the variance divided by a threshold $\Gamma$, substituting the variance by its upper bounds preserves the truth of the statement:

$$\Pr\left( \left|Y_\omega - \hat{S}(\omega)\right|^2 \geq \Gamma \right) \leq \frac{\|S\|_2^2}{L.\Gamma} \qquad 4.58$$

By expressing the threshold $\Gamma$ as a proportion of the total signal energy $\Gamma = \varepsilon^2\|S\|_2^2$. Then,

$$\Pr\left( \left|Y_\omega - \hat{S}(\omega)\right|^2 \geq \varepsilon^2\|S\|_2^2 \right) \leq \frac{1}{L.\varepsilon^2} = p_{fy} \qquad 4.59$$

The probability calculated in the equation above represents *the probability of failure for $Y_\omega(p_{fy})$*. The mean sample $Y_\omega$ is considered successful if the difference between this sample and the actual coefficient value $\hat{S}(\omega)$ is less than a threshold $\Gamma = \varepsilon^2 \|S\|_2^2$.

By knowing the probability for any $Y_\omega$ to exceed the threshold $\Gamma = \varepsilon^2 \|S\|_2^2$, it is possible to calculate the probability that at least half of the $Y_\omega$ samples, i.e. *K/2* samples) exceed this limit (i.e. the coefficient estimation algorithm probability of failure $\delta$ :

$$\begin{aligned}
&\Pr\left( \left| \tilde{\hat{S}}(\omega) - \hat{S}(\omega) \right|^2 \geq \varepsilon^2 \|S\|_2^2 \right) \leq \\
&\sum_{i=K/2}^{K} \binom{K}{i} . \left( \Pr\left( \left| Y_\omega - \hat{S}(\omega) \right|^2 \geq \varepsilon^2 \|S\|_2^2 \right) \right)^i \left( \Pr\left( \left| Y_\omega - \hat{S}(\omega) \right|^2 < \varepsilon^2 \|S\|_2^2 \right) \right)^{K-i} \\
&= \sum_{i=K/2}^{K} \binom{K}{i} \left( \frac{1}{L.\varepsilon^2} \right)^i \left( 1 - \frac{1}{L.\varepsilon^2} \right)^{K-i} \\
&= \sum_{i=K/2}^{K} \binom{K}{i} (p_{fy})^i (1 - p_{fy})^{K-i} \\
&= I_{p_{fy}}\left( \frac{K}{2}, K - \frac{K}{2} + 1 \right) = I_{p_{fy}}\left( \frac{K}{2}, \frac{K}{2} + 1 \right)
\end{aligned} \qquad 4.60$$

It is more efficient to choose *K* as an odd number for simplifying the median operation. In this case, the probability above becomes:

$$\boxed{\begin{aligned}
&\Pr\left( \left| \tilde{\hat{S}}(\omega) - \hat{S}(\omega) \right|^2 \geq \varepsilon^2 \|S\|_2^2 \right) \leq \sum_{i=(K+1)/2}^{K} \binom{K}{i} (p_{fy})^i (1 - p_{fy})^{K-i} \\
&= I_{p_{fy}}\left( \frac{K+1}{2}, K - \frac{K+1}{2} + 1 \right) = I_{p_{fy}}\left( ceil\left(\frac{K}{2}\right), floor\left(\frac{K}{2}\right) + 1 \right)
\end{aligned}} \qquad 4.61$$

Hence,

$$\boxed{\Pr\left( \left| \tilde{\hat{S}}(\omega) - \hat{S}(\omega) \right|^2 \geq \varepsilon^2 \|S\|_2^2 \right) \leq I_{p_{fy}}\left( ceil\left(\frac{K}{2}\right), floor\left(\frac{K}{2}\right) + 1 \right) = \delta} \qquad 4.62$$

for $K = 1, 3, 5, \ldots.$, where,

$I_p(x,\ y) \equiv \dfrac{B(p;\ x,\ y)}{B(x,\ y)}$ is the regularized incomplete beta function.

$B(x, y)$ is the beta function , $B(p;\ x,\ y)$ is the incomplete beta function.

Figures 4.40 and 4.41 show the relations between the coefficient estimation algorithm probability of failure $\delta$, the number of samples $K$ and the probability of failure for $Y_\omega[p_{fy}]$.

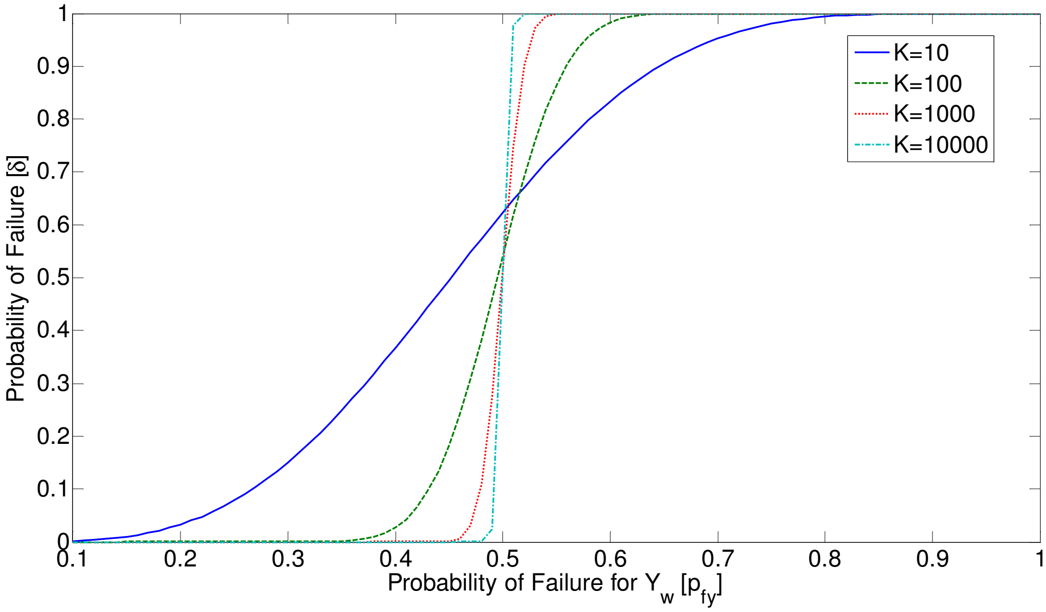

Figure 4.40. The probability of estimation failure as a function of probability of failure for *Yw* at different median window sizes

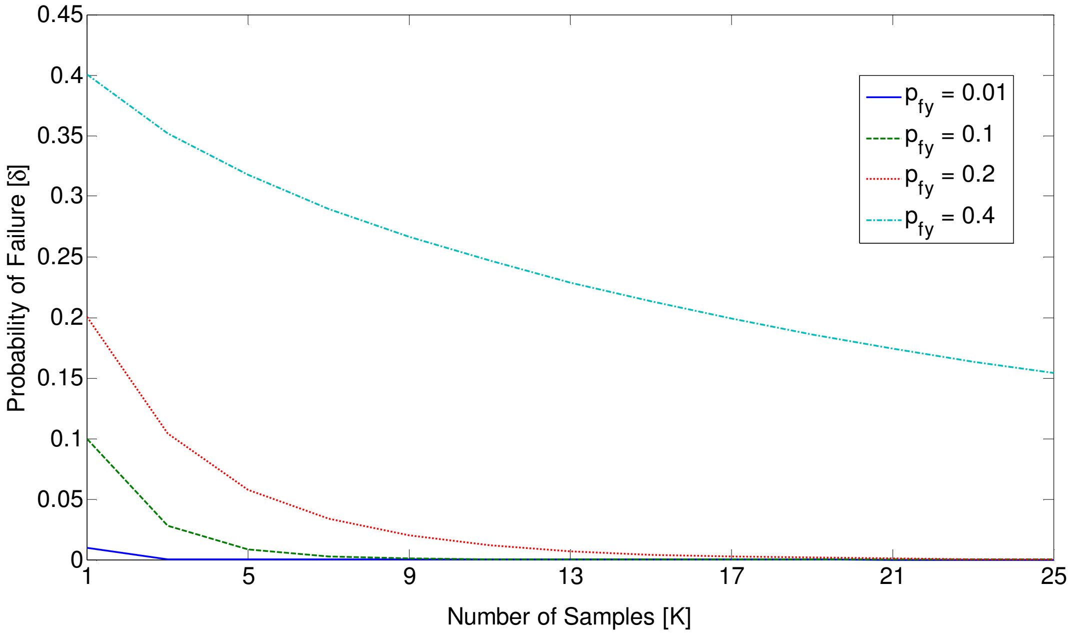

Figure 4.41. The probability of estimation failure as a function of median window sizes at different probabilities of failure for *Yw*

It should be noted that the probability of failure for $Y_\omega$ is directly related to the parameters $L$ and $\varepsilon$ through the following relation:

$$p_{fy} = \frac{1}{L.\varepsilon^2} \tag{4.63}$$

Generally, decreasing the probability of failure for $Y_\omega$ (by increasing $L$ or $\varepsilon$) or increasing the parameter $K$ will result in a reduction in the estimation algorithm probability of failure $\delta$. However, if the probability of failure for $Y_\omega$ is more than 0.5, then the estimation algorithm will be more likely to fail $\delta$ >0.5 regardless of the value of *K,* as can be shown in Figure 4.40. In fact, the larger the K, the more likely the estimation algorithm will fail if $\delta$ > 0.5. The opposite is also true. If $\delta$ < 0.5, then the larger the *K*, the more likely the estimation algorithm will succeed. In Figure 4.41, it can be noted that the reduction rate in probability of failure increases as $p_{fy}$ decreases. While $\delta$ decreases slowly as $K$ increases when $p_{fy}=0.4$, it drops faster when $p_{fy}$ is as small as 0.2 or slower as $K$ increases.

If the target probability of failure for $Y_\omega$ is relatively close to 0.5 (e.g. 0.4), then even as much as $K$=25 is not sufficient to achieve this target if $p_{fy}>0.4$. For $p_{fy}=0.2$, $K=3$ is sufficient to obtain a probability of failure of 0.1. The median operation is not necessary for $p_{fy} \leq 0.1$. While this may be appealing, it is important to observe the number of samples required, because $L$ increases as $p_{fy}$ decreases. More precisely, for estimation accuracy bounds of 0.01, the sufficient values of parameter $L$ are approximately 1000000, 100000, 50000, and 25000 for $p_{fy}$= 0.01, 0.1, 0.2, 0.4 respectively.

Figures 4.42 depicts the total number of samples required for different probability values of failure $\delta$ and the accuracy bound $\varepsilon$. Figure 4.42 shows that the smaller the median window size $K$, the less the total number of samples $K.L$ for most of probabilities of failure < 0.5. However, for small probabilities of failure < 0.04, an optimum value of $K$ exists to achieve the minimum number of samples $L.K$. Table 4.9 summarizes the optimum $K$ values and their associated ranges of probability of failures derived from Figures 4.43 and 4.44.

Table 4.9 The optimum median window size *K* for different probability of failures

| Probability of failure ranges | Optimum *K* value to achieve minimal total # of samples *L.K* |
|---|---|
| < 0.002 | 9 |
| 0.002-0.006 | 7 |
| 0.006-0.01 | 5 |
| 0.01-0.04 | 3 |
| >0.04 | 1 |

In the next section, the optimum values of $L$ and $K$ are determined as a function of design parameters $\varepsilon$ and $\delta$ to achieve the least number of samples required by the coefficient estimation algorithm.

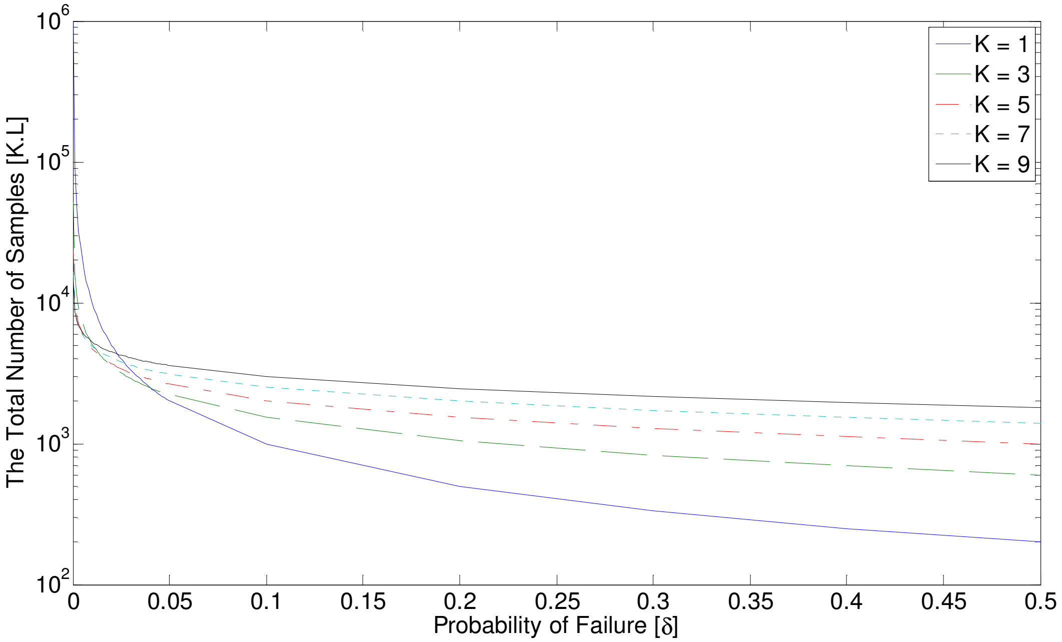

Figure 4.42. The total number of samples as a function of the probability of failures at different median window size and accuracy of factor of 0.1

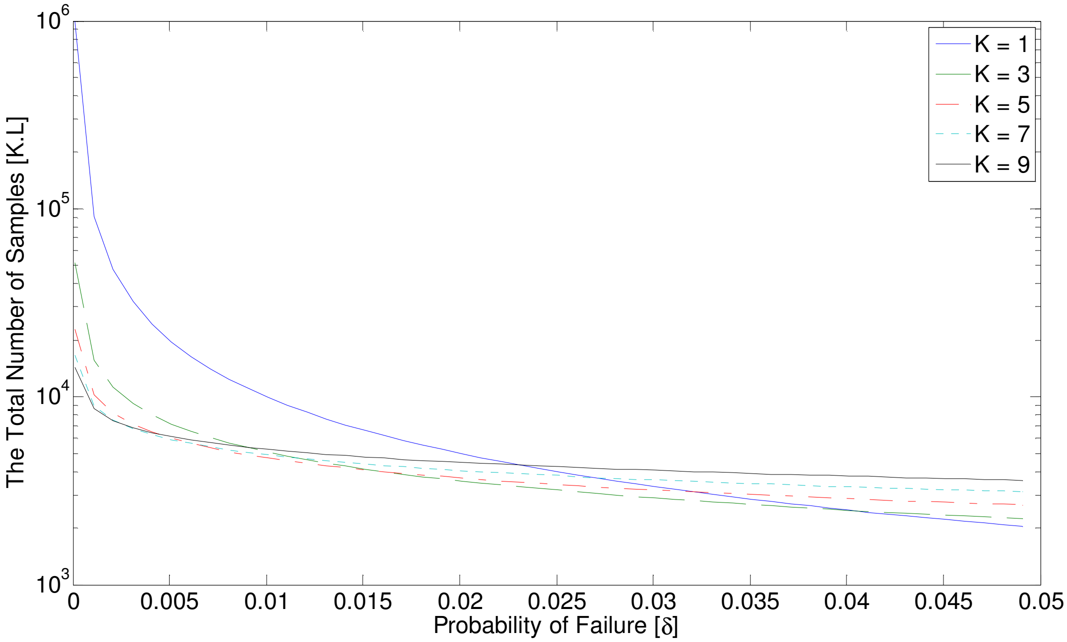

Figure 4.43. The total number of samples as a function of small probability of failures at different median window size and accuracy of factor of 0.1

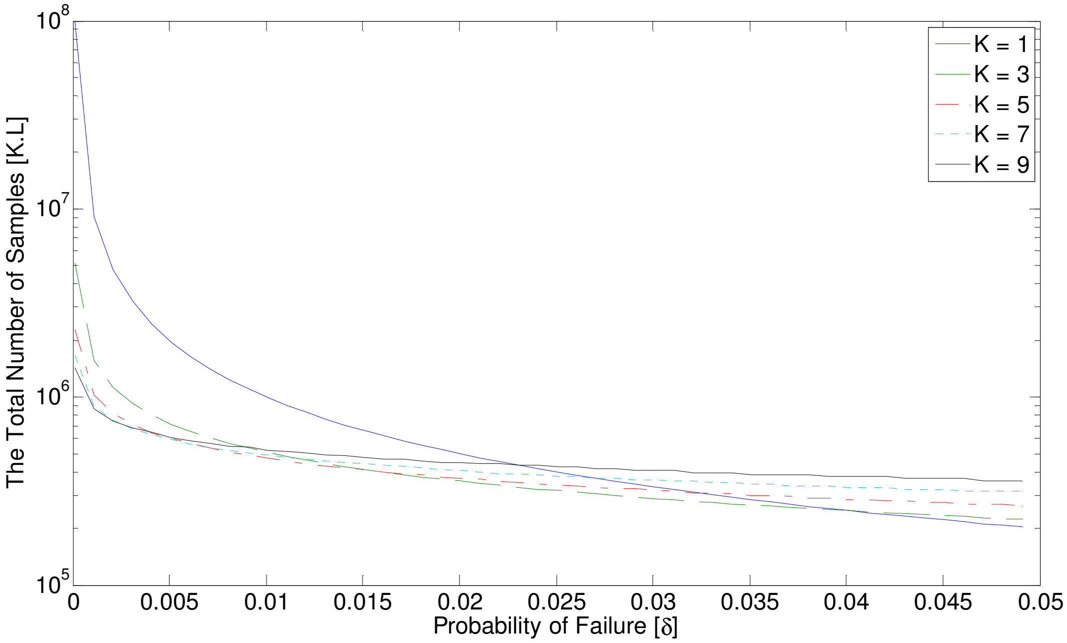

Figure 4.44. The total number of samples as a function of small probability of failures at different median window size and accuracy of factor of 0.01

### *4.5.5.2. Finding Optimal Values for Estimation Parameters L, K*

Section 4.5.5.1 found that the probability of failure for the coefficient estimation algorithm is

$$\delta \le \sum_{i=(K+1)/2}^{K} \binom{K}{i} (p_{fy})^{i} (1 - p_{fy})^{K-i} \qquad 4.64$$

The objective of this section is to find the values of *K* and *L* at which their product *K.L* is minimal under the constraints $\varepsilon, \delta$.

For small $p_y$ (e.g., $p_y$=*0.001*) and a small *K* (e.g., *K<9*), we can approximate the inequality 4.64 to be:

$$\delta \le p_{fy}^{\frac{K+1}{2}} (1 - p_{fy})^{\frac{K-1}{2}} \sum_{i=(K+1)/2}^{K} \binom{K}{i} \approx p_{fy}^{\frac{K+1}{2}} \sum_{i=(K+1)/2}^{K} \binom{K}{i} \quad : \quad (1 - p_{fy})^{\frac{K+1}{2}} \approx 1 \qquad 4.65$$

Assuming that $K$ is an odd number, the function $f(i)=\begin{pmatrix} K \\ i \end{pmatrix}$ is symmetric around the value *K/2,* as shown in the Figure 4.45. Also, since $\sum_{i=0}^{K}\begin{pmatrix} K \\ i \end{pmatrix} = 2^K$ , then $\sum_{i=(K+1)/2}^{K}\begin{pmatrix} K \\ i \end{pmatrix} = \frac{2^K}{2} = 2^{K-1}$

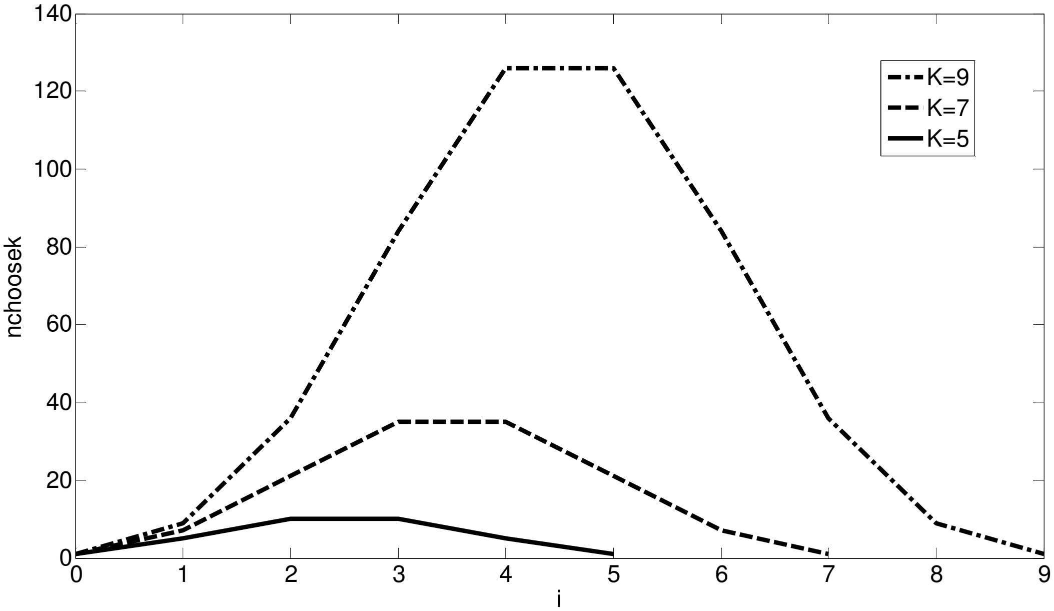

Figure 4.45. The combination (*K, i*) as a function of i at different values of *K*

Therefore,

$$\delta \leq p^{\frac{K+1}{2}} 2^{K-1} \qquad 4.66$$

After some mathematical manipulation, it is easy to find that,

$$K \geq \frac{\ln\left(\frac{4\delta^2}{p_y}\right)}{\ln(4p_y)} = \frac{\ln(4\delta^2\varepsilon^2 L)}{\ln\left(\frac{4}{\varepsilon^2 L}\right)} : p_y = \frac{1}{\varepsilon^2 L} \qquad 4.67$$

To minimize the total number of samples *K.L*:

$$\frac{\partial K.L}{\partial L} = \frac{\partial K}{\partial L} L + K = 0 \qquad 4.68$$

$$\frac{\partial K}{\partial L} = \frac{\partial}{\partial L}\left(\frac{\ln\left(4\delta^2\varepsilon^2 L\right)}{\ln\left(\frac{4}{4\varepsilon^2 L}\right)}\right) = \frac{\ln\left(\delta^2\right)}{L.\left[\ln\left(\frac{\varepsilon^2 L}{4}\right)\right]^2} \qquad 4.69$$

Substituting 4.69 in 4.68:

$$\frac{\ln\left(\delta^2\right)}{\left[\ln\left(\frac{\varepsilon^2 L}{4}\right)\right]^2} + \frac{\ln\left(4\delta^2\varepsilon^2 L\right)}{\ln\left(\frac{4}{\varepsilon^2 L}\right)} = 0 \qquad 4.70$$

$$\ln\left(\delta^2\right) - \ln\left(4\delta^2\varepsilon^2 L\right).\ln\left(\frac{\varepsilon^2 L}{4}\right) = 0 \qquad 4.71$$

$$\ln\left(\delta^2\right) - \ln\left((4\delta)^2\right).\ln\left(\frac{\varepsilon^2 L}{4}\right) + \left(\ln\left(\frac{\varepsilon^2 L}{4}\right)\right)^2 = 0 \qquad 4.72$$

Solving the quadratic equation for $\ln\left(\frac{\varepsilon^2 L}{4}\right)$:

$$\ln\left(\frac{\varepsilon^2 L}{4}\right) = \frac{-\ln\left((4\delta)^2\right) \pm \sqrt{\left(\ln\left((4\delta)^2\right)\right)^2 - \ln\left(\delta^2\right)}}{2} = \ln(4\delta) \pm \sqrt{\left(\ln(4\delta)\right)^2 - \ln\left(\delta^2\right)} \qquad 4.73$$

$$\ln(L) = \ln(4\delta) - \ln\left(\frac{\varepsilon^2}{4}\right) \pm \sqrt{\left(\ln(4\delta)\right)^2 - \ln\left(\delta^2\right)} \qquad 4.74$$

However, $\left(\ln(4\delta)\right)^2 >> \ln\left(\delta^2\right)$ for very small $\delta$, hence,

$$\ln(L) = \ln\left(\frac{4}{\varepsilon^2}\right) \qquad 4.75$$

$$\boxed{L = \frac{4}{\varepsilon^2} \sim \mathrm{O}\left(\frac{1}{\varepsilon^2}\right)} \qquad 4.76$$

And, hence, $p_{fy} = 1/4$ and $K$ can be derived from equation 4.77:

$$\delta \approx p_{fy}^{\frac{K+1}{2}}\left(1 - p_{fy}\right)^{\frac{K-1}{2}} 2^{K-1} \qquad 4.77$$

Thus,

$$K \approx \frac{\ln\left(\delta^2\right)}{\ln\left(4p_{fy}\left(1 - p_{fy}\right)\right)} + 1 \qquad 4.78$$

$$\boxed{K \approx 7\ln\left(\frac{1}{\delta}\right) + 1 \sim \mathrm{O}\left(\ln\left(\frac{1}{\delta}\right)\right)} \qquad 7.79$$

Therefore, for accuracy 0.1 and probability of failure of 0.1, the required number of samples are in the order of thousands. Note that the number of samples is calculated based on loose bounds. Nevertheless, the number of samples needed to achieve a reasonable performance in one estimation stage is relatively high. In practice, it was found that three steps, each using 10 samples per mean and five means per median for a total of 150 samples per coefficient determine the coefficient with accuracy $\varepsilon$ =1e-4 [130].

In the following section, considerable complexity reduction is achieved by recursive estimation of multiple iterations and relaxed accuracy and probability of failure constraints.

### *4.5.5.3. Iterative Estimation Method*

As discussed in the previous section, the required number of samples for a single-step coefficient estimation is relatively high to achieve an acceptable failure probability and accuracy factor (e.g. $\delta < 0.1$ and $\varepsilon < 0.1$). One technique employed to reduce the number of samples is to relax the accuracy and probability of failure constraints of the estimation procedure and iterate the estimation in a greedy fashion. With each iteration, the estimated coefficients found from the previous iteration are subtracted from the original signal spectrum and the coefficient estimation is conducted on the residual spectrum. The iterative coefficient estimation procedure ends after a fixed number of iterations or when the total energy of the residual signal spectrum is below a given threshold.

The following analysis finds the collective estimation accuracy and probability of failure of the iterative estimation procedure as a function of the relaxed accuracy and probability of failure constraints.

Let us denote

- $\varepsilon'$, $\delta'$ the relaxed accuracy and failure probability constraints for each estimation iteration. Note that these parameters are fixed for each estimation iteration.
- $\tilde{\hat{S}}_j(\omega)$ the estimated coefficient of frequency $\omega$ at iteration j.
- $R_n = \sum_{j=1}^{n} \tilde{\hat{S}}_j(\omega).\phi_\omega$ the estimated signal in time domain after n iteration.

In this context, the iterative estimation procedure is as follows. At each iteration, the coefficient estimation is conducted as described in section 2.4, under relaxed accuracy and failure probability constraints $\varepsilon'$, $\delta'$. The estimation found in this iteration is subtracted from the signal *S*. This subtraction is performed in the time domain. In the next iteration, the coefficient estimation is carried out on the residual signal after the subtraction. Hence, the accuracy of the subsequent estimation is no longer defined as a fraction of the total signal energy. Instead, the estimation accuracy in the next stage is a fraction of the energy of the residual signal. Mathematically, this can be expressed as follows:

$$\left|\hat{S}(\omega) - \tilde{\hat{S}}_n(\omega)\right|^2 \leq \varepsilon'^2 \left\|S - R_{n-1}\right\|^2 \qquad 4.80$$

By adding the noise floor amount $\|S\|^2 - \left|\hat{S}(\omega)\right|^2 = \eta\|S\|^2$ to both sides of the inequality, we get:

$$\|S - R_n\|^2 \le \|S\|^2 - \left|\hat{S}(\omega)\right|^2 + \varepsilon'^2\left\|S - \tilde{S}_{n-1}(\omega)\right\| = \eta\|S\|^2 + \varepsilon'^2\|S - R_{n-1}(\omega)\| \qquad 4.81$$

This inequality can be viewed as $X_n \le \eta\|S\|^2 + \varepsilon'^2 X_{n-1}$.

Note:

Solving a linear inhomogeneous recursion equation of the first order is as follows:

Let $x_n = ax_{n-1} + b_1$ for all $n \ge 1$ and $x_0 = b_0$

The solution is

$$x_n = \begin{cases} b_1 \dfrac{\left(1 - a^n\right)}{1 - a} + b_0 a^n & a \ne 1 \\ nb_1 + b_0 & a = 1 \end{cases}$$

Assuming that the initial state of $X$ (estimation error) is the total signal energy $R_0 = ||S||^2$, we can easily see that:

$$\|S - R_n\|^2 \le \eta\|S\|^2 \sum_{i=0}^{n-1}\left(\varepsilon'^2\right)^i + \varepsilon'^{2n}\|S\|^2 = \eta\|S\|^2 \frac{1 - \varepsilon'^{2n}}{1 - \varepsilon'^2} + \varepsilon'^{2n}\|S\|^2 \qquad 4.82$$

and by removing the noise floor $\eta\|S\|^2$, again from both sides of the inequality, we get:

$$\boxed{\left|\hat{S}(\omega) - \tilde{\hat{S}}_n(\omega)\right|^2 \le \eta\|S\|^2 \frac{\varepsilon'^2 - \varepsilon'^{2n}}{1 - \varepsilon'^2} + \varepsilon'^{2n}\|S\|^2 = \varepsilon^2\|S\|^2} \qquad 4.83$$

with success probability of $(1 - \delta')^n$.

In summary, the collective estimation accuracy $\varepsilon$ is

$$\varepsilon^2 = \eta \frac{\varepsilon'^2 - \varepsilon'^{2n}}{1 - \varepsilon'^2} + \varepsilon'^{2n} \qquad 4.84$$

Note that as $n \rightarrow \infty$, $\varepsilon^2 \rightarrow \eta \dfrac{\varepsilon'^2}{1 - \varepsilon'^2}$

and the collective success probability:

$$\boxed{1 - \delta = (1 - \delta')^n} \qquad 4.85$$

As $n \rightarrow \infty$, $1 - \delta \rightarrow 0$

In estimating q frequency coefficients simultaneously, then the estimation accuracy will be:

$$\boxed{q\varepsilon^2 = \eta \frac{q\varepsilon'^2 - q\varepsilon'^{2n}}{1 - q\varepsilon'^2} + \left(q\varepsilon'^2\right)^n} \qquad 4.86$$

and the probability of success:

$$\boxed{(1 - \delta)^q = (1 - \delta')^{qn}} \qquad 4.87$$

As the number of iteration increases, the effective estimation accuracy increases and the probability of success of the estimation algorithm decreases.

However, the gain in accuracy diminishes quickly following only a few iterations (three to five iterations) due to the fast convergence of the accuracy parameter to its steady state value $\eta \dfrac{\varepsilon'^2}{1 - \varepsilon'^2}$.

The number of iterations for the iterative estimation algorithm is decided based on two considerations. First, the total number of samples required by the iterative method is less than the those needed by the single-step method. Second, an appropriate trade off between the probability of failure and accuracy factor should be achieved.

### *4.5.6. Norm Estimation*

Norm estimation is a critical and challenging operation in the FFS algorithm, as it affects the performance of several FFS stages, including the isolation and group testing as well as the FFS convergence to a valid spectrum estimation result. The challenge is to provide norm estimations based on merely a few random signal samples reliable and accurate enough to ensure the correct algorithm operation. In the following analysis, the upper and lower norm bound is calculated for the norm estimation technique to estimate the norm of a noisy single frequency $S(n) = a\phi_\omega(n) + e(n)$ where $|a|^2 \geq \eta\|S\|^2$

1- Calculate the number of samples $r = floor\left(C.\ln\left(\frac{1}{\delta}\right)\right)$

2- Take r independent samples of *S: S(n₁), S(n₂), S(n₃), … , S(nᵣ)*

3- Return *NORM=N.*$\xi$ percentile of the Samples S($n_i$) where *i = 1, 2, 3, … , r*

Lower Bound

Based on the trigonometric inequality:

$$|S(n)| \geq |a\phi_\omega(n)| - |e(n)| \qquad 4.88$$

Since $\phi_\omega = \frac{1}{\sqrt{N}} e^{j\omega n}$

$$\sqrt{N}|e(n)| \geq |a| - \sqrt{N}|S(n)| \geq \sqrt{\eta}\|S\| - \sqrt{N}|S(n)| \qquad 4.89$$

Let us define a set of random time instants $\Omega = \left\{n : N|S(n)|^2 < \lambda\|S\|^2\right\}$. These time instants are those that if they are selected by the estimation algorithm at step three will return an estimation that is below the lower bound defined in this set.

Then,

$$\sqrt{N}|e(n)| \geq \sqrt{\eta}\|S\| - \sqrt{\lambda}\|S\| \qquad 4.90$$

since $|a|^2 \geq \eta\|S\|^2$, then $\|e\|^2 \leq (1-\eta)\|S\|^2$

it follows,

$$(1-\eta)\|S\|^2 \geq \|e(n)\|^2 \geq \sum_{n\in\Omega}|e(n)|^2 \geq |\Omega|.\frac{\left(\sqrt{\eta}-\sqrt{\lambda}\right)^2}{N}\|S\|^2 \text{ for all } n \in \Omega \qquad 4.91$$

It follows that:

$$|\Omega| \leq \frac{N(1-\eta)}{\left(\sqrt{\eta}-\sqrt{\lambda}\right)^2} \ , \qquad 4.92$$

Hence, the probability of a time instant sample $n \in \Omega$ becomes $\alpha = \frac{|\Omega|}{N}$

It should be noted that the parameter $\alpha = \frac{|\Omega|}{N} = \frac{(1-\eta)}{\left(\sqrt{\eta}-\sqrt{\lambda}\right)^2}$ depends on the lower bound level set by $\lambda$.

In order to calculate the probability that $\xi$ time samples belong to the set $\Omega$, consider now the random variable $\chi_\Omega$:

$$\chi_\Omega(n) = \begin{cases} 1 & n \in \Omega \\ 0 & n \notin \Omega \end{cases} \qquad 4.93$$

Based on the Chernoff's bound ( $P(X \geq a) \leq \min\left(e^{-za}E\left(e^{zX}\right)\right) : z > 0$ ), it follows that:

$$\Pr(Norm > \lambda\|S\|^2) = \Pr(\xi.r \text{ or more of the samples } \mathrm{n} \in \Omega)$$

$$= \Pr\left(\sum_{i=1}^{r}\chi_\Omega(n_i) > \zeta.r\right) \leq e^{-\xi.r.z}E\left(e^{z\sum_{i=1}^{r}\chi_\Omega(n_i)}\right) \qquad 4.94$$

but,

$$E\left(e^{z\chi\Omega}\right) = e^o \Pr(\chi_\Omega(i) = 0) + e^z \Pr(\chi_\Omega(i) = 1) = 1 - \alpha + \alpha e^z \qquad 4.95$$

and since $\chi_\Omega(n_1), \chi_\Omega(n_2),..., \chi_\Omega(n_r)$ are independent variables, then:

$$E\left(e^{\sum_{i=1}^{r} z\chi\Omega(n_i)}\right) = \prod_{i=1}^{r} E\left(e^{z\chi\Omega(n_i)}\right) = \left(1 - \alpha + \alpha e^{z}\right)^{r} \qquad 4.96$$

Therefore,

$$\Pr(Norm < \lambda\|S\|^{2}) \leq e^{-\xi.r.z}\left(1 - \alpha + \alpha e^{z}\right)^{r} = \left[(1 - \alpha)e^{-\xi..z} + \alpha e^{(1-\xi)z}\right]^{r} \qquad 4.97$$

Let us now find $z$ that minimizes $\left[(1 - \alpha)e^{-\xi..z} + \alpha e^{(1-\xi)z}\right]$:

$$\frac{d\left[(1 - \alpha)e^{-\xi..z} + \alpha e^{(1-\xi)z}\right]}{dz} = 0 \qquad 4.98$$

$$\rightarrow \xi(1 - \alpha)e^{-\xi..z} = (1 - \xi)\alpha e^{(1-\xi)z} \rightarrow e^{z} = \frac{\xi(1 - \alpha)}{\alpha(1 - \xi)} \qquad 4.99$$

$$\begin{aligned}\Pr(Norm < \lambda\|S\|^{2}) &\leq \left[(1 - \alpha)\left(\frac{\alpha(1 - \xi)}{\xi(1 - \alpha)}\right)^{\xi} + \alpha\left(\frac{\xi(1 - \alpha)}{\alpha(1 - \xi)}\right)^{(1-\xi)}\right]^{r} \\ &= \left[(c_1 + c_2)\alpha^{\xi}(1 - \alpha)^{1-\xi}\right]^{r}\end{aligned} \qquad 4.100$$

where: $c_1 = \left(\frac{1 - \xi}{\xi}\right)^{\xi}, c_2 = \left(\frac{\xi}{1 - \xi}\right)^{1-\xi}$

so

$$\boxed{\Pr(NORM < \lambda\|S\|^{2}) \leq \left[(c_1 + c_2)\alpha^{\xi}(1 - \alpha)^{1-\xi}\right]^{r} = \delta} \qquad 4.101$$

The number of samples r required by this norm estimation algorithm is derived from this probability formula.

$$r = \left[(c_1 + c_2)\alpha^{\xi}(1 - \alpha)^{1-\xi}\right]\ln(\delta) \qquad 4.102$$

Upper Bound

For the balance of this analysis, the probability of exceeding an upper bound $U\|S\|^{2}$ is investigated.

$$\Pr(NORM > U.\|S\|^2) \le \frac{N.E\left(|S(t_i)|^2\right)}{U\|S\|^2} = \frac{N.\frac{\|S\|^2}{N}}{U\|S\|^2} = \frac{1}{U} \qquad 4.103$$

Since NORM is the $\xi$ percentile of the sequence : *NS(n*$_1$*), NS(n*$_2$*), NS(n*$_3$*), … , NS(n*$_r$*)* ,

$$\boxed{\Pr(NORM > U.\|S\|^2) \le \sum_{j=(1-\xi).r}^{r} \binom{r}{j}\left(\frac{1}{U}\right)^{j}\left(1-\frac{1}{U}\right)^{r-j}} \qquad 4.104$$

# 5. Conclusion

This dissertation explores the design and implementation of cognitive radio on small form factor software radio platforms based on hybrid DSP/FPGA processing architecture. Configurability and spectrum sensing are fundamental aspects of this technology. The study undertaken was two fold. One investigated the efficient design and implementation of a configurable cognitive radio transceiver using hardware reuse and task partitioning techniques. The other examined the feasibility of using a fast, resource-efficient sensing technique, namely Fast Fourier Sampling (FFS), for spectrum sensing.

Wireless transceivers are comprised of several signal-processing blocks, including coding, modulation and synchronization, among others. The implementation of the configurable wireless transceivers on small form factor software radio is achieved by incorporating configurability in the signal processing blocks and using a mode-switching mechanism to facilitate switching between various communication modes. Because the limited capabilities of the platform impose restrictions on the configurable transceiver design, an efficient design framework that exploits the hardware resources and processing capabilities is required. This dissertation proposes a framework for building such a configurable transceiver on DSP/FPGA processing architectures. It details the design and implementation of two fundamental signal-processing blocks in wireless transceivers, namely modulation and synchronization. The transitioning between configurations is controlled by a mode-switching architecture that incorporates circular buffers to store processed samples during the mode-transitioning period.

The speed and accuracy of spectrum sensing techniques are important factors in cognitive radio network performance. As such, it is important to note that limitations imposed by computational complexity and shortened monitoring times impede the success of the spectrum sensing operation performed by cognitive radio nodes. Compressive sensing techniques are considered novel solutions for scalability problems in a number of signal processing operations. Fast Fourier Sampling is a compressive sensing technique that can reduce barriers to current spectrum sensing computational requirements. Its successful application results in faster sensing operations, less complex sensing modules, or wider spectrum sensing capabilities. This dissertation examines the feasibility of using Fast Fourier Sampling for spectrum sensing applications in a wireless microphone signal detection application. It explores the impact of various configuration parameters on the sensing performance and proposes a cooperative sensing scheme to expand FFS capabilities in spectrum sensing applications for cognitive radio networks. It also provides a comprehensive mathematical analysis that addresses the performance of each signal processing stage for an algorithm and their optimum configuration settings.

The analysis and design framework, developed in this dissertation, formulates a technical foundation for building cognitive radios on small form factor DSP/FPGA platforms. The designs presented serve as a research platform for various wireless transceiver design aspects, including the examination of cross-layer design between PHY and MAC layers with an aim to optimize wireless communication performance. The work presented herein focuses on narrowband wireless communications. Future research should extend the proposed design to include wideband spread spectrum techniques and the use of configurable filters with reloadable coefficients to support different pulse-shaping roll-off factors.

The design challenges for a mode-switching control call for a complex system that is beyond the scope of pure physical-layer implementation. The need for cross-layer design techniques to address problems that physical layers cannot overcome, including spectrum sensing and communication session negotiations, as well as channel condition evaluation will be important. Future work can apply a cognitive element that executes alternate operation management tasks, e.g. power management and spectrum sensing, to the proposed configurable radio design approach.

The FFS technique has both unique advantages and technical challenges that should be addressed in future work. These include:

- Parallelism. The semiconductor industry favors multi-core and multiprocessor architecture, making parallel algorithm research appealing. Some FFS stages can be parallelized by decomposing them into independently run processes, including a bit testing operation and a coefficient estimation for each identified frequency. In future work, we suggest to use a parallelized version of the code using OpenMP on multi-core with shared memory architecture.
- Parameterizable. Because the frequency identification section can be parameterized, a specific range of frequencies can be defined in bit testing so that spectrum sensing can be targeted for predefined sub-bands of interests rather than scanning for the entire spectrum range from zero to half the sampling frequency. This includes bands known to be previously occupied that are able to build upon adaptive FFS algorithms for scanning bands based on wireless activity.

The appealing features come at the cost of several technical challenges. These include

- Constructing a random sampling unit. One study proposed two implementations of the random sampling unit using either an array of data converter or analog registers [17]. The collected samples used in FFS are not dependent on the signal or algorithm progress. Hence, it’s possible to decide in advance , e.g. prior to algorithm execution, the sampling time instants. A study to determine the minimum time difference between samples is needed for a given channel condition (SNR) to estimate implementation cost. The time difference is expected to decrease for lesser SNR values to achieve an equivalent detection performance.
- The effect of limited bit size and quantization error propagation on algorithm stability and accuracy.

Useful extensions of this work could include studying the combination of two cooperative schemes, namely narrowband and wideband sensing schemes, in a hybrid cooperative sensing for large wireless cognitive network, and then analyzing the cooperative algorithm complexity in terms of number of samples used for spectrum estimation and the control channel bandwidth for information exchange in the cooperative sensing network.

## 6. References

[1] J.H.Reed, *Software Radio: A modern approach to radio engineering* Prentice Hall, 2002.

[2] FCC, "ET Docket No. 03-108: Facilitating Opportunities for Flexible, Efficient, and Reliable Spectrum Use Employing Cognitive Radio Technologies. URL: http://hraunfoss.fcc.gov/edocs_public/attachmatch/FCC-05-57A1.doc. ," 2005.

[3] J. Mitola, "Cognitive radio: An integrated agent architecture for software defined radio." vol. Doctor of Technology Stockholm, Sweden: Royal Institute of Technology (KTH), 2000.

[4] N. Hayes, "The JTRS SCA specification-the past, the present, and the future," in *Military Communications Conference, 2005. MILCOM 2005. IEEE*, 2005, pp. 2713-2719 Vol. 5.

[5] J. Mitola, "The software radio architecture," *Communications Magazine, IEEE,* vol. 33, pp. 26-38, 1995.

[6] J. R. MacLeod, T. Nesimoglu, M. A. Beach, and P. A. Warr, "Enabling technologies for software defined radio transceivers," in *MILCOM 2002. Proceedings*, 2002, pp. 354-358 vol.1.

[7] M. S. Safadi and D. L. Ndzi, "Digital Hardware Choices For Software Radio (SDR) Baseband Implementation," in *Information and Communication Technologies, 2006. ICTTA '06. 2nd*, 2006, pp. 2623-2628.

[8] W. H. W. Tuttlebee, *Software Defined Radio: Baseband Technologies for 3G Handsets and Basestations*, 1st ed.: Wiley, 2004.

[9] R. I. Lackey and D. W. Upmal, "Speakeasy: the military software radio," *Communications Magazine, IEEE,* vol. 33, pp. 56-61, 1995.

[10] B. Tarver, E. Christensen, and A. Miller, "The Wireless Information Transfer System (WITS) architecture for the Digital Modular Radio (DMR) software defined radio (SDR)," in *MILCOM 2000. 21st Century Military Communications Conference Proceedings*, 2000, pp. 226-230 vol.1.

[11] P. W. Fox, "Software defined radios-Motorola's Wireless Information Transfer System (WITS)," in *EUROCOMM 2000. Information Systems for Enhanced Public Safety and Security. IEEE/AFCEA*, 2000, pp. 43-46.

[12] D. L. Tennenhouse, T. Turletti, and V. G. Bose, "The SpectrumWare testbed for ATM-based software radios," in *Universal Personal Communications, 1996. Record., 1996 5th IEEE International Conference on*, 1996, pp. 915-917 vol.2.

[13] Website: www.gnu.org/software/gnuradio/.

[14] Website: www.lyrtech.com.

[15] Website: www.mathworks.com.

[16] Website: www.ti.com.

[17] Website: www.xilinx.com.

[18] T. Shono, Y. Shirato, H. Shiba, K. Uehara, K. Araki, and M. Umehira, "IEEE 802.11 wireless LAN implemented on software defined radio with hybrid programmable architecture," *Wireless Communications, IEEE Transactions on,* vol. 4, pp. 2299-2308, 2005.

[19] K. E. Mohamed, B. M. Ali, S. S. Jamuar, S. Khatun, and A. Ismail, "Implementation of CDMA transmitter for a multi-standard SDR base band platform," in *Communications, 2007. APCC 2007. Asia-Pacific Conference on*, 2007, pp. 303-306.

[20] J. Chapin, V. Lum, and S. Muir, "Experiences implementing GSM in RDL (the Vanu Radio Description Language)," in *Military Communications Conference, 2001. MILCOM 2001. Communications for Network-Centric Operations: Creating the Information Force. IEEE*, 2001, pp. 213-217 vol.1.

[21] J. Glossner, D. Iancu, L. Jin, E. Hokenek, and M. Moudgill, "A software-defined communications baseband design," *Communications Magazine, IEEE,* vol. 41, pp. 120-128, 2003.

[22] Y. Zhuan, J. Grosspietsch, and G. Memik, "An FPGA Based All-Digital Transmitter with Radio Frequency Output for Software Defined Radio," in *Design, Automation & Test in Europe Conference & Exhibition, 2007. DATE '07*, 2007, pp. 1-6.

[23] A. Di Stefano, G. Fiscelli, and C. G. Giaconia, "An FPGA-Based Software Defined Radio Platform for the 2.4GHz ISM Band," in *Research in Microelectronics and Electronics 2006, Ph. D.*, 2006, pp. 73-76.

[24] S. Abendroth, R. Heuze, and S. Weiss, "A Software defined radio front-end implementation," in *IEE DSPEnabled Radio Conference*, Livingston (Scotland), 2003.

[25] H. Harada, "A Software Defined Cognitive Radio Prototype," in *Personal, Indoor and Mobile Radio Communications, 2007. PIMRC 2007. IEEE 18th International Symposium on*, 2007, pp. 1-5.

[26] H. Harada, "Software defined radio prototype toward cognitive radio communication systems," in *New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005. 2005 First IEEE International Symposium on*, 2005, pp. 539-547.

[27] H. Harada, "A Feasibility Study on Software Defined Cognitive Radio Equipment," in *New Frontiers in Dynamic Spectrum Access Networks, 2008. DySPAN 2008. 3rd IEEE Symposium on*, 2008, pp. 1-12.

[28] L. Belanger, J. Ahern, and P. Fortier, "Prototyping wireless base stations or edge devices on a DSP/FPGA architecture using high-level tools," in *International Conference on Signal Processing Applications and Technology ICSPAT '00* Dallas, Tex, USA, 2000.

[29] L. Belanger, "Prototyping advanced 3G/4G wireless and SDR (Software Defined Radio) systems on the DSP/FPGA SignalMaster platform using a system-level approach," in *Global Signal Processing Conferences and Expos for the Industry (GSPx)*, 2003.

[30] A. Kwan, S. Boumaiza, M. Smith, and F. Ghannouchi, "Automating the Verification of SDR Base band Signal Processing Algorithms Developed on DSP/FPGA Platform," in *Signal Processing Systems Design and Implementation, 2006. SIPS '06. IEEE Workshop on*, 2006, pp. 5-9.

[31] J.-P. Delahaye, G. Gogniat, C. Roland, and P. Bomel, "Software Radio and Dynamic Reconfiguration on a DSP/FPGA platform," *Frequenz Journal of Telecommunications,* pp. 152-159, 5-6/2004 2004.

[32] R. Pfeifer and C. Scheier, *Understanding Intelligence*. Cambridge,MA: MIT Press, 1999.

[33] S. Haykin, "Cognitive radio: brain-empowered wireless communications," *Selected Areas in Communications, IEEE Journal on,* vol. 23, pp. 201-220, 2005.

[34] R. D. Hinman, "Application of Cognitive Radio Technology to Legacy Military Waveforms in a JTRS (Joint Tactical Radio System) Radio," in *Military Communications Conference, 2006. MILCOM 2006. IEEE*, 2006, pp. 1-5.

[35] Website: http://e2r2.motlabs.com.

[36] C. Doerr, D. C. Sicker, and D. Grunwald, "Experiences Implementing Cognitive Radio Control Algorithms," in *Global Telecommunications Conference, 2007. GLOBECOM '07. IEEE*, 2007, pp. 4045-4050.

[37] H. Arslan, *Cognitive Radio, Software Defined Radio, and Adaptive Wireless Systems* vol. XVIII: Springer, 2007.

[38] C. Bostian and J. H. Reed, "Understanding the Issues in Software Defined Cognitive Radio," in *tutorial presented at IEEE DySPAN*, Baltimore, MD, 2005.

[39] C. Cordeiro, K. Challapali, D. Birru, and N. Sai Shankar, "IEEE 802.22: the first worldwide wireless standard based on cognitive radios," in *New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005. 2005 First IEEE International Symposium on*, 2005, pp. 328-337.

[40] K. Challapali, "Spectrum Agile Radios: Real-time Measurements," in *Cognitive Radio Conference* Washington DC, 2004.

[41] A. Sahai, N. Hoven, and R. Tandra, "Some fundamental limits on cognitive radio," in *42nd Allerton Conference on Communication, Control, and Computing*, 2004.

[42] FCC, "Revision of Parts 2 and 15 of the Commissions Rules to Permit Unlicensed National Information Infrastructure (U-NII) Devices in the 5GHz Band," 18 November 2003.

[43] S. J.Shellhammer, "spectrum sensing in IEEE 802.22," in *1st IAPR workshop on Cognitive Information Processing CIP* Greece 2008.

[44] S. Haykin, D. J. Thomson, and J. H. Reed, "Spectrum Sensing for Cognitive Radio," *Proceedings of the IEEE,* vol. 97, pp. 849-877, 2009.

[45] I. F. Akyildiz, W.-Y. Lee, M. C. Vuran, and S. Mohanty, "NeXt generation/dynamic spectrum access/cognitive radio wireless networks: A survey," *Computer Networks,* vol. 50, pp. 2127-2159, 2006.

[46] H. Arslan and T. Yücek, "Spectrum Sensing for Cognitive Radio Applications," in *Cognitive Radio, Software Defined Radio, and Adaptive Wireless Systems* Netherland: Springer, 2007, pp. 263- 289.

[47] M. Höyhtyä, A. Hekkala, M. Katz, and A. Mämmelä, "Spectrum Awareness: Techniques and Challenges for Active Spectrum Sensing," in *Cognitive Wireless Networks* Netherlands Springer, 2007, pp. 353- 372.

[48] Y. Zeng, Y. C. Liang, A. T. Hoang, and R. Zhang, "A Review on Spectrum Sensing for Cognitive Radio: Challenges and Solutions," *EURASIP Journal on Advances in Signal Processing,* vol. 2010, 2010.

[49] H. Tang, "Some physical layer issues of wide-band cognitive radio systems," in *New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005. 2005 First IEEE International Symposium on*, 2005, pp. 151-159.

[50] A. Sahai, R. Tandra, S. M. Mishra, and N. Hoven, "Fundamental design tradeoffs in cognitive radio systems," in *Proceedings of the first international workshop on Technology and policy for accessing spectrum* Boston, Massachusetts: ACM, 2006.

[51] S. Geirhofer, T. Lang, and B. M. Sadler, "A Measurement-Based Model for Dynamic Spectrum Access in WLAN Channels," in *Military Communications Conference, 2006. MILCOM 2006. IEEE*, 2006, pp. 1-7.

[52] N. S. Shankar, C. Cordeiro, and K. Challapali, "Spectrum agile radios: utilization and sensing architectures," in *New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005. 2005 First IEEE International Symposium on*, 2005, pp. 160-169.

[53] F. F. Digham, M. S. Alouini, and M. K. Simon, "On the energy detection of unknown signals over fading channels," in *Communications, 2003. ICC '03. IEEE International Conference on*, 2003, pp. 3575-3579 vol.5.

[54] M. P. Wylie-Green, "Dynamic spectrum sensing by multiband OFDM radio for interference mitigation," in *New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005. 2005 First IEEE International Symposium on*, 2005, pp. 619-625.

[55] S. D. Jones, N. Merheb, and I. J. Wang, "An experiment for sensing-based opportunistic spectrum access in CSMA/CA networks," in *New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005. 2005 First IEEE International Symposium on*, 2005, pp. 593-596.

[56] P. Papadimitratos, S. Sankaranarayanan, and A. Mishra, "A bandwidth sharing approach to improve licensed spectrum utilization," *Communications Magazine, IEEE,* vol. 43, pp. supl.10-supl.14, 2005.

[57] A. E. Leu, K. Steadman, M. McHenry, and J. Bates, "Ultra sensitive TV detector measurements," in *New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005. 2005 First IEEE International Symposium on*, 2005, pp. 30-36.

[58] T. Yucek and H. Arslan, "Spectrum Characterization for Opportunistic Cognitive Radio Systems," in *Military Communications Conference, 2006. MILCOM 2006. IEEE*, 2006, pp. 1-6.

[59] A. E. Leu, M. McHenry, and B. L. Mark, "Modeling and analysis of interference in listen-before-talk spectrum access schemes," *Int. J. Netw. Manag.,* vol. 16, pp. 131-147, 2006.

[60] G. Ganesan and L. Ye, "Agility improvement through cooperative diversity in cognitive radio," in *Global Telecommunications Conference, 2005. GLOBECOM '05. IEEE*, 2005, pp. 5 pp.-2509.

[61] G. Ganesan and Y. Li, "Cooperative spectrum sensing in cognitive radio networks," in *New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005. 2005 First IEEE International Symposium on*, 2005, pp. 137-143.

[62] N. Sai Shankar, "Overview of Blind Sensing Techniques Considered in IEEE 802.22 WRANs," in *Sensor, Mesh and Ad Hoc Communications and Networks Workshops, 2008. SECON Workshops '08. 5th IEEE Annual Communications Society Conference on*, 2008, pp. 1-4.

[63] Z. Tian and G. B. Giannakis, "A Wavelet Approach to Wideband Spectrum Sensing for Cognitive Radios," in *Cognitive Radio Oriented Wireless Networks and Communications, 2006. 1st International Conference on*, 2006, pp. 1-5.

[64] Y. Hur, J. Park, W. Woo, K. Lim, C. H. Lee, H. S. Kim, and J. Laskar, "A wideband analog multi-resolution spectrum sensing (MRSS) technique for cognitive radio (CR) systems," in *Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium on*, 2006, p. 4 pp.

[65] J. Park, Y. Hur, T. J. Song, K. Kim, J. Lee, K. Lim, C. H. Lee, H. S. Kim, and J. Laskar, "Implementation Issues of A Wideband Multi-Resolution Spectrum Sensing (MRSS) Technique for Cognitlve Radio (CR) Systems," in *Cognitive Radio Oriented Wireless Networks and Communications, 2006. 1st International Conference on*, 2006, pp. 1-5.

[66] S. M. Kay, *Fundamentals of statistical signal processing: Detection theory* vol. Printice Hall PTR, 1998.

[67] R. Tandra and A. Sahai, "Fundamental limits on detection in low SNR under noise uncertainty," in *Wireless Networks, Communications and Mobile Computing, 2005 International Conference on*, 2005, pp. 464-469 vol.1.

[68] ATSC, "*ATSC Digital Television Standard (A/53) Revision E, with Amendments No. 1 and 2*," September 2006.

[69] J. M. Mendel, "Tutorial on higher-order statistics (spectra) in signal processing and system theory: theoretical results and some applications," *Proceedings of the IEEE,* vol. 79, pp. 278-305, 1991.

[70] C. Hou-Shin, G. Wen, and D. G. Daut, "Spectrum Sensing Using Cyclostationary Properties and Application to IEEE 802.22 WRAN," in *Global Telecommunications Conference, 2007. GLOBECOM '07. IEEE*, 2007, pp. 3133-3138.

[71] Z. Ye, J. Grosspietsch, and G. Memik, "Spectrum Sensing Using Cyclostationary Spectrum Density for Cognitive Radios," in *Signal Processing Systems, 2007 IEEE Workshop on*, 2007, pp. 1-6.

[72] H. Ning, S. SungHwan, C. Jae Hak, and K. Jae Moung, "Spectral correlation based signal detection method for spectrum sensing in IEEE 802.22 WRAN systems," in *Advanced Communication Technology, 2006. ICACT 2006. The 8th International Conference*, 2006, pp. 6 pp.-1770.

[73] M. Oner and F. Jondral, "Cyclostationarity based air interface recognition for software radio systems," in *Radio and Wireless Conference, 2004 IEEE*, 2004, pp. 263-266.

[74] M. Ghozzi, F. Marx, M. Dohler, and J. Palicot, "Cyclostatilonarilty-Based Test for Detection of Vacant Frequency Bands," in *Cognitive Radio Oriented Wireless Networks and Communications, 2006. 1st International Conference on*, 2006, pp. 1-5.

[75] A. Fehske, J. Gaeddert, and J. H. Reed, "A new approach to signal classification using spectral correlation and neural networks," in *New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005. 2005 First IEEE International Symposium on*, 2005, pp. 144-150.

[76] S. Shellhammer, V. Tawil, G. Chouinard, M. Muterspaugh, and M. Ghosh, "Spectrum Sensing Simulation Model," September 2006.

[77] C. Cordeiro, K. Challapali, and M. Ghosh, "Cognitive PHY and MAC layers for dynamic spectrum access and sharing of TV bands," in *Proceedings of the first international workshop on Technology and policy for accessing spectrum* Boston, Massachusetts: ACM, 2006.

[78] Y. Hur, J. Park, K. Kim, J. Lee, K. Lim, C. H. Lee, H. S. Kim, and J. Laskar, "A Cognitive Radio (CR) Testbed System Employing a Wideband Multi-Resolution Spectrum Sensing (MRSS) Technique," in *Vehicular Technology Conference, 2006. VTC-2006 Fall. 2006 IEEE 64th*, 2006, pp. 1-5.

[79] D. Taubenheim, W. Chiou, N. Correal, P. Gorday, S. Kyperountas, S. Machan, P. Minh, S. Qicai, E. Callaway, and R. Rachwalski, "Implementing an Experimental Cognitive Radio System for DySPAN," in *Global Telecommunications Conference, 2007. GLOBECOM '07. IEEE*, 2007, pp. 4040-4044.

[80] D. Cabric, S. M. Mishra, and R. W. Brodersen, "Implementation issues in spectrum sensing for cognitive radios," in *Signals, Systems and Computers, 2004. Conference Record of the Thirty-Eighth Asilomar Conference on*, 2004, pp. 772-776 Vol.1.

[81] D. Cabric, A. Tkachenko, and B. W. Brodersen, "Experimental Study of Spectrum Sensing Based on Energy Detection and Netwok Cooperation," in *The 2nd Annual International Wireless Internet Conference (WICON)* TAPAS Workshop, 2006.

[82] A. C. Gilbert, S. Guha, P. Indyk, S. Muthukrishnan, and M. Strauss, "Near-optimal sparse fourier representations via sampling," in *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing* Montreal, Quebec, Canada: ACM, 2002.

[83] A. C. Gilbert, S. Muthukrishnan, and M. Strauss, *Improved time bounds for near-optimal sparse fourier representations* vol. 5914. Bellingham, WA, ETATS-UNIS: Society of Photo-Optical Instrumentation Engineers, 2005.

[84] A. C. Gilbert, M. J. Strauss, and J. A. Tropp, "A Tutorial on Fast Fourier Sampling," *Signal Processing Magazine, IEEE,* vol. 25, pp. 57-66, 2008.

[85] J. Crocket, "DSP Architectures for Wireless Communications," in *International Symposium on Advanced Radio Technologies*, 1998.

[86] O. Agazzi, T. Chin-Pyng, D. Messerschmitt, and D. Hodges, "Timing Recovery in Digital Subscriber Loops," *Communications, IEEE Transactions on,* vol. 33, pp. 558-569, 1985.

[87] Website: www.rfdesign.com.

[88] M. Grecki, T. Jezynski, and A. Brandt, "Estimation of IQ Vector Components of RF Field - Theory and Implementation," in *12th Int. Conf. Mixed Design of Integrated Circuits and Systems*, Cracow, Poland, 2005, pp. 783-788.

[89] J. G. Proakis, *Digital Communications*, 4th ed. New York: McGraw-Hill, 2000.

[90] C. Dick, B. Egg, and F. Harris, "Architecture and simulation of timing synchronization circuits for the FPGA implementation of narrowband waveforms," in *the SDR 06 technical conference and product exposition*, 2006.

[91] J. Zhang, N. Wu, J. Kuang, and H. Wang, "High speed all digital symbol timing recovery based on FPGA," in *Wireless Communications, Networking and Mobile Computing, 2005. Proceedings. 2005 International Conference on*, 2005, pp. 1402-1405.

[92] Z. Yang, C. Yun, W. Jianming, Z. Xiaoyang, and Z. Dian, "Optimization of symbol timing recovery for DTMB receivers," in *ASIC, 2007. ASICON '07. 7th International Conference on*, 2007, pp. 822-825.

[93] Z. Minjian, H. Aiping, Z. Zhaoyang, and Q. Peiliang, "All digital tracking loop for OFDM symbol timing," in *Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th*, 2003, pp. 2435-2439 Vol.4.

[94] Z. Yongxue, F. Haidong, Y. Lixin, and X. Yangzhen, "Practical implementation of blind equalization, carrier recovery and timing recovery for QAM cable receiver chip," in *ASIC, 2003. Proceedings. 5th International Conference on*, 2003, pp. 886-889 Vol.2.

[95] R. De Gaudenzi and V. Vanghi, "All-digital carrier phase and clock timing recovery for 8PSK," in *Global Telecommunications Conference, 1991. GLOBECOM '91. 'Countdown to the New Millennium. Featuring a Mini-Theme on: Personal Communications Services*, 1991, pp. 375-379 vol.1.

[96] Z. Wu, Z. Yin, G. Ren, and S. Yang, "A research on adaptive PCM clock recovering," in *Microwave and Millimeter Wave Technology, 2002. Proceedings. ICMMT 2002. 2002 3rd International Conference on*, 2002, pp. 406-409.

[97] P. Zicari, P. Corsonello, and S. Perri, "An Efficient Bit-Detection and Timing Recovery Circuit for FPGAs," in *Electronics, Circuits and Systems, 2006. ICECS '06. 13th IEEE International Conference on*, 2006, pp. 168-171.

[98] J. Noseworthy and J. Kulp, "Standard Interfaces for FPGA Components," in *Military Communications Conference, 2007. MILCOM 2007. IEEE*, 2007, pp. 1-5.

[99] K. Skey, J. Bradley, and K. Wagner, "A Reuse Approach for FPGA-Based SDR Waveforms," in *Military Communications Conference, 2006. MILCOM 2006. IEEE*, 2006, pp. 1-7.

[100] D. Scaperoth, B. Le, T. Rondeau, D. Maldonado, C. W. Bostian, and S. Harrison, "Cognitive Radio Platform Development for Interoperability," in *Military Communications Conference, 2006. MILCOM 2006. IEEE*, 2006, pp. 1-6.

[101] T. Hentschel, T. Hentschel, M. Henker, and G. Fettweis, "The digital front-end of software radio terminals," *Personal Communications, IEEE [see also IEEE Wireless Communications],* vol. 6, pp. 40-46, 1999.

[102] P. Mackenzie, L. E. Doyle, K. E. Nolan, and D. O'Mahony, "An Architecture for the Development of Software Radios on General Purpose Processors," in *the ISSC Irish Signals and Systems Conference*, University College Cork and Cork Institute of Technology, Cork, Rep. Of Ireland, 2002, pp. 275-280.

[103] J. Lotze, S. A. Fahmy, J. Noguera, L. Doyle, and R. Esser, "An FPGA-based cognitive radio framework," in *Signals and Systems Conference, 208. (ISSC 2008). IET Irish*, 2008, pp. 138-143.

[104] L. Jun-Seo, P. Jong-Hyun, K. Sang-Woo, L. YingShan, and R. Heung-Gyoon, "Implementation of DSP-based digital receiver for the SDR application," in *Communications, 2004 and the 5th International Symposium on Multi-Dimensional Mobile Communications Proceedings. The 2004 Joint Conference of the 10th Asia-Pacific Conference on*, 2004, pp. 6-10 vol.1.

[105] S. Watanabe, Y. Kunisawa, D. Kamisaka, T. Inoue, and Y. Takeuchi, "A Software Radio Implementation of CDMA2000 1xEV-DO on a Single DSP Chip Designed for Mobile Handset Terminal," in *Vehicular Technology Conference, 2006. VTC-2006 Fall. 2006 IEEE 64th*, 2006, pp. 1-5.

[106] J. Bertrand, J. W. Cruz, B. Majkrzak, and T. Rossano, "CORBA delays in a software-defined radio," *Communications Magazine, IEEE,* vol. 40, pp. 152-155, 2002.

[107] P. Ekas and B. Jentz, "Developing and Integrating FPGA Coprocessors," in *Embedded Computing Design*, 2003.

[108] M. Brogioli, P. Radosavljevic, and J. R. Cavallaro, "A General Hardware/Software Co-design Methodology for Embedded Signal Processing and Multimedia Workloads," in *Signals, Systems and Computers, 2006. ACSSC '06. Fortieth Asilomar Conference on*, 2006, pp. 1486-1490.

[109] S. Rajagopal, B. A. Jones, and J. R. Cavallaro, "Task Partitioning Wireless Base-station Receiver Algorithms on Multiple DSPs and FPGAs," in *International Conference on Signal Processing Applications and Technology (ICSPAT)*, 2000.

[110] Website: http://cas.web.cern.ch/cas/Sweden-2007/Lectures/Sigtuna-lectures.htm.

[111] Website: http://en.wikipedia.org/wiki/Error_correcting_code.

[112] D. L. Donoho, "Compressed sensing," *Information Theory, IEEE Transactions on,* vol. 52, pp. 1289-1306, 2006.

[113] Y. L. Polo, W. Ying, A. Pandharipande, and G. Leus, "Compressive wide-band spectrum sensing," in *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, 2009, pp. 2337-2340.

[114] T. Zhi and G. B. Giannakis, "Compressed Sensing for Wideband Cognitive Radios," in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, 2007, pp. IV-1357-IV-1360.

[115] Y. Zhuizhuan and S. Hoyos, "Compressive spectrum sensing front-ends for cognitive radios," in *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*, 2009, pp. 1899-1904.

[116] S. J. Shellhammer, A. K. Sadek, and Z. Wenyi, "Technical challenges for cognitive radio in the TV white space spectrum," in *Information Theory and Applications Workshop, 2009*, 2009, pp. 323-333.

[117] M. Ghosh, V. Gaddam, G. Turkenich, and K. Challapali, "Spectrum Sensing Prototype for Sensing ATSC and Wireless Microphone Signals," in *Cognitive Radio Oriented Wireless Networks and Communications, 2008. CrownCom 2008. 3rd International Conference on*, 2008, pp. 1-7.

[118] C. Clanton, M. Kenkel, and Y. Tang, "Wireless Microphone Signal Simulation Model," IEEE 802.22March 2007.

[119] M. A. Iwen, A. Gilbert, and M. Strauss, "Empirical evaluation of a sub-linear time sparse DFT algorithm " *Communications in Mathematical Science,* vol. 5, pp. 981-998, 2007.

[120] C. Komninakis, "A fast and accurate Rayleigh fading simulator," in *Global Telecommunications Conference, 2003. GLOBECOM '03. IEEE*, 2003, pp. 3306-3310 vol.6.

[121] S. M. Mishra, A. Sahai, and R. W. Brodersen, "Cooperative Sensing among Cognitive Radios," in *Communications, 2006. ICC '06. IEEE International Conference on*, 2006, pp. 1658-1663.

[122] C. Hou-Shin, G. Wen, and D. G. Daut, "Spectrum Sensing for Wireless Microphone Signals," in *Sensor, Mesh and Ad Hoc Communications and Networks Workshops, 2008. SECON Workshops '08. 5th IEEE Annual Communications Society Conference on*, 2008, pp. 1-5.

[123] L. Sunmin, K. Sangwon, P. Changhyun, and S. Myungsun, "The detection and classification of the Wireless Microphone signal in the IEEE 802.22 WRAN system," in *Microwave Conference, 2007. APMC 2007. Asia-Pacific*, 2007, pp. 1-4.

[124] Q. Zhi, S. J. Shellhammer, Z. Wenyi, and A. H. Sayed, "Spectrum Sensing by Cognitive Radios at Very Low SNR," in *Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE*, 2009, pp. 1-6.

[125] T. Yucek and H. Arslan, "A survey of spectrum sensing algorithms for cognitive radio applications," *Communications Surveys & Tutorials, IEEE,* vol. 11, pp. 116-130, 2009.

[126] Q. Zhi, C. Shuguang, H. Poor, and A. Sayed, "Collaborative wideband sensing for cognitive radios," *Signal Processing Magazine, IEEE,* vol. 25, pp. 60-73, 2008.

[127] P. K. Varshney, *Distributed Detection and Data Fusion*. New York: Springer-Verlag, 1997.

[128] C. R. C. da Silva, C. Brian, and K. Kyouwoong, "Distributed Spectrum Sensing for Cognitive Radio Systems," in *Information Theory and Applications Workshop, 2007*, 2007, pp. 120-123.

[129] T. Ragheb, S. Kirolos, J. Laska, A. Gilbert, M. Strauss, R. Baraniuk, and Y. Massoud, "Implementation models for analog-to-information conversion via random sampling," in *Circuits and Systems, 2007. MWSCAS 2007. 50th Midwest Symposium on*, 2007, pp. 325-328.

[130] J. Zou, A. Gilbert, M. Strauss, and I. Daubechies, "Theoretical and experimental analysis of a randomized algorithm for Sparse Fourier transform analysis," *J. Comput. Phys.,* vol. 211, pp. 572-595, 2006.

[131] website: http://www.mathpages.com/home/kmath199.htm.

[132] A. H. Sayed, *Adaptive Filters*: Wiley-IEEE Press, 2008.

[133] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions (with Formulas, Graphs, and Mathematical Tables*: Dover Publications, 1972.

# Appendix A. Lyrtech Small Form Factor Software Defined Radio Platform

The development platform is a modular small form factor (SFF) SDR platform produced by Lyrtech. Figure A. 1 shows a simplified hardware block diagram of the SDR platform, which consists of three modules: digital signal processing module, data conversion module and RF module.

## Digital Signal Processing Module

The cognitive radio design is implemented in the digital signal processing module. The other two modules are configurable through a number of control signals generated from the digital signal processing module.

The digital signal processing module uses a Virtex-4 SX35 FPGA and DM6446 DMP. The DMP chip features an advanced Very Long Instruction Word (VLIW) DSP portion, which is responsible for a selected number of signal processing tasks, and a Reduced Instruction Set Computer (RISC) ARM9 core, which is used to run a real-time operating system (RTOS).

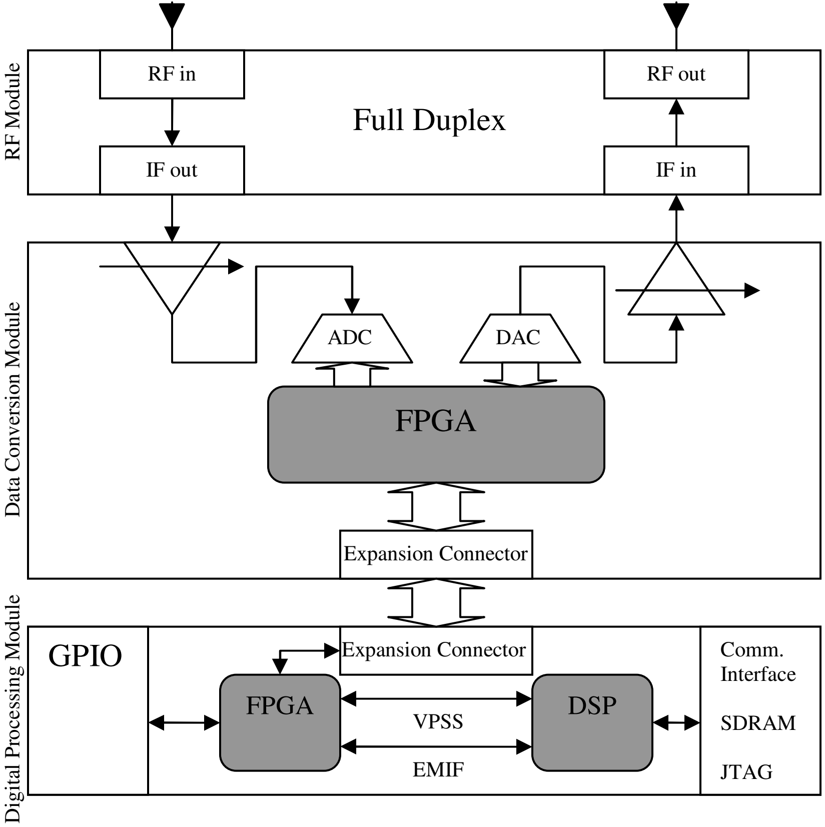

Figure A. 1 Hardware block diagram of the SDR platform tasks in the transceiver design.

## Data Conversion Module

The data conversion module is equipped with a 14-bit dual channel ADC with sampling rate up to 125 MSPS and a 16-bit dual channel interpolating DAC with a sampling rate up to 500 MSPS. It is also equipped with programmable gains at the input and output of the ADC and DAC, respectively. This enables the implementation of automatic gain control AGC and transmission power control. The signal digitization at the receiver side is performed at fixed IF level equal to 30 MHz.

## RF Module

The RF module is comprised of a super-heterodyne receiver and direct quadrature transmitter. The transceiver is full duplex with transmission frequency range of 200-930 MHz and receiving frequency range of 30-928 MHz. The receiver is configured to have either a 5 or 20 MHz bandwidth through a set of pass-band filters.

## DSP/FPGA Interface

Interfacing between DSP and FPGA is achieved using different connections. The main bridge between the two processing engines is the Video Processing Sub-system (VPSS) data port, i.e. a DM6446 DSP 16-bit synchronous video data transfer port. This connection is used for high speed data exchange between DSP and FPGA. The VPSS is comprised of the video processing front end (VPFE) and the video processing back end (VPBE), where the VPFE is used as an input interface to the DSP and the VPBE as an output interface from the DSP (See Figure A.2). The VPSS is adapted to be used on the digital processing module of the SDR platform as an interface to transfer bulk data other than video between DSP and FPGA. In order to emulate video signals, Vsync and Hsync signals are generated by the VPFE of FPGA interface. The FPGA VPBE uses the Vsync and Hsync signals generated by the DSP to synchronize the incoming data transfer.

The VPSS blocks in FPGA are clocked at 75 MHz. The clock signal is processed by the Output Dual Data Rate register (ODDR) before passing the clock signal to the DSP SoC. The maximum data rate between DSP and FPGA that can be achieved using VPSS connection is:
FPGA to DSP : 150 MBps (16 bits, 75 MHz).
DSP to FPGA: 75 MBps (16 bits, 37.5 MHz).

As can be noted from Figure A.2, the interface between the custom logic in the FPGA and the VPSS bus is through a First Input First Output (FIFO) buffers that support up to 512 32-bit samples.

Another interfacing method is achieved by FPGA custom registers using External Memory Interface (EMIF). These registers are shared memory blocks of eight 32-bit words between DSP and FPGA On-Chip Peripheral Bus (OPB) as shown in Figure A.2. The EMIF connection is used for asynchronous low data rate communication between DSP and FPGA. In the setup shown in Figure A.2, the EMIF connection is used to access the custom registers in FPGA which are used as control flags or status registers.

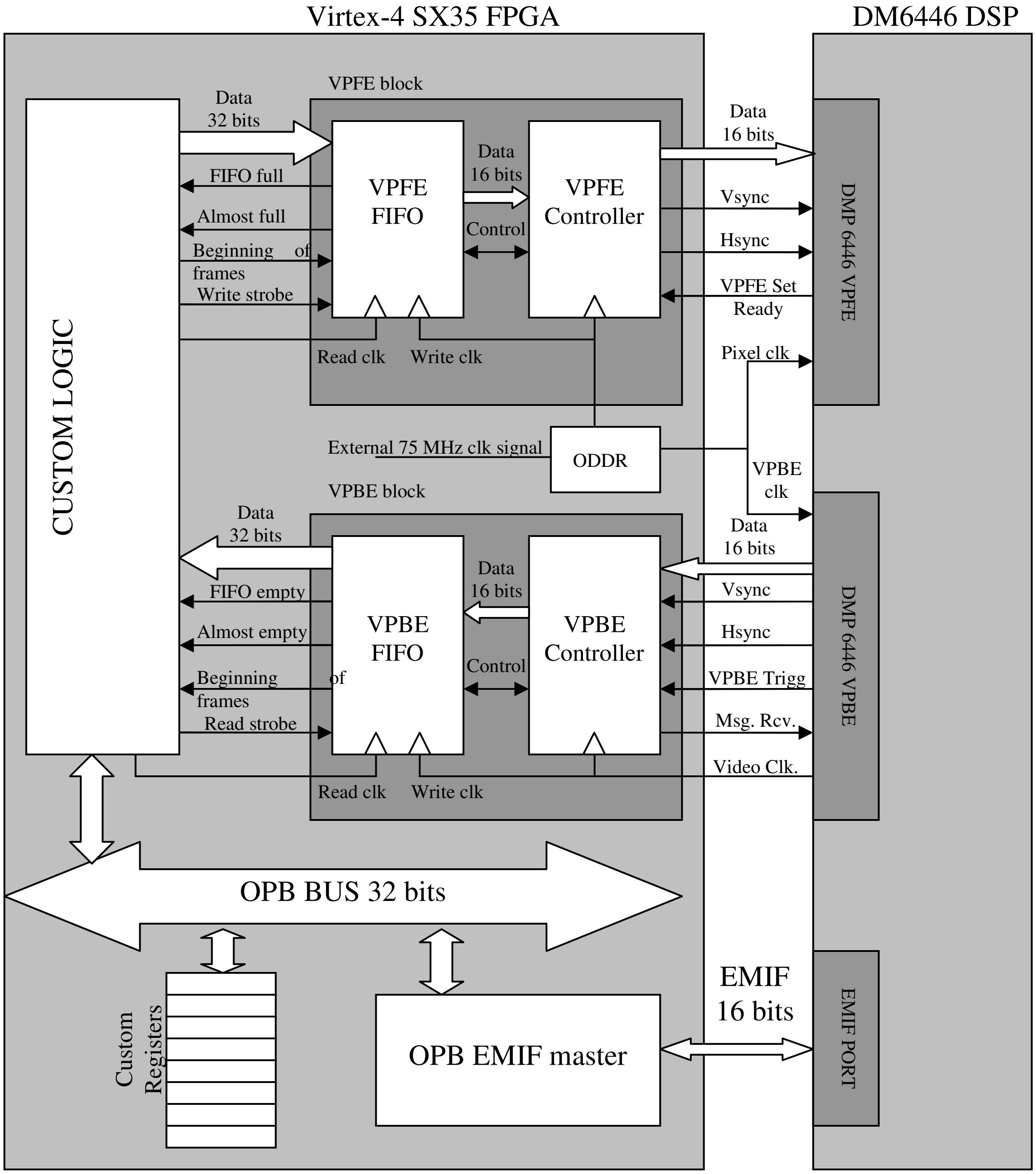

Figure A.2 Communication interfaces between the DM6446 DSP and the Virtex-4 SX35 FPGA