AMShare: Distributed Frequency Synchronization of Small Cell Base Transceiver Stations Utilizing Commercial AM Wireless Signals

by

Milad Mazi Esfahani

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2017

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Exponential growth of connected users in recent years and the increasing demands for ubiquitous coverage of wireless services has pushed the mobile operators to invest in innovative technologies that can support more connected users with higher quality of service. The idea of overlay network in cellular networks by adding small cells, has been proposed in recent years to improve network capacity and address the varying demands of mobile users in different areas. In deploying small cells as an overlay network, lots of technical challenges should be addressed. Stringent timing and synchronization requirements of the small cells are one of the challenges in implementing small cell radios. Currently, macro base transceiver stations use GNSS[1] receivers to comply with timing and synchronization requirements of the air interface standard (2G, 3G or LTE). However, the GNSS based synchronization solutions are not economically feasible in small cell implementations. Additionally, GNSS may not work well or at all in some indoor environments or in dense urban areas with high rise buildings where the canyon effect limits the sky view of the reviver. In this research, a new low cost frequency synchronization solution based on the commercial AM wireless signals is proposed. Each BTS in a small cell utilizes the carrier component embedded in a commercial AM wireless signal as the reference clock. Different small cell transceivers can then correct the relative CFO[2] between them using a common over the air reference clock. As part of the MW[3] band, AM wireless signals can penetrate buildings and due to their ground-wave propagation have a relatively stable path that does not affect the transmitted signal. So they can be a good candidate for a synchronization solution. The proposed distributed frequency synchronization method is implemented using off the shelf components and its performance is evaluated in short term and long term measurement periods.

[1] Global Navigation Satellite Systems
[2] Carrier Frequency Offset
[3] Medium Wave

Acknowledgements

First of all, I would like to highly appreciate for all of the supports of my beloved supervisor professor Amir Keyvan Khandani who helped me in every stage of my research. I never forget his kindness and generous support by providing me with lots of resources and instruments to learn more and gain experience in the field of wireless communication systems. His support and vision for practical projects in the field of wireless communication really helped me to gain valuable insight that I could never gain if I were anywhere else in Canadian universities. The CST research lab has tremendous potential to be in the list of top research facilities in North America due to its advanced instruments and devices as well as its human resources and talents. I highly appreciate for all of the supports and guidance of professor Khandani who created such a great environment for progress and success.

Also, I would like to sincerely appreciate all of the supports of one of the best engineers that I have ever met in my life, Ali Nik Akhtari. I owe Ali lots of my practical viewpoints and experiences that I gained in the CST research lab.

Finally, I say a big thank you to all of the past and current team members of CST research lab including Alireza Khodaian and Aidin Reihani who played an important role in shaping my academic personality.

Dedication

This thesis is not definitely the last milestone in the journey of my life. My wonderful journey in pursuing Master of Applied Science in Electrical Engineering started on September 2015 and I am very pleased that I could finish one of the biggest chapters in the book of my life. I would like to dedicate this thesis and all of my success to the two angels who helped me during every stage of my life, my dear parents Mohammad Ali and Zahra. I would also dedicate my thesis to my beloved brother, Matin, my best friend in the life.

It would be my pleasure to dedicate my thesis to all of my teachers in the past and most importantly, Professor Amir Keyvan Khandani who helped me to learn more and gain more experiences from the beautiful world of communication systems engineering.

Also, I never forget my teachers in the past, professors Mohsen Maddahali, Mohammad Javad Omidi and Mahmoud Modarres Hashemi at Isfahan University of Technology for their generous support during the years of my bachelor studies in Iran. I dedicate my thesis and my success to them for all of their efforts in shaping my academic personality.

Table of Contents

List of Tables

List of Figures

Chapter 1

Synchronization Aspects in Cellular Mobile Networks

The rapid development of cellular mobile networks in recent years demanded for the continual innovation in the synchronization solutions of cellular mobile networks. Additionally, the migration from Time Division Multiplexing (TDM) to packet-based technologies in mobile networks pushed the telecom industry to define new methodologies to distribute accurate time and frequency references in the network backhaul to be accessible at the radio base station sites. Furthermore, the promises and advantages of using coordination features at the radio interface in small cell environments demanded for new synchronization technologies in telecom industry. Specifically, new synchronization methods are needed to address the synchronization requirements of small cell deployments. The required synchronization accuracy at the radio base station depends on the Third Generation Partnership Project (3GPP) radio access technology used and types of radio features (e.g. Inter Cell Interference Coordination) being deployed in the mobile network. In this chapter, synchronization needs of different radio access technologies and the available synchronization solutions are discussed. Additionally, the applicability of current synchronization solutions to the small cell deployments is addressed. Finally, the urgent need for innovative synchronization solutions in small cell environments is discussed.

1.1 Synchronization Requirements in the Mobile Networks Backhaul

International Telecommunication Standardization Sector (ITU-T) has defined three types of synchronization as follows: [1, 21] [1]

- Frequency synchronization: In a cellular mobile network, two nodes are said to be frequency synchronized with each other when their local operations including the signal generation, modulation and transmission are controlled by a reference signal whose significant instants occurs at nominally the same rate. For each type of application, any variation in the rate of significant instants of clock signals should be within a specified limit.
- Phase synchronization: The nodes in a cellular mobile network are said to be phase synchronized when all of them have access to a reference timing signal whose rising edges occur at the same instant. It must be noted that the different in the timing of occurrence of rising edges in the timing signal must be within a specified limit for different air interface radio specifications and the type of interference coordination features implemented in the network.
- Time synchronization: The nodes in a cellular network are said to be synchronous in time when all of them have access to a common absolute time reference. In other words, all nodes have information about the absolute time and share a common time scale.

Each of the three types of synchronization leads to different synchronization technologies in mobile networks. Frequency synchronization is the fundamental requirement of all mobile systems. Particularly, to minimize interference, facilitate hand over between mobile base stations, and comply with the regulatory requirements on signal generation, the generated RF signal at the radio interface must be in a strict compliance with frequency accuracy requirements of the specific radio access technology used at the air interlace. The frequency synchronization requirement of LTE standard at the air interface can be found in [3]. As described in the 3GPP Technical Specification (TS) 36.104 [3], the same clock source must be used for generating both RF signal and data clock. For the Wide Area Network (WAN) base stations, the modulated carrier frequency of the base transceiver station shall be accurate within ± 50 parts per billion (ppb) when the signal is observed over a

[1] Refer to ITU-T G.810 and G.8260 for further information

period of one subframe which is 1 milliseconds in LTE standard. The frequency accuracy requirement for pico base stations is relaxed to 100 ppb. The other 3GPP radio access technologies, including Global System for Mobile Communications (GSM) and Wideband Code-Division Multiple Access (WCDMA) have similar frequency synchronization requirements as LTE standard.

In Time Division Duplexing (TDD) systems, phase synchronization is required because the same frequency band is used for both uplink and downlink transmission and the signal is transmitted in different time slots. Hence to avoid interference between adjacent cells, the base stations need to be in phase alignment with each other. Particularly, in LTE-TDD radio access technology, base stations need to be synchronous in time with an accuracy of less than $3\mu s$ for cells of equal or less than 3km radius and $10\mu s$ for cells of more than 3 km radius. [4]

Different synchronization requirements for the most popular radio access technologies and some of the radio features that could be implemented in a small cell deployment are shown in Table 1.1 .

1.2 Demands of Small Cells on the Network Backhaul

With the advent of smart phones in recent years and the exponential growth of the mobile users, mobile network operators are facing growing demand for higher capacity in the network. To address the unprecedented increasing demand for higher capacity in the mobile networks, the overlay network idea is proposed by leaders in telecom industry. In the overlay network idea, mobile network capacity is increased by implementing small cell base stations beside the traditional macro cells in the service area. To provide excellent data rates and offload the macro cells traffic, small cells should be installed close to known traffic hot spots in the servicing area. Also, to provide good indoor data rates and coverage, indoor small cells can be easily installed in large numbers due to their small size and flexible backhaul solutions available at indoor service areas. In dense urban areas, outdoor small cells are deployed to offload the traffic from both indoor and outdoor users.

It must be noted that small cells should be deployed on the same carrier frequency as the macro cellular layer hence the radio spectrum utilization is maximized. On the other hand, deployment of small cells on the same carrier frequency as the underlying macro network will cause inter cell interference in the network. Specifically, LTE standard is designed to operate in interference limited environments and full radio spectrum can be reused in all cells. There are some techniques to improve overall system performance

Air Interface Technology/ Coordination Feature	**Frequency Accuracy**	**Time/Phase Accuracy**
WCDMA	± 50 *ppb* (Wide Area Networks) ± 100 *ppb* (Local Area Networks) ± 250 *ppb* (Small Cells)	N/A
TS-SCDMA	± 50 *ppb* (Wide Area Networks) ± 100 *ppb* (Local Area Networks) ± 250 *ppb* (Small Cells)	± 1.5 μs
LTE-FDD	± 50 *ppb* (Wide Area Networks) ± 100 *ppb* (Local Area Networks) ± 250 *ppb* (Small Cells)	N/A
LTE-TDD	± 50 *ppb* (Wide Area Networks) ± 100 *ppb* (Local Area Networks) ± 250 *ppb* (Small Cells)	± 1.5 μs ($r_{cell} \leq 3km$) ± 5 μs ($r_{cell} > 3km$
eICIC	N/A	± 1 μs
Dynamic Point Blanking (Coordinated Scheduling)	N/A	± 1.5 μs
Downlink Joint Transmission	N/A	± 1.5 μs

Table 1.1: Synchronization Requirements of Most Popular Radio Access Technologies and Coordination Features

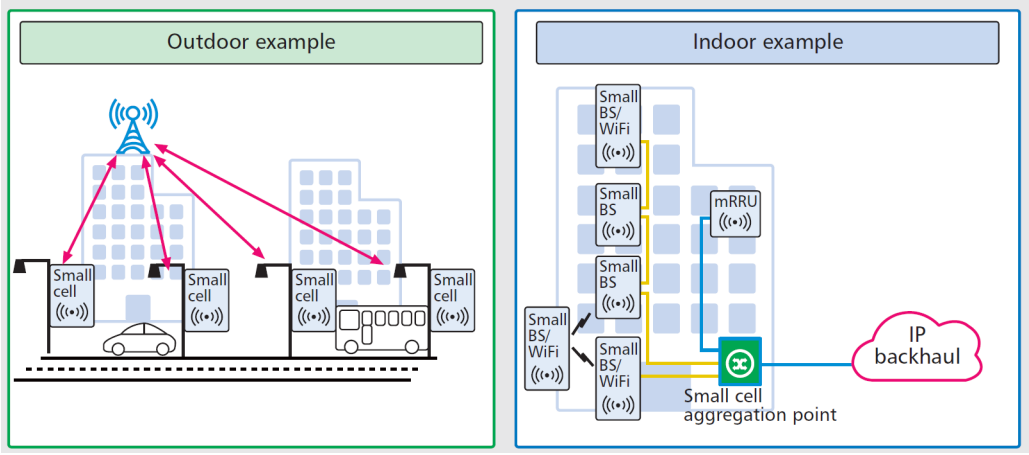

Figure 1.1: Different Deployment Environments for Small Cell Base Transceiver Stations [10]

in LTE standard by coordinating the interference between the cells. Coordinated MultiPoint (CoMP) and intercell interference coordination (ICIC) are two such interference coordinating mechanisms.

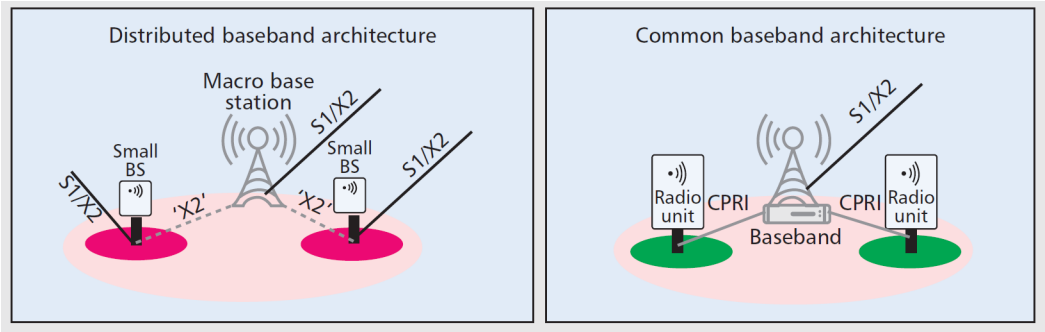

Figure 1.2: Distributed Baseband and Common Baseband Architecture of Small Cells [10]

There are two different small cell architectures that are used in industry namely the distributed baseband architecture and common baseband architecture. These two different architectures are shown in Figure 1.2. The common baseband architecture which is usually referred to as distributed radio access network (D-RAN) comprises of a baseband unit that is connected to two or more radio units. These units can be at the macro size or a low power

small cell radio unit. Radio units are connected to the baseband unit via the CPRI [2] link. It is worth noting that the concept of small cell implies the use of low power radio units. These radio units can be located at the same location as baseband unit or geographically separated from the baseband unit.

CPRI is an interface that connects an RU[3] or RRU[4] to the BBU[5]. It is an internal interface in base stations and should not be considered as part of the network backhaul. No additional synchronization is required for the radio units or remote radio units since CPRI delivers synchronization to the Radio Units. Additionally, the synchronization method that is provided to the base station via the backhaul or a locally GNSS [6] receiver is independent of CPRI synchronization.

In a distributed baseband architecture, a macro base station which manages the underlying cellular network is connected to multiple low power base stations within the macro cell coverage area. All of these base stations have their own baseband and radio units, and dedicated S1 and X2 interfaces. In LTE standard, the S1 interface connects the eNodeB and the packet core, and X2 interface connects different eNodeBs to each other. It must be noted that both S1 and X2 interfaces are considered as part of the network backhaul and are external eNodeB interfaces.

The requirements on synchronization accuracy in small cell deployments depends on the air interface standard and radio features that are planned to be deployed to coordinate intercell interference. For example, low power base stations such as pico base stations may be deployed with radio features that require no additional synchronization demands on the backhaul which are beyond the requirements of existing macro base stations. Features such as range extension, which is used to improve uplink coverage, and load-based handover, which is used to offload the macro network, place no new demands on synchronization accuracy compared to the already existing demands from the LTE technology. [10].

In a small cell deployment scenario, time synchronization between the small cell base stations and the underlying macro base stations is required if the interference coordination features that planned to be used in the system have tight synchronization requirements. For example, interference coordination features such as eICIC [7] and dynamic point blanking require the timing synchronization between base stations to be within $\pm 1\ \mu s$ to $\pm 1.5\ \mu s$.

[2] Common Public Radio Interface
[3] Radio Unit
[4] Remote Radio Unit
[5] Base Band Unit
[6] Global Navigation Satellite System
[7] enhanced Inter Cell Interference Coordination

1.3 Challenges in Small Cell Synchronization

In designing synchronization solutions for small cell networks, a number of issues have to be considered for designing the system. These issues are as follows:

1. Deployment environment: The most common method for synchronizing base stations in the cellular mobile networks are GNSS[8]-based synchronization solutions. In a GNSS-based synchronization scenario, a local oscillator with good short term stability is disciplined by the Rubidium or Cesium oscillators on the GNSS satellites whose signal is available at the earth surface. One of the most important features that must be considered before designing a small cell synchronization solution is deployment environment. For example, if the small cells are designed to operate in indoor environments the GNSS-based synchronization solutions may not be useful. GNSS signal can not penetrate in buildings with a reasonable signal level to be above the receiver sensitivity hence the receiver needs a clear skyview to receive and process GNSS radio signals. If the small cell is located away from the window with limited or no view of the sky, the GNSS-based synchronization may not work at all. Additionally, if the small cell is planned to offload the macro cell network traffic in dense urban areas with tall buildings the GNSS-based synchronization solutions may not work well due to the "canyon effect". Canyon effect causes the small cell base stations to have a restricted view of the sky and prevents the GNSS signal to be received with reasonable signal power.

2. Network backhaul specifications: When adding the overlay network to an existing cellular network by adding small cell base stations, the mobile operators usually have to use the readily available network backhaul in BTS[9] installation site. This network backhaul available for small cell base stations is usually not part of the mobile operator's macro backhaul. The quality of the available backhaul is usually quite different from the macro backhaul of the operator's network. For example, it may have high PDV[10] or unreliable class of service support. Reliability issues and versatile nature of different backhauls at different small cell deployment sites make it impossible to distribute timing and frequency synchronization across the network using packet-based technologies.

[8] Global Navigation Satellite System
[9] Base Transceiver Station
[10] Packet Delay Variation

3. Cost: As part of the business model for small cell deployments, the base stations that are planned to operate as the overlay network should be low-cost and low-power. To fulfil this requirement, the synchronization method used in the small cell should also be low-cost and not power hungry. This implies that complex and high power consuming solutions like the GNSS-everywhere or local time servers are not an appropriate option for small cell synchronization.

4. Security considerations: Since small cell base stations are physically available in different indoor and outdoor environments, this makes them more exposed to attackers for disrupting synchronization of the mobile network. There are some methods like IPsec that can be used to avoid such attacks in small cell base stations however using security algorithms like IPSec will introduce additional delays to the network and degrades the performance of packet-based synchronization solutions.

1.4 Existing Synchronization Solutions for Small Cells

Although some issues and concerns are needed to be addressed in small cell synchronisation solutions, the synchronization requirements of small cells are analogous to the ones required by macro cell base stations. It means that macro cell synchronization solutions can be readily used in small cells without any modification. However, in using the traditional synchronization solutions some specific concerns and requirements are needed to be considered as mentioned in section 1.3. Additionally, some special solutions might need to be designed just for small cells. The existing synchronisation solutions of small cells can be generally categorized in two sections namely frequency synchronization and, time and phase synchronization. Synchronization technologies in each of them are discussed in the following sections.

1.4.1 Frequency Synchronization

There are two solutions that are being used in packet based networks to distribute frequency synchronization across all network elements. These approaches are as follows:

- Physical-layer-based methods: In Ethernet-based networks, synchronous Ethernet (SyncE) is the most common physical-layer-based synchronization solutions[11]

[11]For more information on SyncE refer to ITU-T G.8261, G.8262 and G.8264 [1]

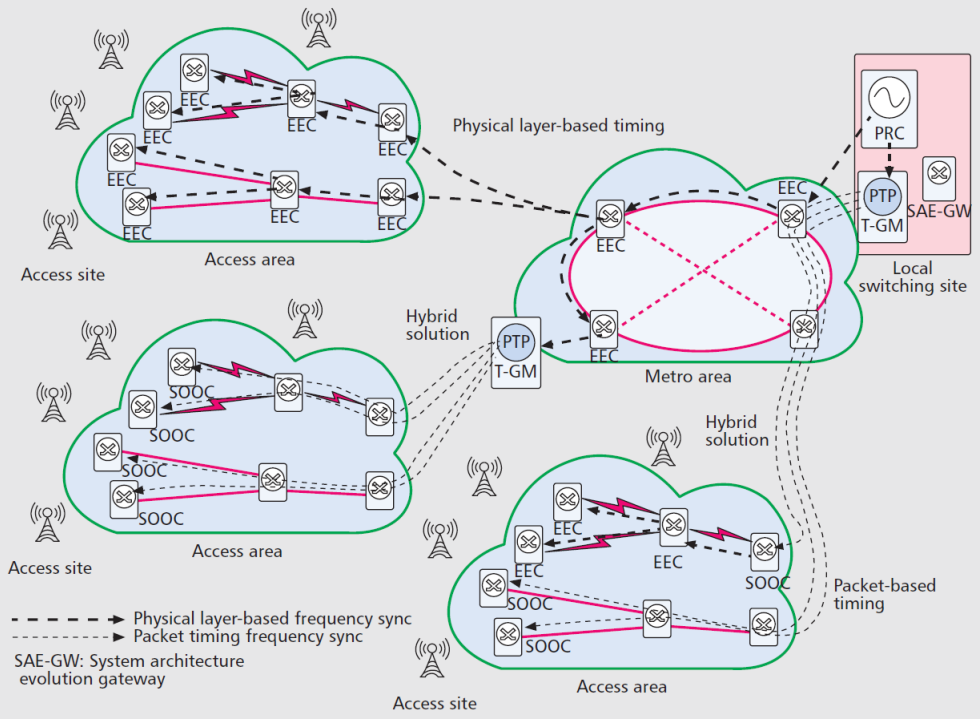

Figure 1.3: Possible Approaches in Frequency Synchronization of Cellular Networks [10]

- Packet-based methods: These kind of synchronization solutions are generally used in end-to-end architectures as defined in [1]. The most widely known packet-based synchronization methods are PTP[12] and NTP[13]

It must be noted that each of the above methods can be used separately or a hybrid solution can be developed that a physical-layer-based solution provides the required references for packet-based synchronization methods. An overview of possible approaches in frequency synchronization of mobile networks is shown in Figure 1.3. In the syncE method, every node between the master primary reference clock (PRC) must handle the synchronous Ethernet by means of the Ethernet Equipment Clock (EEC). These nodes can be a switching site or the end user equipment (end user equipment can include the base stations at the edge of access layer).

[12]Precision Time Protocol
[13]Network Time Protocol

In the PTP packet-based synchronization solution, an IEEE 1588-based telecom grand master (T-GM) is responsible for generating timing packets that are received by the slave only ordinary clocks (SOOC) installed in the end nodes of the network. It muse be noted that T-GM in PTP synchronization method has access to a very accurate primary reference clock that is usually provided by transfer standards discussed in section 2.4 or GNSS-based solutions. In the case of frequency synchronization using NTP packets, an NTP time server is used in the network backhaul instead of telecom grand master(T-GM).

1.4.2 Time and Phase Synchronization

When phase synchronization is required in a mobile network, the typical approach is distributing a timing signal across the network as a common time synchronization reference. As it was mentioned in Table 1.1, the timing requirements of commonly used mobile standards are within the microseconds range. In existing standard specifications, the following two approaches has been defined to distribute timing synchronization across the network [2].

- Distributed primary reference timing clock (PRTC): In this approach the accurate timing signal is usually provided by a GNSS receiver which is traceable to NIST[14] and UTC[15]
- Packet-based time synchronization: PTP protocol is used in this method to distribute time synchronization across the network elements [15]

In today's mobile networks, a hybrid solution including both the above methods is usually used for time and phase synchronization across the network. In a hybrid solution, the PRTC which is usually a GPS receiver (GPS receiver is commonly used in North America) and the T-GM which is the grand master in PTP technology are deployed closer to the network edge [2]. For example PRTC and T-GM are usually located in a small cell cluster. In this case, the hybrid solution is called local time synchronization distribution [10].

Detailed specifications of the packet-based timing synchronization methods is still under research and development stage in the telecom standardization bodies. Although the specifications are not defined yet, some considerations and assumptions can be made based

[14]National Institute of Standards and Technology
[15]Universal Coordinated Time

on the timing requirements of mobile networks. Using packet-based solutions to address the stringent timing requirements of air interface standards in small cell deployments dictates that all nodes in the chain between the master (PRTC) and the base stations should be able to process PTP packets using either a telecom boundary clock (T-BC), or a telecom transparent clock (T-TC).

Combining packet-based timing synchronization solutions with a physical-layer-based frequency synchronization method like SyncE will usually lead to the improved performance and reliability of timing synchronization. For example when the link to PRTC is lost (in the case of GNSS signal lost), the stable and accurate frequency reference provided by the physical layer can be used to enhance performance during holdover period. [10]

Distributing time synchronization across the network using packet-based solutions relies on the two-way packet exchange between the PRTC and the end nodes (e.g. small cell base stations). Furthermore, the path delay is estimated using round-trip delay measurements in PTP algorithms. So any asymmetry in the network topology would result in erroneous path delay estimation and that leads to decreased performance of PTP synchronization method. So one of the key aspects in PTP solutions is controlling source of asymmetry in the network architecture between timing master and slave nodes[16]. This dependence of packet-based timing solutions on network asymmetry is one of the major drawbacks of packet-based methods.

1.5 Emerging Synchronization Solutions for Small Cells

There are numerous synchronization solutions that are still being reviewed by standardization bodies to be deployed in small cell scenarios. These solutions include G.fast, local time distribution, partial timing support, and over the air synchronization.

The evolved version of G.fast standard as defined in ITU-T, is one of the proposed frameworks for the new high speed copper-based transmission lines. G.fast includes methods and algorithms to transport frequency and time synchronization for copper-based networks.

Depending on the deployment environment of small cells, it may be required that the accurate time synchronization be distributed across the base stations. For example in pico base stations supporting eICIC, accurate timing distribution is needed. On the other hand, PTP solution is heavily dependent on network asymmetries and using that

[16]Refer to ITU-T G.8271 for more information on controlling asymmetries in the network backhaul [2]

for time synchronization requires controlling any source of asymmetry in the network. Asymmetry control in the network requires that every node in the network supports IEEE 1588. On the other hand, backward compatibility of synchronization solution with the legacy infrastructure makes it impossible to have timing synchronization with full on-path support since supporting of IEEE 1588 in every network node is a relatively new method and has not yet fully standardized for telecom equipment manufacturers.

There are some methods to address security concerns in synchronization solutions for small cells deployment. However using such methods like IPSec secure tunnels is contradictory to distributing accurate timing synchronization across the network. Specifically, IPSec processing algorithms would add unacceptable jitter and delay to the packet-based solutions. Additionally, using IPSec makes it impossible for intermediate nodes to process PTP packets as part of the full on path support technology of PTP. So addressing the stringent synchronization demands of small cells and security considerations at the same time is not a trivial task in designing synchronization solutions for small cell deployments.

To address some of the issues explained in the above paragraph, special synchronization architectures have been proposed. These methods rely on distributing PRTC and T-GM closer to the mobile network edge hence allowing a local time reference to be distributed in a much smaller network domain. Implementing a local accurate time and frequency source implies implementing a GNSS receiving equipment in small cell aggregation nodes. This accurate time and frequency reference can then be distributed across the small cell base stations that are connected to the aggregation node.

The other method that is still under standardization is the PTP with partial on path support. In this method, there is no longer a requirement for all nodes to support IEEE 1588. In PTP partial on path support, a T-GM is proposed to be placed somewhere in the network aggregation layer. Implementing this method requires addressing lots of challenges that arise form asymmetries in the network due to queuing delays.

The final feasible synchronization solution that can be implemented to address stringent requirements of the air interface standards is using GNSS signals at the cell site. This method which is called "GNSS Everywhere" needs a GNSS receiver to be installed in every base station. As discussed earlier, there are serious concerns and constraints on using GNSS everywhere method including but not limited to availability of GNSS signals in different environments, cost consideration and security concerns. Specifically, GNSS everywhere solution is not a feasible option for indoor deployments of small cells. Combining the GNSS everywhere with other synchronization methods like SyncE and IEEE 1588 PTP is proposed to mitigate such issues [10]. However, there are lots of ongoing research and development in this sector to address synchronization challenges for small cell deployments.

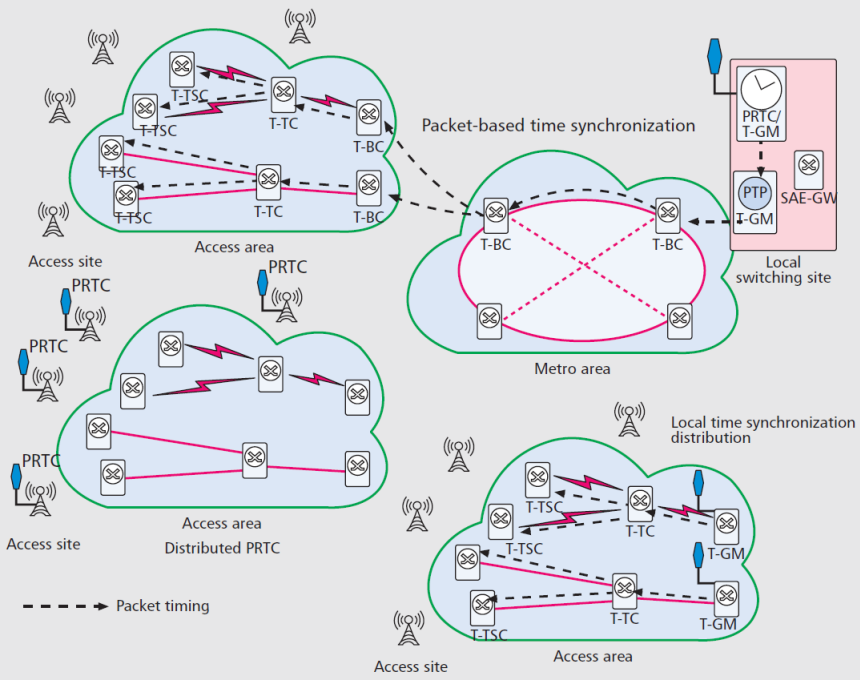

Figure 1.4: Different Time Sycnrhonization Solutions in the Backhaul of Mobile Networks [10]

New synchronization solutions are under development to address some of the challenges in small cell synchronization that existing technologies can not address.

1.6 Need for New Synchronization Solutions

As it was explained in the previous sections in this chapter, there are numerous open challenges in synchronization of overlay networks. As it was mentioned earlier, existing synchronization technologies can not address requirements of small cell base stations. There is a strong need for new synchronization solutions that can address small cell requirements in a technically feasible, low-cost and low-power method. In this thesis, a new frequency synchronization solution for small cells is proposed. This method relies on the idea of over-the-air synchronization. In the proposed method, small cell base stations can listen to the over-the-air signal and tune their local oscillators using the over-the-air signal as a reference.

Chapter 2

Frequency Measurement and Calibration Techniques

2.1 Chapter Overview

Different synchronization aspects in small cell deployments have reviewed in the previous chapter. Additionally, existing synchronization solutions and their applicability to small cell deployments were discussed and the strong need for new synchronization methods has been discussed. Before explaining the proposed synchronization solution for small cell base stations, it is required to have a brief review of different frequency measurement and calibration techniques. In this chapter, technical terms in defining performance of frequency references are defined. Then different oscillator and frequency sources are studied. Finally, transfer standards as the reliable sources of traceability to national standards are discussed.

In the most basic definition, frequency is the rate at which an event repeats itself. If the repetition interval of the event is considered to be T, then frequency will be the reciprocal of this time interval that equals to $f = 1/T$. According to the International System of Units (SI), frequency is always expressed in Hz. By means of appropriately selected measurement techniques, average frequency of a signal can be measured very precisely over specific time intervals.

Electrical devices that produce a known frequency are called frequency standards. The frequency standards should be calibrated to force their output frequency to be within some bounds. These bounds are defined by tolerance requirements that are defined based on the specific application. I

Frequency calibration is referred to a set of steps by which the performance of a frequency standard is measured. The frequency standard that is under measurement is called DUT (Device Under Test). In most applications, DUT is a quartz, rubidium or a cesium oscillator. At the heart of most frequency calibration methods is the comparison of the frequency output of DUT to a precise reference frequency. For the calibrations to be valid, the reference should outperform the DUT by a specified ratio that is usually determined by the performance tolerance of user application. This ratio is called Test Uncertainty Ratio (TUR). The lower the value of TUR the longer measurement period is required for the calibration to be valid. For typical frequency synchronization standards, TUR of 10:1 is preferred due to the short amount of time required for the calibration to take place. It muse be noted that lower values of TUR can be used at the expense of longer frequency measurement and processing intervals.

In a calibration method, the difference between the actual frequency and the nominal frequency of a frequency standard is measured. This difference is called frequency offset and is one of the fundamental factors in the performance analysis of frequency calibration methods. With a high probability, the frequency offset of the DUT will remain within a certain range values called frequency uncertainty. User application determines the frequency uncertainty values and thresholds that must be met or exceed in different scenarios.

In frequency calibration methods, the frequency standard used as reference should be traceable. The International Organization for Standardization (ISO) defines traceability as follows:

"The property of the result of a measurement or the value of a standard whereby it can be related to stated references, usually national or international standards, through an unbroken chain of comparisons all having stated uncertainties." [20]

The "unbroken chain of comparisons" in the above definition should trace back to the National Institute of Standards and Technologies (NIST) located in the United States.

There are a number of transfer standards that can be used to deliver a frequency reference from the national standard to the site where DUT calibration is taking place. Transfer standards are devices that receive and process radio signals that provide frequency that is traceable to NIST. The radio signal is actually a link back to the national standard laboratories. There are a number of signals that can be used as links in frequency transfer standards. WWV, WWVH and WWVB are broadcast radio signals from NIST stations in the United States that contains timing and frequency information for calibrating DUTs that are located in remote places. Radio navigation signals from LORAN-C and GPS can also provide an appropriate radio signal to be used as the frequency transfer standard. Each of the mentioned signals, delivers the NIST traceability at a known level of uncertainty. The availability of such transfer signals is a tremendous advantage for frequency calibration in remote places. By using such radio signals, DUTs in different locations can be calibrated using over the air signals as long as the calibration stations equipped with a radio receiver.

In the following section, the specifications that are generally used to describe a frequency calibration method are introduced. Then various types of frequency standards and transfer standards are explained.

2.2 Frequency Calibration Specifications: Frequency Offset and Stability

Two main technical specifications of any frequency calibration method are frequency offset and stability. The definition of frequency offset and stability as well as their measurement techniques are introduced in this section. Throughout this section, frequency offset is often referred to frequency accuracy, and the stability is considered the same thing as uncertainty.

In a calibration method, the DUT is supposed to produce a specific frequency. Due to some physical phenomena, the frequency produced by the DUT is different from the nominal frequency. Once the calibration is completed, the frequency offset and uncertainty of DUT can be stated precisely. To measure frequency offset, the output signal of DUT should be compared to a precise frequency reference. Frequency offset measurement comprises of a phase comparison between the signal produced by DUT and the signal received from a frequency transfer standard. Once the amount of phase deviation and measurement period are known, frequency offset can be estimated. The measurement period is the time interval in which the phase comparisons are made. Frequency offset can be estimated using the following relation where ΔT is the amount of phase deviation and T is the measurement period.

$$f_{offset} = -\Delta T/T \tag{2.1}$$

Small values of frequency offset show that the frequency produced by the DUT is close to the frequency of the reference. As an example, an oscillator that accumulates $+1$ μs of phase deviation per day has a frequency offset of about -1×10^{-11} *with respect to the reference*. It must be noted that the frequency offset value can be converted to Hz if the nominal frequency of DUT is known. For example, the frequency offset of 1.27×10^{-12} for a DUT with nominal frequency of 10 MHz translates to $+0.0000127Hz$. So the actual frequency produced by the DUT is $10000000.0000127Hz$.

The measurement period should be selected long enough to ensure that the frequency offset of DUT is measured precisely and other sources are not contributing a significant amount of uncertainty to the measurement. As a result, we must make sure that ΔT is really a measure of only the DUT's phase deviation from the reference and is not being contaminated by noise from the reference or the components of measurement system. Here is the reason that why a 10:1 TUR is desirable. Actually, by selecting a TUR of 10:1, appropriate measurement periods can be selected in a way that calibration does not take much time. Many frequency calibration methods are able to measure a frequency offset of 1×10^{-10} in a 1 seconds measurement period [23]. Due to the limitations and conditions of the frequency transfer signals, a TUR of 10:1 is not always possible. For example, if the transfer standards based on the radio navigation signals such as LORAN-C or GPS are used, radio path noise contributes to the phase deviation. To overcome this problem, a measurement period of at least 24 hours is required to compensate for the path noise in the transfer standard signal. Such a measurement period is selected because changes in the path delay between the source and receiver often have a cyclical variation that averages out over 24 hours. In addition to averaging, curve-fitting algorithms and other statistical methods are often used to improve the uncertainty of frequency estimation. [32]

2.2.1 Frequency Offset

In most of the specification sheets of oscillators, the term frequency accuracy is used instead of frequency offset. Basically, frequency accuracy and frequency offset are equivalent terms that refer to the result of a measurement at a given point of time. Frequency uncertainty actually shows the upper and lower limits of the frequency offset. ISO[1] defines uncertainty as follows:

[1]International Organization for Standardization

"Parameter, associated with the result of a measurement, that characterizes the dispersion of values that could reasonably be attributed to the measurement." [20]

It can be said that the frequency uncertainty shows the possible range of values (or limits) for the frequency offset. As a rule of thumb, a 2σ uncertainty test is used in practice. This means that there is 95.4% probability that the frequency offset will stay within the mentioned range during the measurement period. The largest contributor to the frequency uncertainty is usually the stability of the DUT. In the next section, stability is explained.

2.2.2 Stability

First of all, it should be mentioned that there is an important distinction between frequency offset and stability. Frequency offset is a measure of how close the output frequency of DUT is to the frequency produced by the reference. It does not have any information about the quality of an oscillator. As an example, a stable oscillator may produce a frequency that has a large offset and needs to be adjusted by comparing its output to the frequency reference. On the other hand, an unstable oscillator may produce a frequency that is very close to the reference at a point of time but has large fluctuations over time. Stability, is a measure of the ability of an oscillator to produce the same frequency over time. Hence, there is no right or wrong frequency in the context of oscillator stability. It is just a measure of consistency of the frequency produced by DUT.

Stability is defined as the statistical estimate of the frequency fluctuations of a signal over a given period of time. Short-term stability usually refers to fluctuations over time intervals less than 100 s. Long-term stability is referred to measurement intervals greater than 100 s, but usually refers to periods longer than 1 day [28]. There are a number of statistical methods to estimate the frequency stability of a DUT. These methods are either in the frequency domain or time domain and statistical tools exist that can convert the estimates in each domain to the other and vice versa. Usually, time domain estimates are used since time interval counters are mostly used to measure frequency of signals in laboratories.

In the time-based stability estimation, a time series of frequency offset measurements is required. Then a number of statistical tools can be used to determine the dispersion or scatter of this time series as an indication of oscillator phase noise. It is evident that larger dispersion of frequency offset measurements translates to greater frequency instability of the oscillator.

Usually, classical statistics such as standard deviation or variance are used to measure the dispersion of a time series. However, they cannot be used in the case of frequency stability estimation. Variance is only valid for stationary time series, where the data is time-independent. When using the variance as a measure of dispersion of data, it is assumed that the noise is white. In other words, it is assumed that noise power is evenly distributed across the frequency band of measurement. Time series resulting from the frequency offset measurement of oscillator output are usually non-stationary. In other words, it contains time-dependent noise. For stationary time series, the mean and variance converge to particular values as the number of points in calculation are increased. On the other hand, in the case of non-stationary data, the mean and variance converge to nothing specific. Instead, they just fluctuate over time as a new measurement is added to the time series.

Due to the aforementioned reasons, a non-classical statistical tool is used to estimate frequency stability in the time domain. This statistic is called Allan variance. Since it is actually the square root of the variance, the term Allan deviation is better to be used. IEEE[2] recommended the Allan deviation to be used by manufacturers of frequency generation devices as a standard specification for stability. Allan deviation can be calculated as follows:

$$\sigma_x(\tau) = \sqrt{\frac{1}{2(M-1)} \sum_{i=1}^{M-1} \overline{x}_{i+1} - \overline{x}_i} \tag{2.2}$$

Where M is the number of elements in the x_t time series. It must be noted that Allan deviation is a function of the measurement interval hence the data samples in the time series should be separated by τ seconds. As it is evident in the above equation, Allan deviation subtracts the value of previous measurement from the current one to eliminate time dependency between successive data points.

As it is shown in Figure 2.1, the stability improves as the measurement period gets longer. Measurement system noise contributes to the fluctuations of frequency offset measurements. However, there is a lower bound for the Allan deviation of any oscillator which is called Flicker floor. After a largely enough measurement period, the oscillator reaches to its flicker floor after which no gain is achieved by increasing the measurement period. The flicker floor is where non-stationary processes in the frequency output of the oscillator dominates the white noise process. It is worth noting that most quartz and rubidium oscillators reach their flicker floor at a measurement period of 1000 seconds or less but cesium

[2]Institute of Electrical and Electronics Engineers

oscillators may not reach their flicker floor for 10^5 seconds (more than 1 day). Figure 2.2 shows different regions in the Allan deviation graph of an oscillator.

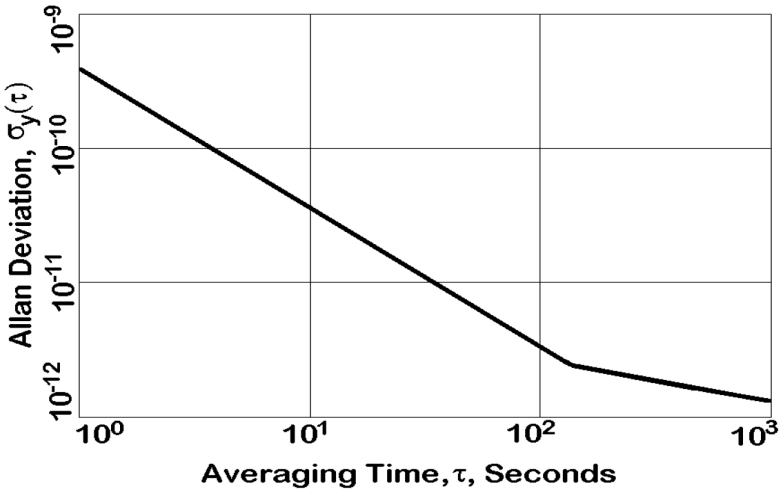

Figure 2.1: Sample Allan Deviation Graph
[28]

2.3 Frequency Standards

All of the frequency standards have a device that produces periodic repetitive signals. These signals can be sinusoidal signals or pulse trains. Such a device is called a resonator. A resonator needs energy to produce periodic signals. The combination of the resonator and the energy source is called an oscillator. Generally, there are two main types of oscillators: quartz oscillators and atomic oscillators.

2.3.1 Quartz Oscillators

Quartz crystal oscillators are the most common types of oscillators used in electronic devices and related industries. More than 10^9 quartz oscillators are produced annually for applications ranging from wristwatches and clocks to communications networks and space tracking systems. Those quartz oscillators that are used as stand-alone in electronic and telecommunication systems should be calibrated periodically and their frequency difference

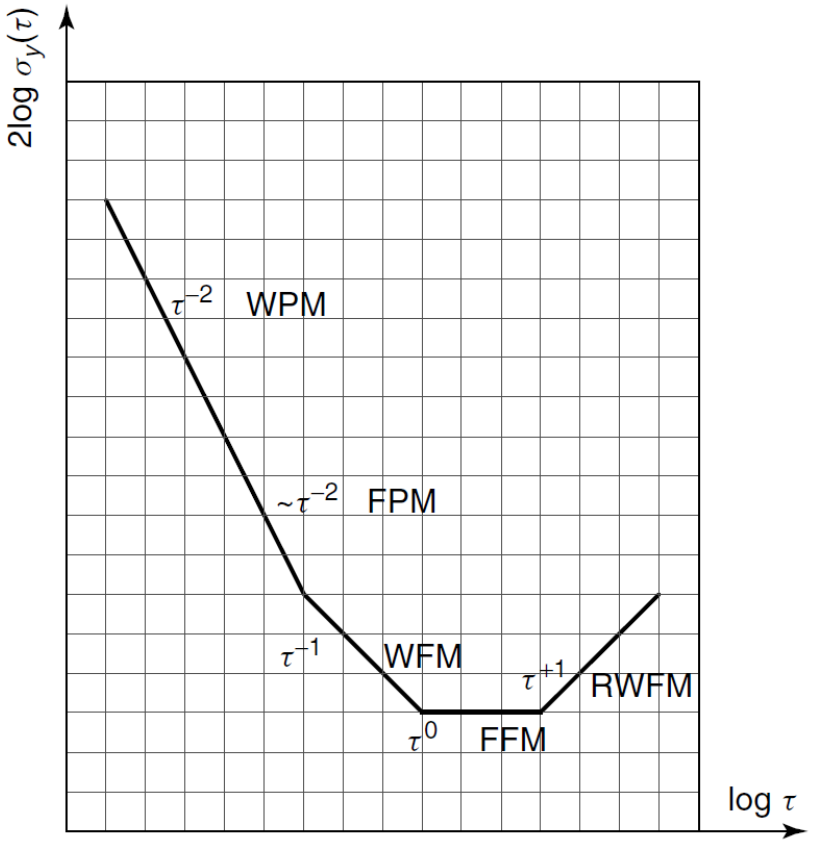

Figure 2.2: Regions of Different Noise Processes at the Oscillator Output

compensated over time. It must be noted that a typical quartz oscillator costs from a few hundred to a few thousand dollars depending on the quality of oscillator.

There is a quartz crystal inside every quartz oscillator. The internal crystal can be made from synthetic or natural quartz. It must be noted that almost all modern quartz oscillators are made up of synthetic materials. The quartz crystal inside the oscillator acts as mechanical resonator that produces periodic voltage due to the piezoelectric effect. This effect causes the quartz crystal to expand and contract when an external voltage is applied. Every crystal has a resonance frequency which can be determined by its physical dimensions and the type of material used in the crystal. Due to the uncertainties in crystal cuts and impurities of the materials used, no two crystals can be found that produce the same oscillating voltage hence it is impossible that two crystals produce exactly the same frequency. The output frequency of a quartz oscillator is either its fundamental frequency

or a multiple of that. One of the key features of quartz oscillators is their dependence on the environmental conditions such as temperature, humidity, pressure and vibration. [28, 33]

So in a typical quartz oscillator, the fundamental frequency changes when any of the environmental parameters changes. There are a number of ways that the environment-dependence of the fundamental frequency of quartz oscillators can be compensated for. The Oven Controlled Crystal Oscillator (OCXO) maintains the temperature of crystal in a relatively constant level by enclosing it in a temperature-controlled chamber called an oven. When an OCXO turns on, it goes through a process called warm-up period. In the warm-up period, the temperature of the crystal is stabilized to reduce the temperature dependence of the output frequency. After the warm-up period, the temperature inside the oven remains constant even in the case that the temperature outside the oscillator changes.

The other technological solution to solve the temperature dependence of the crystal oscillators is Temperature Controlled Crystal Oscillator (TCXO). In a TCXO, the temperature of the crystal is measured by a thermistor and the generated output voltage of that thermistor is applied to a voltage-variable reactance that is called varactor. This varactor changes the output frequency of the oscillator as a function of the output voltage of thermistor in a way that the effect of temperature is removed. This technique does not work well like OCXO but is much less expensive. That is why in most electronic circuits that high performance temperature independence is not required, TCXO is preferred over OCXO due to its lower cost. The other type of crystal oscillators that is less usual in electronic circuits and systems is Microprocessor Controlled Crystal Oscillator (MCXO). The MCXO uses a digital temperature sensor and digital algorithms that run on a microprocessor to compensate for the variations in the output frequency of the crystal oscillator. The MCXO falls between a TCXO and a OCXO in terms of cost and performance.

Oscillator Ageing

The output frequency of crystal oscillators changes over time due to the effects inside the oscillator. This systematic change in frequency with time is called ageing. Ageing can be positive or negative and is observed as a nearly linear change in the resonance frequency of the oscillator. Usually, the resonance frequency decreases over time which might indicate that the crystal is getting larger. Ageing has many possible causes including to the contamination of crystal due to deposits of foreign material, changes in the oscillator circuitry, or changes in the quartz material or crystal structure. High quality crystal oscillators have an ageing rate no more than 5×10^{-9} per year. The best in class crystal oscillators are known for their excellent short term stability. An OCXO might be stable to 1×10^{-13} at 1 second. The limiting factor in short term stability is often noise from the

electronic components in the oscillator circuitry. On the other hand, long term stability of crystal oscillators is poor due to ageing and other factors. Even the best OCXO must be periodically adjusted to stay within 1×10^{-10} of their nominal frequency.

2.3.2 Atomic Oscillators

The resonance frequency of atomic oscillators is based on the quantized levels of energies in atoms and molecules. According to the laws of quantum mechanics, the energies of a bounded system such as an atom has certain discrete values. Once an atom absorbs an electromagnetic energy, its energy level boosts to a higher level. Similarly, the energy level in an atom drops to a lower level by emitting electromagnetic energy. The resonance frequency of an atomic oscillator is the difference between the two energy levels divided by Planck's constant:

$$f = \frac{E_2 - E_1}{h} \tag{2.3}$$

Since the mechanism of atomic oscillators is based on a fundamental natural phenomenon, all of the them are considered intrinsic standards. There are generally three main types of atomic oscillators including rubidium standards, cesium standards and hydrogen masers. In all three types of atomic oscillators, there is a quartz crystal oscillator which is locked to the resonance frequency of an atom of interest (like rubidium or cesium). By locking the quartz crystal oscillator to an atomic resonance, most of the factors that degrade the long-term performance of a quartz oscillator disappear. That is because the atomic oscillators are much less sensitive to environmental perturbations than the quartz crystal oscillators. As a result, the long-term stability and uncertainty of an atomic oscillator are much better than those of a quartz oscillator, but the short-term stability is unchanged.

Rubidium Oscillators

Among all of the three main types of atomic oscillators, rubidium oscillators are of the lowest price. Rubidium oscillators might have been placed in the best point of cost/performance graph. They perform much better than quartz oscillators and are much less expensive than cesium oscillators.

The resonance frequency of a rubidium oscillator is 6834682608 Hz which is the resonance frequency of the Rb atom. This frequency is synthesized from a lower frequency

that is generated by a quartz crystal and the quartz frequency is steered by the rubidium resonance. The method of steering the frequency of the quartz crystal using the rubidium resonance yields a very stable frequency source with short-term stability of quartz oscillators but much better long-term stability due to rubidium resonance.

The initial price of a rubidium oscillator (typically from $3,000 to $8,000) is higher than that of a quartz oscillator but needs much less frequency adjustments over time. As a result, the long term cost of a rubidium oscillator is less than that of a quartz oscillator if it is intended to be used as a frequency standard.

Frequency offset of rubidium oscillators typically lies in the ranges from 5×10^{-10} to 5×10^{-12}. Stability is typically about 1×10^{-12} at one day. Maintaining the frequency within 1×10^{-11} can be easily done using a rubidium oscillator but is nearly impossible using even the best in class quartz oscillators. It must be noted that the performance of a well-maintained rubidium oscillator can approach the performance of a cesium oscillator considering the fact that rubidium oscillators are much smaller in size, more reliable and less expensive.

Cesium Oscillators

The primary frequency standards are based on cesium oscillators because the SI second is based on the resonance frequency of the cesium atom which is 9192631770 Hz. It must be noted that the output frequency of a cesium oscillator is very close to its nominal value and the ageing effects that cause change of output frequency over years is negligible. The time scale followed by all major countries, Coordinated Universal Time (UTC) is derived primarily from averaging the performance of a large ensemble of cesium oscillators, although some hydrogen masers also contribute to UTC. [28]

Cesium oscillators are at the heart of most modern time and frequency distribution systems. The primary frequency standard for the United States is a large cesium fountain oscillator named NIST-F1 that has a frequency uncertainty of about 1.7×10^{-15}. Cesium beam technology is used in commercially available cesium oscillators due to its smaller size. Commercial cesium oscillators have different qualities but their frequency uncertainty is bounded to $\pm 5 \times 10^{-12}$ after a short warm up period. After the warm-up period, the frequency uncertainty of a cesium oscillator is usually some parts in 10^{13}. Stability generally reaches parts in 10^{14} at one day, and it might take days or weeks before a cesium oscillator reaches its flicker floor. It must be noted that the initial purchase price of a cesium oscillator ranges from $30,000 to $80,000.

Hydrogen Masers

The most complex and most expensive commercially available frequency standard is the hydrogen maser. A few number of hydrogen masers are built and most of them are owned by observatories or national standard laboratories. "Maser" is an acronym that stands for *"Microwave Amplification by Stimulated Emission of Radiation"*. Masers are based on the resonance frequency of hydrogen atom which is 1420405752 Hz. Hydrogen masers are categorized into two types namely active masers and passive masers. In an active maser, a quartz oscillator is phase-locked to an spontaneous oscillation of a hydrogen atom. In the second type, passive maser, a quartz oscillator is frequency locked to the frequency reference of hydrogen atom. Since active masers derive the output frequency more directly from the atomic resonance, they have better short-term stability than passive masers. Both types of hydrogen masers have better short-term stability than a cesium oscillator. On the other hand, the frequency uncertainty of a hydrogen maser is greater than that of a cesium oscillator due to the inherent dependency of maser's performance on a complex set of environmental conditions. It must be noted that hydrogen masers typically cost about $200,000 or more due to their complex structure and low volume of production.

To summarize the material discussed in the previous sections, characteristics of different types of oscillators are shown in Table 2.1.

Oscillator Type	**Quartz TCXO**	**Quartz MCXO**	**Quartz OCXO**	**Rubidium**	**Cesium**	**Hydrogen Maser**
Primary Standard	No	No	No	No	Yes	No
Intrinsic Standard	No	No	No	Yes	Yes	Yes
Resonance Frequency	Mechanical	Mechanical	Mechanical	6.834682608 GHz	9.19263177 GHz	1.42040575 GHz
Stability $\sigma_y(\tau)$,$\tau = 1s$	1×10^{-9}	1×10^{-10}	1×10^{-12}	5×10^{-11} to 5×10^{-12}	5×10^{-11} to 5×10^{-12}	1×10^{-12}
Noise Floor $\sigma_y(\tau)$	1×10^{-9} ($\tau = 1$ to 10^2 s)	1×10^{-10} ($\tau = 1$ to 10^2 s)	1×10^{-12} ($\tau = 1$ to 10^2 s)	1×10^{-12} ($\tau = 10^3$ to 10^5 s)	1×10^{-14} ($\tau = 10^5$ to 10^7 s)	1×10^{-15} ($\tau = 10^3$ to 10^5 s)
Ageing/year	5×10^{-7}	5×10^{-8}	5×10^{-9}	2×10^{-10}	None	1×10^{-13}
Frequency Offset (After warm-up)	1×10^{-6}	1×10^{-7} to 1×10^{-8}	1×10^{-8} to 1×10^{-10}	1×10^{-10} to 1×10^{-12}	1×10^{-12} to 1×10^{-14}	1×10^{-12} to 1×10^{-13}
Warm-Up Time	$<$ 10s to 1×10^{-6}	$<$ 10s to 1×10^{-8}	$<$ 5min to 1×10^{-8}	$<$ 5min to 5×10^{-10}	$<$ 30min to 5×10^{-12}	$<$ 24hours to 1×10^{-12}
Cost	$ 100	$ 1000	$ 2000	$ 3000 to $8000	$ 30,000 to $80,000	$ 200,000 to $300,000

Table 2.1: Technical Specifications of Different Oscillator Types

2.4 Transfer Standards

As it was mentioned in previous sections, in a frequency calibration procedure, the output frequency of DUT (Device Under Test) is compared to a reference with known precision and guaranteed level of stability. The oscillator that is used in the DUT can be a quartz crystal oscillator or the other two atomic standards, namely rubidium and cesium oscillators. The reference of comparison in all of calibration methods is usually an oscillator with higher performance than the DUT oscillator or a transfer standard that receives a radio signal as the source of comparison [28]. All of transfer standards receive a radio signal that has a cesium oscillator at its source. The signal received using a transfer standard provides a cesium-based frequency to the user as the source of comparison.

In most of the transfer standards, the received radio signal is traceable to the national frequency standard maintained by NIST. Among the transfer standards, those that are transmitted in the HF band[3], including WWV and WWVH, and those that are in the LF band[4], namely WWVB, are directly controlled by NIST hence the radio signals received from these stations is directly traceable to the national frequency standard. Other transfer standards including LORAN-C and Global Positioning System (GPS) are traceable because their signals are regularly compared with the NIST frequency standard.

It must be noted that using the transfer standards has a disadvantage that even if a nearly perfect oscillator is used as source of transmitted radio signal over the air, the effects of radio path between transmitter and receiver degrades the performance of the precise clock source. As the radio signal traverses the wireless link between transmitter and the receiver, the frequency fluctuates due to the different paths that signal traverses along the way to the receiver. When the received signal is observed over long periods of time, these fluctuations will average out. However, frequency fluctuations in transfer standards will degrade the short-term performance of the standard. Actually, the constantly changing radio path introduces frequency fluctuations even if a cesium-based oscillator with negligible frequency offset and high accuracy is used at the transmitter. As a result, it can be concluded that transfer standards are not suitable for short-term stability measurements. However, they can be used as a source of comparison for long-term stability measurements when the frequency fluctuations can be reduced by means of averaging the observed signal over long enough measurement intervals. If a cesium oscillator is used at the transmitter, observing long enough periods of received signal and averaging can yield the performance of the cesium oscillator at the receiver.

[3]High Frequency Band (3-30 MHz)
[4]Low Frequency Band (30-300 kHz)

Transfer Standard	**Frequency Uncertainty over 24h Measurement Period**
HF receiver (WWV and WWVH)	$\pm 5 \times 10^{-9}$
LF receiver (LORAN-C and WWVB)	$\pm 1 \times 10^{-12}$
GPS receiver	$\pm 2 \times 10^{-13}$

Table 2.2: Frequency Uncertainty of the Most Common Transfer Standards [28]

Among the frequency standards, some of them have path variations such that makes them unsuitable for high level frequency calibrations. As an example, consider the broadcast signal from WWV station, located in Fort Collins, Colorado. WWV is a HF radio station that transmits on 2.5, 5, 10 and 20 MHz. Although the WWV station is referenced to NIST, much of its performance is lost when received in a remote station. The receiver that listens to a WWV signal, practically receives its skywave component. Skywave component is part of the signal that travels up to the ionosphere and is reflected back to the earth. The path delay of the WWV signal constantly changes due to the constant changes of the height of ionosphere in different times of day and different locations. The path delay can be as much as 500 to 1000 microseconds. Since there is so much variability in the path, averaging leads to only limited improvement. Therefore, although WWV is traceable to NIST, its frequency uncertainty is limited to parts in 10^9 when averaged over 24 hour intervals. [28]

The other transfer standards, have more stable paths and much lower uncertainty values. Low Frequency (LF) radio stations such as WWVB and LORAN-C can provide traceability to NIST with a frequency uncertainty of 1×10^{-12} per 24 hours. The wireless path of an LF signal is much more stable than a HF signal but an LF signal still experiences a variable path delay to the changes of ionosphere height at sunset and sunrise. Due to the reliability issues of frequency standards in both LF and HF path, the most widely used signals as the source of frequency are those transmitted by Global Positioning System (GPS) satellites. GPS signals have the advantage of unobstructed path between the transmitter and the receiver. The frequency uncertainty of GPS is about 2×10^{-13} per day.

The most common transfer standards with their frequency uncertainty is shown in Table 2.2. It must be noted that the frequency uncertainty in this table has been calculated with respect to NIST when averaged for at least 24 hours. [28]

2.4.1 WWVB

Many countries broadcast time and frequency signals in the LF band from 30 to 300 KHz, as well as in the VLF band from 3 to 30 KHz [28]. Due to the fact that part of the LF signals is the ground-wave that traverses the curvature of the earth, the path stability of these signals is quite good. One such transfer standard is NIST's WWVB station located at Fort Collins, Colorado that transmits on 60 KHz and provides coverage for most of the North America.

It must be noted that environmental conditions as well as daily and seasonal changes still influence the WWVB path although it is far more stable than an HF path. Path length is important in LF band radiation because part of the signal travels along the ground (ground-wave component) and another part is reflected from the ionosphere (sky-wave component). The ground-wave path is far more stable than the sky-wave path. If the path is relatively short (less than 1000 km), the receiver might continuously track the ground-wave signal since it always arrives faster. For longer paths, a mixture of ground-wave and sky-wave is received. It must be noted that over a very long path, the ground-wave can become so weak that it will be possible to receive only the sky-wave component of the transmitted signal. In the former case, the path becomes much less stable.

In an LF path, characteristics vary at different times of the day. The receiver can usually distinguish between ground-wave and sky-wave components of the signal during daylight and night-time hours and as a result, path stability can vary by only a few hundred nanoseconds per day. However, if some sky-wave is being received and cannot be identified, diurnal phase shift occurs at sunrise and sunset. For instance, as the path changes from all darkness to all daylight, the ionosphere lowers and the path gets shorter. The path length then stabilizes until either the transmitter or receiver enters darkness. At this point ionosphere rises and the path gets longer.

Using WWVB signals as transfer standard has lots of advantages over the other two alternative methods. By using a good antenna and RF front end to boost the received signal, WWVB can provide traceability to NIST with a frequency uncertainty of 1×10^{-12} when averaged over one day. Such a high level of performance can be achieved with very low cost and simple receivers. [24]

2.4.2 LORAN-C

LORAN-C is a radio navigation system that operates in the LF band. U.S. Department of Transportation (DOT) is responsible for the operation of most of LORAN-C systems.

The system is comprised of a number of stations called chains. Each chain has one master station and two to five secondary stations. The stations operate at a high power, typically between 275 to 800 kW, and broadcast on a carrier frequency of 100 kHz using a bandwidth of 20 kHz. In order to identify the transmitted signal of each chain, each LORAN-C station transmits a unique stream of pulses. Actually, each chain transmits a group of pulses that includes the pulses of the individual stations. The pulse group is then sent at a unique group repetition interval (GRI). As the transmitter radiates the pulse groups, they traverse to the receiver in all directions. The ground-wave component travels parallel to the surface of the earth and the sky-wave component travels upward to the ionosphere and is reflected back to the earth surface from the ionosphere. In the LORAN-C, the pulse shape is designed so that the receiver can distinguish between ground-wave and sky-wave components. So the LORAN-C receiver can lock to the more stable ground-wave component of the transmitted signal. Actually, a typical LORAN-C receiver acquires the ground-wave component of the signal and locks to it by tracking the third cycle of the received pulse signal. In the technical design of LORAN-C, third cycle is chosen for two reasons. First, it arrives early in the pulse so we know that it is ground-wave. Second, the receiver can lock to the third cycle easier since its amplitude is higher than the first and second cycles in the pulse. Generally, a receiver within 1500 km of the transmitter can track the same ground-wave cycle indefinitely and avoid sky-wave reception. If the path exceeds 1500 km, the receiver might lose lock and jump to another cycle of the carrier. [28]

The radio path variations cause the frequency uncertainty of LORAN-C to be degraded. The size of these variations depends on the signal strength, distance between transmitter and the receiver, weather and atmospheric conditions and the quality of receiver circuitry and antenna. These variations will average out over time hence the long-term stability of LORAN-C is intact. Actually, path length variations cause the short-term stability of LORAN-C to be degraded. Due to the aforementioned factors, a measurement period of at least 24 hours is recommended when using LORAN-C to calibrate atomic oscillators.

2.4.3 Global Positioning System (GPS)

U.S. Department of Defence (DOD) developed and operated GPS as a radio navigation system. GPS consists of a constellation of at least 24 earth-orbiting satellites including 21 primary satellites and 3 in-orbit spares. A specific atomic frequency standard (rubidium or cesium oscillator) is installed in each orbiting satellite. These frequency standards are referenced to the United States Naval Observatory (USNO) and are traceable to NIST. The on-board rubidium and cesium oscillators of GPS satellites are steered from USDoD

ground stations and are referenced to the Universal Coordinated Time (UTC) maintained by USNO.

The GPS satellites broadcast on two carrier frequencies: L1 at 1575.42 MHz and L2 at 1227.6 MHz. GPS satellites broadcast a spread-spectrum waveform called a pseudo-random noise (PRN) code on L1 and L2, and each satellite is identified by the unique PRN code it transmits. There are two types of PRN codes. The first type is a coarse acquisition (C/A) code with a chip rate of 1023 chips per millisecond. The second type is a precision (P) code with a chip rate of 10230 chips per millisecond. The C/A code is broadcast on L1 and the P code is broadcast on both L1 and L2 [18, 25]. In a GPS receiver, the antenna must have a clear view of the sky to be able to acquire and track the broadcast signal of GPS satellites. The GPS receiver requires a clear view of the sky since GPS propagation is line-of-sight.

Once a GPS receiver is turned on, it searchers for the available GPS satellites that are visible from the antenna site. After that, the receiver computes its three-dimensional coordinate including latitude, longitude and altitude as long as at least 4 GPS satellites are available. Then, the receiver produces a frequency signal. By averaging data from multiple satellites, a receiver can reduce the frequency uncertainty at the output of GPS receiving equipment. [25]

Compared to the LF radio signal, GPS has lots of technical advantages. The GPS signals are easier to receive by using a less-expensive receiver. Additionally, the GPS coverage is much larger and the performance is better. However, the short-term stability of a GPS receiver is not good to calibrate a DUT during short amount of time. In a GPS-assisted calibration system, a measurement period of at least 24 hours is recommended when calibrating atomic frequency standards.

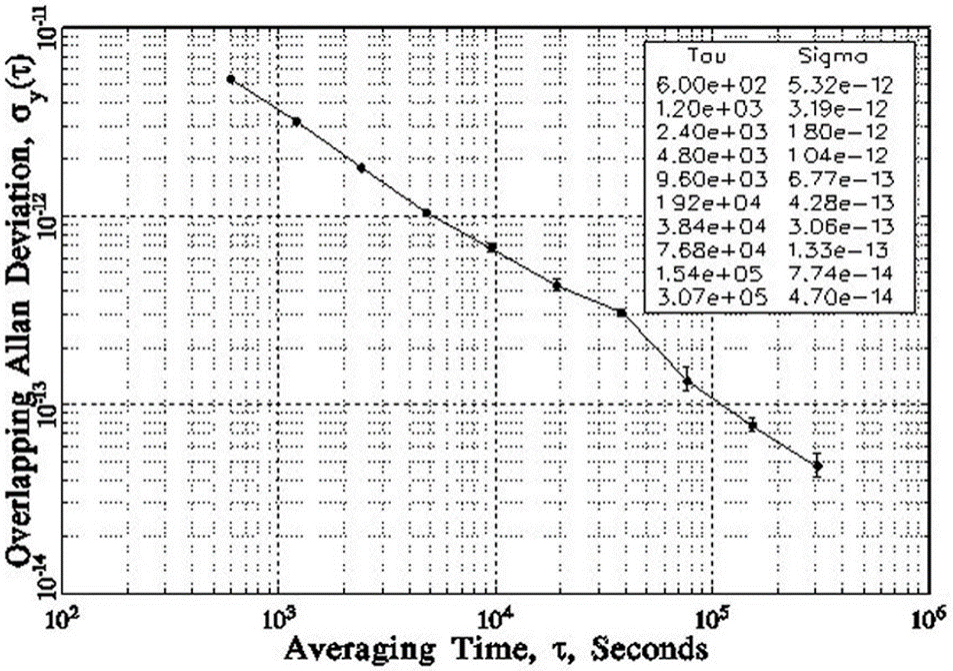

Figure 2.3: Allan Deviation of a Typical GPS Disciplined Oscillator

Chapter 3

Frequency Estimation Algorithms

3.1 Chapter Overview

As it will be discussed in the following sections of the thesis, an important part of the proposed frequency synchronization solution for small cell base transceiver stations is the frequency estimation subsystem that processes data samples of the signal of interest. In this chapter, different frequency estimation algorithms are discussed and the selected frequency estimator that is implemented in AMShare prototype is explained in details. This chapter is organized as follows. First a background review on the statistical signal processing required for frequency estimation of sinusoidal signals is presented. Then different DFT-based frequency estimators are compared. Finally, the algorithm of interest that is selected to be applied on the real-world signal samples is analyzed thoroughly.

3.2 Frequency Estimation Problem

Frequency estimation of sinusoidal signals is a fundamentally important non-linear parameter estimation problem arising in several applications. Estimating the frequency and phase of a sinusoidal signal corrupted by Additive White Gaussian Noise (AWGN) has been well-studied in the literatures of signal processing and communication systems. It has lots of applications including but not limited to the carrier recovery in a communication system, object localization in radar and sonar systems and carrier synchronization in distributed beam-forming wireless telecommunication systems. [27]

In recent years, numerous algorithms for estimating frequency and phase of a signal has been developed that can be categorized based on their estimation accuracy, processing latency and computational complexity. One of the fundamental tools that will facilitate the estimation of frequency and phase of a signal is Discrete-Time Fourier (DFT) transform or its efficient implementation Fast Fourier Transform (FFT). A big class of estimators use DFT as one of the main processing tools in their signal processing chain due to its relationship with Maximum Likelihood Estimation (MLE) of frequency. An ML estimator achieves the Cramer Rao Lower Bound (CRLB) which provides the best possible frequency estimation that leads to the minimum achievable estimation error over a wide range of Signal to Noise Ratios (SNR). Different ML estimators have been proposed in [5, 26, 29]. It must be noted that by increasing the number of DFT points in an estimation algorithm, the estimation accuracy can be improved but the latency introduced to the system may be prohibitive for using more DFT points. Several approaches have also been proposed that rely on the interpolation between DFT samples to increase estimation accuracy instead of increasing number of DFT points.

3.3 Complex Signal Model

The discrete-time signal model used in estimating the unknown parameters of a single tone corrupted by noise is as follows. It is assumed that the signal values is observed at discrete positions in time and the signal values along the vertical axis are continuous (no quantization is used)

$$z[n] = b_0 exp(j\omega_0 nT + \phi_0) + \eta[n] \tag{3.1}$$

for $n = n_0, n_1, ..., n_0 + N - 1$. In the above model, frequency ω_0 amplitude b_0 and phase offset θ_0 are unknown constants. n_0 indicates the index of first observed sample, T is the sampling period and η_0 is a zero-mean complex valued Gaussian random variable with $var\{Real\eta[n]\} = var\{Imag\eta[n]\} = \sigma^2$. It must be noted that the in-phase and quadrature noise components are uncorrelated. So it is assumed that $\eta[n]$ is independent identically distributed for $n = n_0, ...n_0 + N - 1$. Signal to Noise Ratio (SNR) is defined as $\frac{b_0^2}{2\sigma^2}$

In a parameter estimation problem, the discrete-time observations of the complex-valued signal is fed at the input of an estimator algorithm. The estimator then produces an estimate of the unknown parameters b_0, ω_0 and ϕ_0. If the estimated parameters for the amplitude, frequency and phase are shown as $\widehat{b_0}$, $\widehat{\omega_0}$ and $\widehat{\phi_0}$, respectively then the resulting estimation errors can be shown as $\overline{b} := b_0 - \widehat{b}$, $\overline{\omega} := \omega_0 - \widehat{\omega}$ and $\overline{\phi} := \phi_0 - \widehat{\phi}$.

3.4 Cramer-Rao Lower Bound

Finding the lower bound of the variance of an unbiased estimator is extremely useful in practical applications. It can actually provide a benchmark that the performance of any unbiased estimator can be compared against it. There are lots of such variance bounds in the literature However, the Cramer-Rao Lower Bound is the easiest one to determine. In this thesis, Cramer-Rao Lower Bound has been used as a reference of the performance of an estimator algorithm. CRLB is the best achievable performance bound for an unbiased estimator in terms of mean squared error cost function.

As described in [22], the CRLB for a vector parameter estimation problem in which more than one parameter of the signal is unknown is found by inverting the $[i, i]$ element of the Fischer information matrix defined as:

$$Var(\widehat{\theta}_i) \geq [\mathbf{I}^{-1}(\theta)]_{ii} \tag{3.2}$$

where $I(\theta)$ is defined as:

$$[I(\theta)]_{ij} := -E\left[\frac{\partial^2 lnp(x;\theta)}{\partial\theta_i \partial\theta_j}\right] \tag{3.3}$$

The θ represents the vector of unknown parameters and the $p(x;\theta)$ is the probability density function of the signal x which depends on the vector of unknown parameters θ.

The Fischer information matrix for the CRLB of the complex valued signal model 3.1 can be proved to be as follows:

$$I(\theta) := \frac{1}{\sigma^2}\begin{bmatrix} b_0^2T^2(n_0^2N + 2n_0^2P + Q) & 0 & b_0^2T(n_0N + P) \\ 0 & N & 0 \\ b_0^2T(n_0N + P) & 0 & b_0^2N \end{bmatrix} \tag{3.4}$$

where:

$$P := \sum_{n=0}^{N-1} n = \frac{N(N-1)}{2} \tag{3.5}$$

$$Q := \sum_{n=0}^{N-1} n^2 = \frac{N(N-1)(2N-1)}{6} \tag{3.6}$$

$$\theta := [\omega_0, b_0, \phi_0]^T \tag{3.7}$$

Using 3.2, the lower bound on the variance of the parameter estimator applied on the signal model 3.1 can be derived as follows:

$$Var\{\widehat{b_0}\} \geq \frac{\sigma^2}{N} \tag{3.8}$$

$$Var\{\widehat{\omega_0}\} \geq \frac{12\sigma^2}{b_0^2 T^2 N(N^2-1)} \tag{3.9}$$

$$Var\{\widehat{\phi_0}\} \geq \frac{12\sigma^2(n_0^2 N + 2n_0 P + Q)}{b_0^2 N^2(N^2-1)} \tag{3.10}$$

In the above section, the topic of parameter estimation and the lower bound on its performance was evaluated and described. However, estimating the parameters of a complex-valued signal is of no practical interest. In lots of practical situations, we need to estimate the parameters of a real valued sinusoidal signal. From now on, in the following sections of the thesis, we restrict our attention on the real valued signal model. To have a benchmark for comparing different parameter estimators, we need to derive the Cramer-Rao Lower Bound of the real signal model. The real signal model and the CRLB for sinusoidal parameter estimation are introduced in the following sections.

3.5 Real Signal Model

The real valued signal model that is actually a sinusoidal signal is defined as follows:

$$x[n] = A_0 cos(2\pi f_0 n + \phi_0) + \omega[n] \tag{3.11}$$

where $A_0 > 0$ and $0 < f_0 < \frac{1}{2}$. Additionally, $\omega[n]$ is the Additive White Gaussian Noise with zero mean and the variance of σ^2. It must be noted that the f_0 in the 3.11 is the relative frequency of the signal, meaning that it can be represented as $f_0 = \frac{f_{signal}}{f_{sampling}}$ where f_{signal} is the analog frequency of the signal and $f_{sampling}$ is the sampling frequency.

3.6 Cramer Rao Lower Bound for Real Sinusoidal Parameter Estimation

To derive the Cramer Rao Lower Bound of the sinusoidal parameter estimator, it is assumed that the f_0 in the signal model 3.11 is not near 0 or 1/2. This assumption helps to simplify the lower bound based on the following approximations [31]

$$\frac{1}{N^{i+1}} \sum_{n=0}^{N-1} n^i sin(4\pi f_0 n + 2\phi) \approx 0 \tag{3.12}$$

$$\frac{1}{N^{i+1}} \sum_{n=0}^{N-1} n^i cos(4\pi f_0 n + 2\phi) \approx 0 \tag{3.13}$$

for $i = 0, 1, 2$. To derive the Fischer information matrix, we use the following for the unknown vector parameter $\theta = [A_0, f_0, \phi_0]^T$:

$$[I(\theta)]_{ij} = \frac{1}{\sigma^2} \sum_{n=0}^{N-1} \frac{\partial x[n;\theta]}{\partial \theta_i} \frac{\partial x[n;\theta]}{\partial \theta_j} \tag{3.14}$$

Using the 3.14, the Fischer information matrix becomes:

$$I(\theta) = \frac{1}{\sigma^2} \begin{bmatrix} \frac{N}{2} & 0 & 0 \\ 0 & 2A^2\pi^2 \sum_{n=0}^{N-1} n^2 & \pi A^2 \sum_{n=0}^{N-1} n \\ 0 & \pi A^2 \sum_{n=0}^{N-1} n & \frac{NA^2}{2} \end{bmatrix} \tag{3.15}$$

The CRLB of the parameter estimator of the sinusoidal signal model can be derived using 3.2:

$$Var\{\widehat{A}\} \geq \frac{2\sigma^2}{N} \tag{3.16}$$

$$Var\{\widehat{f_0}\} \geq \frac{12}{(2\pi)^2 \eta N(N^2-1)} \tag{3.17}$$

$$Var\{\widehat{\phi}\} \geq \frac{2(2N-1)}{\eta N(N+1)} \tag{3.18}$$

Where η is the Signal to Noise Ratio (SNR) and is defined as:

$$\eta = \frac{A_0^2}{2\sigma^2} \tag{3.19}$$

3.7 Maximum Likelihood Estimation using Discrete Fourier Transform (DFT)

The Maximum Likelihood (ML) frequency estimator for the signal model 3.11 is as follows: [30]

$$\widehat{f}_{0ML} = Argmax|A(f)| \qquad f \in [0, \infty) \tag{3.20}$$

where $A(f)$ is defined as follows:

$$A(f) = \frac{1}{N}\sum_{n=0}^{N-1} x[n]exp(-jn2\pi fT) \tag{3.21}$$

Once the frequency estimate $\widehat{f}_{0ML}$ has been found using 3.20, phase and amplitude estimates can be computed as follows:

$$\widehat{\phi}_{0ML} = \angle\{exp(-j2\pi\widehat{f}_{0ML}n_0T)A(\widehat{f}_{0ML})\} \tag{3.22}$$

$$\widehat{A}_{0ML} = |A(\widehat{f}_{0ML})| \tag{3.23}$$

As discussed in [30], Discrete Fourier Transform (DFT) can be used to find the f that approximately maximizes $A(f)$. If we analyze 3.21, we notice that this equation can be derived by sampling $A(f)$ at discrete frequencies $f = \frac{k}{MT}$ for $k := 0, ..., M-1$. To calculate the values of $A(f)$ at different discrete frequencies, Fast Fourier Transform is usually used due to its lower complexity and faster processing. After calculating values of

$A(\frac{k}{MT})$, the index value at which sampled $A(f)$ attains its maximum is found. Then the frequency estimate can be computed as follows:

$$\widehat{f}_{0ML} = \frac{k_{max}}{MT} \tag{3.24}$$

Once the frequency estimate is found using 3.24, phase and amplitude estimates can be found using 3.22 and 3.23. According to [30], when all of the signal parameters are not known, the proposed estimation algorithm estimates the frequency at first. As seen in 3.22 and 3.23, phase and amplitude estimates are derived from the frequency estimate and as a result the estimation accuracy of frequency estimate can directly affect phase and amplitude estimations. To improve the estimation accuracy of the frequency estimator, a two-step search algorithm has been introduced in [30]. The first step is finding the index value corresponding to the maximum value of the DFT of the signal in 3.11. This step is called coarse search that results in a course estimate of the f_0. In the next step which is called fine-search, the local maximum that is closest to the coarse estimate is found. If the output of coarse search part is accurate, fine search estimator will provide the global maximum of $A(f)$ and as a result the whole two-step estimator can provide the best Maximum Likelihood estimate of the frequency. The secant method has been used in [30] as the fine-search algorithm and no details have been provided about its implementation approach. In the following sections different two-step frequency estimators are discussed.

3.8 DFT-Based Frequency Estimators

Maximum Likelihood Estimators are one type of parameter estimators that received lots of interest in different applications including but not limited to range and bearing estimation in radar and sonar applications as well as carrier frequency estimation in distributed beam forming wireless systems. An ML estimator is asymptotically efficient meaning that it achieves the Cramer Rao Lower Bound as the number of observations or signal samples gets large. Since the ML estimator is asymptotically efficient, it can be used in applications in which the parameters of a complex signal are unknown and the "true value" of the unknown parameters is needed. As discussed in the previous section, the frequency estimate of an ML estimator $\widehat{f}_{ML}$ requires numerical methods to be computed. In this chapter, five approximate ML estimators based on DFT and different fine search algorithms are introduced and their performance is evaluated with simulations. The ML estimators that are considered in this chapter are as follows:

1. Approximate ML estimator using DFT and no post processing
2. Approximate ML estimator using DFT and quadratic interpolation algorithm
3. Approximate ML estimator using DFT and Secant recursive method
4. Approximate ML estimator using DFT and Newton's recursive method
5. Approximate ML estimator using DFT and bisection method

AS it is evident in the above list, all of the ML estimators discussed in this chapter rely on DFT as part of their coarse search method.

3.9 Approximate ML Frequency Estimation using DFT and no Post Processing

As discussed in the previous section, the frequency estimate of the maximum likelihood estimator is

$$\widehat{f}_{ML} = Argmax|A(f)| \qquad\qquad f \in [0, \infty) \tag{3.25}$$

where

$$A(f) = \frac{1}{N}\sum_{n=0}^{N-1} x[n]exp(-jn2\pi fT) \tag{3.26}$$

It is evident that the M-point DFT is the sampled version of $A(f)$ at frequencies $f = \frac{k}{MT}$ for $k \in \{0, 1, 2, ..., M-1\}$ where T is the sampling period. Usually, the Fast Fourier Transform (FFT) is used to compute the $A(\frac{k}{MT})$. So the 3.25 can be rewritten as follows:

$$\widehat{k} = Argmax|A(\frac{k}{MT})| \tag{3.27}$$

where k belongs to the set $\kappa = \{0, 1, ..., M-1\}$. Once the $\widehat{k}$ is found using 3.27, the approximate frequency estimate $\widehat{f}$ can be found using the following equation.

$$\widehat{f} = \frac{\widehat{k}}{MT} \tag{3.28}$$

Finding the frequency estimate using the DFT as part of the coarse search algorithm requires an exhaustive search over a finite set of points in the discrete frequency domain.

3.10 Approximate ML Frequency Estimation using DFT and Quadratic Interpolation

In finding the peak of $|A(\omega)|$ using 3.27, the exhaustive search is done on the discrete set $\kappa = \{0, 1, ..., M-1\}$. As shown in Figure 3.1, the peak value of $A(\omega)$ can be located between two integer values of k. On the other hand, the DFT index cannot be a non-integer value. To improve the estimation accuracy, one approach is to interpolate the values near the DFT index at which the $A(\frac{k}{MT})$ attains its maximum value. One of the common interpolation algorithms is the Quadratic interpolation which has been used in the literature to improve the estimation accuracy in the phase and frequency estimation of single tone signals. [19] In the quadratic interpolation approach, a quadratic polynomial in the form of $y = a+bx+cx^2$ can be fitted on the discrete frequencies in the set $x \in \{\frac{\widehat{k}-1}{MT}, \frac{\widehat{k}}{MT}, \frac{\widehat{k}+1}{MT}\}$ and the DFT magnitudes given by $y = |A(x)|$. Once the polynomial parameters a, b and c are found, then the fine frequency estimate which is the peak of the quadratic function can be computed as $\widehat{f}_{Quad} = -\frac{b}{2c}$.

$$\begin{bmatrix} (\frac{\widehat{k}-1}{MT})^2 & \frac{\widehat{k}-1}{MT} & 1 \\ (\frac{\widehat{k}}{MT})^2 & \frac{\widehat{k}}{MT} & 1 \\ (\frac{\widehat{k}+1}{MT})^2 & \frac{\widehat{k}+1}{MT} & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} |A(\frac{\widehat{k}-1}{MT})| \\ |A(\frac{\widehat{k}}{MT})| \\ |A(\frac{\widehat{k}+1}{MT})| \end{bmatrix} \tag{3.29}$$

Once the 3.29 is solved for the parameters a, b and c, the fine frequency estimate which is the peak of the quadratic fit can be computed using $\widehat{f}_{Quad} = -\frac{b}{2c}$

3.11 Approximate ML Frequency Estimation using DFT and Secant Method

As discussed earlier, the maximum likelihood frequency estimation has a two step search routine. The first part is the called the coarse search in which the index of the maximum value of the DFT magnitude values is found. As it was discussed in 3.9, the estimation accuracy of the coarse search heavily depends on the number of FFT points. To improve the estimation accuracy, the coarse search is followed by a fine search method. The quadrature interpolation method discussed in 3.10 is one type of the fine search methods. The Secant method is another type of fine search methods that is discussed in this section along with its application to ML frequency estimation.

The Secant method is an iterative algorithm that is used to find the roots of an equation g. It is assumed that the domain and range of f is the real number set R. The iteration formula for the Secant method is:

$$x^{(m)} := x^{(m-1)} - g(x^{(m-1)})\frac{x^{(m-1)} - x^{(m-2)}}{g(x^{(m-1)}) - g(x^{(m-2)})}, \qquad m \geq 2 \tag{3.30}$$

For the Secant method to converge to the root of function f, the two initial values $x^{(0)}$ and $x^{(1)}$ must be chosen close enough to the desired root.

Finding the f that maximizes the $|A(f)|$ is equivalent to finding the root of the first order derivative of $|A(f)|$ or the root of the first order derivative of $|A(f)|^2$. The equation 3.21 can be written in the rectangular form as follows:

$$A(f) := B(f) + jC(f) \tag{3.31}$$

Where $B(f)$ and $C(f)$ are defined as follows:

$$B(f) := \frac{1}{N}\sum_{n=0}^{N-1} x[n]cos(2\pi nfT) \tag{3.32}$$

$$C(f) := \frac{-1}{N}\sum_{n=0}^{N-1} x[n]sin(2\pi nfT) \tag{3.33}$$

If we define $F(f)$ as follows:

$$F(f) := |A(f)|^2 = B(f)^2 + C(f)^2 \tag{3.34}$$

Then, finding the maximum value of $A(f)$ is equivalent to finding the root of the first derivative of $F(f)$. The first order derivative of $F(f)$ can be computed as:

$$F'(f) = \frac{dF}{df} := 2B\frac{dB}{df} + 2C\frac{dC}{df} \tag{3.35}$$

where $\frac{dB}{df}$ and $\frac{dC}{df}$ can be computed as follows:

$$\frac{dB}{df} := \frac{T}{N}\sum_{n=0}^{N-1} -nx[n]sin(2\pi fTn) \tag{3.36}$$

$$\frac{dC}{df} := \frac{-T}{N}\sum_{n=0}^{N-1} nx[n]cos(2\pi fTn) \tag{3.37}$$

To find the roots of the 3.35 using the Secant method by using 3.30 we can write the iterative relationship for $\widehat{f}^{(m)}$ as:

$$\widehat{f}^{(m)} = \widehat{f}^{(m-1)} - F'(\widehat{f}^{(m-1)})\frac{\widehat{f}^{(m-1)} - \widehat{f}^{(m-2)}}{F'(\widehat{f}^{(m-1)}) - F'(\widehat{f}^{(m-2)})} \tag{3.38}$$

In the Secant method we need the criteria that controls the number of iterations. If any of the following criteria is satisfied, the iteration should be stopped and latest $\widehat{f}^{(m)}$ should be provided at the output. The latest value found by iteration $\widehat{f}^{(m)}$ is the output of the fine search part in the Secant method and hence the approximate ML frequency estimate of the Secant method $\widehat{f}_{Secant}$.

3.12 Approximate ML Frequency Estimation using DFT and Newton's Method

The Newton-Raphson method which is usually called the Newton method is an algorithm to find the roots of a function $g : R \to R$. The iterative formula of the Newton's method is defined as follows:

$$x^{(m)} = x^{(m-1)} - \frac{g(x^{(m-1)})}{g'(x^{(m-1)})} \qquad m \geq 1 \tag{3.39}$$

where $g'(x)$ is the first order derivative of $g(x)$ with respect to variable x. To find the frequency f that maximizes $|A(f)|$ we can find the roots of the first order derivative of $F(f)$ which is defined in 3.34. In order to apply the Newton algorithm, we need to compute the second order derivative of $F(f)$ which can be calculated as follows:

$$\frac{d^2F}{df^2} = (2\pi)^2\{2(\frac{dB}{df})^2 + 2B\frac{d^2B}{df^2} + 2(\frac{dC}{df})^2 + 2C\frac{dC^2}{d^2f}\} \tag{3.40}$$

where $\frac{dB}{df}$, $\frac{dC}{df}$, $\frac{d^2B}{df^2}$ and $\frac{d^2C}{df^2}$ can be derived as follows:

$$\frac{dB}{df} := \frac{2\pi T}{N}\sum_{n=0}^{N-1} n(-x[n]sin(2\pi Tfn)) \tag{3.41}$$

$$\frac{dC}{df} := \frac{-2\pi T}{N}\sum_{n=0}^{N-1} n(x[n]cos(2\pi Tfn)) \tag{3.42}$$

$$\frac{d^2B}{df^2} := \frac{4\pi^2T^2}{N}\sum_{n=0}^{N-1} n^2(-x[n]cos(2\pi Tfn)) \tag{3.43}$$

$$\frac{d^2C}{df^2} := \frac{4\pi^2T^2}{N}\sum_{n=0}^{N-1} n^2(-x[n]sin(2\pi Tfn)) \tag{3.44}$$

So in the case of finding the maximizer frequency of $A(f)$ using the Newton's method, the iterative formula becomes

$$\widehat{f}^{(m)} = \widehat{f}^{(m-1)} - \frac{F'(\widehat{f}^{m-1})}{F''(\widehat{f}^{(m-1)})} \tag{3.45}$$

The stopping criteria for Newton's method are same as those mentioned for the Secant method in the previous section.

3.13 Approximate ML Frequency Estimation using DFT and Bisection Method

In the previous sections, two different iterative ML estimator were introduced. The Newton and Secant methods can track the CRLB over a wide range of SNR values however both of them have a common drawback. The estimator may end up finding the points that are not close enough to the peak of $A(f)$ function defined in 3.21. To guarantee the convergence of the estimator another method is introduced in this section which is called bisection method. To find the roots of a function g that are in the interval $[a, b]$ and by having the condition that $g(a)g(b) < 0$ the bisection method converges to the root value by dividing the expected range in every round of iteration. Finally, the best frequency estimate is the midpoint of the smallest range found. The absolute error of the function root after iterating over n steps can be calculated as follows:

$$|x^{(n)} - x^{(n-1)}| = \frac{|b-a|}{2^n} \tag{3.46}$$

To find the approximate ML frequency estimate using bisection method, first the the peak index of DFT is located as the coarse search method. Then the range bounds of the bisection method is initialized as $f^{(0)} = \dfrac{k-1}{MT}$ and $f^{(1)} = \dfrac{k+1}{MT}$. Since the two points are chosen on the two opposite sides of the approximate ML frequency estimate $\widehat{f}_{Bisection}$, the convergence is guaranteed. It must be noted that the stopping criteria for the bisection method are the same as those mentioned for the Secant and Newton methods.

3.14 Fine Resolution Frequency Estimator Based on DFT Interpolation in Windowed Data Samples

As discussed in the previous sections of this chapter, there are numerous methods and algorithms to estimate the frequency of a sinusoidal signal corrupted with noise and interference. As it was shown by simulations, the DFT with no post processing method requires huge number of DFT points to provide fine estimation of frequency. To overcome the inherent limitation of DFT with no post processing method, the idea of DFT interpolation was introduced and was shown that can provide fine estimates with lower DFT points with respect to DFT with no post processing approach. However, the quadratic interpolation can not follow the Cramer-Rao lower bound due to the inherent limitations of quadratic

formula. To improve the estimation accuracy, three iterative frequency estimators were introduced and it was shown that they can perform better in terms of mean squared error in the estimated frequency. The major drawback of Secant and Newton methods, is their convergence time. Generally, they are computationally intensive and depending on the coarse estimate value, they may require large number of iterations to converge to the final value. Additionally, in some cases they may not converge to a specific value due to the misplacement of coarse estimate value. The bisection method were introduced to overcome this limitation and it was discussed that the bisection method always converges to final estimate value with the cost of linear convergence. So, in a typical application bisection method should be used as a backup method to check the convergence of the problem while one of the other two iterative methods, namely Secant and Newton, are processing the data samples.

In the proposed frequency synchronization solution in this thesis, a quasi-real time with fine resolution algorithm is required to process the data samples of the signal of interest. It means that the data samples should be processed in the least possible amount of time considering the precision required by the specific application. In other words, the estimation errors in the digital signal processing section should be minimized for the synchronization method to be reliable in the real world scenarios. To fulfil the requirement of fine resolution and low computational complexity with near real time operation, a frequency estimator based on the interpolation of DFT samples is selected to be used in the AMShare prototype. The details of selected method are discussed in this section.

The frequency estimator that is selected to be used in AMShare prototype is mainly based on the estimation method given in [12] and [13]. The method described in [12] and [13] is a dichotomous frequency estimation algorithm like the estimation algorithms discussed earlier in this chapter. In the first stage, the N-point FFT of the input signal is calculated and the FFT bin with maximum magnitude is found. This stage is called coarse estimate. Once the coarse estimate of the frequency is found, it needs to be refined with a fine estimation stage. In the fine estimation stage, the fine part of the frequency is calculated using the FFT bin with maximum amplitude(k_c) and its immediate left and right neighbours, $k_c - 1$ and $k_c + 1$. The fine part of frequency $\widehat{\delta}$ is estimated using the following interpolation formula [12] [13]:

$$\widehat{\delta} = c_N Real\{\frac{X[k_c - 1] - X[k_c + 1]}{2X[k_c] - X[k_c - 1] - X[k_c + 1]}\} \tag{3.47}$$

Where c_N equals to $\frac{tan(\pi/N)}{(\pi/N)}$ for the rectangular window. In this method, the final

frequency estimate is the combination of coarse and fine estimation stages $\widehat{f} = 2\pi F_s(k_c + \widehat{\delta})/N$. Further details of this method can be found in references [12] and [13].

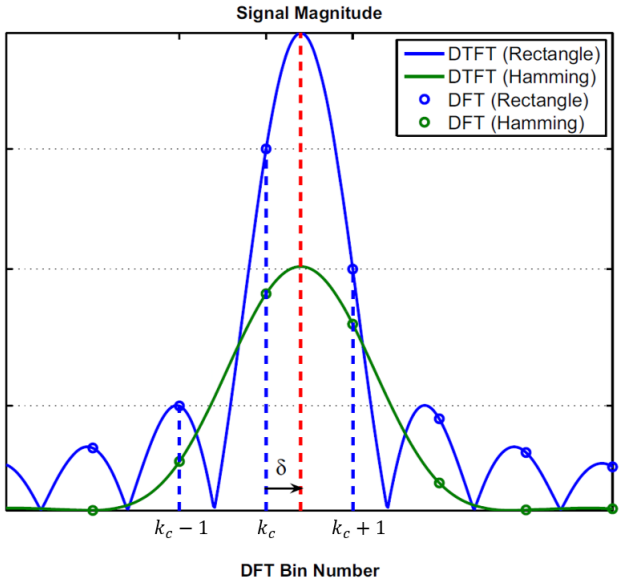

Figure 3.1: DFT Interpolation to Find the best Fractional Frequency δ

The major drawback of the estimation method in [12] and [13] is the requirement of FFT calculation with the rectangular window without any zero-padding. In many practical applications, it is necessary to apply a specific type of window function (i.e. Hamming or Blackman-Harris) on the data samples. For example, a properly selected window function may be required to suppress the interference caused by the unwanted components of the spectrum. [8] [9]. In the AMShare synchronization solution, a properly selected window function is required to be applied on the data samples to reduce the effect of neighbouring interference on the signal of interest. In other words, since the proposed synchronization solution relies on the spectrum to calculate the frequency estimate it is necessary to reduce the effects of interfering components on the desired signal. In the instrumentation and measurement literature, a number of frequency estimation methods with specific window types are proposed. [6,7,16,17]. Except the method proposed by Duda [16], all of the other estimation methods in the literature are derived for specific windows. Duda proposed a novel approach to that compensates the estimation bias resulting from applying the specific window function through a high order polynomial interpolation. The method used in implementing the AMShare prototype is the one proposed in [14]. Candan fine resolution frequency estimator is very similar to the Duda's method. The main advantage of Candan

work is its improved performance than Duda's method while having lower computational complexity.

In many real world applications, the signal of interest is observed in the presence of interfering signals. In such applications, the DFT should be calculated with a properly selected window function to reduce the effect of interfering signals on the frequency estimate. The method proposed in [14] uses the same dichotomous search method in [12] and [13] in which the N-point DFT of the signal $x[n]$ is calculated followed by a peak search in the magnitude spectrum. Then the fractional part of the frequency δ is estimated using formula 3.47. The method in [14] selects the bias-correction coefficient c_N such that the inherent bias of the estimator is reduced according to the specific type of window function applied on data samples. It then uses a novel approach by modulating the signal to the baseband and estimating the residual fractional frequency δ_2 that practically removes the estimation bias. The method proposed in [14] is selected to be the estimator in AMShare prototype due to multiple reasons. First of all, it practically removes the contribution of estimation bias from the mean squared error of the frequency estimate. Additionally, it selects the bias correction factor according to the specific window function by a simple method with low computational complexity. Furthermore, Candan method is a kind of DFT interpolation method that calculates the frequency estimate once the DFT samples are ready. In terms of practical constraints, this method can be used to track the frequency of signal of interest in real-time provided that the DFT points of incoming data samples are calculated in hardware to reduce the latency in the algorithm. Details of the selected method for AMShare prototype implementation are discussed in the next section.

3.14.1 Candan Fine Resolution Frequency Estimator

In the first stage of Candan method, the real valued window function $w[n]$ is applied on data samples $x[n]$ and then the resulting windowed data samples are transformed into discrete frequency domain as shown in 3.48.

$$X[k] = \sum_{n=0}^{N-1} w[n]x[n]e^{-j(2\pi/M)kn} \qquad k = \{0, 1, 2, ..., M-1\} \tag{3.48}$$

where M is the number of DFT points, which can be larger than the number of samples N if the samples are zero-padded.

In this method, the fractional frequency δ can be estimated by processing the DFT samples $X[k_c - 1]$, $X[k_c]$ and $X[k_c + 1]$ using the relation 3.47 when the value of c_N is

known. In [14], it has been shown that the bias correction factor can be selected as follows to reduce the bias for sufficiently small values of fractional frequency δ.

$$c_N = \frac{B_0^2}{A_1 B_0 + A_0 B_1} \tag{3.49}$$

where A_0, A_1, B_0 and B_1 are real valued window dependent constants and can be computed as follows:

$$A_0 = Imag\{f_w(1) - f_w(-1)\} \tag{3.50}$$

$$A_1 = f'_w(1) - f'_w(-1) \tag{3.51}$$

$$B_0 = 2f_w(0) - f_w(1) - f_w(-1) \tag{3.52}$$

$$B_1 = Imag\{2f'_w(0) - f'_w(1) - f'_w(-1)\} \tag{3.53}$$

and the window dependent function $f_w(\alpha)$ is defined as:

$$f_w(\alpha) = \sum_{n=0}^{N-1} w[n] e^{j(2\pi n M)\alpha} \tag{3.54}$$

where $w[n]$ is the window function applied on the data samples before DFT calculation.

As stated in [14], for sufficiently small values of δ, the proposed correction method practically removes the estimation bias. In other words, for small values of δ for example $|\delta| < 0.01$ the contribution of estimation bias in the mean squared estimation error is removed completely. In [14] a new approach is proposed to reduce estimation bias by appropriately selecting the operation point to be around $\delta \approx 0$ for all values of δ. The main idea is using the relation 3.47 twice to eliminate the contribution of estimation bias in the mean squared estimation error for all values of δ. The approach that is proposed in [14] and is implemented in the prototype of AMShare synchronization solution is shown in Table 3.1.

It must be noted that in the 4th step of the method proposed in [14], the input data samples $x[n]$ are modulated such that the discrete frequency of the signal after modulation

Algorithm Step	Description
1	Calculate the windowed M-point DFT of the data samples $x[n]$
2	Find the DFT index (k_c) with maximum magnitude of the spectrum ($0 \leq k_c \leq M-1$)
3	Estimate $\widehat{\delta_1}$ using $X[k_c]$, $X[k_c-1]$, $X[k_c+1]$ and the c_N calculated from 3.49
4	Compute modulated data samples $x_2[n] = x[n]exp(-j(2\pi/M)\widehat{\delta_1}n)$
5	Calculate the windowed M-point DFT of the modulated data samples $x_2[n]$
6	Estimate $\widehat{\delta_2}$ using $X_2[k_c]$, $X_2[k_c-1]$, $X_2[k_c+1]$ and the c_N calculated from 3.49
7	Provide the final fine resolution frequency estimate $\widehat{\delta} = \widehat{\delta_1} + \widehat{\delta_2}$ and $\widehat{f} = \frac{2\pi F_s}{M}(k_c + \widehat{\delta})$

Table 3.1: Algorithm Steps of the Frequency Synchronization Method Used in AMShare Prototype

becomes $k_c + (\delta - \widehat{\delta_1}) = k_c + \widehat{\delta_2}$ in terms of DFT bins. In the next steps 5 and 6, the residual component after modulation (δ_2) is estimated. It must be noted that since the true value of δ_2 is expected to be small, the estimated value $\widehat{\delta_2}$ is expected to be an accurate representation of δ_2 [14]. The final frequency estimate is then produced by combining the coarse frequency estimate k_c and the other two fractional frequencies $\widehat{\delta_1}$ and $\widehat{\delta_2}$. The overall implementation cost of the proposed method is the calculation of an M-point DFT, calculation of $\widehat{\delta_1}$ including 1 complex division and 1 real multiplication, recalculation of $X_2[k_c]$, $X_2[k_c-1]$ and $X_2[k_c+1]$ in the second iteration of algorithm which includes $3M+N4$ complex multiplications and finally calculation of $\widehat{\delta_2}$ which has 1 complex division and 1 real multiplication.

Chapter 4

Proposed Frequency Synchronization Solution for Small Cell Base Stations

As it was discussed in the previous chapters, there are numerous challenges and concerns in deployment of small cell synchronization solutions. Some of those challenges are briefly reviewed as follows:

- Deployment environment of small cell base station: As mentioned earlier, if the base station is planned to be deployed in indoor environments the GNSS-based synchronization solutions are not a feasible option any more. Due to high penetration loss of GNSS signals, their power level in indoor environments is not enough to be detected at GNSS receiver. Additionally, in dense urban environments with tall buildings, GNSS signals can not be received due to the canyon effect. It can be concluded that GNSS everywhere method is not a practical option for indoor small cell base stations.

- Reliability issues of network backhaul at cell site: The other existing synchronization solutions for small cell base stations are packet-based methods. SyncE is the frequency synchronizing solution that is currently used to distribute accurate frequency sources over the physical layer of Ethernet networks. This method is usually used as part of a hybrid solution beside GNSS-based method to improve its reliability and performance. The major drawback of SyncE is its strong dependence on the network backhaul architecture at the cell site. In indoor small cell sites like enterprise buildings and shopping malls, the network backhaul is usually operated by third party companies and their quality and architecture is not known to the mobile operator that provides services at the cell site. Additionally, the network backhaul of small

cell base stations is different from the macro cell base stations and that introduces errors in frequency synchronization between small cells base stations and macro cell base stations. It can be concluded that packet-based methods can not be a successful candidate for frequency synchronization in small cell base stations.

- Cost considerations: The other option for frequency synchronization in small cell base stations is using high quality stand alone clock sources like OCXOs or even Rubidium oscillators. Using such frequency sources with high performance in both short term and long term can guarantee compliance with strict standard requirements on frequency synchronization. Although this method seems as a technically feasible option, it can not be implemented in small cell deployments due to the economic considerations. At the heart of the business model for small cell deployments is providing an overlay network with low-cost and low-power base stations to improve the capacity and coverage of mobile network operator. Installing a high quality oscillator in every small cell base station will lead to huge CAPEX[1] costs for the operator and is not economically feasible.

Based on the aforementioned considerations in small cell deployments, there is a strong need for a new low-cost and low-power synchronization solution to address the standard requirements on frequency synchronization of small cell base stations. The method should be able to be used everywhere regardless of the deployment environment and can be combined with existing synchronization solutions to provide a hybrid method. Additionally, it should not depend on the specific cell site features like the quality of network backhaul.

In this chapter a new frequency synchronization solution is proposed to be used in small cell base stations. The overall method is explained and the system model is provided. The pros and cons of the solution are discussed and its applicability to small cell deployments is studied.

[1] CAPital EXpenditure

4.1 Proposed Frequency Synchronization Solution

The proposed synchronization solution in this thesis utilizes the idea of over the air clock as its base of operation. In this method, the carrier component of the broadcast AM wireless signals is used as the common sinusoidal signal to be used as frequency reference in different small cell base stations. Each base station, then listens to this shared AM signal and continually compensates for the CFO[2] at its RF[3] circuitry for the air interface standard. Since different base stations are locked to a common AM signal and track its behaviour over time, the proposed method is called AMShare from now on in this thesis.

AMShare has lots of advantages over the state of the art synchronization solutions. First of all, it can address the challenges in indoor small cell deployments as well as in dense urban environment. The frequency band of commercial AM broadcast signals covers part of MW[4] band from 540 KHz to 1600 KHz. MW signal radiates from the transmitter antenna in two different components, namely ground-wave component and sky-wave component. The ground-wave radiation of AM signals as part of MW band, makes them an ideal candidate for frequency synchronisation in indoor environments and dense urban areas since they can be easily received and detected inside the buildings and between tall terrain. Additionally, the carrier component of an AM broadcast signal contains half power of the transmitter and the other remaining power is distributed over the bandwidth of the audio signal modulated by the carrier component which is 10 KHz. So, the carrier component can be easily detected and separated from the AM audio source. In AMShare synchronization solution, the carrier component is proposed to be used as the common over the air clock signal for CFO compensation in small cell base stations.

4.2 Signal Model in AMShare Synchronization Method

The received signal at the small cell base station can be modelled as follows:

$$s(t) = B(t)A(1 + \alpha m(t))cos(2\pi f_c t + \phi_0 + \theta(t)) + \omega(t) \tag{4.1}$$

Where $B(t)$ and $\theta(t)$ represent the channel effects on the transmitted signal. A represents the signal amplitude at the transmitter output. $m(t)$ is the audio signal that

[2]Carrier Frequency Offset
[3]Radio Frequency
[4]Medium Wave

modulates the carrier signal centred at f_c with a bandwidth of 4 kHz. α represents the modulation index at the transmitter and ϕ_0 shows the initial phase of the carrier signal at the transmitter. Additionally, $\omega(t)$ is the Additive White Gaussian Noise (AWGN) with zero mean and variance of σ^2. It must be noted that there is a guard interval of 1 kHz for different commercial AM radio stations so the total bandwidth of a specific AM radio station is 10 kHz.

To compare different components in the received signal, model 4.1 can be rewritten as follows:

$$s(t) = B(t)Acos(2\pi f_c t + \phi_0 + \theta(t)) + \alpha B(t)m(t)cos(2\pi f_c t + \phi_0 + \theta(t)) + \omega(t) \qquad (4.2)$$

In AMShare method, our main focus is on the first component in the signal model 4.2. The first component contains the information about the frequency of transmitter. Specifically, in the proposed synchronisation solution the useful information is embedded in the frequency of carrier component instead of the amplitude of carrier as opposed to the traditional AM radio receivers.

4.3 System Model of AMShare

AMShare synchronization solution comprises of several parts including signal amplification, filtering, signal acquisition and digital signal processing. The system model of the proposed method is shown in Figure 4.1.

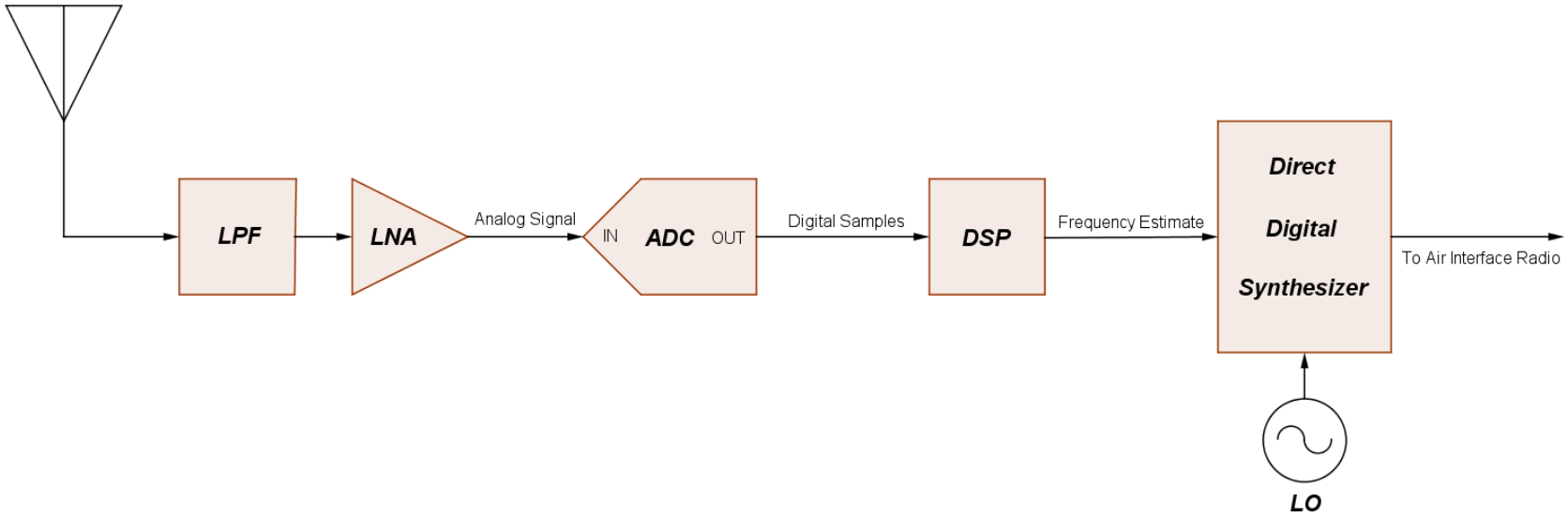

Figure 4.1: System Model of the AMShare Frequency Synchronization Solution

As it is shown in Figure 4.1, the proposed method comprises of an analog front end and a digital subsystem. The analog front end includes the antenna and RF front end circuits. At the intersection of analog and digital domains, there is an ADC[5] that converts the analog signal from the output of analog front end to the digital data to be processed on a digital signal processor. Details of each block in the block diagram and the interfaces are discussed in the next chapter.

The AMShare method works as follows:

1. The received signal from the antenna is amplified using low noise amplifier with excellent noise figure performance
2. A low pass filter is applied on the amplified signal to attenuate out of band components and interferers at the analog domain
3. The whole spectrum from DC up to the cut off frequency of low pass filter is sampled and digitized using a high precision analog to digital converter (ADC)
4. Digital samples are fed through the estimator algorithm that runs on a digital signal processor or a host computer
5. The estimator algorithm provides an estimate of the frequency of carrier component embedded in the received AM signal
6. The estimate frequency is provided to a DDS[6] to compensate for the CFO[7] at the air interface radio

It must be noted that the block diagram shown in , is for one base station that listens to the shared AM signal. The same structure should be used for the other base stations in AMShare synchronization solution. In other words, all of the small cell base stations deployed in an area perform the same set of analog and digital operations on the received signal to compute the estimate frequency of the carrier component of AM signal. Then, different base stations can use the computed estimate for compensating for the CFO at their air interface radio.

[5] Analog to Digital Converter
[6] Direct Digital Synthesizer
[7] Carrier Frequency Offset

Chapter 5

Hardware Implementation and Observations

5.1 Chapter Overview

In the previous section, the system model of AMShare was discussed and its mechanism of operation was explained. To verify the AMShare capabilities in a real world scenario, a prototype based on the proposed idea has been implemented and tested. In other words, the AMShare idea has been applied on real signals by designing and implementing a complete hardware prototype. In this chapter, details of hardware implementation for testing the AMShare performance are discussed. Additionally, observations that are a result of implemented hardware and software components are discussed in this chapter.

To evaluate the performance of AMShare on the real signals and to verify its performance in real world scenarios, the system model proposed in is implemented for two completely separate receiver prototypes. In other words, the implementation details discussed in this chapter applies to both receiver chains. Two separate chains in this prototype represent two separate receiver circuitry of the synchronization subsystem embedded in the small cell base station.

5.2 Antenna and RF Front End

In this section, the implementation details of the antenna and RF front end that used for signal reception are discussed.

5.2.1 Antenna

For prototype purposes, we have decided to use a passive loop antenna with tuning capability for better out-of-band rejection at the reception point. The antenna used in the prototype is a loop antenna with 9 inches of diameter by Tecsun. Tecsun AN-100 is a tunable small form factor antenna that is designed for MW band.

Figure 5.1: Tecsun AN-100 Tunable Loop Antenna

5.2.2 RF Front End: LPF and LNA

The RF front end of the implemented prototype is comprised of a low pass filter, a three stage low noise amplifier and two attenuators that are used to prevent low noise amplifiers from entering saturation region.

The low pass filter that is used in the prototype is a coaxial filter by Mini-Circuits. The filter pass band covers DC to 2.5 MHz with a loss of less than 1 dB. 3 dB cut-off frequency of the used filter is $f_{co} = 2.75$ MHz. The stop-band covers 3.8 MHz to 5.0 MHz with more than 20 dB loss and 5.0 MHz to 200 MHz with the loss of more than 40 dB.

Figure 5.2: Mini-Circuits SLP-2.5+ Coaxial Low Pass Filter

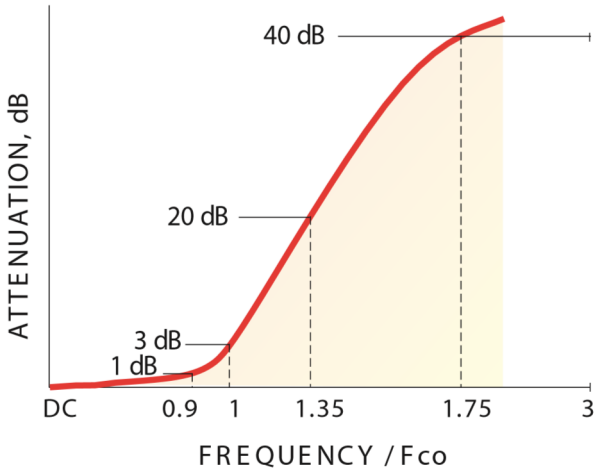

Figure 5.3: Frequency Response of SLP-2.5+ Filter

As part of the analog front end of AMShare prototype, the low noise amplification of the received signal is done in three stages. In each stage, the low noise amplifier ZFL-1000LN+ by Mini-Circuits is used. At every stage the input signal is amplified with a gain of about 23 dB. The gain value is set by the level of supply voltage. In the prototype,

supply voltage of 15 V is used for amplifiers and gain is set to about 23 dB at each amplifier stage. It muse be noted that the typical noise figure at each stage is about 2.9 dB. The typical performance curves of the low noise amplifiers are shown in the following figures.

Figure 5.4: Mini-Circuits ZFL-1000LN+ Low Noise Amplifier

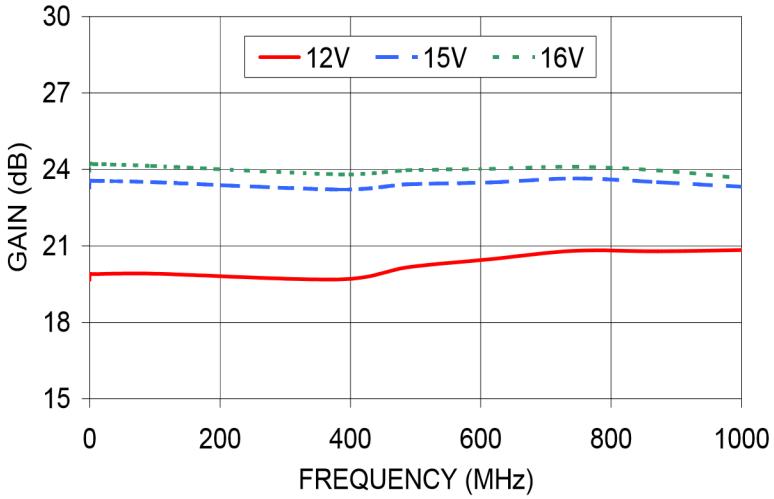

Figure 5.5: Gain vs. Frequency Curve of ZFL-1000LN+ for Three Different Supply Voltage Options

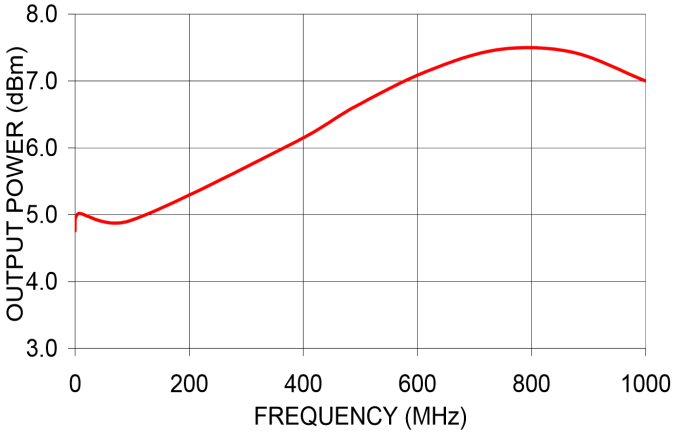

Figure 5.6: Output Power of ZFL-1000LN+ at 1 dB Compression

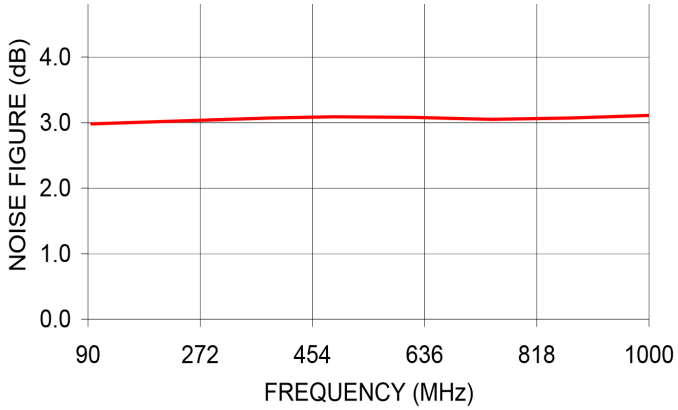

Figure 5.7: Noise Figure of ZFL-1000LN+ at Supply Voltage of 15 V

It must be noted that two coaxial attenuators are used between three LNA stages to prevent each stage from saturation. An attenuator of 3 dB is used between the first two stages and another attenuator of 10 dB is used between the last two stages of amplification. So the total gain of the RF front end in the prototype can be calculated as shown in 5.1. It must be noted that the loss due to coaxial cables can is neglected in the gain computations of amplification stage since the operating frequency is part of the MW band and the coaxial cable loss in this band is negligible.

$$G = 3 \times 20dB - 3dB - 10dB = 47dB \tag{5.1}$$

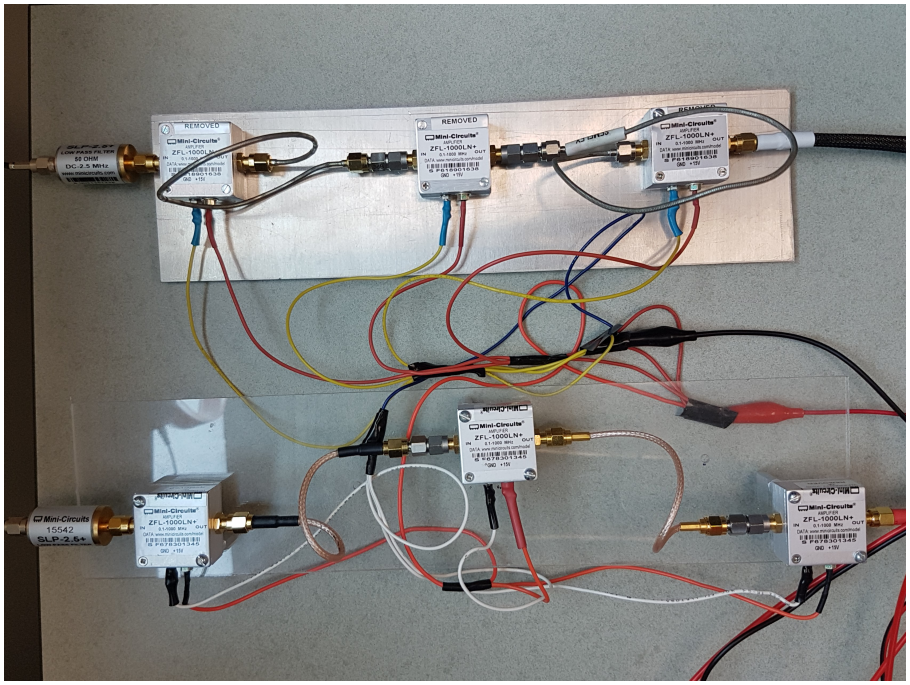

Figure 5.8: Implemented Prototype of the RF Front End Section for Two Separate Receiver Chains

5.3 Analog to Digital Conversion

Once the received signal is passed through a low pass filter and amplified by three LNAs, it is ready to be converted to digital domain for further processing. To convert the analog signal at the output of the analog front end section to digital samples, three different hardware platforms have been used in this research. The first two platforms are the commercial off the shelf circuits and evaluation boards by Analog Devices and the last one is two real time digital oscilloscopes by Rohde & Schwarz (R & S). Each of these platforms are discussed in detail in the following sections.

5.3.1 Analog Devices AD7760 and CED-1Z Platform

The first analog to digital conversion platform that is used to implement the prototype of AMShare is the combination of the AD7760 high precision 24 bit sigma-delta ADC and the CED-1Z converter and evaluation FPGA board. The CED-1Z board is a platform from

Analog Devices that is intended for use in evaluation, demonstration and development of systems that use Analog Devices precision converters as part of their conversion platform. In the first version of AMShare prototype, the CED-1Z FPGA board acts as the controller and data acquisition platform between the converter board and the host PC or digital signal processor. It must be noted that it transmits and receives data over the the USB link.

AD7760 is a 24-bit sigma-delta analog to digital converter (ADC) from Analog Devices. It has an excellent performance of 100 dB SNR at 2.5 mega samples per second due to its sigma-delta architecture. That means it is an ideal solution for high speed data acquisition systems that demand for high precision data samples. Additionally, it has a wide dynamic range with reduced anti-aliasing requirements that simplifies the design process. AD7760 is the ideal analog to digital conversion solution for applications that require high SNR without a complex front-end signal processing design.

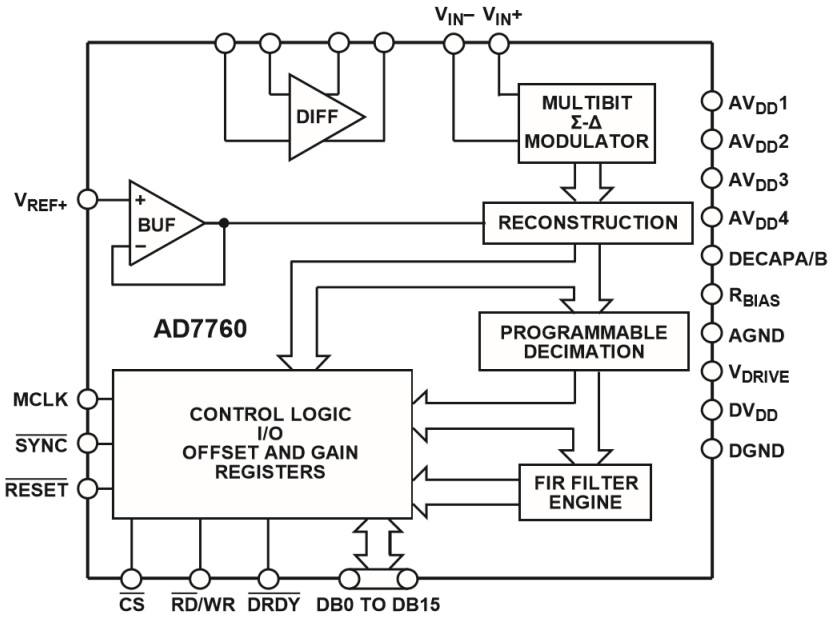

Figure 5.9: Functional Block Diagram of AD7760 High Precision ADC

As described in the datasheet of AD7760, it is a useful approach to drive the ADC with differential signals to get the maximum performance from the converter. In the AMShare prototype, the circuit shown in Figure 5.10 has been used to convert the bipolar single-ended signal swinging around GND to two differential signals that drive AD7760. It must

be noted that the recommended circuit has been implemented using available components in the CST research lab to convert the single-ended signal from the RF front end to the differential outputs that drive AD7760. The circuit built for single-ended to differential conversion is shown in Figure 5.11.

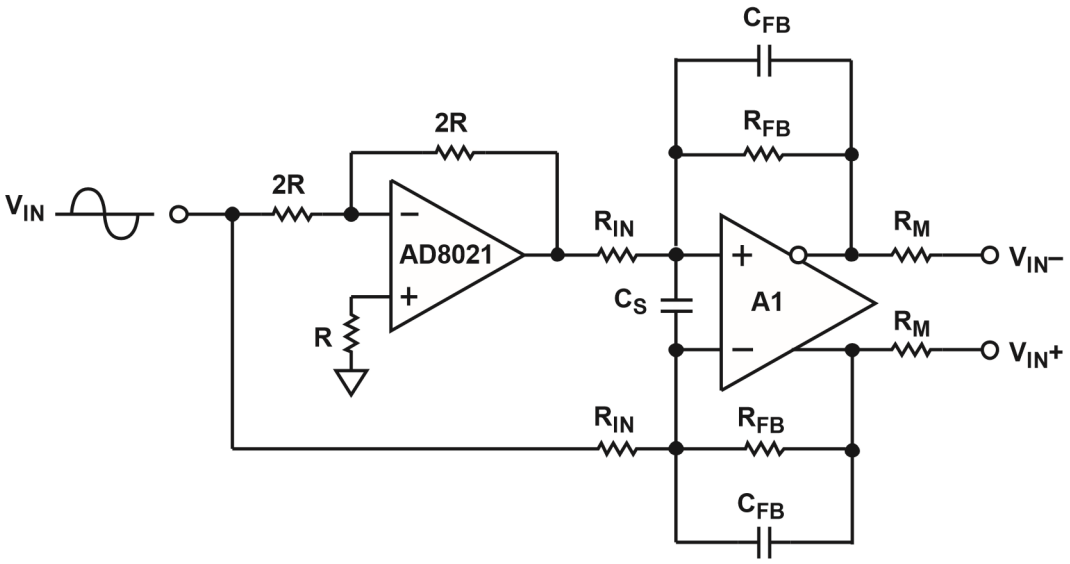

Figure 5.10: Recommended Circuit for Single-Ended to Differential Conversion

To exploit the full feature capabilities of AD7760 and reduce any points of uncertainty in the prototype design and implementation process, the Analog Devices board for evaluating AD7760 has been used in the AMShare prototype. There is a FPGA device on AD7760 board that acts as an intermediate device to interface with CED-1Z evaluation and development platform. The combination of AD7760 evaluation board and CED-1Z platform along with the software provided by Analog Devices prepares a complete data acquisition platform that let the user to upload the acquired samples to a host computer via the USB link. The block diagram of the complete data acquisition subsystem used as the first data conversion platform in AMShare prototype is shown in Figure 5.12.

As shown in Figure 5.12, the Analog Devices CED-1Z evaluation and development platform has been used to interface the AD7760 evaluation board to the host computer. Functional block diagram of CED-1Z board is shown in Figure 5.13.

It must be noted that the AD7760 evaluation board is connected to the CED-1Z platform via PPI headers. Additionally, the CED-1Z is responsible for communicating with AD7760 to trigger sample acquisition and stores them on the on-board buffer. It then

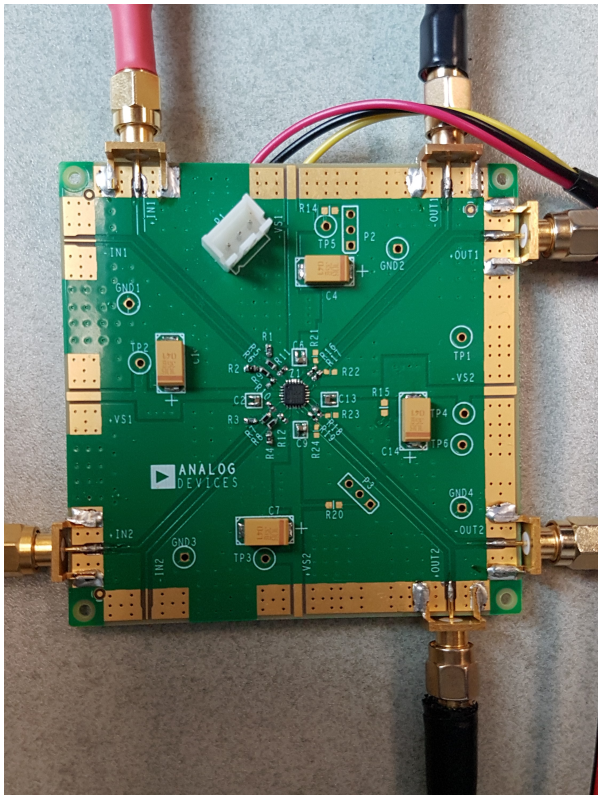

Figure 5.11: Single-Ended to Differential Circuit Implemented in CST Research Lab

transfers the acquired digital samples to host computer via the USB interface. In AMShare prototype, we have used the platform shown in Figure 5.12 to acquire data samples over different measurement periods via the host computer connected to CED-1Z with a USB 2.0 cable. It is worth mentioning that CED-1Z board has three 90-way connectors that can directly connect the CED-1Z to Analog Devices Blackfin EZ-Kit. By combining the CED-1Z with Blackfin EZ-Kit, a very powerful standalone system can be implemented in which acquired data samples can be processed on Blackfin DSP instead of the host computer.

In the first version of AMShare prototype, short-term stability of the carrier component embedded in the broadcast AM signal has been measured. Details of the selected broadcast station are discussed in. Results and observations of the first AMShare prototype based on AD7760 acquisition platform are as follows.

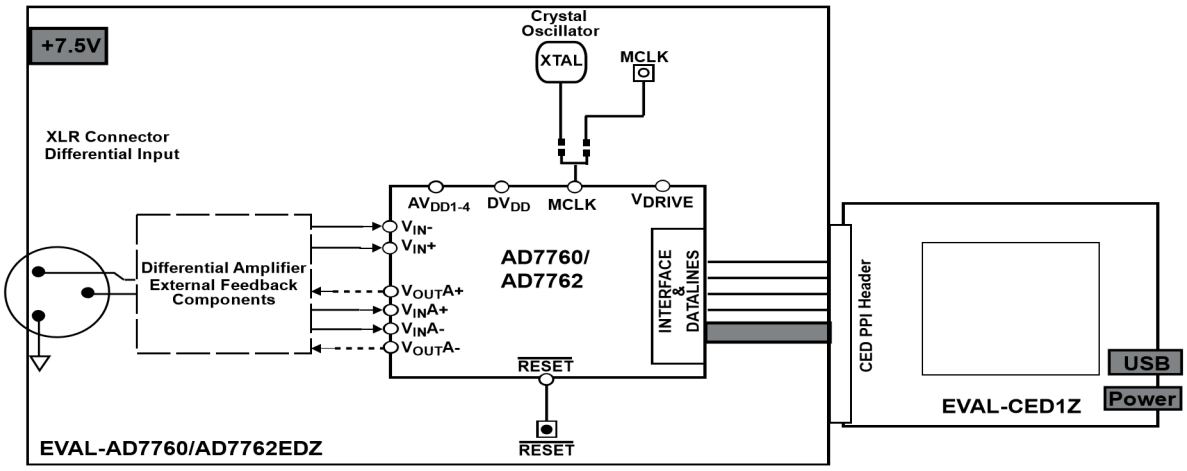

Figure 5.12: Block Diagram of Complete Data Acquisition System Including AD7760 Evaluation Board and CED-1Z Platform

5.3.2 Analog Devices AD7626 and SDP-H1 Platform

In the second version of AMShare prototype, a combination of AD7626 evaluation board and SDP-H1 high speed controller board is used.

AD7626 from Analog Devices is a charge redistribution successive approximation register (SAR) based analog to digital converter with 16 bits of resolution and throughput of 10 mega samples per second. AD7626 has unmatched performance in noise with an SNR of 91.5 dB and has excellent linearity over its entire operation range (INL[1] of ±45 LSB and DNL[2] of ±0.35 LSB).

AD7626 accepts differential inputs IN+ and IN- and samples the voltage difference between them at the CNV edge. It has the capability of converting 10 mega samples per second from its differential analog inputs to 16-bit digital data. Its power consumption is typically around 136 mW and the device can work with both 5 V and 2.5 V power supply voltages. The functional block diagram and the internal schematic of AD7626 are shown in 5.14 and 5.15 respectively. It muse be noted that AD7626 uses LVDS signalling in its digital interface to a host board for enabling high data transfer rates. Also, it is worth mentioning that it uses a 4.096 V as reference voltage. AD7626 converts the differential voltage of its analog inputs (IN+ and IN-) to a digital output word. The analog inputs IN+

[1]Integral Non Linearity
[2]Differential Non Linearity

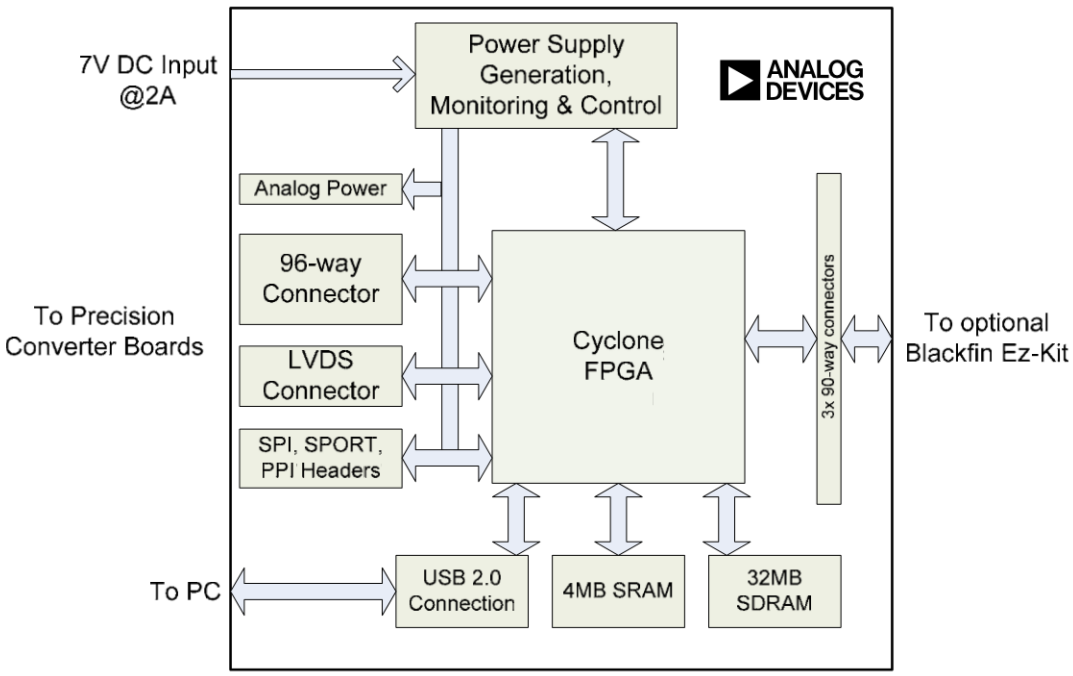

Figure 5.13: Functional Block Diagram of Analog Devices CED-1Z Evaluation and Development Platform

and IN- need to be biased around $V_{REF}/2 = 2.048V$ as the common mode voltage for the converter to work properly. This kind of differential analog input architecture causes the common signals to both inputs such as in-band interferences to be rejected. The output word from AD7626 is in twos complement format with MSB first.

In the AD7626, all of analog to digital conversions are manipulated by two signals CNV+ and CNV-. They can be either a CNV+/CNV- LVDS signal or a 2.5 V CMOS logic signal that is only applied to CNV+ pin of the converter. The conversion process is initiated on the rising edge of CNV± signal. Additionally, it requires a sequential set of operations to be done for acquiring samples from the digital interface of AD7626. To reduce the development time and improve the overall system reliability, the evaluation board of AD7626 is used in the AMShare prototype along with SDP-H1 controller board. By using this complete platform, the host computer can control the evaluation through the SDP-H1 platform over the USB link. To further amplify the input signals for improved converter performance, the differential amplifier (ADA4932-1) installed on the AD7626 evaluation platform is configured to be in the input signal path. It must be noted that

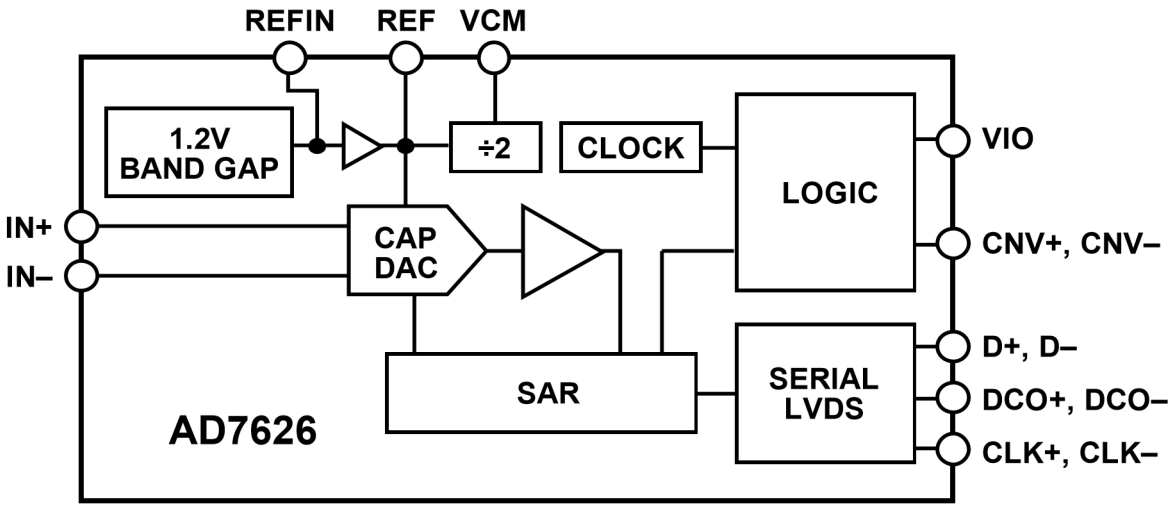

Figure 5.14: Functional Block Diagram of Analog Devices AD7626 16-bit 10 MSPS ADC

AD7626 evaluation board connects to SDP-H1 boards via the 160 pin FMC[3] connector. The complete data acquisition platform that is used in the second version of AMShare prototype is shown in Figure 5.16. The functional block diagram of the platform is shown in Figure 5.17.

The SDP-H1 controller board that is used in the second version of AMShare prototype, is part of the Analog Devices system demonstration platforms. It has a Xilinx Spartan-6 FPGA that is responsible for communication with the AD7626 evaluation board via an FMC low pin count connector. SDP-H1 is an ideal solution for system engineers who seeks to develop a demonstration platform or prototype in the shortest possible time. The SDP-H1 connects to the host computer via a USB 2.0 high speed connection allowing users to communicate with the board and upload the acquired samples to the host computer using the USB connection. There is also an ADSP-BF527 Blackfin processor on the SDP-H1 board that provides the USB controller for the board and allows the user to configure the FPGA via the USB connection. Specifically, it acts as an interface connecting the host computer to the converter evaluation board.

Data acquisition subsystem of the second version of AMShare that is built and tested in CST research lab is shown in Figures 5.18 and 5.19.

[3]FPGA Mezzanine Card

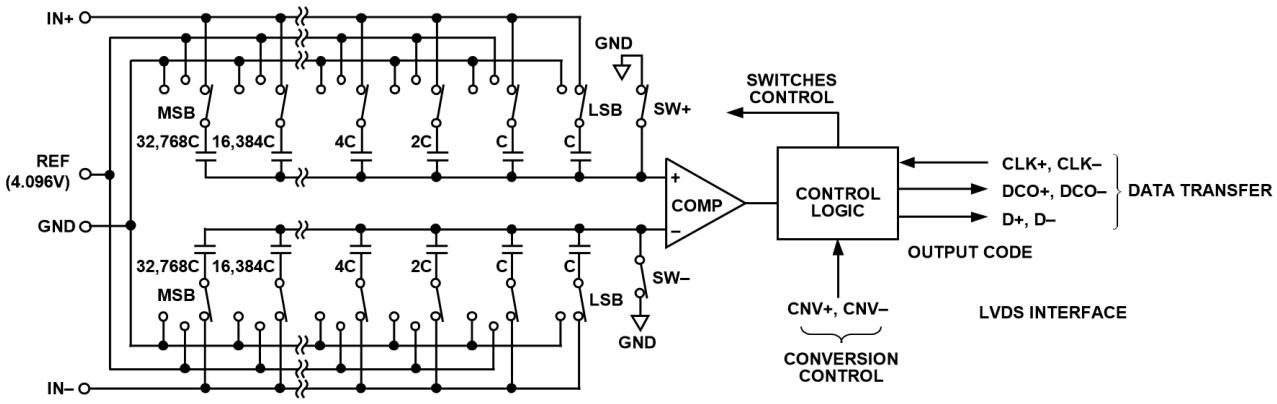

Figure 5.15: Equivalent Schematic of Analog Devices AD7626 Analog to Digital Converter

5.3.3 Rohde & Schwarz RTO 1044 and RTE 1034 Real Time Digital Oscilloscopes

In the third version of AMShare prototype, a combination of two real time digital oscilloscopes from Rohde & Schwarz has been used as data acquisition platform. The main reason to use R& S real time digital oscilloscopes is their well developed hardware drivers and API functions that allows the user to write C programs on the host computer to control both devices and acquire samples based on the specific needs. In the third version of AMShare prototype, a comprehensive C program has been developed in National Instruments LabWindows CVI development environment that controls the devices connected to the computer and acquires real time data samples based on specific needs (i.e. sampling rate, time interval between measurements). It must be noted that both digital oscilloscopes are connected to the host computer via the LAN connection. Additionally, a LAN hub is used in between to connect the instruments to host computer. A simplified schematic of the third version of AMShare is shown in Figure 5.20. Additionally, the snapshots of both devices during the measurements are shown in Figures 5.21 and 5.22.

5.4 Different AMShare Prototypes At a Glance

As it was discussed in the previous sections, three different prototype versions of AMShare has been implemented to evaluate the performance of system in real world scenarios. To understand differences between three versions of AMShare prototype, their main features and specifications are briefly mentioned in Table 5.4.

Figure 5.16: Complete Data Acquisition Platform Comprising AD7626 Evaluation Board and SDP-H1 Controller Board

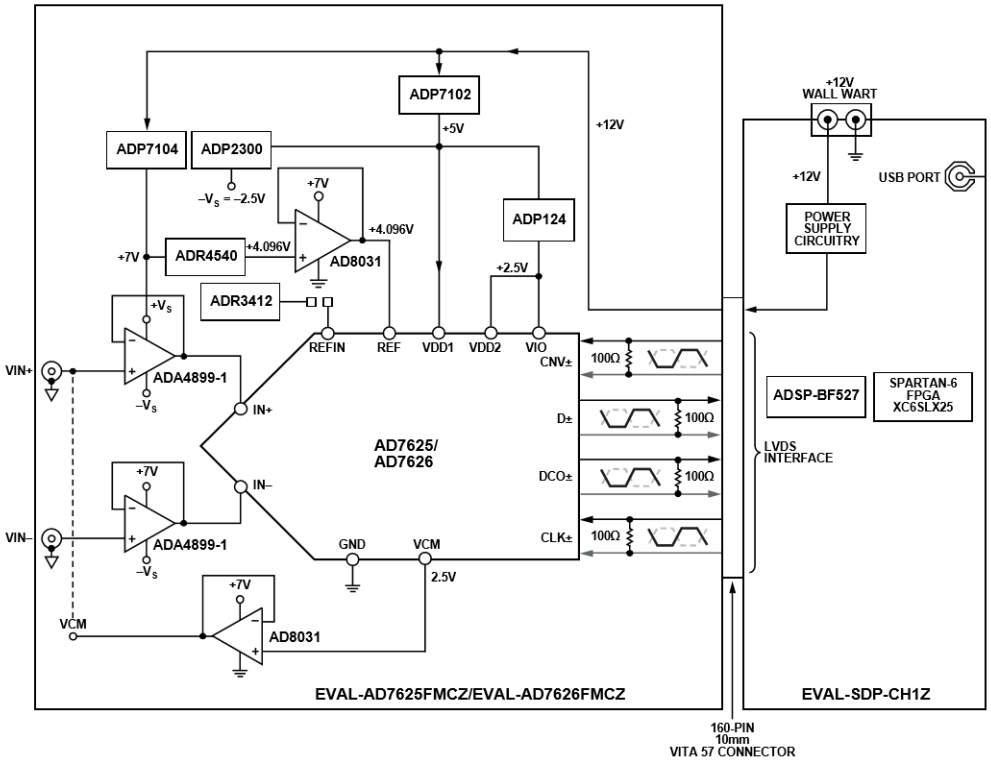

Figure 5.17: Functional Block Diagram of Data Acquisition Platform Used in Second Version of AMShare Prototype

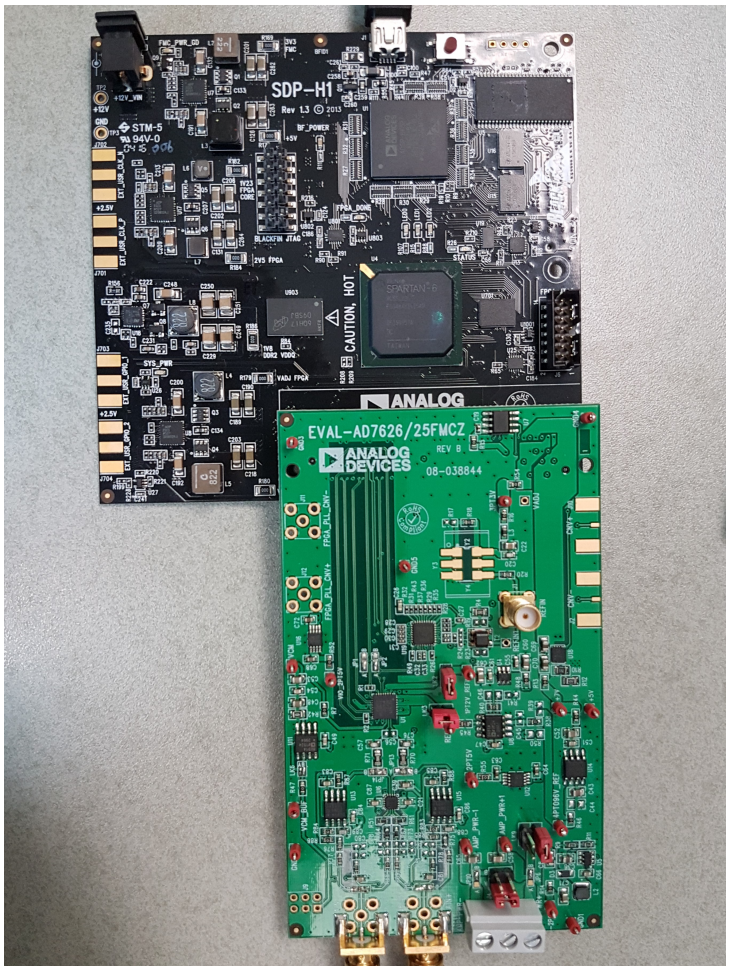

Figure 5.18: Data Acquisition Platform using AD7626 Evaluation Board and SDP-H1 Controller Board

Figure 5.19: Complete Data Acquisition Platform of AMShare for Two Different Receiver Chains

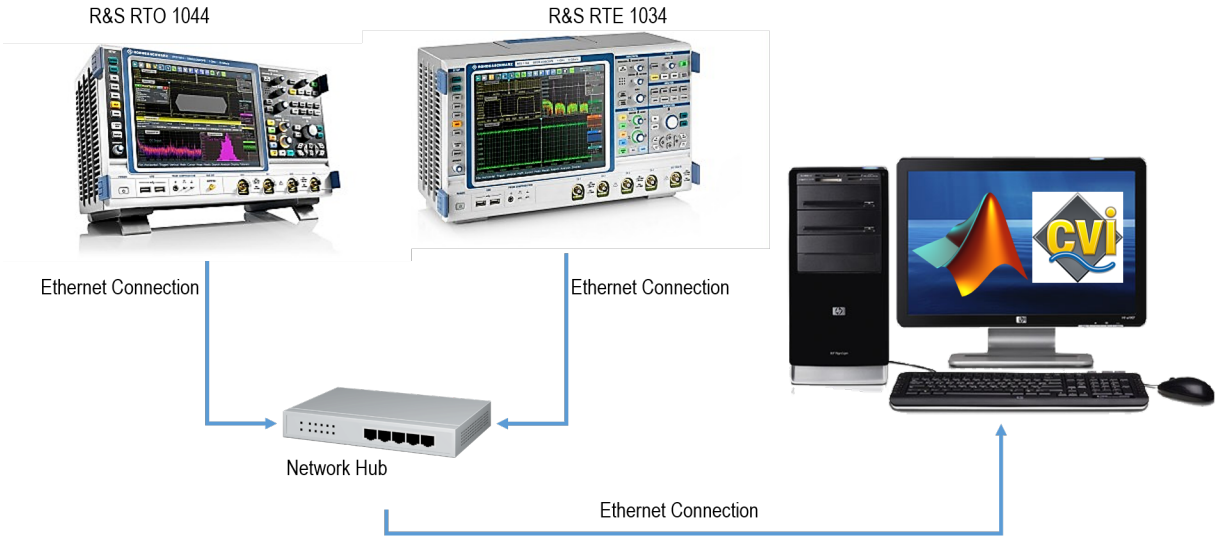

Figure 5.20: Schematic of the Complete Data Acquisition Platform of Third Version of AMShare Comprising R & S RTO 1044 and RTE 1034

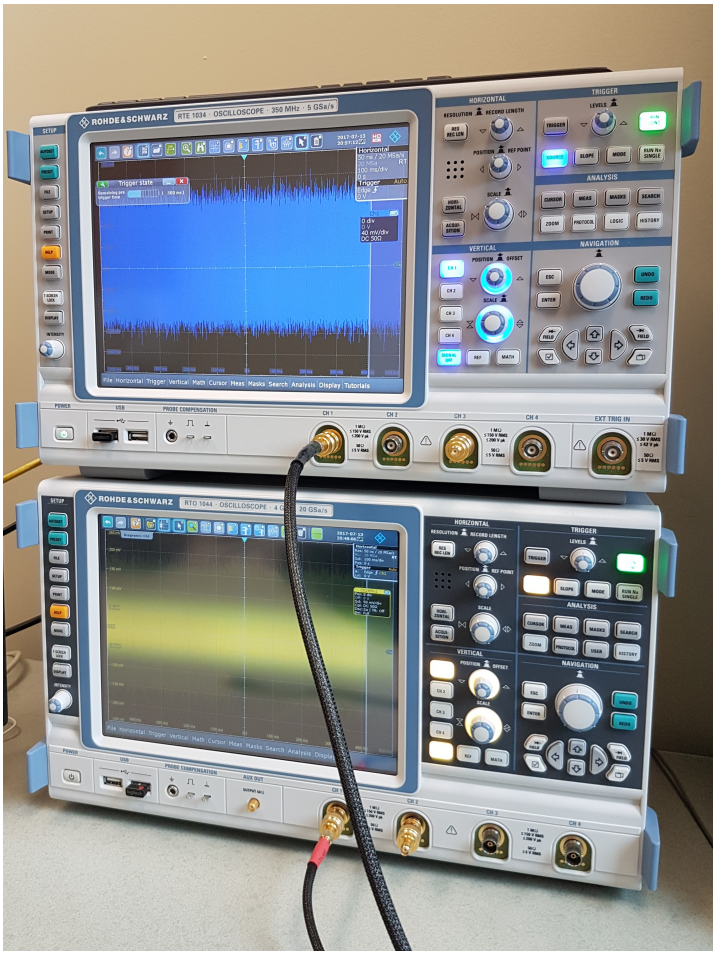

Figure 5.21: Real-Time Digital Oscilloscopes RTO 1044 and RTE 1034 In Data Acquisition Process

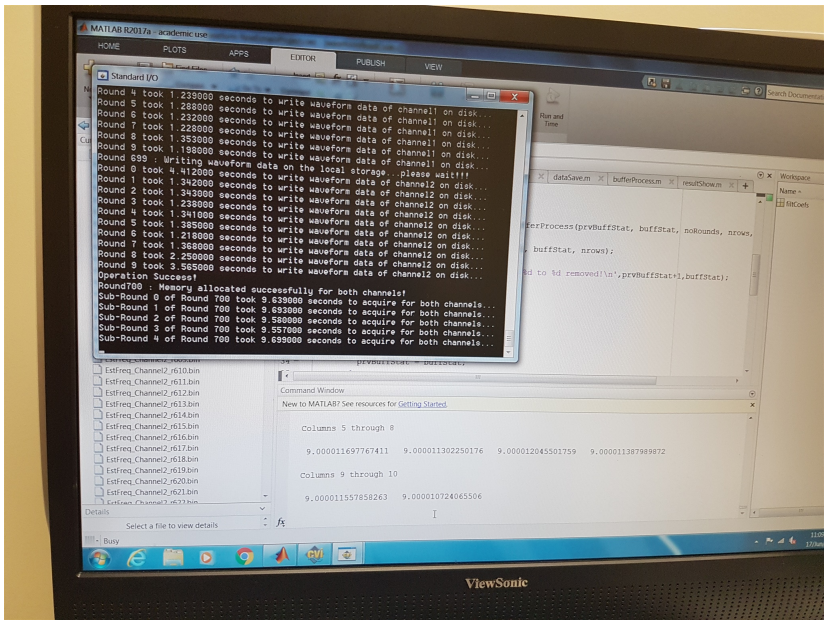

Figure 5.22: Measurement Program Developed in C language using National Instruments LabWindows CVI

Table 5.1: Detailed Specifications of Different AMShare Prototypes

Platform	**ADC Resolution**	**Dynamic Range**	**SNR**	**Clock Frequency**	**ODR**	**Buffer Size**	**Connection**	**ADC Architecture**	**Total Power Dissipation**
AD7760	24 bit	100 dB @ 2.5 MHz Rate	100 dB @ 2.5 MHz Rate	40 MHz	2.5 MSPS	2^{19} Samples	USB 2.0	$\Sigma - \Delta$	958 mW @ Normal Mode
AD7626	16 bit	91.5 dB	91 dB	100 MHz	10 MSPS	2^{20} Samples	USB 2.0	PulSAR	150 mW @ conversion
R & S RTO 1044/RTE 1034	8 bit	45 dB	45 dB	20 MHz	20 MSPS	20M $\approx 2^{24}$ Samples	Ethernet 100Mbps	N/A	N/A

5.5 Digital Signal Processing

In previous sections of this chapter, different AMShare prototypes were discussed and the implementation details were elaborated. Specifically, the radio front end circuit and the analog to digital conversion platform were studied in detail. As it was discussed earlier, once the digital samples are acquired and uploaded to the host computer they need to be processed to extract the frequency information of incoming radio signal from them. In this section, the digital signal processing part of AMShare synchronization solution is discussed.

5.6 Observations and Results

As described in the previous sections of this thesis, three different versions of the AMShare prototype has been implemented and tested successfully. In each prototype, there is two distinct receiver chains that receive and process the incoming signal broadcast by the commercial AM radio transmitter. In each receiver chain, the output bit stream of the analog to digital conversion platform is processed on a host computer to compute the frequency estimate of the carrier component of the received AM signal. In this research, the method proposed by Candan [14] is used to compute ML frequency estimate. In this sections, the observations and processing results of each prototype are discussed.

5.6.1 AD7760 and CED-1Z Platform

As discussed in earlier, the complete data acquisition system of the first version of AMShare prototype comprises the AD7760 evaluation board and CED-1Z platform both from the Analog Devices. The results and observations of the first version of AMShare prototype are discussed in this section.

In the first experiment of AMShare prototype, $2^{19} = 524288$ samples have been acquired with the AD7760 and CED-1Z platform. Then, the samples were passed through a digital low pass filter with cut-off frequency of 1 MHz to remove the out of band spectrum components in the received signal. Candan fine frequency estimator then was applied on the acquired samples to estimate the frequency of the carrier component embedded in the received AM signal in each receiver chain. The estimated frequency in each receiver chain is shown in Figure 5.23.

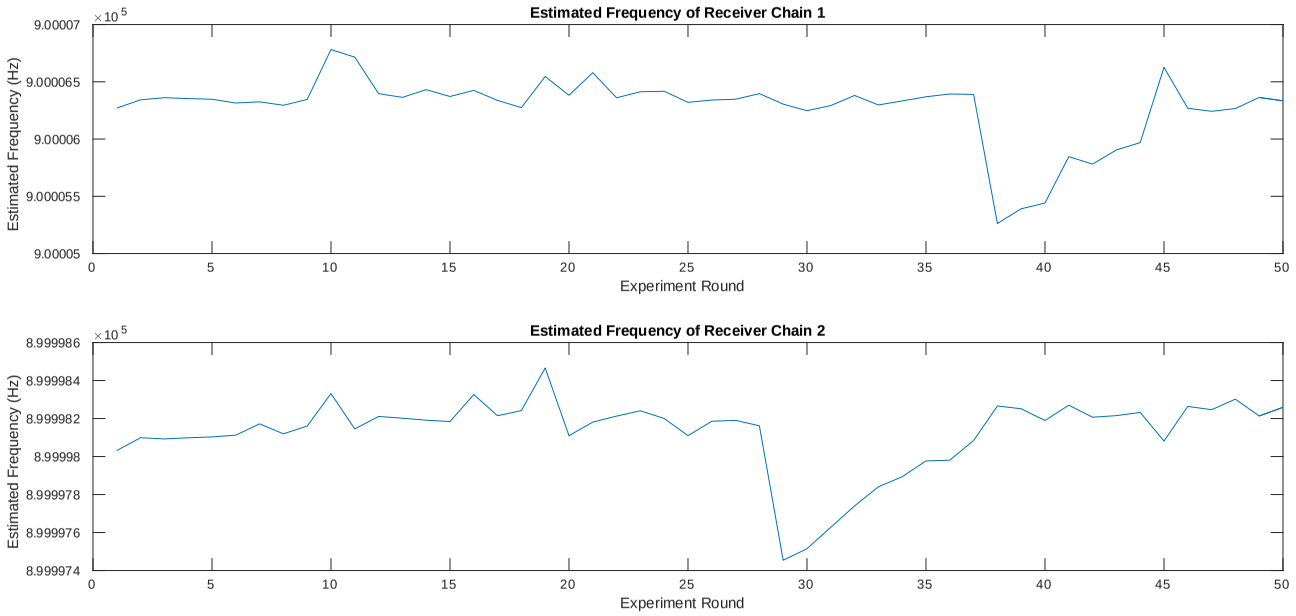

Figure 5.23: Estimated Frequency $\widehat{f}$ in Each Receiver Chain of First Version of AMShare Prototype

To evaluate the efficiency of the AMShare frequency synchronization solution, the difference between the estimated frequency in each receiver chain of the first version of AMShare prototype is measured and the result is shown in Figure 5.24.

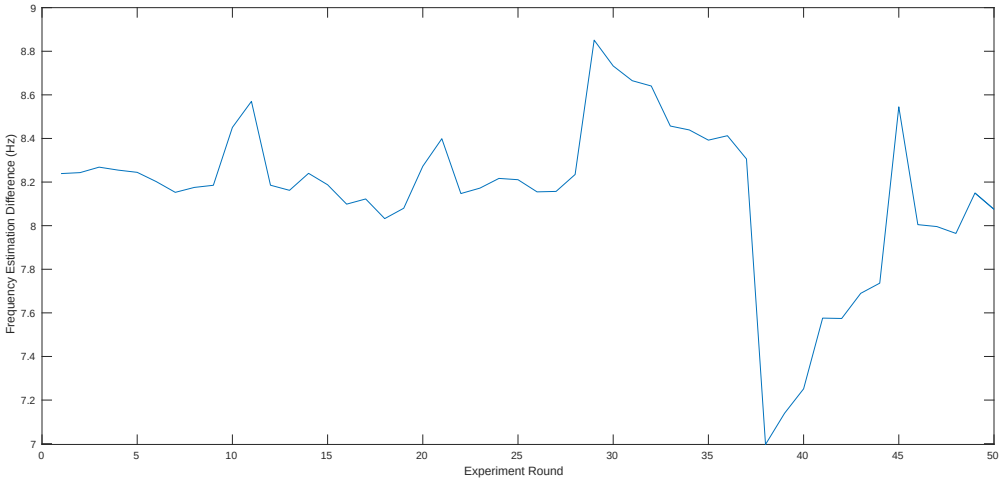

Figure 5.24: Frequency Estimation Difference in Two Separate Nodes of First Version of AMShare Prototype Measured Over 7200 Seconds

To have a better understanding of the estimation difference trend between two separate receiver chains, the first order difference of the frequency estimation difference vector between two chains is measured and the result is shown in Figure 5.25.

The experiment parameters that used in evaluating the first version of AMShare prototype are shown in Table 5.2 .

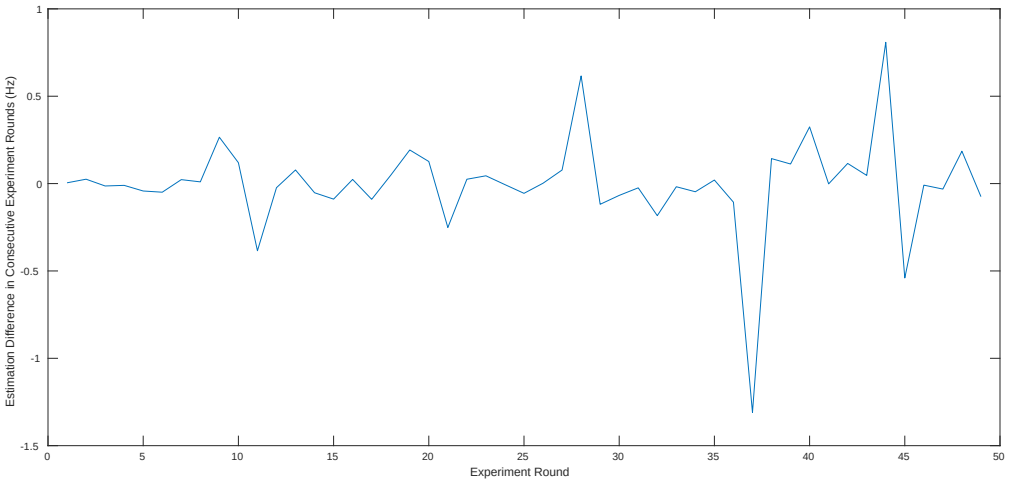

Figure 5.25: Estimation Trend Between Two Separate Nodes of AMShare Prototype Computed By First Order Difference of Estimation Vector Measured Over 7200 Seconds

$E(\widehat{f}_1)$	$E(\widehat{f}_2)$	σ^2_{f1}	σ^2_{f2}	$E(\widehat{f}_1 - \widehat{f}_2)$	σ^2_{f1-f2}
900.006277337 kHz	899.998124234 kHz	0.08503429921	0.03826582691	8.15310311605 Hz	0.13384047249

(a) Measurement Results of First Version of AMShare Prototype Comprising AD7760 Evaluation Board and CED-1Z

Platform	**Sample Size**	**Fs**	**Experiment Duration**	**Experiment Rounds**	**Duration Per Experiment**
AD7760 and CED-1Z	524288 $=2^{19}$	2.5 MSPS	7200 sec	50	144 sec

(b) Parameters of the First Version of AMShare Prototype

Table 5.2: Parameters and Measurement Results of First Version of AMShare Prototype

5.6.2 AD7626 and SDP-H1 Platform

In the second version of AMShare prototype, the same analog front end is used along with the second data acquisition platform comprising AD7626 evaluation board and SDP-H1 development board. The raw digital samples at the output of AD7626 are transferred to the host computer via a USB 2.0 link for further processing. In the host computer, the data samples of both receiver chains are processed by Candan fine frequency estimator to estimate the frequency of carrier component embedded in the received AM signal by each receiver chain. The observations and results of the second version of AMShare prototype are shown in Figures 5.26 and 5.27.

Different technical parameters and observation results of the second version of AMShare prototype are shown in Table 5.3.

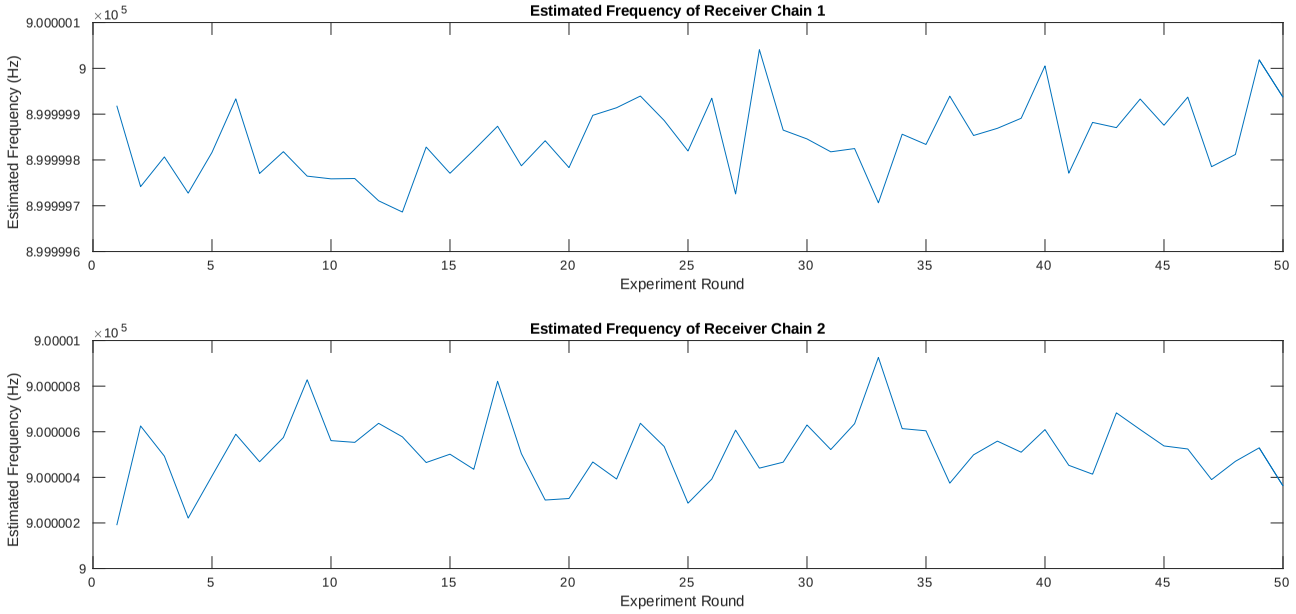

Figure 5.26: Estimated Frequency $\widehat{f}$ in Each Receiver Chain of Second Version of AMShare Prototype

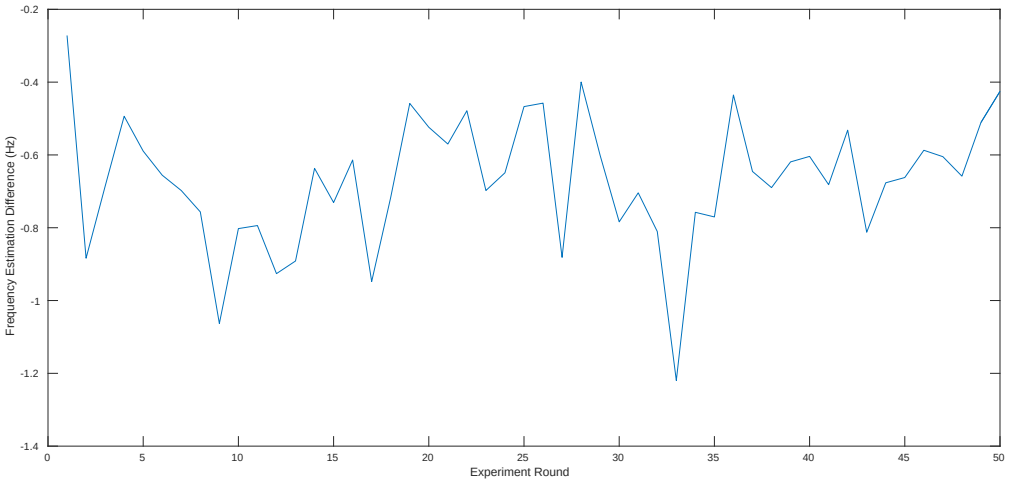

Figure 5.27: Frequency Estimation Difference in Two Separate Nodes of Second Version of AMShare Prototype

$E(\widehat{f}_1)$	$E(\widehat{f}_2)$	σ^2_{f1}	σ^2_{f2}	$E(\widehat{f}_1 - \widehat{f}_2)$	σ^2_{f1-f2}
899.999844155 kHz	900.000514831 kHz	0.00674739970	0.02029219360	0.67067547572 Hz	0.03092194412

(a) Measurement Results of Second Version of AMShare Prototype Comprising AD7626 Evaluation Board and SDP-H1 Development Board

Platform	**Sample Size**	**Fs**	**Experiment Duration**	**Experiment Rounds**	**Duration Per Experiment**
AD7626 and SDP-H1	1048576 $=2^{20}$	10 MSPS	7200 sec	50	144 sec

(b) Parameters of the Second Version of AMShare Prototype

Table 5.3: Parameters of Second Version of AMShare Prototype

5.6.3 Rohde & Schwarz RTO 1044 and RTE 1034

As described earlier, in third version of AMShare prototype two different digital oscilloscopes are used to acquire data samples. For third version of AMShare prototype, an automated program was developed by C language in NI[4] LabWindows CVI software to acquire data samples. LabWindows CVI is a development environment by National Instruments that provides a means to develop C programs for controlling the instruments and measurement tools that are connected to the host computer. To acquire data samples from both real time digital oscilloscopes, they were connected to a network hub and the hub was connected to host computer via a fast Ethernet connection. Once the data samples were acquired, they were processed in the host computer to estimate the frequency of the carrier component in the received AM signal. The observations and results of third version of AMShare prototype are shown in Figure 5.28 and 5.29. It must be noted that the results shown in Figures 5.28 and 5.29 are acquired over 24 hour measurement period.

Different technical parameters and observation results of the third version of AMShare prototype are shown in Table 5.4.

[4]National Instruments

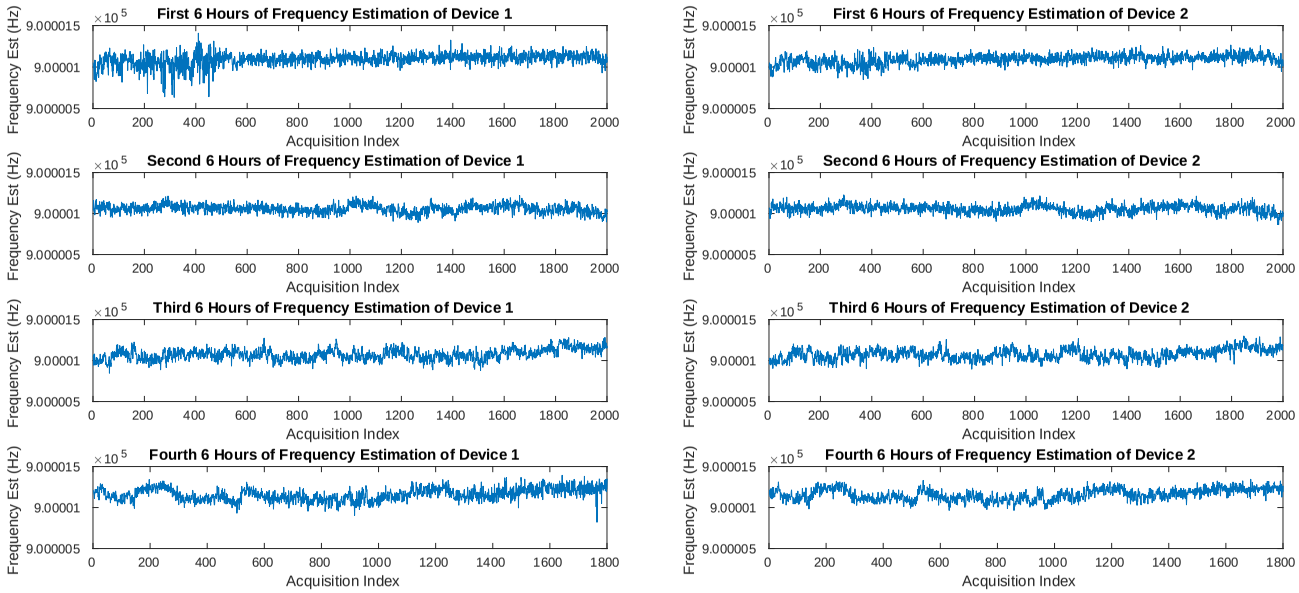

Figure 5.28: Estimated Frequency $\widehat{f}$ in Each Receiver Chain of Third Version of AMShare Prototype

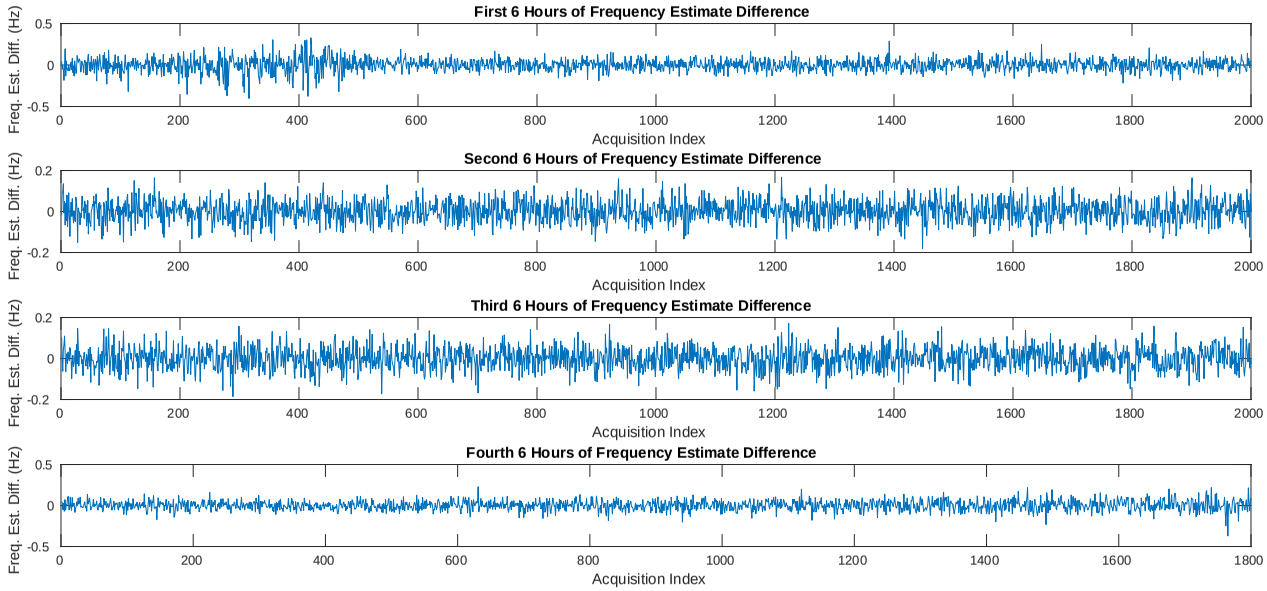

Figure 5.29: Frequency Estimation Difference in Two Separate Nodes of Third Version of AMShare Prototype

$E(\widehat{f}_1)$	$E(\widehat{f}_2)$	σ^2_{f1}	σ^2_{f2}	$E(\widehat{f}_1 - \widehat{f}_2)$	σ^2_{f1-f2}
900.001094738 kHz	900.001094136 kHz	0.00608048567	0.00528514293	6.02554539949 E-04 Hz	0.00382803429

(a) Measurement Results of Third Version of AMShare Prototype Comprising R & S RTO 1044 and RTE 1034 Real Time Digital Oscilloscopes

Platform	**Sample Size**	**Fs**	**Experiment Duration**	**Experiment Rounds**	**Duration Per Experiment**
R& S RTO 1044 and RTE 1034	20 MSamples	20 MSPS	24 Hours	7800	11 sec

(b) Parameters of the Third Version of AMShare Prototype

Table 5.4: Parameters of Third Version of AMShare Prototype

Chapter 6

Applications and Future Research

In this thesis, a new synchronization solution has been proposed that uses the idea of over the air clock as its base of operation. AMShare enables independent base transceiver stations to compensate for their CFO by listening to a common reference signal that is transmitted as an embedded component in commercial AM radio signals. Since the AMShare synchronization solution provides a method to synchronize independent local oscillators in different wireless nodes in the system.

In a wireless communication system, each node transmits its information signal at a particular carrier frequency. In almost all of the modern air interface standards in mobile communication systems, the signal is up-converted from baseband to the carrier RF frequency at the transmitter and down-converted back to the baseband at the receiver. It must be noted that both up-conversion and down-conversion are performed by a process called mixing in which the signal is multiplied by the carrier frequency. For a successful data exchange between transmitter and the receiver, both of transmitter and receiver need to generate the RF carrier signal with a precise and stable frequency.

Each wireless node has a local crystal oscillator[1] that produces a low frequency sine wave, which is used as the reference clock. The reference clock in each wireless transceiver is fed to a Phase Locked Loop (PLL) that uses the reference clock to generate the desired frequency. Instead of PLL, a direct digital synthesizer can also be used to generate the desired RF frequency from the reference clock. In a wireless communication network comprising several transceiver nodes, the key problem is that reference clocks in different nodes

[1]Due to economic considerations, atomic oscillators are less common type of oscillators that used in wireless standards

have slight differences in their frequencies. The slight frequency differences between different wireless transceivers is mainly because of the frequency difference in crystal oscillators that are used as references in wireless nodes. Since the PLL circuits in different transceivers are locked to different reference clocks with different frequencies, the generated frequency at the output of PLLs of each node are slightly different and that leads to CFO[2] between nodes. It must be noted that the CFO is not a constant value in wireless communication systems. Even variations of 0.1 degree in the temperature can cause CFO variations of a few hundred hertz. [11]

The proposed frequency synchronization solution in this thesis called AMShare provides a method to synchronize the local oscillators in different wireless transceiver nodes. As described earlier, the main application of the proposed method is maintaining frequency synchronization of indoor small cell base transceiver stations by means of sharing a common over the air clock that is used as reference. Beside of that, the proposed method can provide a means for coherent transmission wireless transmission from independent nodes. Specifically, by listening to a common reference clock across different nodes, AMShare provides a coherent radio abstraction that enables implementation of distributed PHY algorithms such as distributed MIMO, distributed modulation, distributed compressive sensing over the air and transmitter cooperation for cognitive networks.

6.0.1 Future Research

In this research, a new frequency synchronization solution for indoor small cell base stations is proposed that has the inherent potentials to address lots of challenges in the small cell deployments. It must be noted that AMShare frequency synchronization solution needs lots of future research on different aspects of proposed method. First of all, the AMShare needs a rigorous study in the field of propagation models of AM signal as part of the MW band. The propagation models are needed to study the effect of path length differences on the quality of the clock signal used at AM radio transmitter. It must be noted that the AM radio transmitter station that is used as the main reference in this research is the broadcast station located in Hamilton, Ontario, Canada. It must be noted that the selected radio transmitter station broadcasts on 900 kHz carrier frequency. Hence each receiving chain in the proposed platform, estimates the frequency of carrier component embedded in the received AM signal in the range very close to the 900 kHz carrier frequency. As described in section 2.4, the path length differences and other wireless link conditions can potentially affect the quality of the carrier component in the transmitted AM signal. The

[2]Carrier Frequency Offset

frequency fluctuations due to changing wireless link conditions can affect the precision of the transmitted signal. So, a deep research is needed to analyze effects of path noise on the received signal and detect the patterns and periods of occurrence of wireless link frequency fluctuations.

The other possible area of future research is analyzing the effects of phase noise patterns of the clock signal that feeds the ADC on the output spectrum. Since the main idea of AMShare method relies on the frequency estimation using spectral samples, any spectral leakages and interferences caused by neighbouring signals or due to pulse width modulation caused by phase noise of clock signal can deteriorate the AMShare performance.

Additionally, the fourth and fifth versions of AMShare prototype are proposed to be implemented using two high performance data acquisition platforms from Analog Devices. The proposed platforms are AD9681 and AD9253 evaluation boards. These evaluation boards should be used along with HSC-ADC-EVALC and HSC-ADC-EVALD FPGA-based development platforms.

AD9681 is an octal 14-bit 125 MSPS Analog to Digital Converter with excellent power consumption performance (110 mW per channel at 125 MSPS). The AD9681 has an SNR of 74 dBFS and SFDR of 90 dBc. It is designed for low cost, low power, small size deployment environments. The AD9681 ADC contains several features designed to maximize flexibility and minimize system cost, such as programmable clock and data alignment. The functional block diagram of AD9681 is shown in Figure 6.1.

For the ease of implementation and to reduce design development time, AD9681 evaluation board should be used with HSC-ADC-EVALDZ FPGA-based data capture kit. HSC-ADC-EVALDZ has 256 kB FIFO path and is based on Virtex-6 FPGA. It supports multiple ADC channels up to 18 bits and connects to the host computer via a simple USB 2.0 interface. The complete data acquisition platform comprising AD9681 evaluation board and HSC-ADC-EVALDZ is shown in Figure 6.2.

The other data acquisition platform that can be used to implement fifth version of AMShare prototype is based on AD9253 and HSC-ADC-EVALCZ. The AD9253 is a quad 14 bit 125 MSPS serial LVDS Analog to Digital Converter that is optimized for outstanding dynamic performance (with 74 dB SNR and 90 dBc SFDR both measured to Nyquist rate) and low power (110 mW per channel at 125 MSPS) in applications where a small size package is critical. The functional block diagram of AD9253 is shown in Figure 6.3.

To reduce design development time and to improve reliability of the fifth version of AMShare prototype, it can be used along with HSC-ADC-EVALC high speed converter evaluation platform.

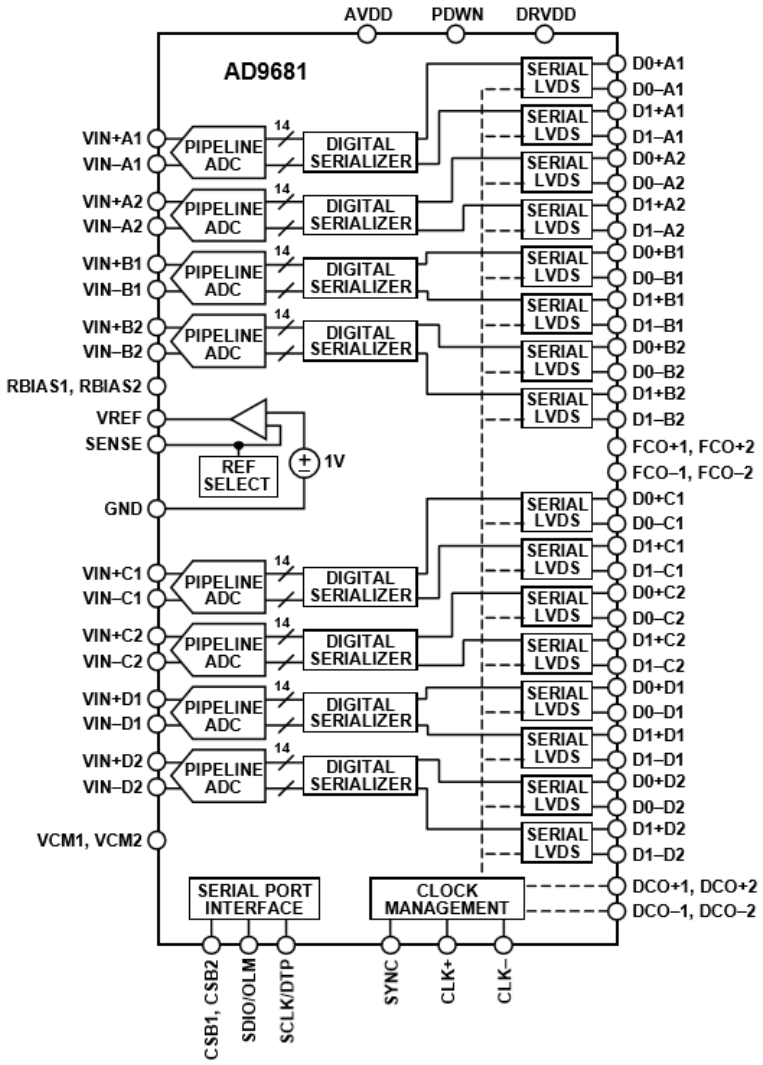

Figure 6.1: Functional Block Diagram of AD9681 octal 14-bit 125 MSPS ADC

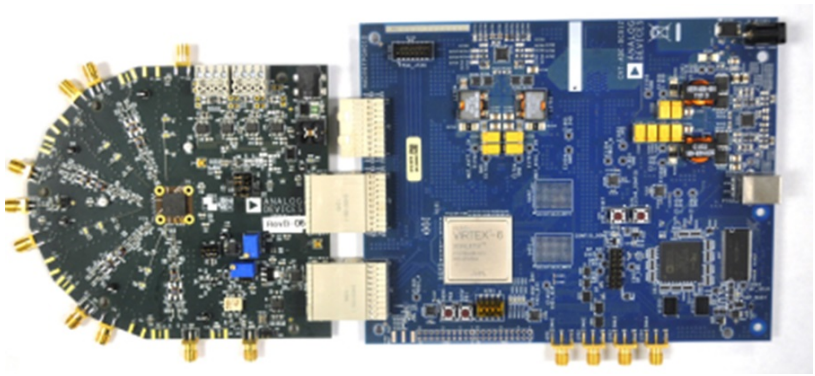

Figure 6.2: Complete Data Acquisition Platform Comprising AD9681 Evaluation Board and HSC-ADC-EVALD high speed converter and evaluation platform

The HSC-ADC-EVALC board is an FPGA-based data capture kit that is designed to communicate with single and multi-channel ADC platforms. It has a FIFO with depth of 64 kB. It must be noted that HSC-ADC-EVALC high speed converter evaluation board uses an FPGA based buffer memory to capture blocks of digital data from the Analog Devices high speed analog-to-digital converter (ADC) evaluation boards. It connects to the host computer via a USB 2.0 port.

The complete data acquisition platform comprising AD9253 evaluation board and HSC-ADC-EVALC converter evaluation platform that can be used to implement fifth version of AMShare method is shown in Figure 6.4.

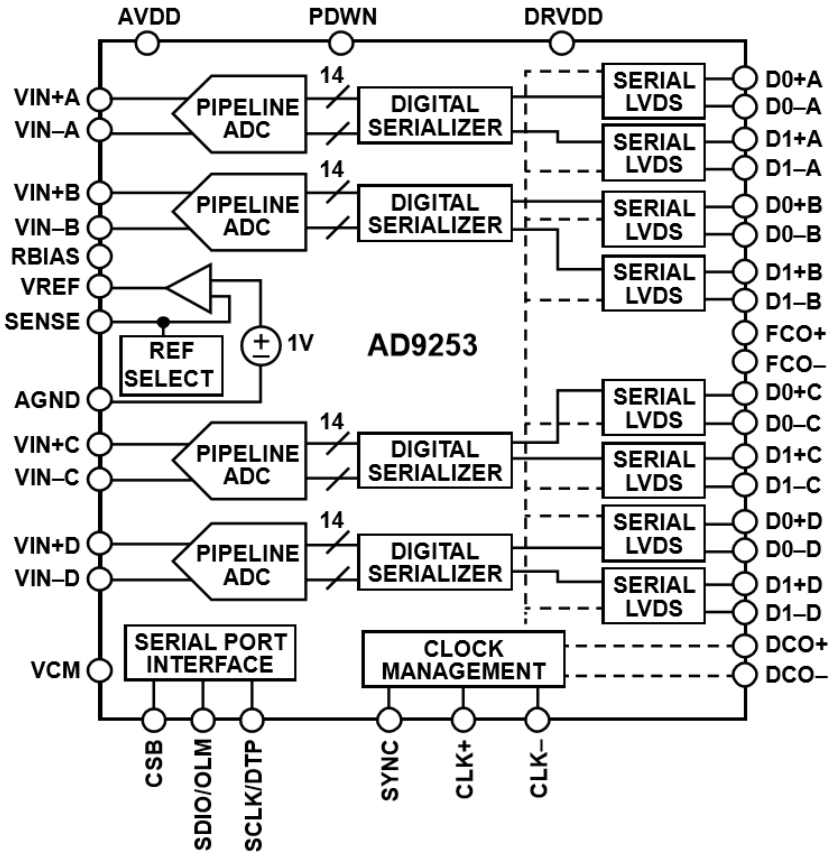

Figure 6.3: Functional Block Diagram of AD9253 quad 14-bit 125 MSPS Serial LVDS ADC

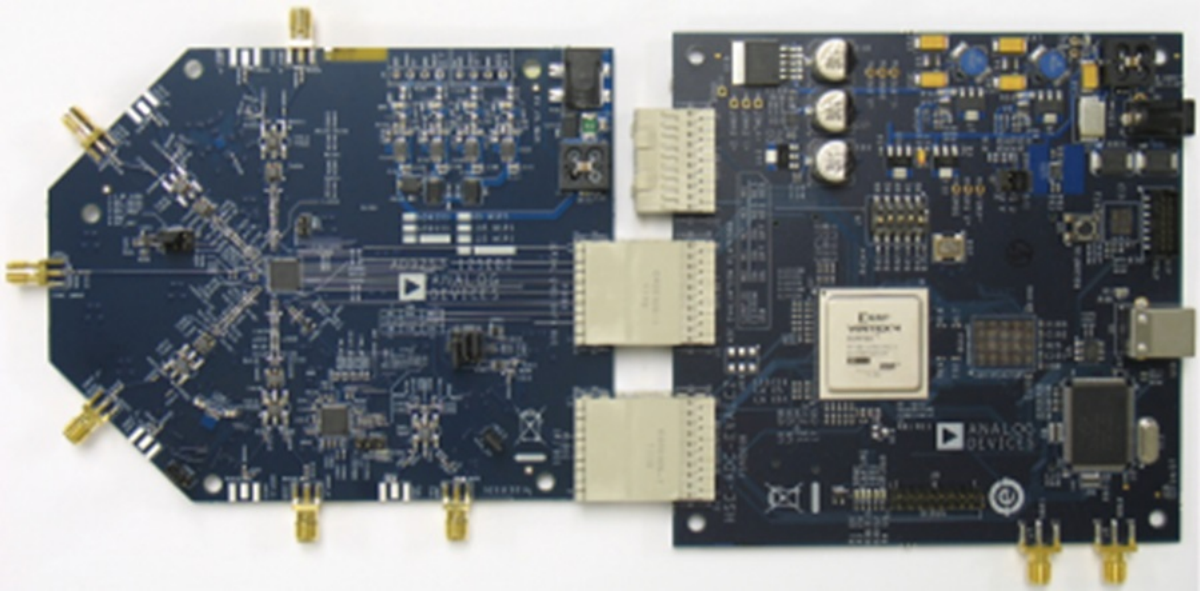

Figure 6.4: Complete Data Acquisition Platform Comprising AD9253 Evaluation Board and HSC-ADC-EVALC high speed converter and evaluation platform

References

[1] *ITU-T G.826x series.*

[2] *ITU-T G.827x series.*

[3] 3GPP TS 36.104. *Base Station (BS) Radio Transmission and Reception.*

[4] 3GPP TS 36.133. *Requirements for Support of Radio Resource Management.*

[5] T. Abatzoglou. A fast maximum likelihood algorithm for frequency estimation of a sinusoid based on newton's method. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 33(1):77–89, Feb 1985.

[6] D. Agrez. Weighted multipoint interpolated dft to improve amplitude estimation of multifrequency signal. *IEEE Transactions on Instrumentation and Measurement*, 51(2):287–292, Apr 2002.

[7] G. Andria, M. Savino, and A. Trotta. Windows and interpolation algorithms to improve electrical measurement accuracy. *IEEE Transactions on Instrumentation and Measurement*, 38(4):856–863, Aug 1989.

[8] Daniel Belega and Dario Petri. Sine-wave parameter estimation by interpolated DFT method based on new cosine windows with high interference rejection capability. *Digital Signal Processing*, 33:60–70, 2014.

[9] Daniel Belega, Dario Petri, and Dominique Dallet. Frequency estimation of a sinusoidal signal via a three-point interpolated DFT method with high image component interference rejection capability. *Digital Signal Processing*, 24:162–169, 2014.

[10] D. Bladsjö, M. Hogan, and S. Ruffini. Synchronization aspects in lte small cells. *IEEE Communications Magazine*, 51(9):70–77, September 2013.

[11] C. A. Boano, M. Zúñiga, J. Brown, U. Roedig, C. Keppitiyagama, and K. Römer. Templab: A testbed infrastructure to study the impact of temperature on wireless sensor networks. In *IPSN-14 Proceedings of the 13th International Symposium on Information Processing in Sensor Networks*, pages 95–106, April 2014.

[12] Ç. Candan. A method for fine resolution frequency estimation from three dft samples. *IEEE Signal Processing Letters*, 18(6):351–354, June 2011.

[13] Ç. Candan. Analysis and further improvement of fine resolution frequency estimation method from three dft samples. *IEEE Signal Processing Letters*, 20(9):913–916, Sept 2013.

[14] Ç. Candan. Fine resolution frequency estimation from three DFT samples : Case of windowed data. *Signal Processing*, 114:245–250, 2015.

[15] CPRI Spec. v. 5.0. *http://www.cpri.info/*.

[16] K. Duda. Dft interpolation algorithm for kaiser x2013;bessel and dolph x2013;chebyshev windows. *IEEE Transactions on Instrumentation and Measurement*, 60(3):784–790, March 2011.

[17] T. Grandke. Interpolation algorithms for discrete fourier transforms of weighted signals. *IEEE Transactions on Instrumentation and Measurement*, 32(2):350–355, June 1983.

[18] B. Hoffman-Wellenhof, H. Lichtenegger, and J. Collins. *Global Positioning System*. Springer-Verlag, 1997.

[19] D Brown III, Yizheng Liao, and Neil Fox. Low-Complexity Real-Time Single-Tone Phase and Frequency Estimation. *IEEE Military Communication*, (2), 2010.

[20] International Organization for Standardization (ISO). *International Vocabulary of Basic and General Terms in Metrology*, 1993. 59p.

[21] ITU-T Rec. G.810. *Definitions and Terminology for Synchronization Networks*.

[22] Steven M. Kay. *Fundamentals of Statistical Signal Processing, Volume I: Estimation Theory*. Prentice Hall, 1993.

[23] MA Lombardi. An introduction to frequency calibration: Part II. *Cal Lab Int. J. Metrology*, (January 1996):55–86, 1996.

[24] Michael A Lombardi. NIST Time and Frequency Services. *Nist Special Publication*, page 80, 2002.

[25] Michael A Lombardi, Lisa M Nelson, Andrew N Novick, and Victor S Zhang. Time and Frequency Measurements Using the Global Positioning System. *Cal Lab Int. J.*, pages 26–33, 2001.

[26] M. D. Macleod. Fast nearly ml estimation of the parameters of real or complex single tones or resolved multiple tones. *IEEE Transactions on Signal Processing*, 46(1):141–148, Jan 1998.

[27] R. Mudumbai, D. R. Brown Iii, U. Madhow, and H. V. Poor. Distributed transmit beamforming: challenges and recent progress. *IEEE Communications Magazine*, 47(2):102–110, February 2009.

[28] NIST. *An Introduction to Frequency Calibrations.*

[29] D. Rife and R. Boorstyn. Single tone parameter estimation from discrete-time observations. *IEEE Transactions on Information Theory*, 20(5):591–598, Sep 1974.

[30] David C. Rife and Robert R. Boorstyn. Single-Tone Parameter Estimation from Discrete-Time Observations. *IEEE Transactions on Information Theory*, 20(5):591–598, 1974.

[31] P. Stoica, R. L. Moses, B. Friedlander, and T. Soderstrom. Maximum likelihood estimation of the parameters of multiple sinusoids from noisy measurements. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(3):378–392, Mar 1989.

[32] Barry N. Taylor and Chris E. Kuyatt. Guidelines for Evaluating and Expressing the Uncertainty of NIST Measurement Results. *NIST Technical Note*, 1297:20, 1994.

[33] F. L. Walls and J. J. Gagnepain. Environmental sensitivities of quartz oscillators. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 39(2):241–249, March 1992.

APPENDICES

C Code of AMShare V3

The C code developed in NI LabWindows integrated development environment is shown in this section. This code opens a session to each connected instruments, configures them and acquire required data samples based on the specific user requirements.

```
1 //
    ==========================================================================

2 //
3 // Title:    waveformReadCode.c
4 // Purpose:    Data acquisition using R&S RTO 1044 and RTE 1034.
5 //
6 // Created on:  2017-05-21 at 6:56:28 PM by Milad Mazi Esfahani.
7 // Copyright: Milad Mazi Esfahani and CST Research Lab. All
    Rights Reserved.
8 //
9 //
    ==========================================================================

10
11 //
    ==========================================================================

12 // Include files
13
14 //#include "waveformReadCode.h"
15
16 #include "rsscope.h"
17 #include <cvirte.h>
```

```
18 #include <visa.h>
19 #include <stdlib.h>
20 #include <stdio.h>
21 #include <time.h>
22
23
24 #define ARRAY_SIZE     20000000          /* for
       rsscope_ReadWaveform() variable: waveForm */
25 #define STRING_LENGTH 512
26 #define WINDOW         1
27 #define INSTRUMENT1    1
28 #define INSTRUMENT2    2
29
30 /* Useful macros */
31 #define CHECKERR(fCal) \
32     if (status = checkError((fCal)), status < VI_SUCCESS) \
33         return status; else
34
35
36 /* Global variables */
37 static     ViSession  instrSessionDev1 = VI_NULL;
38 static     ViSession  instrSessionDev2 = VI_NULL;
39 ViReal64**  waveFormCh1_Dev1 = NULL;
40 ViReal64**  waveFormCh2_Dev1 = NULL;
41 ViReal64**  waveFormCh1_Dev2 = NULL;
42 ViReal64**  waveFormCh2_Dev2 = NULL;
43 ViInt32      length = 20000000;
44 ViInt32    nrows = 5;
45 ViInt32    roundsNo = 36;
46 FILE     *fp1;
47 FILE     *fp2;
48 FILE     *fps;
49 ViInt32    fileSize;
50 ViInt32    i;
51 ViInt32    j;
52 ViInt32    k;
53
```

```
54 ViChar       instr1_desc [STRING_LENGTH] = "TCPIP0::192.168.10.1::
       INSTR";
55 ViChar     instr2_desc [STRING_LENGTH] = "TCPIP0::192.168.10.2::
       INSTR";
56 ViInt32      id                  = VI_FALSE;
57 ViInt32      reset               = VI_FALSE;
58 ViInt32      arithmetics         = RSSCOPE_VAL_OFF;
59 ViInt32      acqType             = RSSCOPE_VAL_HI_RES;
60 ViStatus     status              = VI_SUCCESS;
61 ViReal64     acqTime             = 0.1;
62 ViReal64     timeout             = 5000;
63 ViInt32      actualPoints        = 0;
64 ViReal64     initalX             = 0.0;
65 ViReal64     xIncrement          = 0.0;
66
67
68
69 /* Utility functions declarations */
70 ViStatus checkError(ViStatus status);
71 int    getWaveform(ViInt32, ViInt32, ViInt32);
72 void   cleanup (ViBoolean);
73 int    fileWrite(ViInt32, ViInt32, ViInt32);
74 void   waitFor (unsigned int);
75 int    readWaveform(ViInt32, ViInt32, ViInt32);
76 int    memorySet();
77 void   memoryClean();
78 int    initSession(ViInt32);
79 void   statUpdate (ViInt32);
80 void   bufferFlagUpdate(ViInt32);
81
82
83
84 int getWaveform(ViInt32 instrId, ViInt32 round, ViInt32 channel)
85 {
86
87   initSession(instrId);
88   /* configures how the oscilloscope acquires data and fills the
        waveform record */
```

```
89      //CHECKERR (rsscope_ConfigureAcquisitionType (instrSession ,
            RSSCOPE_VAL_CHANNEL_1, 1, acqType));
90
91    /* configures the method to reduce the data stream of the ADC
         to a stream of waveform points with lower sample rate */
92      //CHECKERR (rsscope_ConfigureWaveformArithmetic (instrSession
            , RSSCOPE_VAL_CHANNEL_1, 1,arithmetics));
93
94    /* configures the time of one acquisition, that is the time
         across the 10 divisions of the diagram */
95      //CHECKERR (rsscope_ConfigureAcquisitionTime (instrSession,
            acqTime));
96
97    readWaveform(instrId ,round,channel);
98
99
100   return 0;
101 }
102
103 int initSession(ViInt32 instrId)
104 {
105   /* initialize driver for the device passed by the input
         argument */
106   if (instrId == 1)
107   {
108
109       rsscope_init (instr1_desc, id, reset, &instrSessionDev1);
110   }
111
112   if (instrId == 2)
113   {
114
115     rsscope_init (instr2_desc, id, reset, &instrSessionDev2);
116   }
117
118   return 0;
119 }
120
```

```
121 int readWaveform(ViInt32 instrId, ViInt32 round, ViInt32 channel)
122 {
123   /* This function initiates an acquisition on the channels that
        you
124      enable with the rsscope_ConfigureChannel function.  It then
125      waits for the acquisition to complete, and returns the
           waveform
126      for the channel you specify*/
127
128   if (instrId == 1)
129   {
130
131     if (channel==1)
132     {
133         rsscope_ReadWaveform (instrSessionDev1, channel, 1,
              ARRAY_SIZE, timeout, waveFormCh1_Dev1[round], &
              actualPoints, &initalX, &xIncrement);
134
135       //printf("Actual points of channel %d are %d\n",channel,
            actualPoints);
136
137       /*if((sizeof(waveFormCh1[round])/sizeof(ViReal64)) !=
            actualPoints)
138       {
139         printf("Error! Operation Failed!\n");
140         return 0;
141       }*/
142
143       if (actualPoints == 0)
144       {
145         printf("Error! Operation Failed!\n");
146         return 0;
147       }
148
149       return 1;
150     }
151
152     if (channel==2)
```

```
153     {
154       rsscope_ReadWaveform (instrSessionDev1, channel, 1,
              ARRAY_SIZE, timeout, waveFormCh2_Dev1[round], &
              actualPoints, &initalX, &xIncrement);
155
156       //printf("Actual points of channel %d are %d\n",channel,
              actualPoints);
157
158       /*if((sizeof(waveFormCh2[round])/sizeof(ViReal64)) !=
              actualPoints)
159       {
160         printf("Error! Operation Failed!\n");
161         return 0;
162       }*/
163
164       if (actualPoints == 0)
165       {
166         printf("Error! Operation Failed!\n");
167         return 0;
168       }
169
170       return 1;
171     }
172   }
173
174
175   if (instrId == 2)
176   {
177
178     if (channel==1)
179     {
180         rsscope_ReadWaveform (instrSessionDev2, channel, 1,
              ARRAY_SIZE, timeout, waveFormCh1_Dev2[round], &
              actualPoints, &initalX, &xIncrement);
181
182       //printf("Actual points of channel %d are %d\n",channel,
              actualPoints);
183
```

```
184         /*if((sizeof(waveFormCh1[round])/sizeof(ViReal64)) !=
                actualPoints)
185         {
186           printf("Error! Operation Failed!\n");
187           return 0;
188         }*/
189
190         if (actualPoints == 0)
191         {
192           printf("Error! Operation Failed!\n");
193           return 0;
194         }
195
196         return 1;
197       }
198
199       if (channel==2)
200       {
201         rsscope_ReadWaveform (instrSessionDev2, channel, 1,
              ARRAY_SIZE, timeout, waveFormCh2_Dev2[round], &
              actualPoints, &initalX, &xIncrement);
202
203         //printf("Actual points of channel %d are %d\n",channel,
              actualPoints);
204
205         /*if((sizeof(waveFormCh2[round])/sizeof(ViReal64)) !=
              actualPoints)
206         {
207           printf("Error! Operation Failed!\n");
208           return 0;
209         }*/
210
211         if (actualPoints == 0)
212         {
213           printf("Error! Operation Failed!\n");
214           return 0;
215         }
216
```

```
217         return 1;
218       }
219     }
220 }
221
222 /*
      ==========================================================================
      */
223 /* Function: checkError
224 /* Purpose:  If passed status indicates an failure, the error
      message will
225 /*           be displayed.
226 /*==========================================================================
      */
227
228 /* ViStatus checkError (ViStatus status)
229 {
230       ViChar  error_message [STRING_LENGTH];
231       ViChar  error_buffer  [STRING_LENGTH*2];
232       ViChar* p2buf;
233       ViInt32 error;
234
235       if (status < VI_SUCCESS)
236       {
237           // Converts a status code returned by an instrument
                driver function into a user-readable string
238           rsscope_error_message (instrSession, status,
                error_message);
239           p2buf = error_buffer + sprintf (error_buffer, "Primary
                Error: 0x%08X, %s\n", status, error_message);
240
241
242           // This function reads an error code and a message from
                the instrument's error queue
243           rsscope_error_query (instrSession, &error, error_message)
                ;
244
245           if (error != VI_SUCCESS)
```

```
246         {
247             sprintf (p2buf, "Secondary Error: 0x%08X, %s\n",
                    error, error_message);
248         }
249
250     printf("Error:", error_buffer);
251
252         // close session
253         cleanup(VI_TRUE);
254
255     }
256     return status;
257 }
258 */
259
260 /*
    ===========================================================================
    */
261 /* Function: Cleanup
262 /* Purpose:  Closes the VISA session to the instrument and in
    case of error
263 /*           queries the error and shows the dialog box with the
    passed error
264 /*           message. In case Device Clear is requested, it is
    necessary to
265 /*           set the argument deviceClear to VI_TRUE.
266 /*===========================================================================
    */
267 void cleanup (ViBoolean deviceClear)
268 {
269
270     /* Device Clear is a low-level command and should be used in
          case the
271      * instrument is waiting for Operation Complete to cancel the
          wait.
272      * It is useful for instance in case of incorrect external
          trigger
```

```
273       * when the instrument does not respond to any other command
              because of
274       * waiting for trigger.
275       */
276      if (deviceClear)
277      {
278          viClear (instrSessionDev1);
279      viClear (instrSessionDev2);
280          viPrintf (instrSessionDev1, "*CLS\n");
281      viPrintf (instrSessionDev2, "*CLS\n");
282      }
283
284      /* close instrument session */
285      rsscope_close (instrSessionDev1);
286    rsscope_close (instrSessionDev2);
287      instrSessionDev1 = VI_NULL;
288    instrSessionDev2 = VI_NULL;
289
290      memoryClean();
291
292    fclose(fp1);
293
294    fclose(fp2);
295 }
296
297 void memoryClean()
298 {
299
300    if (waveFormCh1_Dev1)
301      {
302      for(j=0;j<nrows;j++)
303      {
304            free (waveFormCh1_Dev1[j]);
305      }
306      free (waveFormCh1_Dev1);
307          waveFormCh1_Dev1 = NULL;
308      }
309
```

```
310    if (waveFormCh2_Dev1)
311    {
312      for(j=0;j<nrows;j++)
313      {
314            free (waveFormCh2_Dev1[j]);
315      }
316      free (waveFormCh2_Dev1);
317          waveFormCh2_Dev1 = NULL;
318    }
319
320    if (waveFormCh1_Dev2)
321      {
322      for(j=0;j<nrows;j++)
323      {
324            free (waveFormCh1_Dev2[j]);
325      }
326      free (waveFormCh1_Dev2);
327          waveFormCh1_Dev2 = NULL;
328      }
329
330
331    if (waveFormCh2_Dev2)
332    {
333      for(j=0;j<nrows;j++)
334      {
335            free (waveFormCh2_Dev2[j]);
336      }
337      free (waveFormCh2_Dev2);
338          waveFormCh2_Dev2 = NULL;
339    }
340
341
342
343 }
344
345
346 int fileWrite(ViInt32 instrId, ViInt32 channel, ViInt32 round)
347 {
```

```
348   char strbuff_ch1[80];
349   char strbuff_ch2[80];
350
351   printf("Round %d : Writing waveform data on the local storage
          ... please wait!!!\n",round);
352
353   for (i=0;i<nrows;i++)
354   {
355
356     if (instrId == 1)
357     {
358       if (channel==1)
359       {
360         clock_t start = clock();
361         sprintf(strbuff_ch1,"WaveFormRound%d_Channel%d_r%d_Dev1.
              bin",i,channel,round);
362         fp1=fopen(strbuff_ch1,"wb");
363         fwrite(waveFormCh1_Dev1[i],sizeof(ViReal64),ARRAY_SIZE,
              fp1);
364         fclose(fp1);
365         clock_t end = clock();
366         float seconds = (float)(end - start) / CLOCKS_PER_SEC;
367         printf("Round %d took %f seconds to write waveform data
              of channel%d of device 1 on disk...\n",i,seconds,
              channel);
368
369       }
370
371       if (channel==2)
372       {
373         clock_t start = clock();
374         sprintf(strbuff_ch2,"WaveFormRound%d_Channel%d_r%d_Dev1.
              bin",i,channel,round);
375         fp2=fopen(strbuff_ch2,"wb");
376         fwrite(waveFormCh2_Dev1[i],sizeof(ViReal64),ARRAY_SIZE,
              fp2);
377         fclose(fp2);
378         clock_t end = clock();
```

```
379         float seconds = (float)(end - start) / CLOCKS_PER_SEC;
380         printf("Round %d took %f seconds to write waveform data
                of channel%d of device 1 on disk...\n",i,seconds,
                channel);
381       }
382     }
383
384
385     if (instrId == 2)
386     {
387       if (channel==1)
388       {
389         clock_t start = clock();
390         sprintf(strbuff_ch1,"WaveFormRound%d_Channel%d_r%d_Dev2.
                bin",i,channel,round);
391         fp1=fopen(strbuff_ch1,"wb");
392         fwrite(waveFormCh1_Dev2[i],sizeof(ViReal64),ARRAY_SIZE,
                fp1);
393         fclose(fp1);
394         clock_t end = clock();
395         float seconds = (float)(end - start) / CLOCKS_PER_SEC;
396         printf("Round %d took %f seconds to write waveform data
                of channel%d of device 2 on disk...\n",i,seconds,
                channel);
397
398       }
399
400       if (channel==2)
401       {
402         clock_t start = clock();
403         sprintf(strbuff_ch2,"WaveFormRound%d_Channel%d_r%d_Dev2.
                bin",i,channel,round);
404         fp2=fopen(strbuff_ch2,"wb");
405         fwrite(waveFormCh2_Dev2[i],sizeof(ViReal64),ARRAY_SIZE,
                fp2);
406         fclose(fp2);
407         clock_t end = clock();
408         float seconds = (float)(end - start) / CLOCKS_PER_SEC;
```

```
409           printf("Round %d took %f seconds to write waveform data
                  of channel%d of device 2 on disk...\n",i,seconds,
                  channel);
410         }
411       }
412     }
413
414     return 0;
415 }
416
417 void statusUpdate (ViInt32 roundNo)
418 {
419     ViInt32 *ptr = calloc(1,sizeof(ViInt32));
420     *ptr = roundNo;
421     fps = fopen("status.bin","wb");
422     fwrite(ptr,sizeof(ViInt32),1,fps);
423     fclose(fps);
424 }
425
426
427 void bufferFlagUpdate(ViInt32 roundNo)
428 {
429     ViInt32 *ptr = calloc(1,sizeof(ViInt32));
430     *ptr = roundNo;
431     fps = fopen("bufferFlag.bin","wb");
432     fwrite(ptr,sizeof(ViInt32),1,fps);
433     fclose(fps);
434 }
435
436
437
438 void waitFor (unsigned int secs) {
439       unsigned int retTime = time(0) + secs;    // Get finishing
              time.
440       while (time(0) < retTime);                // Loop until it
              arrives.
441 }
442
```

```
443 int memorySet()
444 {
445     if (waveFormCh1_Dev1)
446       {
447               free (waveFormCh1_Dev1);
448               waveFormCh1_Dev1 = NULL;
449           }
450       waveFormCh1_Dev1 = (ViReal64**) malloc (sizeof (ViReal64) *
              nrows);
451
452     if (!waveFormCh1_Dev1)
453       {
454         printf("out of memory...Can't allocate memory for channel
              1 of device 1\n");
455         return 0;
456       }
457     if (waveFormCh1_Dev1)
458     {
459       for (j=0;j<nrows;j++)
460       {
461         waveFormCh1_Dev1[j] = (ViReal64*) malloc(sizeof (ViReal64
             ) * length);
462         if (waveFormCh1_Dev1[j]==NULL)
463         {
464           printf("Out of memory to allocate for channel 1 of
                device 1!\n");
465           return 0;
466         }
467       }
468     }
469
470
471
472     if (waveFormCh2_Dev1)
473       {
474               free (waveFormCh2_Dev1);
475               waveFormCh2_Dev1 = NULL;
476           }
```

```
477       waveFormCh2_Dev1 = (ViReal64**) malloc (sizeof (ViReal64) *
              nrows);
478
479     if (!waveFormCh2_Dev1)
480       {
481         printf("out of memory to allocate for channel 2 of device
                1\n");
482         return 0;
483       }
484     if (waveFormCh2_Dev1)
485     {
486       for (j=0;j<nrows;j++)
487       {
488         waveFormCh2_Dev1[j] = (ViReal64*) malloc(sizeof (ViReal64
              ) * length);
489         if (waveFormCh2_Dev1[j]==NULL)
490         {
491           printf("Out of memory to allocate for channel 2 of
                  device 1!\n");
492           return 0;
493         }
494       }
495     }
496
497     if (waveFormCh1_Dev2)
498       {
499               free (waveFormCh1_Dev2);
500               waveFormCh1_Dev2 = NULL;
501           }
502       waveFormCh1_Dev2 = (ViReal64**) malloc (sizeof (ViReal64) *
              nrows);
503
504     if (!waveFormCh1_Dev2)
505       {
506         printf("out of memory...Can't allocate memory for channel
                1 of device 2\n");
507         return 0;
508       }
```

```
509     if (waveFormCh1_Dev2)
510     {
511       for (j=0;j<nrows;j++)
512       {
513         waveFormCh1_Dev2[j] = (ViReal64*) malloc(sizeof (ViReal64
              ) * length);
514         if (waveFormCh1_Dev2[j]==NULL)
515         {
516           printf("Out of memory to allocate for channel 1 of
                device 2!\n");
517           return 0;
518         }
519       }
520     }
521
522
523
524     if (waveFormCh2_Dev2)
525       {
526               free (waveFormCh2_Dev2);
527               waveFormCh2_Dev2 = NULL;
528         }
529       waveFormCh2_Dev2 = (ViReal64**) malloc (sizeof (ViReal64) *
            nrows);
530
531     if (!waveFormCh2_Dev2)
532       {
533         printf("out of memory to allocate for channel 2 of device
              2\n");
534         return 0;
535       }
536     if (waveFormCh2_Dev2)
537     {
538       for (j=0;j<nrows;j++)
539       {
540         waveFormCh2_Dev2[j] = (ViReal64*) malloc(sizeof (ViReal64
              ) * length);
541         if (waveFormCh2_Dev2[j]==NULL)
```

```
542         {
543           printf("Out of memory to allocate for channel 2 of
                 device 2!\n");
544           return 0;
545         }
546       }
547     }
548
549   return 1;
550 }
551
552
553 int main()
554 {
555   printf("Program started...\n");
556   waitFor(2);
557
558   initSession(1);
559   printf("Session Initialized with Oscilloscope 1 Successfully!\n
        ");
560   initSession(2);
561   printf("Session Initialized with Oscilloscope 2 Successfully!\n
        ");
562
563   statusUpdate(0);
564   bufferFlagUpdate(0);
565
566   for (k=1;k<roundsNo+1;k++)
567   {
568     int status = memorySet();
569
570     if(status)
571     {
572       printf("Round%d : Memory allocated successfully for both
            channels of both devices!\n",k);
573
574       for (i=0;i<nrows;i++)
575       {
```

```
576         clock_t start = clock();
577         int flagCh1_Dev1 = readWaveform(INSTRUMENT1,i,
                RSSCOPE_VAL_CHANNEL_1);
578         int flagCh2_Dev1 = readWaveform(INSTRUMENT1,i,
                RSSCOPE_VAL_CHANNEL_2);
579
580         int flagCh1_Dev2 = readWaveform(INSTRUMENT2,i,
                RSSCOPE_VAL_CHANNEL_1);
581         int flagCh2_Dev2 = readWaveform(INSTRUMENT2,i,
                RSSCOPE_VAL_CHANNEL_2);
582
583         clock_t end = clock();
584         float seconds = (float)(end - start) / CLOCKS_PER_SEC;
585
586         if ((flagCh1_Dev1==0) || (flagCh2_Dev1==0) || (
                flagCh1_Dev2==0) || (flagCh2_Dev2==0))
587         {
588           printf("Operation could not be completed!\n");
589           return 0;
590         }
591
592         printf("Sub-Round %d of Round %d took %f seconds to
                acquire for both channels...\n",i,k,seconds);
593       }
594
595       fileWrite(INSTRUMENT1, RSSCOPE_VAL_CHANNEL_1, k);
596       fileWrite(INSTRUMENT1, RSSCOPE_VAL_CHANNEL_2, k);
597
598       fileWrite(INSTRUMENT2, RSSCOPE_VAL_CHANNEL_1, k);
599       fileWrite(INSTRUMENT2, RSSCOPE_VAL_CHANNEL_2, k);
600
601       statusUpdate(k);
602
603       memoryClean();
604       //if (k % 2 == 0)
605       //{
606         bufferFlagUpdate(k);
607       //}
```

```
608         printf("Operation Success!\n");
609       }
610
611     }
612
613     return 0;
614 }
```

C Code of The Third Version of AMShare Prototype