

IMPLEMENTATION OF DIGITAL DOWN CONVERTER FOR SOFTWARE RADIO

A DISSERTATION

*Submitted in partial fulfillment of the
requirements for the award of the degree*

of

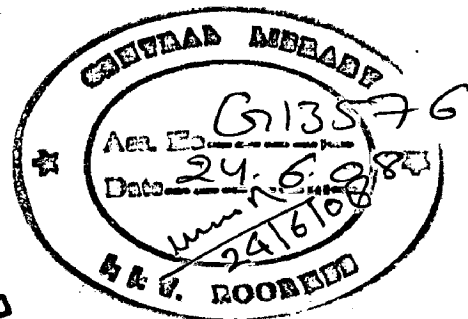
MASTER OF TECHNOLOGY

in

ELECTRONICS AND COMMUNICATION ENGINEERING
(With Specialization in Communication Systems)

By

LT COL A P VERMA



DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE-247 667 (INDIA)

JUNE, 2007


CANDIDATE'S DECLARATION

I hereby declare that the work, which is presented in this project report, entitled "IMPLEMENTATION OF DIGITAL DOWN CONVERTER FOR SOFTWARE RADIO", being submitted in partial fulfillment of the requirements for the award of the degree of MASTER OF TECHNOLOGY with specialization in COMMUNICATION SYSTEMS, in the Department of Electronics and Computer Engineering, Indian Institute of Technology, Roorkee is an authentic record of my own work carried out from July 2006 to June 2007, under guidance and supervision of Dr. D. K. MEHRA, Professor, Department of Electronics and Computer Engineering, Indian Institute of Technology, Roorkee.

The results embodied in this dissertation have not submitted for the award of any other Degree or Diploma.

Date: 29 Jun 2007

Place: Roorkee



LT COL A P VERMA

CERTIFICATE

This is to certify that the statement made by the candidate is correct to the best of my knowledge and belief.

Date: 29.6.07

Place: Roorkee


Dr. D. K. MEHRA
Professor, E & C E Department
Indian Institute of Technology, Roorkee
Roorkee – 247 667, (INDIA)

ACKNOWLEDGEMENTS

It is my privilege and pleasure to express my profound sense of respect, gratitude and indebtedness to my guide, **Dr. D.K. Mehra**, Professor, Department of Electronics and Computer Engineering , Indian Institute of Technology, Roorkee, for his inspiration, guidance, constructive criticisms and encouragement throughout this dissertation work.

I would like to thank **Ms Prerana Gupta**, Ph.D. student, for sharing her knowledge and for her constant support during this course of dissertation work.

Thanks are due to the Lab staff Signal Processing Lab, Department of electronics and Computer Engineering, IIT Roorkee for providing necessary facilities.

I am greatly indebted to all my friends, who have graciously applied themselves to the task of helping me with ample morale support and valuable suggestions. Finally, I would like to extend my gratitude to all those persons who directly or indirectly helped me in the process and contributed towards this work.

Lt Col A P Verma

ABSTRACT

The idea of software radio requires an expansion of Digital Signal Processing (DSP) towards the antenna. Hence, for converting the received signal to baseband, the need of efficient high speed digital down converters (DDC) arises. DDC performs the frequency translation necessary to convert the high sample rates down to lower sample rates for further and easier processing. It is identified as one of the 'critical functionalities' because it has to run at a relative high sample rate, and has to provide high resolution.

For this purpose, a class of digital linear phase finite impulse response (FIR) filters for decimation (sampling rate decrease) is proposed. They require no multipliers and use limited storage making them an economical alternative to conventional implementations for certain applications. A digital filter in this class consists of cascaded integrator-comb (CIC) stages operating at a high sampling rate and an equal number of comb stages operating at a low sampling rate.

Since their inception, CIC filters have become an important building block for DSP systems. They have found a particular niche in digital transmitters and receivers. They are currently used in highly integrated chips from Intersil, Graychip, Analog Devices, as well as other manufacturers and custom designs.

CONTENTS

CANDIDATE'S DECLARATION	i
ACKNOWLEDGEMENTS	ii
ABSTRACT	iii
1 CHAPTER 1 INTRODUCTION	1
1.1 SDR Architecture	4
1.1.1 An Ideal Software Defined Radio Architecture	4
1.1.2 Hardware Architecture	7
1.1.3 Critical SDR Components	8
1.2 Latest Trend	12
1.3 Future Trend	13
1.4 Statement of the Problem	14
1.5 Organization of the Report	14
2 CHAPTER 2 ELEMENTS OF SDR	16
2.1 Main Blocks of SDR	16
2.1.1 Intelligent Antenna Technology	16
2.1.2 Programmable RF Modules	18
2.1.3 Digital Signal Processing Techniques	19
2.1.4 Interconnect Technology	20
2.2 Digital-to-Analog/ Analog-to-Digital Conversion	21
2.3 Upconversion and Downconversion	31
2.3.1 Digital Upconverter (DUC)	31
2.3.1.1 CFIR and PFIR	32

2.3.2	Digital Down Converter (DDC)	32
3	CHAPTER 3 DIGITAL DOWNCONVERTER	37
3.1	Decimation	37
3.1.1	Multistage Design Of Decimator	38
3.1.2	Poly Phase Realization	39
3.1.3	Computationally Efficient Interpolator-Decimator	40
3.1.4	Software Radio Issues in Cellular Base Stations	42
3.2	CIC Decimation Filter	44
3.2.1	CIC Filter Description	44
3.2.2	Frequency Characteristics	46
3.3	Design Considerations for CIC Decimation Filter	49
3.3.1	Register Growth	49
3.3.2	Truncation and Rounding	53
4	IMPLEMENTATION AND RESULTS	57
4.1	General Description	57
4.1.1	PN Sequence	58
4.1.2	Vector Signal Generator (SMIQ02B)	59
4.1.2.1	Features of Vector Signal Generator	59
4.1.2.2	Applications of Vector Signal Generator	59
4.1.2.3	RF Characteristics	60
4.1.3	ICS-660 B Digital to Analog Converter	60
4.1.4	ICS 652 Analog to Digital Converter	61
4.1.4.1	ADC section	62
4.1.4.2	Sampling Clock	63
4.1.4.3	FPDP interface	63
4.1.4.4	PCI bus interface	63
4.1.5	Design Consideration for CIC Filter	64
4.1.5.1	Design of CIC Decimation Filter	65

4.1.6	Design of Chebyshev Filter	66
4.2	Application Software	69
4.3	Results	73
4.3.1	PN Sequence Generation	73
4.3.2	Output of Vector Signal Generator SMIQ-02B	73
4.3.3	Output of ICS-652 ADC card	74
4.3.4	CIC Decimation Filter	75
4.3.5	Chebyshev Low Pass Filter	76
4.4	Conclusion	77
4.4.1	Nonrecursive CIC Decimation Filters	78

REFERENCES

APPENDIX: Code Listing

CHAPTER 1

Introduction

The term "Software Radio" was coined by Joseph Mitola in 1991 to signal the shift from the hardware-intensive digital radios to the multi-band multimode software-based radios. It refers to the class of reprogrammable or reconfigurable radios. The Software Defined Radio (SDR) Forum defines the Ultimate Software Radio (USR) as a radio that accepts fully programmable traffic and control information and supports a broad range of frequencies, air-interfaces, and applications software. The user can switch from one air-interface format to another in milliseconds, use the Global Positioning System (GPS) for location, store money using smartcard technology, or watch a local broadcast station or receive a satellite transmission. A radio that includes a microprocessor or Digital Signal Processor (DSP) does not necessarily qualify as a software radio. However, a radio that defines in software its modulation, error correction, and encryption processes, exhibits some control over the RF hardware, and can be reprogrammed is clearly a software radio.

A good working definition of a software radio is a radio that is substantially defined in software and whose physical layer behavior can be significantly altered through changes to its software. The term software radio generally refers to a radio that derives its flexibility through software while using a static hardware platform. On the other hand, a soft radio denotes a completely configurable radio that can be programmed in software to reconfigure the physical hardware. In other words, the same piece of hardware can be modified to perform different functions at different times, allowing the hardware to be specifically tailored to the application at hand. Nonetheless, the term software radio is sometimes used to encompass soft radios as well. In short, software radios represent a paradigm shift from fixed, hardware-intensive radios to multi-band, multimode, software-intensive radios.

A model of a practical software radio is shown in Figure 1.1. The receiver begins with a smart antenna that provides a gain versus direction characteristic to minimize interference, multipath, and noise [1]. The smart antenna provides similar benefits for the transmitter. Most practical software radios digitize the signal as early as possible in the receiver chain while keeping the signal in the digital domain and converting to the analog

domain as late as possible for the transmitter using a Digital to Analog Converter (DAC). Often the received signal is digitized in the Intermediate Frequency (IF) band.

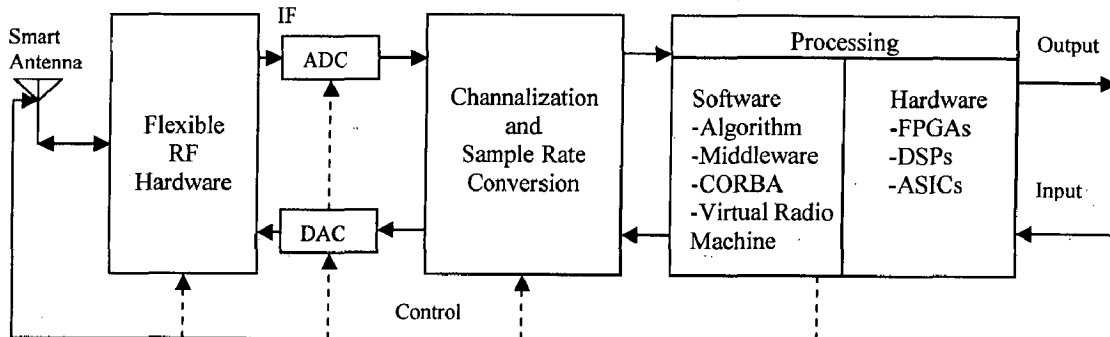


Figure 1.1: Practical Software Radio

Conventional radio architectures employ a super heterodyne receiver, in which the Radio Frequency (RF) signal is picked up by the antenna along with other spurious/unwanted signals, filtered, amplified with a Low Noise Amplifier (LNA), and mixed with a Local Oscillator (LO) to an IF. Depending on the application, the number of stages of this operation may vary. Digitizing the signal with an Analog to Digital Converter (ADC) in the IF range eliminates the last stage in the conventional model in which problems like carrier offset and imaging are encountered. When sampled, digital IF signals give spectral replicas that can be placed accurately near the baseband frequency, allowing frequency translation and digitization to be carried out simultaneously. Digital filtering (channelization) and sample rate conversion are often needed to interface the output of the ADC to the processing hardware to implement the receiver. Likewise, digital filtering and sample rate conversion are often necessary to interface the digital hardware that creates the modulated waveforms to the digital to analog converter.

Processing is performed in software using DSPs, Field Programmable Gate Arrays (FPGAs), or Application Specific Integrated Circuits (ASICs). The algorithm used to modulate and demodulate the signal may use a variety of software methodologies, such as middleware, e.g., Common Object Request Broker Architecture (CORBA), or virtual radio machines, which are similar in function to JAVA virtual machines. This forms a typical model of a software radio. The software radio provides a flexible radio

architecture that allows changing the radio personality, possibly in real-time, and in the process somewhat guarantees a desired Quality of Service (QoS). The flexibility in the architecture allows service providers to upgrade the infrastructure and market new services quickly. This flexibility in hardware architecture combined with flexibility in software architecture, through the implementation of techniques such as object-oriented programming and object brokers, provides software radio with the ability to seam-lessly integrate itself into multiple networks with widely different air and data interfaces. In addition, software radio architecture gives the system new capabilities that are easily implemented with software. For example, typical upgrades may include interference rejection techniques, encryption, voice recognition and compression, software-enabled power minimization and control, different addressing protocols, and advanced error recovery schemes. Such capabilities are well-suited for 3G and 4G wireless requirements and advanced wireless networking approaches. With radio functionality embedded and processed within software modules, there are numerous advantages associated with the SDR technology [2]. These are listed as under:

- *Flexibility & Upgradeability* is the greatest advantage on the implementation of SDR systems. SDR is based on open architecture and consists of a common, generic hardware platform, which allows for flexible installment of different software applications as required for signal transmission. These generic hardware platforms might be used for support of different protocols, services and products. The results are multiband, multimode radio systems able to conform to various protocols such as AMPS, TDMA, CDMA or GSM, which are the most, used air-interface standards currently
- *Interoperable* systems facilitate reduction of the time-to-market duration, since during transition phases to new technologies, legacy and new standards can coexist until the market's final acceptance of the new standard, enabling faster integration of new technologies and resolving the tyranny of legasets.
- *Adaptability* is faster migration towards new standards and technologies through programmability and reconfiguration. Especially toward the evolution of future standards supporting high-speed data services, such as GPRS, EDGE and 3G, software defined base stations will be of advantage enabling *integration* of multiple protocols and dynamic capacity shifting between services as required. The positive impact on future capital costs

for service providers through less required infrastructure deployment could eventually cause lower service charges for subscribers.

- Since the need for hardware modifications and replacements is being eliminated through SDR, provider equipment can be used for an extended period of time as insertion and reconfiguration of new standards becomes much easier, either through over-the-air uploads or directly onsite.
- Less fixed hardware components also means *less maintenance*, as well as the utilization of a generic hardware platform results in *cheaper equipment costs*. SDR also provides the advantage of promoting a more efficient use of the spectrum, for example, the requirement to provide backhaul through leased lines or fixed microwave facilities in traditional architectures presents a huge cost factor for wireless service providers, which gets eliminated by the use of SDR.
- SDR integrated base stations present *great opportunities* to both mobile and cellular services providers as well as subscribers. The same advantages will be available in mobile handsets, such as in cellular phones, PDAs, laptops or other handheld devices. This includes accommodation to multiple communication standards and air-interface protocols, migration capabilities to new emerging standards through software downloads and programmability through software modules. Numerous applications such as cellular phone services, web browsing, email, global positioning or video conferencing could be integrated into one system. Although these kinds of highly convergent systems are not even close to being realistic at the moment, the fact that SDR carries these great potentials makes it even more exiting to follow its future path.

1.1 SDR ARCHITECTURE

1.1.1 An Ideal Software Defined Radio Architecture

The ideal software radio architecture is shown in Figure 1.2. It consists of a digital subsystem & a simple analog subsystem [3]. The analog functions are restricted to those that cannot be performed digitally-that is antenna, RF filtering, RF combination, receive preamplification, transmit power amplification & reference frequency generation. The architecture pushes the analog conversion stage right up as close as possible to the antenna, in this case prior to the power amplifier (PA) in the transmitter and after the LNA in the receiver. The separation of carriers & up down frequency conversion to

baseband is performed by the digital processing resources. Similarly the channel coding & modulation functions are performed digitally at baseband by the same processing resources. Software for the ideal architecture is layered so that the hardware is completely abstracted away from the application software. A middleware layer achieve this functionality by wrapping up the hardware elements into objects and providing services that allow the objects to communicate with each other via a standard interface- for example, CORBA. Middleware includes the OS, hardware drivers, resource management, & other non-application-specific software. The combination of hardware & middleware is often termed a Framework.

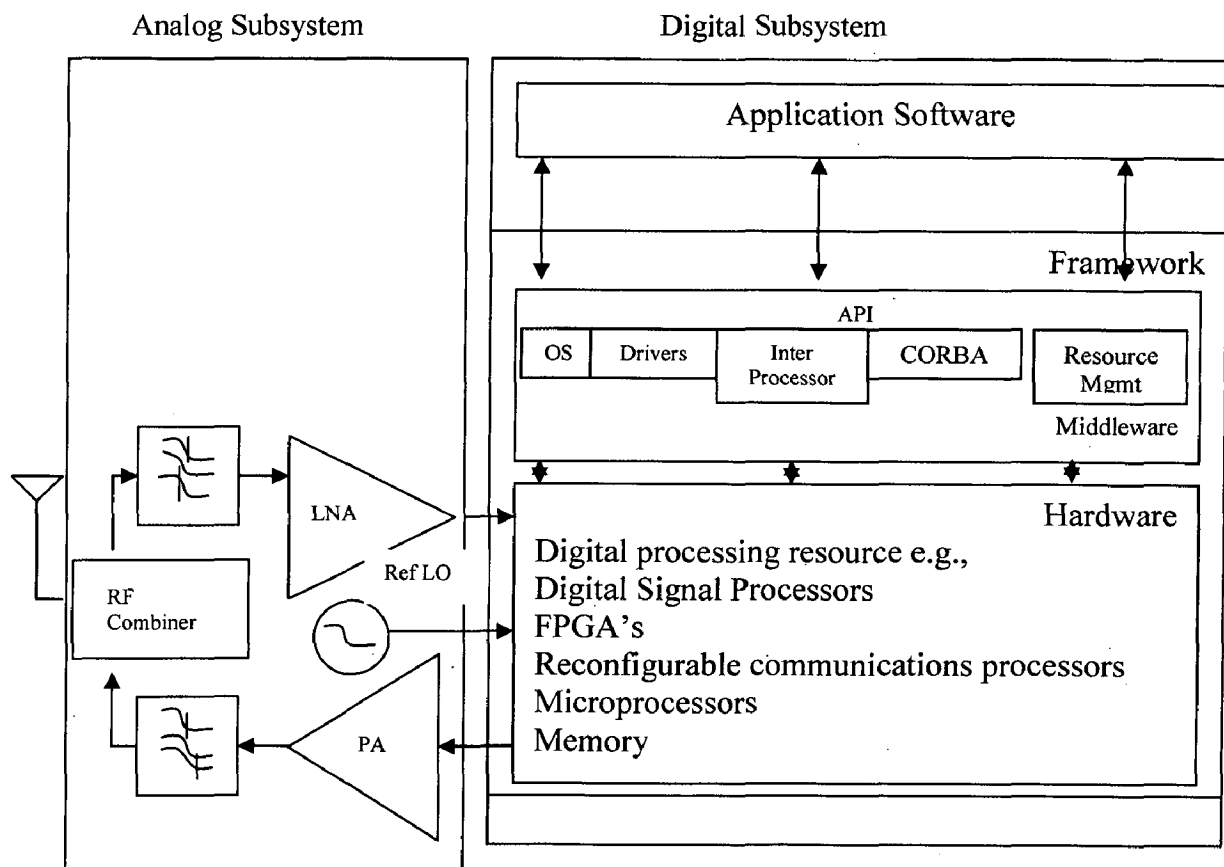


Figure 1.2: Ideal Software Defined Radio with Layered Hardware & Software

SDR architecture is based upon a high-level generic model with specific functional blocks connected via open interface standards recommendations. The software is implemented by controlling the characteristics of equipment/device subsystems through hierarchical and peer level modules that support scalability and flexible extensions of applications. Modularity is the key to successful implementation of

software applications within open systems. Between modules are defined interfaces that are subject to standardization. Within a module the developer is free to implement functionality in the most effective way. Figure 1.3 illustrates a high-level hierarchical functional model for SDR systems. Three views of increasing complexity are presented. The top-level view is a simple representation of an entire information transfer thread. The left side interface is the air interface. The right side interface is the wire side and user interface. The next level view identifies a fundamental ordered functional flow of four significant and necessary functional areas; Front end processing, Information security, Information processing, and Control. The diagrams and processes here are two-way devices i.e. send and receive. The functional model as shown in this figure does not show data or signal flow. These can be explained as follows:

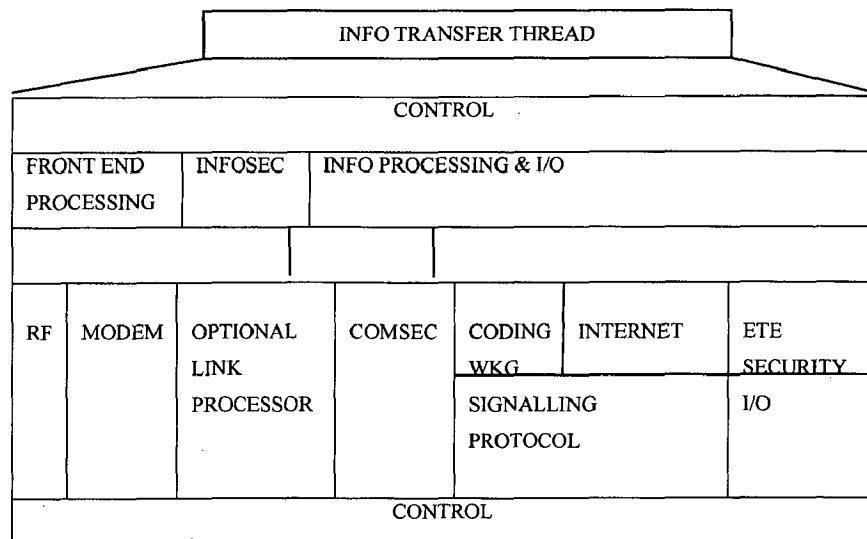


Figure 1.3: Hierarchical Functional Model for Software Defined Radio

- Front End Processing-** It is that functional area of the end user device that consists generically of the physical air (or propagation medium) interface, the front-end radio frequency processing, and any frequency up and down conversion that is necessary. Also, modulation/demodulation processing is contained in this functional block area.

- **Information Security-** It is employed for the purpose of providing user privacy, authentication, and information protection. In the commercial environment, this protection is specified by the underlying service standard while in the defense environment, this protection is of a nature that must be consistent with the various Governmental doctrines and policies in effect.
- **Content or Information Processing-** It is for the purpose of decomposing or recovering the embedded information containing data, control, and timing. Content processing and I/O functions map into path selection (including bridging, routing, and gateway), multiplexing, source coding (including vocoding, and video compression/expansion), signaling protocol, and I/O functions.

1.1.2 Hardware Architecture

The SDR contains a number of basic functional blocks. The radio can be split into three basic blocks, namely the front end, the IF section and the base-band section as shown in the figure 1.4. Each of the sections undertakes different types of functions and therefore is likely to use different circuit technologies.

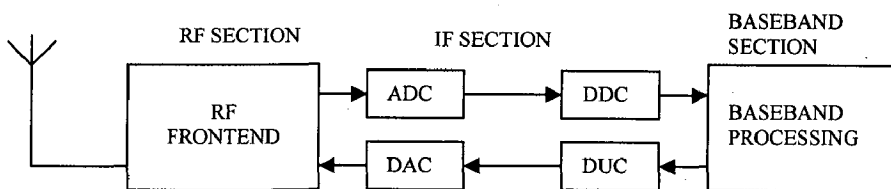


Figure 1.4: Block Diagram of Generic Software Defined Radio

- *The front end section* uses analogue RF circuitry and it is responsible for receiving and transmitting the signal at the operational frequency, coupling the radio to the antenna or its feeder. It also changes the signal to or from the intermediate frequency. Thus on the receive path the front end server is connected to the antenna input using matching circuitry to ensure the optimum signal transfer occurs. It then amplifies the signal and applies it to a mixer with a signal from a local oscillator to down-convert it to the intermediate frequency. On the transmit path the front end takes the signal from the IF, first up-converting it to the final frequency where it is amplified to the required level, passed through suitable matching circuitry to ensure the maximum power transfer and

then presented at the antenna connection to be routed to the antenna either directly or via a feeder.

- The *IF section* performs the digital to and from analogue conversions. It also contains the processing that undertakes what may be thought of as the traditional radio processing elements, including filtering, modulation and demodulation and any other signal processing that may be required. On the receive path the signal enters the DAC where it is digitized and enters the Digital Down Converter (DDC), where the signal is processed and demodulated to provide the baseband signal for the baseband processor. Similarly on the transmit side the signal arrives from the baseband processor and is modulated onto the carrier and conditioned as required. It is then converted from its digital format to analogue using a digital to analogue converter. The DDC and DUC require significant levels of processing. This is required to perform all the processing on the actual signals in digital format. This processing must be achieved in real time for the system to be able to operate satisfactorily. As a result the processors are implemented in either stock DSPs or Application Specific Integrated Circuits (ASICs). In fact to achieve the full programmability and reconfigurability needed for a software defined radio the signal processors may be implemented as Field Programmable Gate Arrays (FPGAs). In this way the circuit can be totally reconfigured if needed.

- The final stage of the radio is the *baseband processor*. It is at this point that the digital data is processed, with protocols being accommodated and the data payload assembled or disassembled from the datastream. Although not as demanding as the DDC and DUC areas, with protocols becoming ever more complicated and demanding, the level of processing is increasing in these areas as well.

1.1.3 Critical SDR Components

It is inarguable that in an ideal setting, the digital conversion of the desired signal would happen immediately upon entry into the system, though numerous road blocks currently prevent a direct-to-digital scenario. There has been growing debate among experts over the most effective combination of ASIC, FPGA, and DSP to handle the steep system requirements for advanced wireless systems. All of these components are essential for operation, but the actual need for each component will be based upon needs assessment of the users. These can be further elaborated as follows:

1. The heart of the software defined radios rests in the *Digital Signal Processing* units. These components implement upper layer protocols in a real-time format, and are manipulated by onboard software, allowing for the interpretation of a variety of signals by simply activating the appropriate software. Using a complex series of algorithms, DSP chips perform functions like channel and source encoding/decoding, filtration, error checking, and modulation/demodulation procedures. Containing a high-speed processing block known as a MAC (Multiply and Accumulate), a DSP processes the signal by retrieving instructions and data from memory, performs requested operations, and stores the results back to memory. The major drawback of DSP units is that they lack the proper processing speeds required for wideband transmissions. To compensate for inadequate performance, some designers suggest that running two DSPs in parallel will result in sufficient function for SDR handsets, and can possibly accommodate the necessary encoding/decoding and symbol processing without the need of ASIC units.

Functions which are often used in wireless applications and are not supported by traditional DSPs are: complex arithmetic, division (such as needed for automatic gain control), coordinate rotation (useful for constellation-based algorithms in a receiver), and square root [4]. Done on a traditional DSP architecture, a combined division and square root operation requires $5N + 12$ cycles for N-bit operands. However, coordinate rotation digital computer (CORDIC)-based approaches to these functions may provide significant speedup of execution.

2. *ASIC* performs signal downconversion, digital filtration, and performs at higher rates of speed than FPGAs. Traditionally smaller in size, ASIC chips are designed for specific purposes, and therefore, are not reprogrammable. This is almost a blessing in disguise, resulting in a dedicated device which has a far more efficient power requirement than general purpose processors. Due to the operation-specific nature of ASIC chips; each chip must be designed for a specific application, and can then be reproduced rather cost effectively.

3. *FPGAs* are completely programmable devices which can perform a variety of user-defined tasks including digital down-conversion, signal processing, and

filtration. Although FPGAs operate slower than ASIC circuits, in terms of embedded memory both are comparable devices. However, when it comes to logic, FPGAs only have a small fraction of the capacity that can be found in ASICs. Many advocate the use of FPGA chips in SDR applications since they can be implemented “off the shelf,” unlike ASICs which need to be created for each specific product. It is suggested that FPGA chips can be integrated along side a DSP, handling a majority of operations; allowing the DSP to operate closer to the level of performance required for wideband applications by leaving only symbol processing to the DSP. Systems which contain both FPGA and DSP chips can deliver signal-processing capabilities ten times more efficiently than with a DSP alone, presenting the welcomed advantages in cost, power, and system footprint as shown in figure 1.5.

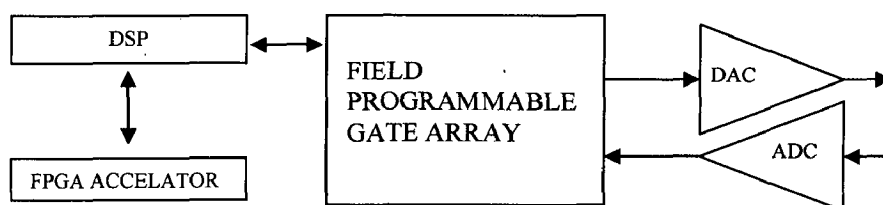


Figure 1.5: Hypothetical Combination of Microprocessors in SDR

4. *Configurable Computing Machine (CCM)* ensures the performance enhancement. The advent of software radios has brought a paradigm shift to radio design [5]. A multimode handset with dynamic reconfigurability has the promise of integrated services and global roaming capabilities. However, most of the work to date has been focused on software radio base stations, which do not have as tight constraints on area and power as handsets. Base station software radio technology progressed dramatically with advances in system design, adaptive modulation and coding techniques, reconfigurable hardware, A/D converters, RF design, and rapid prototyping systems, and has helped bring software radio handsets a step closer to reality. However, supporting multimode radios on a small handset still remains a design challenge. A configurable computing machine, which is an optimized FPGA with application-specific capabilities, shows promise for software radio handsets in optimizing hardware implementations for heterogeneous systems.

Software radios began to obtain popularity largely in base stations rather than handsets. This is due to the fact that unlike handsets, base stations do not have stringent constraints on power, form factor, and weight. Moreover, when the technology was at its inception, base stations provided the perfect platform for experimenting and evaluating different radio designs. However, as the technology matures, software radio designs are migrating to handsets where their true potential can be realized. The additional constraints placed by handsets with limited resources pose one of the main design challenges.

An optimal software radio solution should be able to tune its hardware to the needs of the current system, thereby minimizing power and silicon area to achieve performance similar to an ASIC implementation. Since such optimal systems are still hard to develop; the designer's goal is to maximize the number of supported radio functions with a minimum amount of hardware. Hardware paging presents an elegant solution to this problem, by allowing algorithms to be implemented on physical hardware as needed. While DSPs and FPGAs provide different levels of reconfigurability and aid in implementing flexible designs, CCMs, the latest in reconfigurable technology, is the only class of processor that can currently provide real-time paging of algorithms on hardware, along with partial reconfigurability. Furthermore, in addition to supporting real-time hardware paging, correctly designed CCMs can provide performance enhancements. Extending the flexibility of software radios through the use of CCMs to alter the hardware can render the radio truly "soft." When compared to ASICs and DSPs, CCMs allow designers to achieve flexibility in hardware. CCMs have static hardware for frequently used cores, such as multiplication, filtering, or other communications-oriented algorithms, which result in efficient radio designs. CCMs attempt to customize FPGAs so that system flexibility is retained while taking advantage of the specific properties of communications-oriented cores. CCMs are regarded as extremely powerful processing engines with ASIC-like speeds, which can also be rapidly reconfigured. However, they are not ideally suited for control operations since they lack a microprocessor-like architecture with flexible event-driven operation. Since

control operations can be efficiently programmed on a DSP, CCMs will be most beneficial as co-processors rather than standalone processors. Software radio handsets require extremely flexible and efficient low power hardware. While DSPs and FPGAs provide reconfigurable alternatives for complex high-speed processing, the use of CCM cores can greatly enhance the performance by optimizing the hardware for the application. CCMs show promise in realizing handheld software radios, and with careful design and good simulation tools, it is possible to design CCMs optimized for mobile communications.

1.2 LATEST TREND

Radio technology as we know it is undergoing sweeping changes [6]. Today, even with advancements in RF design and the powerful digital processing available, all radio receivers still use analog parts to tune the radio to a specific carrier frequency. But this is undergoing a change. The Federal Communications Commission (FCC), the SDR Forum and the radio industry are united in the pursuit of the ideal SDR. According to the FCC, "In a SDR, functions that were formerly carried out solely in hardware, such as the generation of the transmitted signal and the tuning and detection of the received radio signal, are performed by software that controls high-speed signal processors." The SDR Forum defines an SDR device as one that functions independently of carrier frequencies and can operate within a range of transmission protocol environments. But the SDR Forum goes a step further by defining the ideal SDR as one that has transceivers that perform upconversion and downconversion between baseband and the RF carrier itself exclusively in the digital domain, reducing the RF interface to a power amplifier in the transmit path, a low noise amplifier in the receive path, and little or no analog filtering. In this ideal radio, it is possible to upgrade or completely change the features by simply uploading new software. This ideal radio defined by the SDR Forum has, until recently, been unachievable due to the lack of very high-frequency RF to digital converters capable of converting carrier frequencies directly to digital data. Now new integrated circuit processes are offering higher speed and lower power. State-of-the art IC design is being applied to these new processes to enable RF to digital conversion directly on carrier

frequencies above 5 GHz. Recently, Vanu Inc. became the first company to successfully complete the FCC's certification process governing software-radio devices. The Vanu Software Radio consists entirely of software applications that support all of the GSM cellular base station functionality running on off-the-shelf servers with RF subsystem. This system can support multiple radio standards, protocols and frequencies on a single industry standard server. It allows modification of the RF planning and assignment of standards through remote software parameter changes. It supports upgrades to new standards through a software download. The availability of high-speed converters has made possible the implementation of true software-radio transmitters and receivers that avoid the use of passive components to tune and modulate/demodulate high-speed wireless signals. These converters, using conventional CMOS and advanced semiconductor and superconductor technologies have performance comparable to or better than conventional analog front-end circuits and enable tunerless architectures wireless transmitters and receivers. Next-generation devices promise to further exceed the performance of analog components enabling the extraction of weak signals in noisier and more interference-ridden environment.

1.3 FUTURE TREND

Reconfigurability in radio development is not such a new technique [7]. Already during the 1980s reconfigurable receivers were developed for radio intelligence in the short wave range. These receivers included interesting features like automatic recognition of the modulation mode of a received signal or bit stream analysis. A *cognitive radio* (CR) is an SDR that additionally senses its environment, tracks changes, and reacts upon its findings. A CR is an autonomous unit in a communications environment that frequently exchanges information with the networks it is able to access as well as with other CRs. Formally, a Cognitive Radio is an intelligent wireless communication system that is aware of its surrounding environment (i.e., outside world), and uses the methodology of understanding-by-building to learn from the environment and adapt its internal states to statistical variations in the incoming RF stimuli by making corresponding changes in certain operating parameters (e.g., transmit-power, carrier-frequency, and modulation strategy) in real-time, with two primary objectives in mind:

- Highly reliable communications whenever and wherever needed;

- Efficient utilization of the radio spectrum.

Six key words stand out in this definition: awareness, intelligence, learning, adaptively, reliability, and efficiency. Implementation of this far-reaching combination of capabilities is indeed feasible today, because of the spectacular advances in digital signal processing, networking, machine learning, computer software, and computer hardware. In the years since Joseph Mitola coined the term “cognitive radio,” there has been much speculation about how an intelligent, aware, adaptive, and learning radio could sweep away old ideas about spectrum access and spawn a new era in wireless communications. Cognitive radio researchers are making significant progress in the development of radios that can observe their environment, take action, and learn from the results. While the boundary between “adaptive” and “cognitive” is ill defined, real radio performance is moving toward Mitola’s vision.

1.4 STATEMENT OF THE PROBLEM

Today’s continuously changing technology brings the need to build “futureproof” radios. If the functions that were formerly carried out by hardware can be performed by software, new functionality can be deployed on a radio by updating the software running on it. The aim of the present work is implementation of Digital Down Conversion for Virtual Radio. Vector Signal Generator is used for generating modulated signal, which is digitized using ADC card. This captured signal is then processed using software to recover the baseband signal.

1.5 ORGANIZATION OF THE REPORT

This thesis is presented in four chapters and is organized as follows:

Chapter 2 explains the various elements of the SDR, covering in detail the ADC/DAC, with due emphasis on Delta-Sigma type of ADC.

Chapter 3 elucidates the decimation rate technique, the *Digital Downconverter* (DDC) that converts the received signal to baseband, a class of linear phase FIR filters called cascaded integrator-comb (CIC) filters that are used for decimation, and finally the *Low Pass Filter*, which is a *Chebyshev* Low Pass Filter used to filter out high frequency components.

Chapter 4 entails the implementation of the dissertation work using the facilities available in the Signal Processing Lab. It also elucidates the results obtained thereafter and finally the conclusion at the end.

CHAPTER 2

Elements of SDR

The choice of building blocks and communications infrastructure are the most critical part of any SDR for both commercial and military applications. They both share similar challenges towards implementation of a true SDR.

In this chapter, the focus is on the hardware technology aspect of the SDR. The main technical objective of a SDR is a versatile reconfigurable platform, which provides interoperability. As depicted in figure 1.1, the practical SDR diagram, the main blocks of a true SDR are namely: intelligent antenna, programmable RF modules, high-performance Digital-to-Analog (DAC) and Analog-to-Digital Converters (ADC), Digital Signal Processing (DSP) techniques/technology and the interconnect technology. In the following sections of the chapter, the current state of the technology and the trend for each of the five mentioned block is briefly reviewed and discussed.

2.1 MAIN BLOCKS OF SDR

As discussed above, SDR comprises of five main blocks. Each block has to provide a considerable amount of flexibility while maintaining performance. They are covered in detail as under:

2.1.1 Intelligent Antenna Technology

Antenna front-end can be split to the basic antenna element and the related array configuration and processing blocks. Today's antenna solutions for SDR systems are based on several separate antennas to cover a broad range of frequency. This is because, the traditional antenna design techniques limits the extent of application and performance of an antenna. In other words, since the antenna size is always proportional to the operating wavelength, a wideband antenna will be less performing and less cost effective than a narrowband antenna. Thus the main activities in this domain remain mainly in array processing blocks and techniques to make the antenna system more performing and intelligent. The word intelligent covers the very basic feature of frequency band selection and adaptation up to advance capabilities such as interference cancellation and mobile tracking. An ideal antenna for a SDR is a self-adapt, self-align and self-healing antenna, which is capable of complete adaptation to its required application and the transmission

environment. Self-adaptability is a capability that the antenna system would adjust its basic parameters such as gain according to the selected frequency band and required system gain of the application. As soon as the system is set up, self-alignment enables the antenna to direct itself based on the maximum signal reception with an aid provided by a telemetry data or a Global Positioning System (GPS). Self-healing can be defined as a feature that helps the system to combat any friendly or hostile interference by proper use of techniques such as array processing, beam-steering and even variable-polarization.

In order to fulfill the above-mentioned features, besides the required elegant signal processing techniques, high performance and flexible radiating elements are essential. Micro Electro-Mechanical System (MEMS) raises the hope for major breakthroughs in the field of broadband reconfigurable antenna design. For example, by employing MEMS, it is possible to change the operating frequency of a rectangular ring slot antenna electronically. In a square slot antenna, a good performance at a frequency is obtained, when the circumference of the ring is approximately one wavelength. By using MEMS switches, as shown in Figure 2.1, in order to reconfigure the antenna for a new frequency band, it is only required to switch in or out different slot elements. The main advantages of MEMS switch over PIN diode switch are low loss, high isolation and smaller size. They are all significant factors for an antenna system.

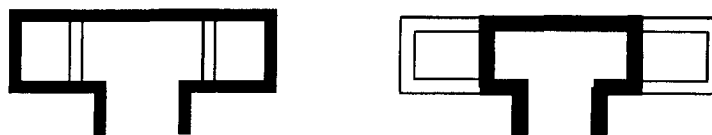


Figure 2.1: Application of MEMS switch for reconfigurable antenna

In current cellular systems with omnidirectional or sectorized antennas, interference limits the choice of frequencies in each cell to one out of every seven. Preliminary analyses indicate that software-radio-based “Smart Antenna” appliquéés can enhance the carrier to interference (C/I) ratio in both the forward and reverse link to allow frequency reuse of 3 to 1. Use of this type of a smart array represents a greater than 2 to 1 increase in capacity and supports a 3 to 1 increase in call arrival rate for the same probability of blocked calls [8]. As the demand for radio spectrum grows, the ability of sophisticated DSP algorithms to combat multipath fading and to reduce interference will become increasingly valuable as a means of adding capacity to cellular and PCS systems. In the

future, software radios will be the technology of choice in an expanding array of mobile radio applications. Today, the real need is to make more voice channels available in a given geographic area in the finite spectrum allocated. When translating this operational need to a technical requirement for an antenna system, the need is not for a steerable “higher gain” array, but instead an array which can increase the C/I ratio at both the base station and the mobile terminal. If this can be achieved, then more RF channels can be added to the base stations in the area without driving the C/I below an acceptable limit. Suppose that the channels in a conventional system without an array have been allocated to maximize capacity subject to a minimum acceptable C/I. If the smart array is employed, the C/I will increase, allowing the addition of more channels until the C/I is again brought to a minimum acceptable level. By increasing the C/I of the desired signal for both the forward and reverse links, the smart array provides more channel capacity.

The antenna(s) of the software radio span multiple bands, with uniform shape and low losses to provide access to available service bands. In military applications, for example, a mobile terminal may need to employ VHF/UHF line of sight frequencies, UHF satellite communications, and HF as a backup mode. Switched access to such multiple bands requires octave bandwidth antennas and/or multiple antennas per band and an agile frequency reference in the RF Segment [9]. The relationship between interference cancellation capacity and the number of antenna elements varies. A single auxiliary element, for example, can reduce interference of a large number of interferers. Algorithms that reduce interference through non-spatial techniques can also reduce a large number of interferers with one or with no auxiliary antenna elements.

2.1.2 Programmable RF Modules

One of the employed techniques for existing SDR systems is to use a bank of RF modules to cover the entire frequency band [10]. By maturing the wideband synthesizer technology, low-noise high-performance semiconductor processes, very flexible RF modules are arising. Implementation of high performance RF devices with a high level of integration of circuits, including switches have been made possible due to low loss and compactness of MEMS technology. The moveable feature of MEMS devices allows dynamic adjustment of the component values. MEMS technology improves the performance and flexibility of several RF components such as:

- Low phase noise Voltage Controlled Oscillators (VCO) by using MEMS-based high Q resonators
- Wideband varactors and phase shifters by using MEMS-based variable capacitors and switch-capacitor networks
- Tunable filters by employing MEMS-based variable reactive elements and switches.

The challenge for designing programmable bandpass filters remains to be resolved. They are essential in both transmitters and receivers, insuring efficient use of channel and high sensitivity. Bandpass filters are usually the most expensive items on the list for RF modules and at the same time the least flexible. A SDR implementation requires these elements to be either electronically configurable or to be stacked to forming a filter bank. Today most of SDR implementations rely on the latter.

Also, superconductive technology has enabled to implement tunable bandpass filters with ultra-sharp roll-off characteristics. The low loss nature of this process enables design and realization of multistage thin film filter with low insertion loss and broadband operation.

2.1.3 Digital Signal Processing Techniques

DSP is the principal enabling element of SDR. Embedded DSP algorithms in the processing engine are responsible to make all the promises of SDR come true. By employing an efficient DSP architecture and set of algorithms, proprietary waveforms and features can be implemented as the applications matures. Among several DSP techniques used in SDR platform, sampling techniques, rate conversion and multirate processing have been instrumental in progress and development of DSP-based systems. Regular implementation choices are based on DSP-processors, ASICs, FPGAs as well as the emerging mixture of these technologies. The most effective combination of Application Specific Integrated Circuits (ASIC), Field Programmable Gate Arrays (FPGA), and Digital Signal Processors (DSP) are used to handle the steep system requirements for advanced wireless systems. All of these components are essential for operation, but the actual need for each component will be based upon needs assessment of the users. These are already discussed in 1.1.3 in detail.

2.1.4 Interconnect Technology

One of the main benefit or objective of the SDR is the ability to connect several independent building blocks to set-up a radio link. An open architecture system obliges a set of interface standards to be developed within framework of an interconnect technology. A successful interconnect approach has to address following critical issues:

- Leveraging open standards
- Addressing multiple protocols
- Meeting increasing speed and throughput requirements
- Connecting to traditional circuit networks.

Mainly, there are three main interconnect architecture: bus architecture, switch fabric architecture and tree architecture. These are briefly discussed as under:

- In *bus architecture*, a bus connects different processing modules. Usually, a high-speed data bus provides high-speed data transmission between the functional units, and a lower speed bus is employed for low rate control data transactions. Based on this architecture, at any time, only one functional unit can transmit data on the bus. In other words, bus is time shared between the functional units. Bus assignment is done by a bus arbitration blocks, which processes requests and sets the priority. Time division limits the bandwidth and also restricts the scalability of the interconnect bus architecture. A new version of PCI, called as PCI-X allows the bus to run up to 133 MHz at 64 bits bus to accommodate a peak data rate of 1 Gbyte/s, however this seems to be still not enough for some applications.
- In *switch fabric architecture*, there is an adapter in each functional unit that packages the data from the functional unit before being sent to the switch fabric. The principle of a switch fabric is similar to an IP or an ATM switch. Switch fabric technology has following advantages compared to the bus architecture:
 - High scalability in both slot count and aggregate performance
 - Flexibility in physical configuration and size
 - Cost-effective, high availability designs with no single points of system failure

- Efficiently addressing the disparate needs of different traffic types and communication protocols.

However it should be noted that the use of switch fabric architecture in SDR calls for strict requirement and performance on the switch fabric in terms of throughput and packet delay.

- In a wireless communication system, the product term of data rate by resolution bits becomes lower and lower as we move from RF to IF and baseband processing. This suggests architecture similar to a *tree*, where from the root node to the leaf node, the channel width becomes wider and wider as shown in figure 2.2. The tree architecture optimizes the use of interconnections and avoids providing interconnects between the blocks that do not require such provision. The main disadvantage of this architecture is its implementation and limited flexibility.

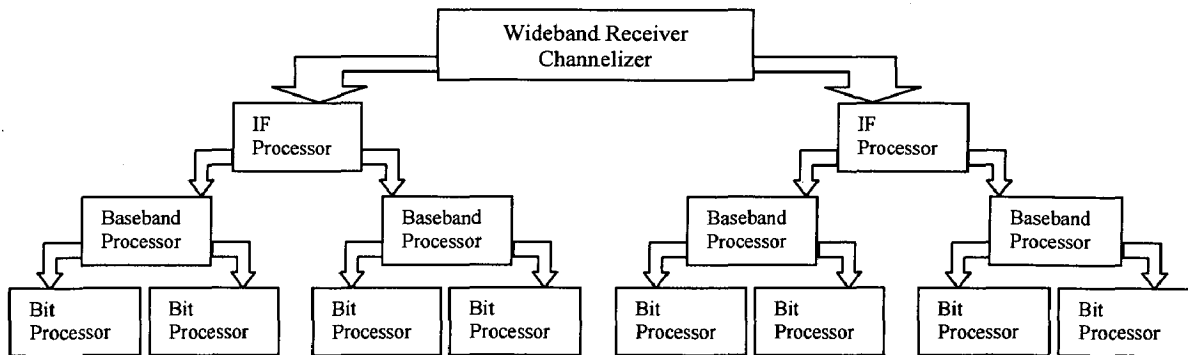


Figure 2.2: Tree Interconnect Architecture

2.2 DIGITAL-TO-ANALOG / ANALOG -TO- DIGITAL CONVERSION

SDR systems hinge on ADCs and DACs components. They have the unique tasks of conversion between analog to digital domain and vice versa. The flexibility of a SDR can be augmented significantly by pushing the converters closer to the antenna. Putting ADC closer to the antenna eliminates multiple stages of downconversion and analog filtering [11]. Traditional electronic converters are pushing the envelope to achieve more resolution and faster conversion rates. The receiver has always more complexity than the transmitter, the importance of operating at higher rates of conversions for ADCs is more

significant and thus has attracted more effort than DACs. The ever-increasing demand from applications such as broadband wireless communication and software radio are stimulating many innovative state-of-the-art ADC developments. Next generation transceiver operating with different standards necessitates the existence of a wide bandwidth and highly linear ADC. Currently, several projects dealing with the design and implementation of ADC architectures to provide the stringent requirements of the wide-bandwidth transreceivers are being developed. The ultimate transceiver target architecture is to perform immediately at the radio frequency (RF) the A/D conversion in order to process the signal completely in the digital domain. In this strategy the ADC should match both high linearity and wide bandwidth. A flash ADC with several GHz bandwidth and more than 6-bit resolution will be designed to meet this specification. The flash ADC is more suitable for base-station transceiver where the whole bandwidth needs to be converted. A different target ADC architecture that is more suitable for handheld devices in terms of power consumption is the bandpass sigma delta. The basic idea of Σ - Δ ADC is to trade sampling speed for number of bits. This is discussed at length subsequently. Here we describe here four major circuit architectures used in A/D converter (ADC) design and outline the role they play in converter choice for various kinds of applications. The most popular ADC architectures available today are successive approximations (sometimes called SAR because a successive-approximations (shift) register is the key defining element), flash (all decisions made simultaneously), pipelined (with multiple flash stages), and sigma-delta (SD), a charge-balancing type.

Conceptually, the *Flash Converters* architecture (illustrated in Figure 2.3) is quite straightforward: a set of 2^n-1 comparators are used to directly measure an analog signal to a resolution of n bits. For a 4-bit flash ADC, the analog input is fed into 15 comparators, each of which is biased to compare the input to a discrete transition value. These values are spaced one least-significant bit ($LSB=FS/2^n$) apart [12]. The comparator outputs simultaneously present 2^n-1 discrete digital output states. If for example the input is just above $1/4$ of full scale, all comparators biased to less than $1/4$ full scale will output a digital '1', and the others will output a digital '0'. Together, these outputs can be read much like a thermometer. The final step is to level-decode the result into binary form.

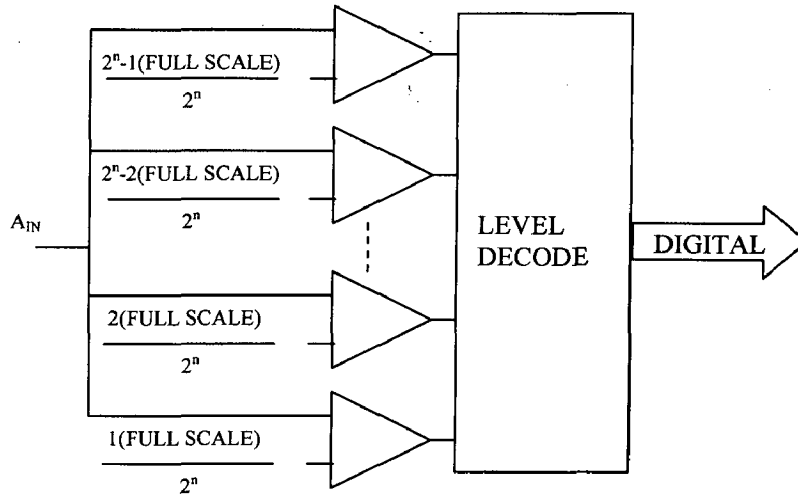


Figure 2.3: Basic flash architecture

The flash architecture has the advantage of being very fast, because the conversion occurs in a single ADC cycle. The disadvantage of this approach is that it requires a large number of comparators that are carefully matched and properly biased to ensure that the results are linear. Since the number of comparators needed for an n -bit resolution ADC is equal to $2^n - 1$, limits of physical integration and input loading keep the maximum resolution fairly low.

Pipelined architecture or pipelined-flash architecture effectively overcomes the limitations of the flash architecture. A pipelined converter divides the conversion task into several consecutive stages. Each of these stages, as shown in figure 2.4, consists of a sample and hold circuit, an m -bit ADC (e.g. a flash converter), and an m -bit DAC. First the sample and hold circuit of the first stage acquires the signal. The m -bit flash converter then converts the sampled signal to digital data. The conversion result forms the most significant bits of the digital output. This same digital output is fed into an m -bit digital-to-analog converter, and its output is subtracted from the original sampled signal. The residual analog signal is then amplified and sent on to the next stage in the pipeline to be sampled and converted as it was in the first stage. This process is repeated through as many stages as are necessary to achieve the desired resolution. In principle, a pipelined converter with p pipelined stages, each with an m -bit flash converter, can produce a high-speed ADC with a resolution of $n = p * m$ bits using $p * (2^m - 1)$ comparators.

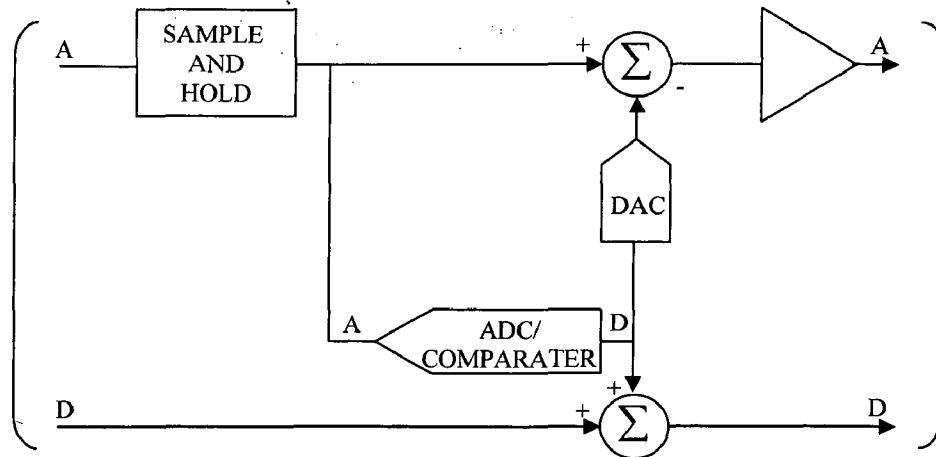


Figure 2.4: A single pipelined converter stage

Pipelined converters achieve higher resolutions than flash converters containing a similar number of comparators. This comes at the price of increasing the total conversion time from one cycle to p cycles. But since each stage samples and holds its input, p conversions can be underway simultaneously. The total throughput can therefore be equal to the throughput of a flash converter, i.e. one conversion per cycle. The difference is that for the pipelined converter, we have now introduced latency equal to p cycles. Another limitation of the pipelined architecture is that the conversion process generally requires a clock with a fixed period. Converting rapidly varying non-periodic signals on a traditional pipelined converter can be difficult because the pipeline typically runs at a periodic rate.

Successive approximations architecture can be thought of as being at the other end of the spectrum from the flash architecture. While a flash converter uses many comparators to convert in a single cycle; a SAR converter, shown in figure 2.5, conceptually uses a single comparator over many cycles to make its conversion. The SAR converter works like an old-fashioned balance scale. On one side of the scale, we place the sampled unknown quantity. On the other side, we place a weight (generated by the SAR and DAC) that has the value of $\frac{1}{2}$ of full-scale and compare the two values. This first weight represents the most significant bit (MSB). If the unknown quantity is larger, the $\frac{1}{2}$ -scale weight is retained; if the unknown quantity is smaller, it is removed. This series of steps is repeated n times, using successively smaller weights in binary

progression (e.g., $1/4$, $1/8$, $1/16$, $1/32$, ... $1/2^n$ of full scale) until the desired resolution, n , is attained. Each weight represents a binary bit, with the largest representing the most significant bit, and the smallest representing the least significant bit.

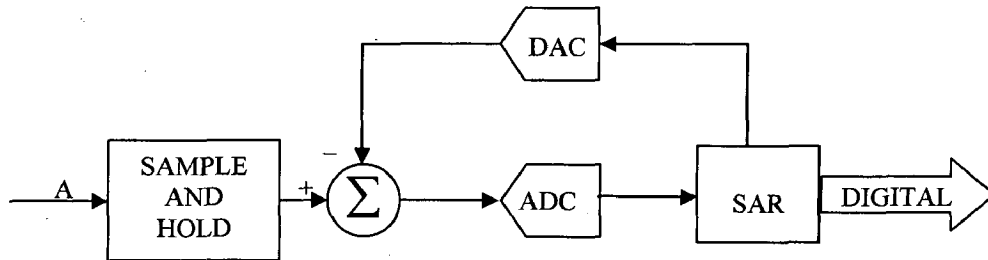


Figure 2.5: Successive-approximations architecture

A SAR converter can use a single comparator to realize a high resolution ADC. But it requires n comparison cycles to achieve n -bit resolution, compared to p cycles for a pipelined converter and 1 cycle for a flash converter. Since a successive-approximations converter uses a fairly simple architecture employing a single SAR, comparator, and DAC, and the conversion is not complete until all weights have been tested, only one conversion is processed during n comparison cycles. For this reason, SAR converters are more often used at lower speeds in higher-resolution applications. SAR converters are also well suited for applications that have non-periodic inputs, since conversions can be started at will. This feature makes the SAR architecture ideal for converting a series of time-independent signals. A single SAR converter and an input multiplexer are typically less expensive to implement than several sigma-delta converters. With dither noise present, SAR and pipelined converters can use averaging to increase the effective resolution of the converter: for every doubling of sample rate, the effective resolution improves by 3 dB or $\frac{1}{2}$ bit.

One consideration when using a SAR or pipelined converter is aliasing. The process of sampling a signal leads to aliasing - the frequency-domain reflection of signals about the sampling frequency. In most applications, aliasing is an unwanted effect that requires a low-pass anti-alias filter ahead of the ADC to remove high-frequency noise components, which would be aliased into the passband. However, undersampling can put aliasing to good use, most often in communications applications, to convert a high-frequency signal to a lower frequency. Undersampling is effective as long as the total bandwidth of a signal meets the Nyquist criterion (less than one-half the sampling rate),

and the converter has sufficient acquisition and signal sampling performance at the higher frequencies where the signal resides. While fast SAR converters are capable of undersampling, the faster pipelined converters tend to be more effective at it.

Sigma-Delta architecture takes a fundamentally different approach from those outlined above. In its most basic form, a sigma-delta converter consists of an integrator, a comparator, and a single-bit DAC, as shown in Figure 2.6. The output of the DAC is subtracted from the input signal. The resulting signal is then integrated, and the integrator output voltage is converted to a single-bit digital output (1 or 0) by the comparator. The resulting bit becomes the input to the DAC, and the DAC's output is subtracted from the ADC input signal, etc. This closed-loop process is carried out at a very high "oversampled" rate. The digital data coming from the ADC is a stream of ones and zeros, and the value of the signal is proportional to the density of digital ones coming from the comparator. This bit stream data is then digitally filtered and decimated to result in a binary-format output. One of the most advantageous features of the sigma-delta architecture is the capability of noise shaping, a phenomenon by which much of the low-frequency noise is effectively pushed up to higher frequencies and out of the band of interest. As a result, the sigma-delta architecture has been very popular for designing low-bandwidth high resolution ADCs for precision measurement. Also, since the input is sampled at a high "oversampled" rate, unlike the other architectures described above, the requirement for external anti-alias filtering is greatly reduced. A limitation of this architecture is its latency, which is substantially greater than that of the other types. Because of oversampling and latency, sigma-delta converters are not often used in multiplexed signal applications. To avoid interference between multiplexed signals, a delay at least equal to the decimator's total delay must occur between conversions. These characteristics can be improved in sophisticated sigma-delta ADC designs by using multiple integrator stages and/or multi-bit DACs. Let us now discuss Sigma-Delta converter in detail. In this type of converter, the analog signal is sampled at a rate much higher than the Nyquist rate, resulting in very closely spaced samples [13]. As a consequence, the difference between the amplitudes of two consecutive samples is very small, permitting it to be represented in digital form using very few bits, usually by one bit. The sampling rate is then decreased by passing the digital signal first through a factor

of M decimator to lower the sampling rate from $M F_T$ to F_T . The decimator is designed by cascading an anti-aliasing low pass M th band digital filter to reduce its bandwidth to π/M and a factor of M down-sampler. The wordlength of the downsampler output determines the resolution of the over sampling A/D converter, and it is much higher than that of the high rate digital signal due to the effect of digital filtering. The basic block diagram representation of the Sigma-Delta A/D converter is shown in figure 2.6.

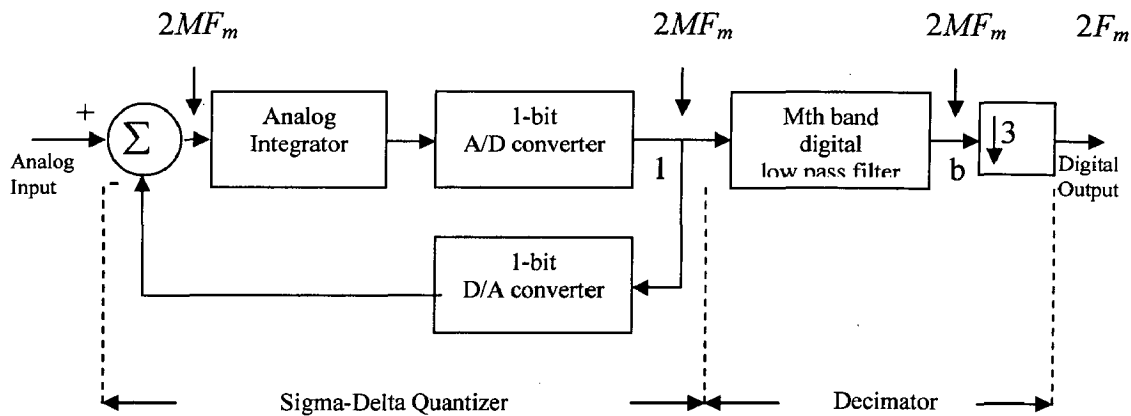


Figure 2.6: Oversampling Sigma-Delta A/D Converter Structure

The figure indicated the sampling rates at various stages of the structure. The 1-bit output samples of the quantizer after decimation become b -bit samples at the output of the Sigma-Delta A/D converter due to the filtering operations involving b -bit multiplier coefficients of the M th band digital low pass filter. Since the oversampling ratio M is typically very large in practice, the Sigma-Delta A/D converter is most useful in low frequency applications such as digital telephony, digital audio and digital spectrum analysers.

To understand the operation of Sigma-Delta A/D converter of the figure 2.6, we need to study the operation of the Sigma-Delta quantizer shown in figure 2.7 (a). To this end it is convenient to use the discrete time equivalent circuit of figure 2.7 (b), where the integrator has been replaced with an accumulator.

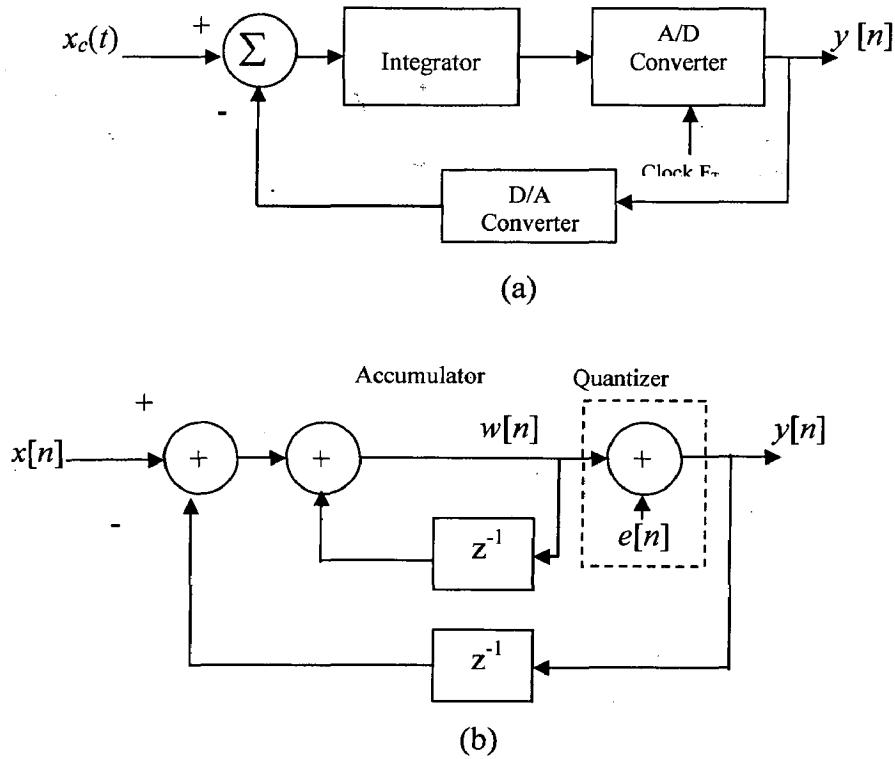


Figure 2.7: Sigma-Delta Quantization Scheme

Here the input $x[n]$ is a discrete time sequence of analog samples developing an output sequence of binary valued samples $y[n]$. From this diagram, we observe that at each discrete instant of time, the circuit forms the difference (Δ) between the input and the delayed output, which is accumulated by a summer (Σ) whose output is then quantized by a 1-bit A/D converter, i.e. a comparator. Even though the input-output relation of the Sigma-Delta quantizer is basically non-linear, the low frequency content of the input $x_c(t)$ can be recovered from the output $y[n]$ by passing it through a digital low pass filter. This property can be easily shown for a constant input analog signal $x_a(t)$ with a magnitude less than +1. In this case, the output $w[n]$ of the accumulator is a bounded sequence with sample values equal to either -1 or +1. This can happen only if the input to the accumulator has an average value of zero. Or in other words, the average value of $w[n]$ must be equal to the average value of the input $x[n]$.

It follows from figure 2.7(b) that the output $y[n]$ of the quantizer is given by

$$y[n] = w[n] + e[n] \quad (2.1)$$

where, $w[n] = x[n] - y[n-1] + w[n-1]$

This gives, $y[n] = x[n] + (e[n] - e[n-1])$

Where the quantity inside the parenthesis represents the noise due to Sigma-Delta modulation. The noise transfer function is simply $G(z) = (1 - z^{-1})$

The power spectral density of the modulation noise is therefore given by

$$P_y(f) = |G(e^{j2\pi fT})|^2 P_e(f) = 4 \sin^2\left(\frac{2\pi fT}{2}\right) P_e(f) \quad (2.2)$$

Where we have assumed the power spectral density $P_e(w)$ of the quantization noise to be the one sided power spectral density defined for positive frequencies only. For a random signal input $x[n]$, $P_e(f)$ is constant for all frequencies and is given by

$$P_e(f) = \frac{(\Delta V)^2 / 12}{F_T / 2} \quad (2.3)$$

Substituting the above in (2.2), we arrive at the power spectral density of the output noise, given by

$$P_y(f) = \frac{2}{3} \frac{(\Delta V)^2}{F_T} \sin^2(\pi fT) \quad (2.4)$$

This can be approximated as

$$P_y(f) \cong \frac{2}{3} \frac{(\Delta V)^2}{F_T} (\pi fT)^2 \quad (2.5)$$

$$= \frac{2}{3} \pi^2 (\Delta V)^2 T^3 f^2, \quad f \ll F_T \quad (2.6)$$

From the above, the in-band noise power of the Sigma-Delta A/D converter is thus given by

$$P_{total, sd} = \int_0^{F_m} P_y(f) df = \frac{2}{3} \pi^2 (\Delta V)^2 T^3 \int_0^{F_m} f^2 df \quad (2.7)$$

$$= \frac{2}{9} \pi^2 (\Delta V)^2 T^3 (F_m)^3 \quad (2.8)$$

The Sigma-Delta A/D converter exhibits an improved noise performance which results from the shape of $|G(e^{j2\pi fT})|$ which decreases the noise power spectral density in band ($0 \leq f \leq F_m$) while increasing it outside the signal band of interest ($f > F_m$). Since this type of converter also employs oversampling, it requires a less stringent analog anti-aliasing filter.

Future SDR receivers should have capability to cover multiple channels each of which have the band width of about 100MHz, so that the band width of the SDR receiver is required to be much greater than 100MHz [14]. The ADC used in the SDR receiver needs to have such a broad band width. When the base stations are constructed, ADCs and digital filters are required for each channel. These architectures have the high flexibility for the change of systems' features such as the modulation scheme, allocated frequency band. Broad-band superconducting RF filters and broad-band ADC are essential for each architecture. The common benefits of using superconducting devices to these architectures come from the employment of the superconductor ADC. As mentioned above, the input bandwidth of greater than 100MHz is needed for a future SDR receiver. However, the band-widths of conventional semiconductor ADCs including sub-ranging ADCs and pipelined ADCs except oversampled ADCs are limited by the clock jitter. Due to this limiting factor, the enhancement rate of the band-width for semiconductor ADCs is suppressed. When we employ the oversampled ADCs, the effect of the clock jitter can be reduced considerably. The superconductor oversampled ADCs based on the single flux- quantum (SFQ) logic can operate at frequencies of several tens of GHz, which is 1- or 2-orders of magnitude higher than the clock frequency of a semiconductor oversampled ADC. This high clock frequency of superconductor ADCs enables to break the barrier of the clock jitter limit and to enhance the input band-width. Another advantage of SFQ-based ADCs is high sensitivity. The minimum power of input signals required to drive SFQ circuits is less than $1 \mu W$, while that of high-speed semiconductor ADCs is around 1mW.

The ADC/DAC blocks perform analog-to-digital conversion (on receive path) and digital-to-analog conversion (on transmit path), respectively. ADC/DAC blocks interface between the analog and digital sections of the radio system. DDC and DUC blocks perform digital-downconversion (on receive path) and digital-up-conversion (on transmit path), respectively. DUC/DDC blocks essentially perform *modem* operations, i.e., modulation of the signal on transmit path and demodulation (also called digital tuning) of the signal on receive path.

2.3 UPCONVERSION AND DOWNCONVERSION

The design of upconverter is somewhat similar to that of downconverter. The notion of up- and down-conversion stands for a shift of a signal towards higher or lower frequencies, respectively. This can be achieved by multiplying the signal $x_a(t)$ with a complex rotating phasor which results in

$$x_b(t) = x_a(t)e^{j2\pi f_c t} \quad (2.9)$$

where f_c stands for the frequency shift, and is called the carrier frequency to which a baseband signal is upconverted, or from which a bandpass signal is downconverted. Digital up- and downconversion is the digital equivalent of (2.9). Depending on the sign of f_c , up- or downconversion results.

The DUC/ DDC can basically have two architectures, one the Direct Conversion Architecture (DCA) and the other is Multiple Conversion Architecture (MCA) [15]. The DCA involves the downconversion directly to the baseband, without converting it to the IF, whereas, MCA performs two explicit downconversions, one in the RF hardware and other in the DSP. This is the best choice for SDR receiver design today due having an edge over the DCA.

2.3.1 Digital Upconverter (DUC)

A direct DUC typically performs the modulation of a baseband signal to an intermediate frequency signal more appropriate for driving a final analog up converter. The low sample rate baseband signal is shaped, interpolated and then mixed with a local oscillator. A block diagram of the digital upconverter core is shown in figure 2.8. Spectral shaping of the complex input signal is performed by the Programmable Finite Impulse Response (PFIR) filter. Typically this filter would be performing a Nyquist transmit filter operation. Bias-free convergent rounding or truncation (selectable by the user) is employed between each processing stage to limit the bit growth through the DUC. Output from each of the PFIR filters is input to each of the Compensation Finite Impulse Response (CFIR) filters, which is used to compensate for the droop within the CIC filter and performs the second rate-change. The CFIR filter's output drives the input to the interpolating CIC filter, which is used for high sample rate changes. The complex data stream from the CIC filter is mixed with a local oscillator generated by the Direct Digital Synthesis (DDS). Results

from the mixers are combined, forming the final DUC result. The DUC result is used as the input to a DAC to generate an intermediate frequency analog signal.

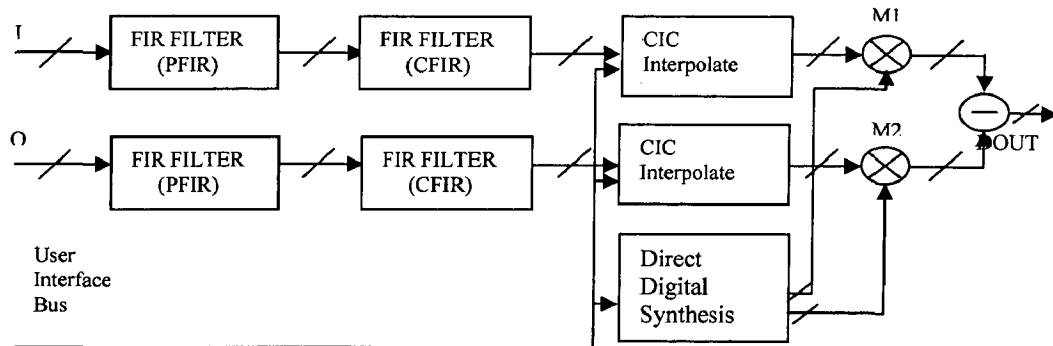


Figure 2.8: DUC Core Architecture Block Diagram

2.3.1.1 PFIR and CFIR

The input data signal stream is filtered by two stages of filtering provided by the PFIR and CFIR FIR filters. The PFIR filter usually performs spectral shaping. The CFIR filter is used to compensate for the droop in the passband of the CIC filter. Both filters utilize a multiplier-accumulator implementation structure and are constructed within System Generator for DSP. The PFIR and CFIR are polyphase multirate filter structures that interpolate by a factor of 2, 4, or 8. The filter length for each filter is configurable. The coefficient precision may also be customized.

Typically, the CFIR filter will have a wide transition band to minimize the filter length. Relative to the input sample rate the CFIR and PFIR both take specific number of clock cycles to process each input sample received by the DUC. When a sample is received both the PFIR and CFIR will output new data samples as soon as the computations have been completed and then sit idle until the next input sample has been received.

2.3.2 Digital Down Converter (DDC)

The DDC is a fully programmable single chip down converter architected to perform intermediate frequency (IF) to baseband processing for communication signal processing [16]. Its primary uses are IF processing in digital radio, cellular telephone, sonar, radar, and similar processing applications. A top level functional block diagram of the DDC architecture is shown in Figure 2.10. The Phase Generator, under the control of input parameters, generates time varying phase words to drive the Sin/Cos Generator. The

Sin/Cos generator produces a quadrature sinusoid at the input sample rate to modulate the input sampled waveform. The modulator is constructed as two real multipliers. The modulation of the real input signal by the quadrature sinusoid forms a spectrally two sided complex signal with one of the sidebands centered at dc. The lowpass filter function eliminates the unwanted sideband and all other signal components except those within the desired band of interest. Since the lowpass operation eliminates one of the sidebands of the output of the modulator, the output of the filter process is a quadrature signal designated as having an inphase (I) and quadrature (Q) component. The Formatter generates the output of the DDC in a user selected format.

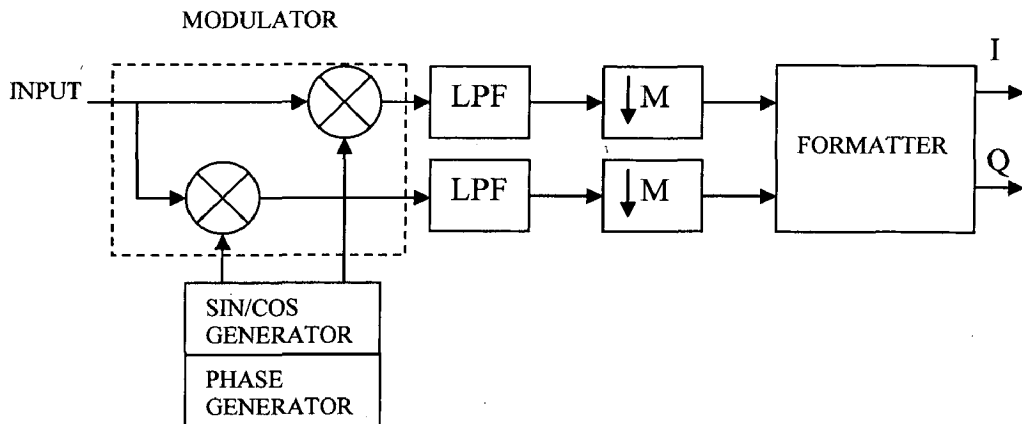


Figure 2.9: DDC Functional Block Diagram

DDCs are grouped into two main categories.

- Wideband DDCs have output channel bandwidths typically above 1 MHz and are appropriate for wideband Code Division Multiple Access (CDMA) and radar applications.
- Narrowband DDCs with bandwidths below 1 MHz are widely used for Frequency Division Multiplexed (FDM) systems including voice and music channels in telecom and commercial broadcast systems.

For wideband channels, a conventional FIR filter, as shown in figure 2.9, is best. For narrowband channels, a multistage Cascaded Integrator-Comb (CIC) filter, followed by an FIR filter to correct frequency droop, is more efficient.

The concept of the DDC is to sample the whole input signal and to use digital techniques to reduce the data. However, that may require an unrealistically fast ADC. For example, Nyquist's theory states that we should sample at a rate "at least double the

bandwidth of interest”. If we do this on a 1GHz carrier, we would need an ADC sampling at well over 2GHz; that ADC would be protected by anti-aliasing filters, removing any signal above 1GHz. However, this is beyond what can be achieved with today’s technology. Frequency shifting is required before the ADC. This can be done in analog, using an IF stage; or it may be possible to use a composite approach. Typically, if the signal is at 1GHz, the bandwidth of interest may be only a MHz wide. The filters required to select such a narrow band would be large and complex; but selecting a band perhaps 40MHz wide could be practical. If we can select this band using analog filters, it is possible to sample at a reduced rate – using the bandpass sampling. In this case, a sample rate (F_s) of 100MHz would mean that our signal could be frequency shifted by $10F_s$, bringing the 1GHz signal in at a much more reasonable 100MHz sample rate. The more aggressively this technique is applied, the greater the strains on the anti-aliasing filters. It also places great emphasis on high analog bandwidth for the ADC, and extremely low jitter for the ADC clocks; any errors here will be magnified greatly. However, it can be used very effectively. This technique is complementary to frequency-shifting the signal to an “Intermediate Frequency”. Typically the approach used—direct conversion, IF or “bandpass sampling” will be chosen dependent on the frequencies involved and other system issues—even down to the type of antenna deployed.

The analog front end performs down conversion of the complete received frequency band of bandwidth B . Inside this band the signal of interest $x_{Tx}(t)$, which should finally be downconverted to baseband. The following signal is produced at the output of the analog downconverter when downconverting by f_1 , where $f_1 < f_c$, which is the carrier frequency.

$$x_{Rx}(t) = x_{Tx}(t)e^{-j2\pi f_1 t} \quad (2.10)$$

The interesting signal component is centered at the IF,

$$f_{IF} = f_c - f_1 \quad (2.11)$$

It is enclosed by several adjacent channel interferes, which are removed by low pass filtering. Thus the digitized signal is

$$x_{dig,IF} = \frac{1}{2} x_{Tx}(kT) e^{j2\pi f_{IF} kT} + A \quad (2.12)$$

where A is adjacent channels after downconversion, anti-aliasing filtering, and digitization. T is the sampling period. The digital IF signal is a complex signal, the interesting signal component is centered at f_{IF} . The objective of DDC is to shift this interesting component from the carrier frequency f_{IF} down to baseband. From the (2.12), it can be found that downconversion can be achieved by multiplying the received signal with the respective exponential function.

$$x_{dig, BB}(kT) = x_{dig, IF}(kT)e^{-j2\pi f_{IF}kT} \quad (2.13)$$

This signal after analog downconversion comprises of the following two components, of real and imaginary parts

$$\begin{aligned} \text{Re} &= \text{Re}(x_{Rx}(t)e^{-j2\pi f_1 t}) \\ &= x_{Rx}(t) \cos(2\pi f_1 t) \end{aligned} \quad (2.14)$$

$$\begin{aligned} \text{Im} &= \text{Im}(x_{Rx}(t)e^{-j2\pi f_1 t}) \\ &= -x_{Rx}(t) \sin(2\pi f_1 t) \end{aligned} \quad (2.15)$$

The analog downconversion can be implemented by means of multiplying the received real signal by a cosine signal and a sine signal. The real part of the complex IF signal is obtained by multiplying the received signal with a cosine signal; whereas the imaginary part is obtained by multiplying with a sine signal.

As seen from (2.12) the input signal to the DDC is in principle a complex signal, hence the DDC described by (2.13) requires a complex multiplication. Since the complex signals are only available in the form of their real and imaginary parts, the complex multiplication of DDC requires four real multiplications. Requiring a resolution of n bits, the lookup table has a size of approximately $2^n \times n$ bits, which together with the four general purpose multipliers results in large chip areas, high power consumption, and considerable costs.

In this chapter, the five main technology elements of SDR are reviewed and the state of technology in each domain is explored. Also implementation approaches in each case are compared. MEMS technology is becoming a critical enabling technology to extend the capability and performance of antennas and several RF building blocks. The resolution and sampling rate of signal converters are being continuously improved. The performance of ADCs is being enhanced by several distinguished effort of advance

semiconductor processes, superconductor based and optical sampling techniques. The final call in selection of a DSP implementation is based on level of programmability and integration, development cycle, performance and power. Among the main three interconnect scheme, the bus interconnect enjoys the widespread use and recognition, however switch fabric with higher speed and more flexibility are being more prominent.

DDC is devoted to filter the input signal at the analog to digital converter rate and to reduce this high rate to an intermediate one [17]. The former can be performed by the means of digital filtering while the latter can be done by downsampling. Even if the filter step could be carried out with traditional FIR filtering techniques, the high rates involved seriously compromise the FIR design, making this approach hardly practicable. From some detailed implementation analysis the main system bottleneck seems to be located in the multiplier stages. These disadvantages can be overcome by an economical class of filters called as Cascaded Integrator Comb or, more briefly, CIC. This shall be covered in detail in chapter three of the thesis.

CHAPTER 3

Digital Down Converter

Fundamental part of many communications systems is DDC. Digital radio receivers often have fast ADC converters delivering vast amounts of data; but in many cases, the signal of interest represents a small proportion of that bandwidth. A DDC allows the rest of that data to be discarded, allowing more intensive processing to be performed on the signal of interest.

This chapter will focus on Decimation, Polyphase realizations of decimators and computationally efficient decimators, and its application in the SDR. It shall also cover DDC and deals with the details of CIC decimation filter and Chebyshev's low pass filter used for the purpose. To begin with, a generic description is dwelled and then specifics are dealt with.

3.1 DECIMATION

Let $x[n]$ is a sequence with a sampling rate of F_T Hz and it is used to generate another sequence $y[n]$ with a desired sampling rate of F'_T Hz, then the sampling rate alteration ratio is given by

$$R = F'_T / F_T$$

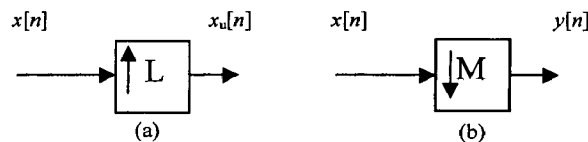


Figure 3.1: Basic Sampling Rate Devices : (a)Up-Sampler (b)Down-Sampler

If $R > 1$, the process is called interpolation and results in a sequence with a higher sampling rate. If $R < 1$, the sampling rate is decreased by a process called decimation. The basic operations employed in the sampling rate alteration process are called up-sampling and down sampling. In up-sampling by an integer factor $L > 1$, $L-1$ equidistant zero-valued samples are inserted by the up-sampler between each two consecutive samples of the input sequence $x[n]$ to develop an output sequence $y[n]$ according to the relation [18]

$$x_u[n] = \begin{cases} x[n/L], & n = 0, \pm L, \pm 2L, \dots \\ 0 & \text{otherwise} \end{cases}$$

Sampling rate of $y[n]$ is L times larger than that of the original sequence $x[n]$. The block diagram representation of the up-sampler, also called a sampling rate expander is shown in the figure 3.1 (a). Conversely, the down-sampling operation by an integer factor $M > 1$ on a sequence $x[n]$, consists of keeping every M th sample of $x[n]$ and removing $M-1$ in between samples, generating an output sequence $y[n]$ according to the relation $y[n] = x[nM]$

This results in a sequence $y[n]$ whose sampling rate is $1/M$ th that of $x[n]$. Basically, all input samples with indices equal to an integer multiple of M are retained at the output and all others are discarded. The block diagram representation of the down-sampler, also called a sampling rate compressor is shown in the figure 3.1(b).

3.1.1 Multistage Design of Decimator

Let us consider a decimator with a single-stage structure, shown in figure 3.2. The basic scheme for the implementation involves a single low pass filter and a single sampling rate alteration device.

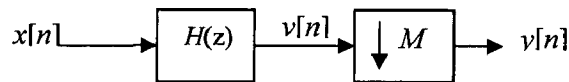


Figure 3.2: Filters in Sampling Rate Alteration System for Decimator

The factor of M decimator of figure 3.2 can be implemented in two stages, as shown in figure 3.3, if the decimation factor M is the product of two integers M_1 and M_2 . The design here involves more than two stages, depending upon the number of factors used to express M . This implies that the computational efficiency is improved significantly by designing the sampling rate alteration system as a cascade of several stages.

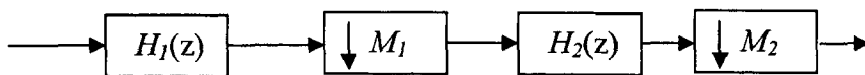


Figure 3.3: Two stage Implementation of Sampling Rate Alteration System for Decimator

The quantity $H(z)$, which is the z-transform of the impulse response sequence $h[n]$, is called as the transfer function of the system function. This may be defined as the ratio of z-transform of output sequence to z-transform of input sequence. The inverse z-transform of the transfer function $H(z)$ yields the impulse response $h[n]$.

3.1.2 Poly Phase Realization

FIR Filter can be realized using polyphase decomposition of its transfer function.

Consider a causal FIR transfer function $H(z)$ of length of 9;

$$H(z) = h[0] + h[1]z^{-1} + h[2]z^{-2} + h[3]z^{-3} + h[4]z^{-4} + h[5]z^{-5} + h[6]z^{-6} + h[7]z^{-7} + h[8]z^{-8} \quad (3.1)$$

The above transfer function can be expressed as a sum of two terms, with one term containing the even indexed coefficients and the other the odd indexed coefficients as shown below:

$$\begin{aligned} H(z) &= (h[0] + h[2]z^{-2} + h[4]z^{-4} + h[6]z^{-6} + h[8]z^{-8}) + (h[1]z^{-1} + h[3]z^{-3} + h[5]z^{-5} + h[7]z^{-7}) \\ H(z) &= (h[0] + h[2]z^{-2} + h[4]z^{-4} + h[6]z^{-6} + h[8]z^{-8}) + z^{-1}(h[1] + h[3]z^{-2} + h[5]z^{-4} + h[7]z^{-6}) \end{aligned} \quad (3.2)$$

By using the notations

$$\begin{aligned} E_0(z) &= (h[0] + h[2]z^{-1} + h[4]z^{-2} + h[6]z^{-3} + h[8]z^{-4}) \\ E_1(z) &= (h[1] + h[3]z^{-1} + h[5]z^{-2} + h[7]z^{-3}) \end{aligned}$$

Equation 2 can be written as

$$H(z) = E_0(z^2) + z^{-1}E_1(z^2) \quad (3.3)$$

Similarly, grouping the terms of (1) differently we can express it as

$$H(z) = E_0(z^3) + z^{-1}E_1(z^3) + z^{-2}E_2(z^3) \quad (3.4)$$

$$\begin{aligned} \text{Where,} \quad E_0(z) &= (h[0] + h[3]z^{-1} + h[6]z^{-2}) \\ E_1(z) &= (h[1] + h[4]z^{-1} + h[7]z^{-2}) \\ E_2(z) &= (h[2] + h[5]z^{-1} + h[8]z^{-2}) \end{aligned}$$

The decomposition of $H(z)$ in the form of (3.3) and (3.4) is more commonly known as the polyphase decomposition. The polyphase structures are often used in multirate digital signal processing applications for computationally efficient realizations. The transfer function of causal FIR filter of order N is given by

$$H(z) = \sum_{k=0}^N h[k]z^{-k},$$

In a general case, an L -branch polyphase decomposition of its transfer function is of the form

$$H(z) = \sum_{m=0}^{L-1} z^{-m} E_m(z^L) \quad (3.5)$$

where $E_m(z) = \sum_{n=0}^{\lfloor (N+1)/L \rfloor} h[Ln+m]z^{-n}$, $0 \leq m \leq L-1$

with $h[n] = 0$ for $n > N$. A realization of $H(z)$ based on the decomposition of (3.5) is called a Polyphase realization.

3.1.3 Computationally Efficient Interpolator and Decimator Structures

Computationally efficient interpolator structures employing linear-phase low pass filters can be derived by applying a polyphase decomposition to the low pass filters

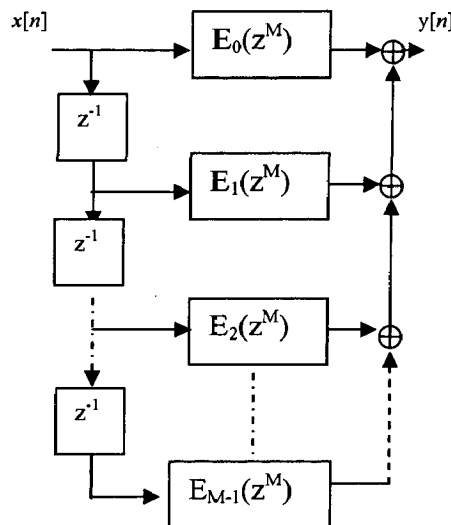


Figure 3.4: Realization of an FIR Filter Based on Polyphase Decomposition

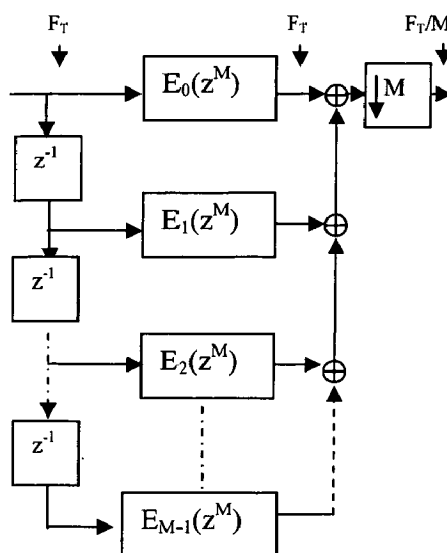


Figure 3.5: Decimator Implementation Based on Polyphase Decomposition

Consider first the use of the polyphase decomposition in the realization of the decimation filter of figure 3.2. If the lowpass filter $H(z)$ is realized as shown in figure 3.4, the overall decimator structure takes the form of figure 3.5.

By invoking the cascade equivalence as shown in figure 3.6



Figure 3.6: Cascade Equivalence

this structure reduces to that indicated in figure 3.7, which is computationally more efficient than the structure of fig 3.2.

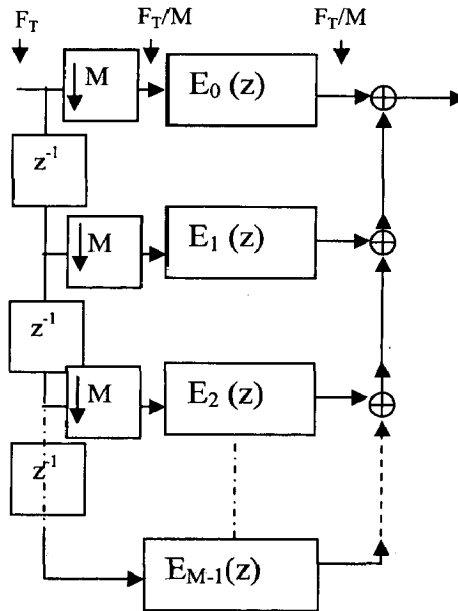


Figure 3.7: Decimator Implementation Based on Polyphase Decomposition

To illustrate this point, assume that the decimation filter $H(z)$ of figure 3.2 is a length N FIR structure and the input sampling period $T=1$. Since the decimator output $y[n]$ is obtained by downsampling the filter output $v_1[n]$ by a factor of M , it is necessary only to compute $v[n]$ at $n=\dots, -2M, -M, 0, M, 2M, \dots$. The computational requirements are therefore N multiplications and $(N-1)$ additions per output sample being computed. However as n increases the stored signals in the delay registers change. As a result, all computations need to be completed in one sampling period, and for the following $(M-1)$

sampling periods the arithmetic units remain idle. Now consider the structure of figure 3.7. If the lengths of the sub filter $E_k(z)$ is N_k , then,

$$N = \sum_{k=0}^{M-1} N_k$$

The computational requirements of the k^{th} sub filter are N_k multiplications and N_k-1 additions per output sample, and that for the overall structure is therefore $N = \sum_{k=0}^{M-1} N_k = N$ multiplications and $N = \sum_{k=0}^{M-1} (N_k - 1) + (M - 1) = N - 1$ additions per decimator output sample. However, in the latter structure, the arithmetic units are operated at all instants of the output sampling period, which is M times that of the input sampling period.

3.1.4 Software Radio Issues in Cellular Base Stations

Software Radio has a dual role, for military, it is the need for a single radio which can communicate with the many types of military radios that use different RF bands and different modulation schemes whereas in commercial arena, its applications can be divided into user terminals and radio base stations [19].

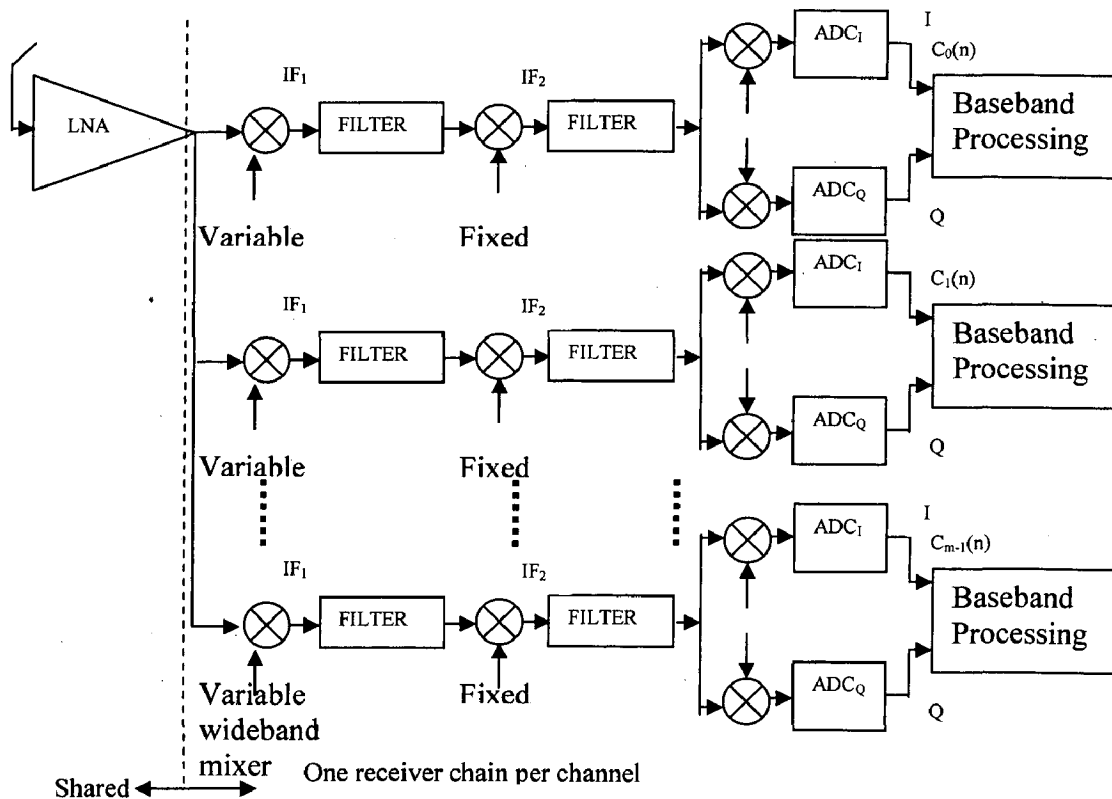


Figure 3.8: Conventional radio Base Station Receivers

The computational load of software radios is on the order of billions of operations per second; hence, with today's state-of-the-art in digital signal processing (DSP) hardware, it is extremely challenging to implement software radio in a battery-powered terminal. Wide-band receivers in software radios can significantly reduce the cost and complexity of today's cellular base stations.

Two of the challenging issues in the implementation of wide-band receivers are digitization of a large block of the cellular spectrum using a single wide-band ADC and extraction of individual channels from the wide-band digitized signal. Currently, cellular base stations employ a bank of narrowband receivers, each having its own complete receiver chain as depicted in figure 3.8 In a base station with a wideband receiver, a single analog front end can be used to receive all channels, hence, the complexity of the analog part of the resulting receiver remains constant, i.e., independent of the number of received channels. The concept of Polyphase implementation can be employed for reducing the computational complexity of the receiver.

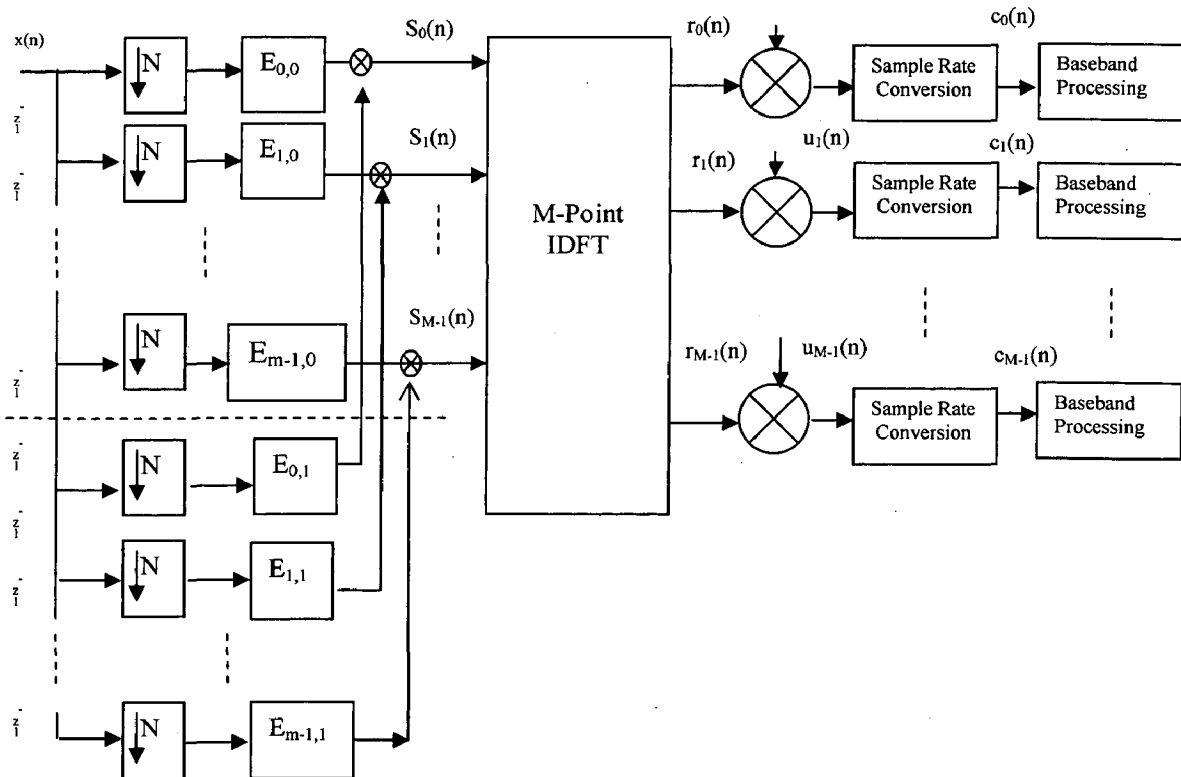


Figure 3.9: Opened Filter bank channelizer

Figure 3.9 depicts opened filter bank channelizer. This structure is computationally efficient because both polyphase filters and IDFT operate at lowest sampling rate in the system.

If M -point IDFT in the OFB channelizer is replaced with an M/L -point IDFT, it results in a channelizer with lower overall complexity. For an M/L -point IDFT to be defined, M must be an integer multiple of L .

3.2 CIC DECIMATION FILTER

The essential function of a decimation or interpolation filter is to decrease or increase the sampling rate and to keep the passband aliasing or imaging error within prescribed bounds. A class of linear phase FIR filters for decimation and interpolation are called cascaded integrator-comb (CIC) filters, designated so because their structure consists of an integrator section operating at the high sampling rate and a comb section operating at the low sampling rate. The filters require no multipliers and use limited storage thereby leading to more economical hardware implementations [20].

Using CIC filters, the amount of passband aliasing or imaging error can be brought within prescribed bounds by increasing the number of stages in the filter. However, the width of the passband and the frequency characteristics outside the passband are severely limited. For critical applications these limitations can be overcome by using CIC filters to make the transition between high and low sampling rates, and to use conventional filters at the low sampling rate to “shape” or “clean-up” the frequency response. In this manner, CIC filters are used at high sampling rates where economy is critical, and conventional filters are used at low sampling rates where the number of multiplies per second is low.

3.2.1 CIC Filter Description

Figure 3.10 shows the basic structure of the CIC decimation filter. The integrator section of CIC filters consists of N ideal digital integrator stages operating at the high sampling rate, f_s . Each stage is implemented as a one-pole filter with a unity feedback coefficient. The system function for a single integrator is

$$H_I(z) = 1/(1 - z^{-1}) \quad (3.6)$$

The comb section operates at the low sampling rate f_s/R where R is the integer rate change factor. This section consists of N comb stages with a differential delay of M samples per stage. The differential delay is a filter design parameter used to control the filter's frequency response. In practice, the differential delay is usually held to $M=1$ or 2 . The system function for a single comb stage referenced to the high sampling rate is

$$H_C(z) = 1 - z^{-RM} \quad (3.7)$$

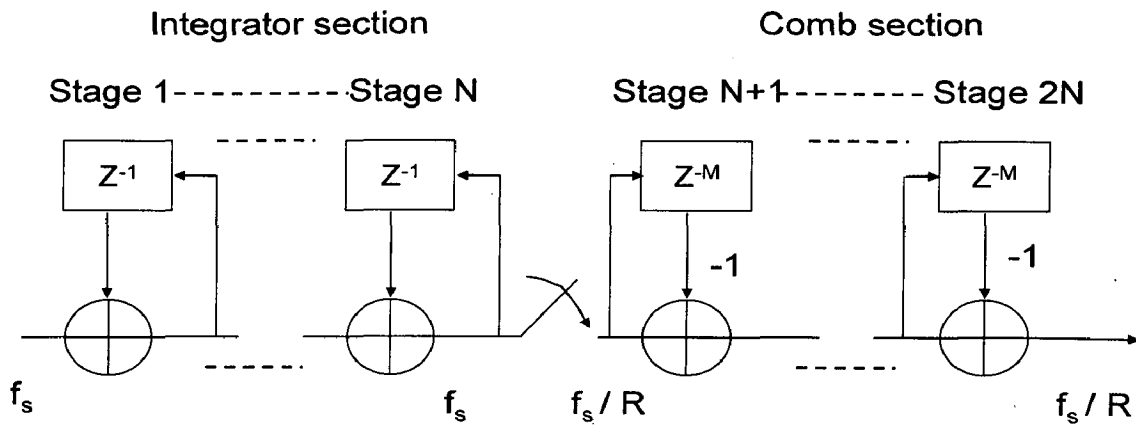


Figure 3.10 Basic structure of CIC decimation filter

There is a rate change switch between the two filter sections. For decimation, the switch sub samples the output of the last integrator stage, reducing the sampling rate from f_s to f_s/R ; and for interpolation, the switch causes a rate increase by a factor of R by inserting $R - 1$ zero valued samples between consecutive samples of the comb section output. It follows from (3.6) and (3.7) that the system function for the composite CIC filter referenced to the high sampling rate, f_s , is

$$H(z) = H_I^N(z)H_C^N(z) = \frac{(1 - z^{-RM})^N}{(1 - z^{-1})^N} = \left[\sum_{k=0}^{RM-1} z^{-k} \right]^N \quad (3.8)$$

It is implicit from the last form of the system function that the CIC filter is functionally equivalent to a cascade of N uniform FIR filter stages. A conventional implementation consists of a cascade of N stages each requiring RM storage registers and one accumulator. Taking advantage of the rate change factor, one of the N stages can be simplified to use only M storage registers. It must be stressed that each integrator has a

unity feedback coefficient; for CIC decimators this results in register overflow in all integrator stages. This is of no consequence if the following two conditions are met:

- The filter is implemented with two's complement arithmetic or other number system which allows "wrap-around" between the most positive and most negative numbers.
- The range of the number system is equal to or exceeds the maximum magnitude expected at the output of the composite filter. For CIC interpolators, the data are preconditioned by the comb section so that overflow will not occur in the integrator stages.

The economics of CIC filters derive from the following sources:

- No multipliers are required;
- No storage is required for filter coefficients;
- Intermediate storage is reduced by integrating at the high sampling rate and comb filtering at the low sampling rate, compared to the equivalent implementation using cascaded uniform FIR filters;
- The structure of CIC filters is very "regular" consisting of two basic building blocks;
- Little external control or complicated local timing is required;
- The same filter design can easily be used for a wide range of rate change factors, R , with the addition of a scaling circuit and minimal changes to the filter timing.

Some problems encountered with CIC filters include the following:

- Register widths can become large for large rate change factors, R .
- The frequency response is fully determined by only three integer parameters (R , M , and N), resulting in a limited range of filter characteristics.

The application for CIC filters is in areas where high sampling rates make multipliers an uneconomical choice and areas where large rate change factors would require large amounts of coefficient storage or fast impulse response generation.

3.2.2 Frequency Characteristics

CIC filters have a low-pass frequency characteristic. The frequency response is given by (3.8) evaluated at

$$z = e^{j(2\pi f / R)} \quad (3.9)$$

Where f is the frequency relative to the low sampling rate f_s/R . As part of the filter design process, R , M , and N are chosen to provide acceptable passband characteristics over the frequency range from zero to a predetermined cutoff frequency f_c expressed relative to the low sampling rate. The power response obtained by substituting (3.9) in (3.8) is given as

$$P(f) = \left[\frac{\sin \pi M f}{\sin \frac{\pi f}{R}} \right]^{2N} \quad (3.10)$$

For large rate change factors R , the power response can be approximated over a limited frequency range by

$$\hat{P}(f) = \left[RM \frac{\sin \pi M f}{\pi M f} \right]^{2N} \quad \text{for } 0 \leq f < \frac{1}{M} \quad (3.11)$$

This approximation can be used for many practical design problems. For example, the error between P and \hat{P} is less than 1 dB for $RM \geq 10$, $1 \leq N \leq 7$ and $0 \leq f \leq 255/(256M)$. For the power response of (3.10) and (3.11), nulls exist at multiples of $f = 1/M$. Thus, the differential delay M can be used as a design parameter to control the placement of nulls. For CIC decimation filters, the region around every M th null is folded into the passband causing aliasing errors; for CIC interpolation filters, imaging occurs in the regions around these nulls. Specifically, these aliasing/ imaging bands are

$$(i - f_c) \leq f \leq (i + f_c) \quad (3.12)$$

for $f \leq \frac{1}{2}$ and $i = 1, 2, \dots, \lfloor R/2 \rfloor$ where $\lfloor x \rfloor$ is the largest integer not greater than x .

For practical design problems, the aliasing/imaging errors can be characterized by the maximum error over all aliasing/imaging bands. For a large class of filter design problems where for $f_c \leq \frac{1}{2M}$, this maximum occurs at the lower edge of the first aliasing/imaging band at

$$f_{AI} = 1 - f_c \quad (3.13)$$

Tables 3.1 and 3.2 are represented from [20] as an aid in determining the tradeoffs between bandwidth, passband attenuation, and aliasing/imaging error.

Relative Bandwidth Differential Delay(Mf_c)	Passband Attenuation at f_c (dB) as a Function of Number of Stages (N)					
	1	2	3	4	5	6
1/128	0.00	0.00	0.00	0.00	0.00	0.01
1/64	0.00	0.01	0.01	0.01	0.02	0.02
1/32	0.01	0.03	0.04	0.06	0.07	0.08
1/16	0.06	0.11	0.17	0.22	0.28	0.34
1/8	0.22	0.45	0.67	0.90	1.12	1.35
1/4	0.91	1.82	2.74	3.65	4.56	5.47

Table 3.1: Passband Attenuation for Large Rate Change Factors

Differential Delay(M)	Relative Band width (f_c)	Aliasing/ Imaging Attenuation at f_{IA} (dB) as a Function of Number of Stages (N)					
		1	2	3	4	5	6
1	1/128	42.1	84.2	126.2	168.3	210.4	252.5
1	1/64	36.0	72.0	108.0	144.0	180.0	215.9
1	1/32	29.8	59.7	89.5	119.4	149.2	179.0
1	1/16	23.6	47.2	70.7	94.3	117.9	141.5
1	1/8	17.1	34.3	51.4	68.5	85.6	102.8
1	1/4	10.5	20.9	31.4	41.8	52.3	62.7
2	1/256	48.1	96.3	144.4	192.5	240.7	288.8
2	1/128	42.1	84.2	126.2	168.3	210.4	252.5
2	1/64	36.0	72.0	108.0	144.0	180.0	216.0
2	1/32	29.9	59.8	89.6	119.5	149.4	179.3
2	1/16	23.7	47.5	71.2	95.0	118.7	142.5
2	1/8	17.3	35.6	53.4	71.3	89.1	106.9

Table 3.2: Passband Attenuation for Large Rate Change Factors

It is assumed that the rate change factor is large, so the power response approximation of (3.11) can be used. In these tables attenuations are calculated relative to the maximum filter response at $f=0$. The passband attenuations given in table 3.1 are constant for a given relative bandwidth-differential delay product (Mf_c); however, this is not the case for the aliasing/imaging attenuation given in table 3.2. Here, two values of differential delay, $M=1$ and 2 are tabulated; differential delays greater than these seem to be of less value.

3.3 DESIGN CONSIDERATIONS FOR CIC DECIMATION FILTER

This section presents design considerations for CIC decimation filters. The most significant bit (MSB) of these decimation filters is determined as a function of the overall register growth. This is followed by use of truncation or rounding at each stage of filtering, the retained number of bits decreasing monotonically from stage to stage. The filter designer can determine the amount of truncation or rounding to apply at each stage, without violating design constraints. It is assumed that the desired frequency characteristics have been determined, resulting in choices for the rate change factor R , differential delay M , and number of stages N . It is also assumed throughout this section and further, that two's complement arithmetic is being used.

3.3.1 Register Growth

The system function from the j th stage up to and including the last stage can be expressed as a fully expanded polynomial in z^{-1} . The resulting function is

$$H_j(z) = \begin{cases} H_I^{N-j+1} H_C^N = \sum_{k=0}^{(RM-1)N+j-1} h_j(k) z^{-k} & j = 1, 2, \dots, N \\ H_C^{j-N} = \sum_{k=0}^{2N+1-j} h_j(k) z^{-kRM} & j = N+1, \dots, 2N \end{cases} \quad (3.14a)$$

Where

$$h_j(k) = \begin{cases} \sum_{l=0}^{\lfloor \frac{k}{RM} \rfloor} (-1)^l \binom{N}{l} \binom{N-j+k-RMl}{k-RMl}, & j = 1, 2, \dots, N \\ (-1)^k \binom{2N+1-j}{k}, & j = N+1, \dots, 2N \end{cases} \quad (3.14b)$$

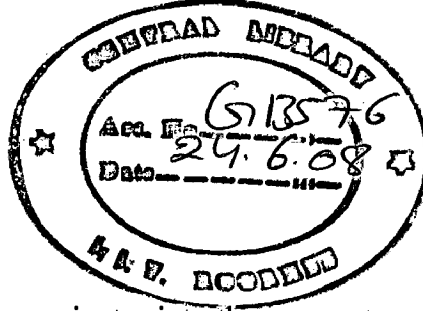
are the impulse response coefficients.

The form of the above function is that of a fully expanded polynomial in z^{-1} . There are two cases expressed: case 1, where j is in the range 1 to N and case 2, where j is in the range $N + 1$ to $2N$. For case 1, there are $N-j+1$ integrators and N combs. The system function is simply

$$H_j(z) = H_I^{N-j+1} H_C^N, \quad j = 1, \dots, N \quad (3.15)$$

Substituting (3.6) and (3.7) into 3.15

$$H_j(z) = \frac{(1 - z^{-RM})^N}{(1 - z^{-1})^{N-j+1}} \quad (3.16)$$



This equation can be expanded by dividing denominator into the numerator resulting in

$$H_j(z) = ((1 - z^{-RM})^{j-1}) \left[\sum_{k=0}^{RM-1} z^{-k} \right]^{N-j+1} \quad (3.17)$$

A dimensional analysis of (3.17) indicates that the order of the polynomial in terms of z^{-1} is

$$RM(j-1) + (RM-1)(N-j+1) = (RM-1)N+j-1 \quad (3.18)$$

Thus, the system function can be expressed as a fully expanded polynomial of the same order. The polynomial has the form

$$H_j(z) = \sum_{k=0}^{(RM-1)N+j-1} h_j(k) z^{-k} \quad (3.19)$$

where $h_j(k)$ are the polynomial coefficients.

Another way of expressing (3.16) is in terms of its binomial expansion. This results in

$$H_j(z) = \left[\sum_{l=0}^N (-1)^l \binom{N}{l} z^{-RMl} \right] \left[\sum_{v=0}^{\infty} \binom{N-j+v}{v} z^{-v} \right] \quad (3.20)$$

and taking the cross product of the two polynomials results in

$$H_j(z) = \sum_{l=0}^N \sum_{v=0}^{\infty} (-1)^l \binom{N}{l} \binom{N-j+v}{v} z^{-(RMl+v)} \quad (3.21)$$

In this expression, terms with identical powers of z^{-1} can be collected together. Thus, for a particular nonnegative value k , where

$$k = RM + v \quad (3.22)$$

it is apparent that l can range over the integers

$$l = 0, 1, \dots, \lfloor k/RM \rfloor \quad (3.23)$$

without forcing v out of range. Using (3.22) and (3.23) we can now collect terms resulting in the fully expanded polynomial

$$H_j(z) = \sum_{k \geq 0} \left[\sum_{l=0}^{\lfloor k/RM \rfloor} (-1)^l \binom{N}{l} \binom{N-j+k-RMl}{k-RMl} \right] z^{-k}, \quad j = 1, 2, \dots, N \quad (3.24)$$

The form of this polynomial is the same as (3.19) where the range of k is established as $k = 0, 1, \dots, (RM-1)N + j - 1$. This results in (3.14a) for case 1.

For case 2, where j is in the range $j = N + 1, \dots, 2N$, there are $2N + 1 - j$ combs and no integrators. The system function is simply

$$H_j(z) = H_C^{2N+1-j}, \quad j = N + 1, \dots, 2N \quad (3.25)$$

and substituting (3.7) into (3.25) results in

$$H_j(z) = (1 - z^{-RM})^{2N+1-j}, \quad j = N + 1, \dots, 2N \quad (3.26)$$

The binomial expansion of (3.26) results in (3.14b) for case 2.

The maximum register growth is defined as the maximum output magnitude resulting from the worst possible input signal relative to the maximum input magnitude. This growth is used in the CIC filter design process to insure that no data are lost due to register overflow. Using this definition, the maximum register growth from the first stage up to and including the last stage is simply

$$G_{\max} = \sum_{k=0}^{(RM-1)N} |h_1(k)| \quad (3.27a)$$

This can be simplified to

$$G_{\max} = (RM)^N \quad (3.27b)$$

The above equation (3.27b) is derived resulting in a simplified expression for the maximum register growth in CIC decimators. This register growth is defined by (3.27a). It is apparent that (3.14a), evaluated at $j = 1$, is the system function for the composite CIC filter and is just an alternate form of (3.8). Combining these two equations results in

$$H_1(z) = \sum_{k=0}^{(RM-1)N} h_1(k) = \left[\sum_{k=0}^{RM-1} z^{-k} \right]^N \quad (3.28)$$

And evaluating (3.28) at $z=1$, results in

$$H_1(1) = \sum_{k=0}^{(RM-1)N} h_1(k) = (RM)^N \quad (3.29)$$

In equation (3.28) it is noted that the system function is the product of N system functions of the form

$$\sum_k z^{-k} \quad (3.30)$$

Since this polynomial has all positive coefficients, it follows that the product of two or more of these polynomials results in a polynomial that also has all positive coefficients. As a result we can equate the coefficients with their absolute values. This results in a version of (3.29) with the form

$$\sum_{k=0}^{(RM-1)N} |h_1(k)| = (RM)^N \quad (3.31)$$

Substituting this expression into (3.27a) results in (3.27b), the equation to be derived. If the number of bits in the input data stream is B_{in} , then the register growth can be used to calculate B_{max} , the most significant bit at the filter output. That is,

$$B_{max} = \lceil N \log_2 RM + B_{in} - 1 \rceil \quad (3.32)$$

Where the least significant bit (LSB) of the input register is considered to be bit number zero and where $\lceil x \rceil$ is the smallest integer not less than x . Not only is B_{max} the MSB at the filter output, but it is also the MSB for all stages of the filter. This can be shown by applying modulo arithmetic to the filter output function. For two's complement arithmetic, the modulo operation can be implemented by simply eliminating bit positions above B_{max} . Since the modulo operation is used at the filter output, the same modulo operation can be applied independently to each integrator and comb stage. This implies that B_{max} is an upper bound for each filter stage. B_{max} is also a lower bound. Since the first N stages of the filter are integrators with unity feedback, it is apparent that the variance of the integrator outputs grow without bound for uncorrelated input data. As seen at the output register, B_{max} is the MSB for each integrator since this is a significant bit and is the highest order bit that can propagate into the output register. Since a propagation path must be provided through the comb section for this MSB, it can be concluded that B_{max} must be the MSB not only for the integrators, but also for the combs that follow.

3.3.2 Truncation and Rounding

B_{\max} is large for many practical cases and can result in large register widths; however, truncation or rounding may be used at each filter stage reducing register widths significantly. To calculate the total error at the filter output due to truncation or rounding, the mean and variance of the error at each error source can be determined and then the corresponding statistics at the filter output due to the source alone can be determined. The total mean and variance at the output is then determined as the sum of the statistics from these individual sources. There are a total of $2N+1$ error sources: the first $2N$ sources are caused by truncation or rounding at the inputs to the $2N$ filter stages. The last error source is due to truncation or rounding going into the output register. The error sources are given indexes corresponding to the filter stage numbers shown in figure 3.10, with $2N+1$ identifying the error source going into the output register. It is often assumed that rounding is always better than truncation; however, in [20] it is shown that except for the first and last error sources, the output error statistics are the same for both truncation and rounding.

Furthermore, to keep the output error within bounds, most practical designs will make use of full precision arithmetic at the first error source. As a result, the only place where the designer need worry about truncation versus rounding is at the last error source going into the output register. It is assumed that each error source produces white noise that is uncorrelated with the input and other error sources. Furthermore, the error at the j th source is assumed to have a uniform probability distribution with a width of

$$E_j = \begin{cases} 0 & \text{no truncation \& rounding} \\ 2^{B_j} & \text{otherwise} \end{cases} \quad (3.33)$$

Where B_j is the number of LSB's discarded at the j th source. It can be shown that since the error has a uniform distribution, the mean of the error is

$$\mu_j = \begin{cases} \frac{1}{2} E_j & \text{if truncation} \\ 0 & \text{otherwise} \end{cases} \quad (3.34)$$

and the variance of the error is

$$\sigma_j^2 = \frac{1}{12} E_j^2 \quad (3.35)$$

To determine the statistics at the output due to the j th error source, the system function from the j th stage up through the last comb as given by (3.14a, 3.14b) can be used. The impulse response coefficients correspond to independent random processes that are summed together to produce one filter output. The error mean and variance corresponding to the k th coefficient are simply $\mu_j h_j(k)$ and $\sigma_j^2 F_j^2(k)$ respectively, and since the processes are independent over k , the total statistics at the j th stage are the sums of the statistics for each impulse response coefficient. That is, the total mean is

$$\mu_{T_j} = \mu_j D_j \quad (3.36a)$$

Where

$$D_j = \begin{cases} \sum_k h_j(k) & j = 1, 2, \dots, 2N \\ 1 & j = 2N + 1 \end{cases} \quad (3.36b)$$

is designated the “mean error gain” for the j th error source. Similarly, the total variance is

$$\sigma_{T_j}^2 = \sigma_j^2 F_j^2$$

Where

$$F_j^2 = \begin{cases} \sum_k h_j^2(k) & j = 1, 2, \dots, 2N \\ 1 & j = 2N + 1 \end{cases} \quad (3.37)$$

is designated the “variance error gain” for the j th error source. The two error gains are used to relate the statistics at the error source to those at the output and are useful in the design process because they are independent of the actual error. It can be demonstrated that the mean error gain given by (3.36b) is zero for all but the first and last error sources and furthermore, the expression for the first error source can be simplified. This results in the form

$$D_j = \begin{cases} (RM)^N & j = 1 \\ 0 & j = 2, 3, \dots, 2N \\ 1 & j = 2N + 1 \end{cases} \quad (3.38)$$

From (3.33) and (3.35) it is noted that the error variance is the same for either truncation or rounding and the total error mean given by (3.36a) and (3.38) is zero for all but the first and last error sources. As a result, the choice of truncation versus rounding does not

affect the error statistics except for the first and last error sources. The total mean and variance at the output due to truncation and/or rounding are

$$\mu_T = \sum_{j=1}^{2N+1} \mu_{T_j} = \mu_{T_1} + \mu_{T_{2N+1}} \quad (3.39)$$

and

$$\sigma_T^2 = \sum \sigma_{T_j}^2 \quad (3.40)$$

Using the foregoing information relating error at the sources to error at the output, we can now work backwards to determine the number of bits to discard given appropriate error constraints. In this process, only the variance is used as a design parameter since it is affected by truncation and rounding at all error sources. On the other hand, the mean is affected by truncation and rounding only at the first and last error sources. It is assumed that the number of bits retained in the output register is B_{out} , so the number of LSB's discarded is

$$B_{2N+1} = B_{max} - B_{out} + 1 \quad (3.41)$$

The resulting error variance $\sigma_{T_{2N+1}}^2$ is defined by (3.37).

A legitimate design decision at this point is to make the variance from the first $2N$ error sources less than or equal to the variance for this last error source, and also to distribute the error about equally among these sources. This results in the following design equation for choosing the number of LSBs to discard at each stage [20]:

$$B_j = \left\lceil -\log_2 F_j + \log_2 \sigma_{T_{2N+1}} + \frac{1}{2} \log_2 \frac{6}{N} \right\rceil \quad (3.42)$$

for $j=1,2,\dots,2N$.

CIC filters are an economical alternative to conventional decimation and interpolation filters. CIC filters are implemented using a cascade of ideal integrator stages operating at a high sampling rate and an equal number of comb stages operating at a low sampling rate. These filters require no multipliers and use limited storage; they can be applied easily to problems requiring a rate change factor that is selectable over a wide operating range. The frequency response of CIC filters is fully determined by only three integer parameters resulting in a limited range of filter characteristics. The

aliasing/imaging error in the passband can be held within arbitrary bounds by appropriate choice of these parameters. However, the bandwidth and the frequency response outside the passband are severely limited. For CIC decimation filters, truncation or rounding may be used at each stage of the filter with a nondecreasing number of LSB's discarded at successive stages. The MSB of each stage is proportional to the maximum register growth expected at the filter output. This requires that all stages have the same MSB.

CHAPTER 4

Implementation and Results

In this chapter we first present various facilities existing in the Signal Processing Lab, available for the implementation of this work. To name them serially, the SMIQ-02B vector signal generator, the ICS-660B DAC and ICS-652 ADC cards for digital to analog conversion and vice versa respectively. We then focus on the design considerations for CIC decimation filter and Chebyshev's Low Pass Filter. We finally present the results of the implementation of DDC.

4.1 GENERAL DESCRIPTION

Figure 4.1 shows the basic block diagram of the system which is implemented. The vector modulated signal (QPSK in particular) is generated using a SMIQ02B vector signal generator (with RF frequency of 500 KHz).

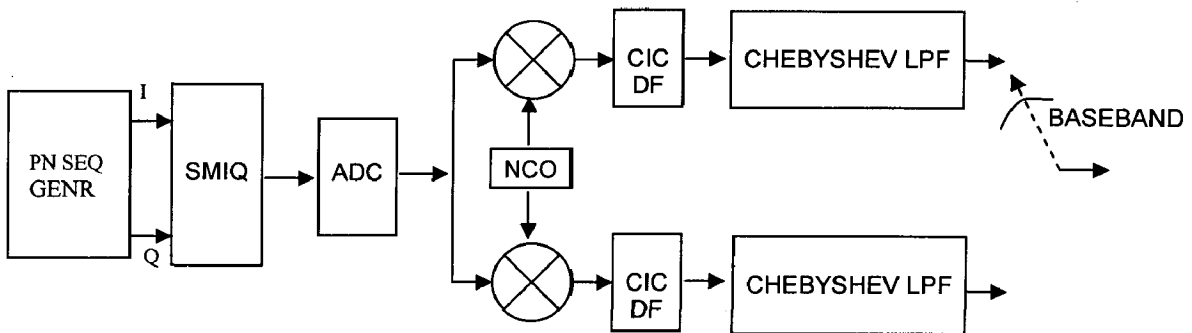


Figure 4.1: Basic Block Diagram of the Implemented System

For the purpose, the input data to SMIQ02B vector signal generator are selected as two pulses generated from PN sequence generator as I and Q components. The vector-modulated signal is digitized using an ICS_652 ADC card having a sampling frequency of 64 MHz. The output from the ADC card is captured and is stored in an array inside the buffer. The data is then multiplied with cosine and sine waveforms, having the freq same as that of the carrier. These I and Q components so obtained are passed through a CIC decimation filter. The impulse response of this CIC filter is calculated and convolved with I and Q signal. This output from CIC is then convolved with the impulse response of the Chebyshev low pass filter which is obtained simultaneously. This gives the

demodulated inphase and quadrature phase signals. These I and Q components are then used to obtain the final output data

4.1.1 PN Sequence

The PN sequence generator used is shown in the figure 4.2. A pseudo-noise (PN) sequence is a periodic binary sequence with a noise like waveform that is usually generated by means of a feedback shift register [21]. A feedback shift register consists of an ordinary shift register made up of m flip-flops and a logic circuit that are interconnected to form a multi-loop feedback circuit.

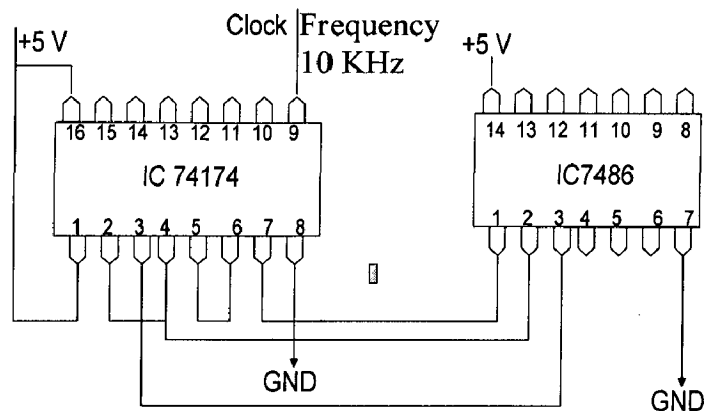


Figure 4.2: PN Sequence Generator

The flip-flops in the shift registers are regulated by a single timing clock. At each pulse of the clock, the state of each flip-flop is shifted to the next one down the line. With each clock pulse the logic circuit computes a boolean function of the states of the flip-flops. The result is then fed back as the input to the first flip-flop, thereby preventing the shift register from emptying. The PN sequence so generated is determined by the length m of the shift register, its initial state, and the feedback logic.

The sequence generator can be realized with a wide range of sequence length. A pseudo random sequence of length 8 is generated and therefore at least three flip flops are employed. A D-type flip flops alongwith one XOR gate is used generating the sequence. The logic is set in such a way that

$$D1 = Q1 \oplus Q3$$

Where Q1 & Q3 are the outputs of first and third flip flop and D1 is the input to the first flip flop.

4.1.2 Vector Signal Generator (SMIQ02B)

The use of modern digital signal processor (DSP) technology, allows the generation of high-precision digital modulation signals at high bit rates without any limitations on modulation modes or standards [22]. The SMIQ feature a hitherto unrivalled versatility regarding signal generation and signal quality and is therefore ideal for use in development and type-approval testing. The wide frequency range from 300 kHz to 2.2 GHz covers almost all major radio bands, including the IF ranges. The high-grade I/Q modulator fitted as standard ensures minimum error vector magnitude and high intermodulation suppression.

4.1.2.1 Features of Vector Signal Generator SMIQ02B

- Covers a frequency range of 300 KHz to 2.2 GHz
- Generation of analog and digital modulation.
- Versatile and broadband generation of digitally modulated signals up to 18 Msymbol/s
- Generation of TDMA, CDMA, WCDMA and CDMA2000 standard signals to all main mobile radio standards
- Broadband I/Q modulator with outstanding vector accuracy
- Optional internal noise generator and distortion simulator
- Optional arbitrary waveform generator

4.1.2.2 Applications of Vector Signal Generator SMIQ02B

- Type-approval testing of digital base and mobile stations
- Sensitivity measurements on digital receivers
- Selectivity measurements on digital receivers
- Base-station transmitter test
- Testing of equalizers
- Components tests

4.1.2.3 RF Characteristics

- Wide output frequency range from 300 kHz to 2.2 GHz
- High (up to 16 dBm) and precise output level (<0.5 dB)
- Frequency hopping (500 μ s)
- RF, AF and level sweep (user-programmable)

4.1.3 ICS-660 B Digital to Analog Converter

The ICS-660B is a 4-channel, 14-bit short PCI Bus DAC card that operates in the 5 to 105 MHz conversion frequency range [23]. The ICS-660B can accept an optional digital upconverter daughter card (DC-60-M2) for programmable upconversion of upto 16 signals. The ICS-660B PCI bus board is designed to simplify complex waveform generation requirements by offering:

- Four separate 14-bit DACs
- Simultaneous conversion at rates upto 105 MHz/ channel
- Synchronous operation of multiple boards
- External or internal conversion clock
- External or internal trigger
- Continuous, one- shot or loop mode data conversion
- 4 MSample on-board swing buffer
- Programmable length PCI bus interrupts
- 200 Megabytes/s Front Panel Data Port (FPDP) interface
- Optional Daughter card for digital upconversion of upto 16 signals.

Figure 4.3 shows a simplified block diagram of the ICS-660B short PCI card. It provides all hardware infrastructure needed for simultaneous conversion of upto four channels of digital data using 14-bit DACs. To allow transfer of configuration information and DAC data from the host CPU, the ICS-660B board includes PCI interface circuitry using the semiconductors and chips. For receiving data from external sources such as DSPs, and FPDP I/II interface is also included. All DAC data is buffered in a 512k x 64 bit swing buffer memory. Data read from the FPDP is buffered in a 4k sample First in first out

(FIFO), before being transferred to the swing buffer. Clock and trigger for the board can be either from an internal or external source.

The frequency of the internal DAC clock is fixed at 100 MHz. Technically, the ICS-660B supports four modes of operations. These are continuous, loop, one-shot and pulse modes. Loop mode and pulse mode can be thought as derivatives of continuous and one-shot modes respectively, where the user supplies data to the card once, rather than constantly updating the card with fresh data.

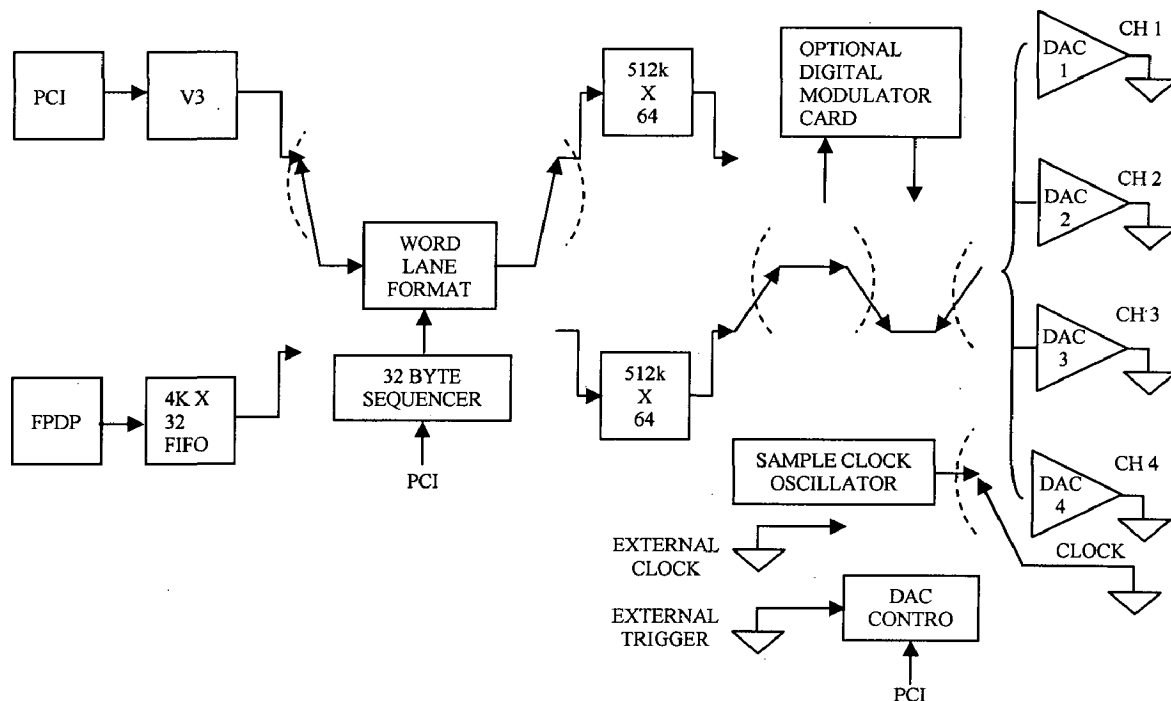


Figure 4.3: Simplified Block Diagram of DAC

4.1.4 ICS 652 Analog to Digital Converter

Figure 4.4 shows the block diagram of the ICS 652 ADC card. The ICS-652 is a 2-channel, 65 MHz/channel, ADC (Analog to Digital) board, designed for a wide range of data acquisition applications [24]. The ICS-652 is a short PCI card that offers over 90 dB SFDR (Spurious Free Dynamic Response) for high-precision and high-frequency applications. Optional daughter cards can be plugged directly into the ICS-652 board for programmable digital down-conversion of broadband or narrowband signals. The ICS-652 ADC and ICS-660B DAC boards are ideally suited for Software Defined Radio receive and transmit applications. The ADC's can be operated in a number of different modes.

- In the *continuous mode*, data is continuously supplied to the selected interface upon application of a trigger signal until the acquisition is disabled.
- In *capture mode*, a fixed number of samples are acquired upon each application of the trigger. The ICS-652 can be configured, in capture mode, to pre-store samples before the trigger and to acquire a programmable number of samples following the trigger (to a maximum of 524,288 samples/ channel).

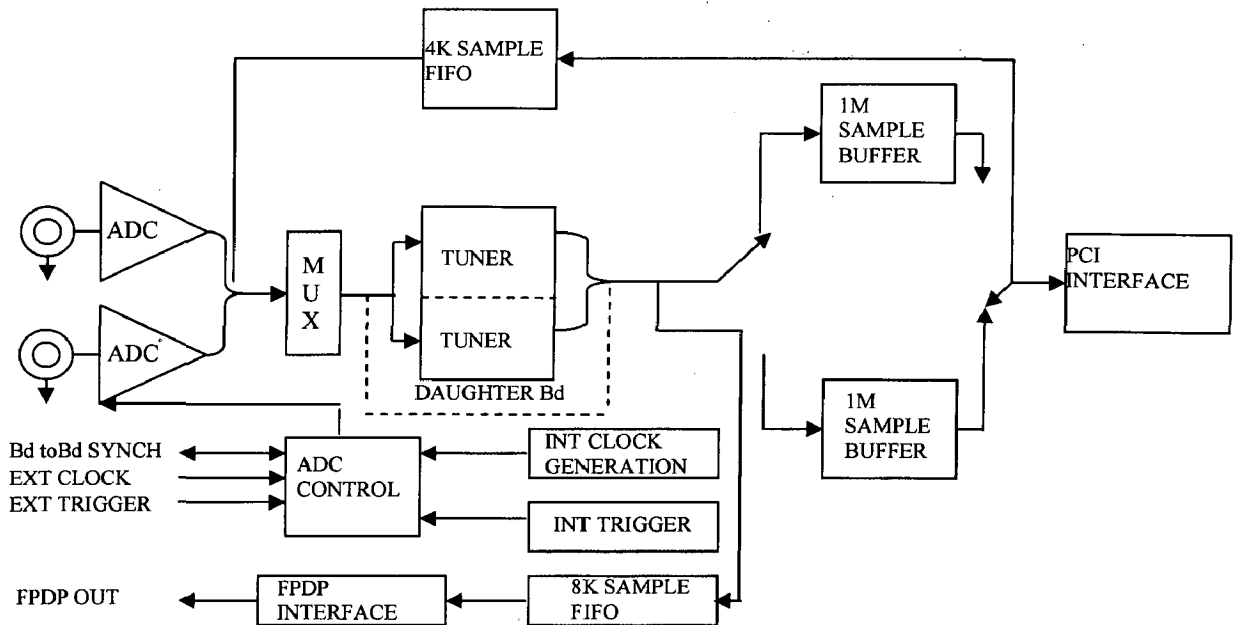


Figure 4.4: Simplified Block Diagram of ADC

Alternatively, the board may be programmed to acquire a fixed number of samples upon each application of the trigger without pre-trigger storage (again the maximum number of samples that may be acquired is 524,288 / channel).

4.1.4.1 ADC section

Figure 4.4 shows a simplified block diagram of the ICS_652 board. The board uses two 14-bit ADC to provide simultaneous sampling at rates of up to 65 MHz/ channels. The minimum sampling rate for the ADC's is 15 MHz, but the ICS-652 provides decimation of ADC data stream by a factor of up to 32, making the effective minimum sampling rate 468.75 KHz. Decimation is accomplished by storing one out of every N samples where N is programmable from 1 to 32. No filtering is performed prior to decimation on the

ICS-652 motherboard. The outputs of the two ADCs are passed to the digital modulator daughter card, if installed and enabled. The outputs are combined to produce a 32 bit word which is placed either in the FDPD FIFO and / or in the PCI swing buffer. The 2's complement samples are located in the most significant bits of each 16-bit portion of the word. To allow fast transfer of ADC data, the ICS-652 board includes a PCI interface capable of burst mode bus cycles, and a 32-bit FPDP interface. The transfer rate of the FPDP interface is fully programmable from 1 to 50 MHz. ADC data is buffered either in the 8 Ksample FIFO when data read out via the FPDP, or the dual-ported memories when data is read out via the PCI bus.

4.1.4.2 Sampling Clock

The sampling clock and the trigger can be either internal or external. The internal ADC clock is a crystal oscillator with a fixed frequency of 64 MHz. The external clock signal can have a TTL-level frequency in the range 15-65 MHz. The trigger can be programmed to be internal or external. The external trigger is a rising-edge TTL signal supplied by the user. The signal must remain high for at least one clock cycle. The trigger is internally synchronized to the sampling clock by the ICS-652; conversion is initiated on the second rising edge of the sampling clock after the application of the trigger. The sampling clock can also be either external or internal. The internal clock is a 64 MHz fixed frequency crystal oscillator.

4.1.4.3 FPDP interface

FPDP is an interconnection for board to board or system to system data transfer. It is a high performance 32 bit parallel interface configured with a ribbon to connect boards or systems together.

4.1.4.4 PCI bus interface

The ICS-652 uses PCI interrupts to indicate swing buffer swap or completion, tuner module overflow, and DMA transfer completion. Two bits in the ICS 652 interrupt mask register allow the user to enable the interrupts for swing buffer swap/ completion (ADC Interrupt) and Tuner Module overflow (Overflow Interrupt). The product specification is appended as follows:

1	No. of Analog Input Channels	2
2	Input Impedance	50 ohm
3	Full Scale Input	+/- 1.2 V
4	Input Signal Bandwidth	DC - 43 MHz (-3 dB point)(ICS-652 version) 2 - 200 MHz (ICS-652T version)
5	Max. Sampling Rate	65 MHz/ch., 2 channels simultaneous
6	ADC Resolution	14 Bits
7	Sampling	Rising edge of internal sample clock, rising or falling edge of external clock (programmable).
8	External trigger	Sampling occurs on second clock edge, following rising edge of external trigger. Trigger must remain high for at least one clock cycle.
9	Power	+5V @ 2.2A (without tuner module) +5V @ 2.9A (with tuner module, typical)
10	SFDR [0 - max. Fs/2]	> 90 dB

The digital signal so obtained is captured and stored in an array inside the buffer and is multiplied with sine and cosine waveforms in order to obtain I and Q components. These I and Q components are then passed through CIC decimation filter for decimation and lowpass filtering of high frequency components.

4.1.5 Design Consideration for CIC Filter

A significant consideration in CIC filters is the size (number of bits) of data that can pass through the filter without loss. The most significant bit (MSB) of the filter represents the maximum number of bits that can be propagated through the filter while maintaining the integrity of the data. Parameters R , M , N and the `InputWordLength` specify the MSB of the filter output. Since the output of the integrator sections of the filter can grow without bounds, the MSB at the filter output is also the MSB for all filter sections. Called B_{max} in the reference, the maximum word length in the filter, or most significant bit (MSB), is both the maximum word length for all of the filter sections as well as the MSB at the filter output. CIC filters include a property that defines how to specify the section word

length for the filter. Called SectionWordLength Mode, this property specifies the specific data format (word length) the filter uses when accumulating data in the integrator sections. SectionWordLength Mode can take one of two values:

- **MinWordLengths**—the filter calculates the optimal section word lengths given the filter parameters R (the rate change factor), M (the differential delay), N (the number of filter sections), and the input and output word lengths.
- **SpecifyWordLengths**—the word lengths for the sections can be specified by entering a scalar or a vector of length $2 \times N$. When a scalar is provided, the filter method expands the scalar into a vector with $2 \times N$ elements, applying the same word length to all sections. If a vector is specified, it must meet these requirements: It must contain $2 \times N$ elements; and the values of the vector elements must be monotonically decreasing.

During filtering least significant bits (LSBs) from each section of the filter can be discarded so long as the error introduced by removing the LSBs is acceptable at the filter output.

4.1.5.1 Design of CIC Decimation Filter

Let F_s denotes the sampling rate of ADC, F_c the carrier freq, F_b the baud rate, the desired output rate of the decimator F_d is

$$F_d = 16 \times F_b \tag{4.1}$$

Also the decimation factor $R = F_s / F_d$ (4.2)

Thus for our case, $F_d = 16 \times 10 \times 10^3 = 160$ KHz

And $R = 64 \times 10^6 / 160 \times 10^3$

Thus, $R = 400$ (1)

The normalized value of the frequency $f_c = F_c / F_s = 500 \times 10^3 / 64 \times 10^6 = 1/128$ (4.3)

Referring the standard table 3.1 and 3.2, as discussed in chapter three, let's design the filter assuming the following:

- Aliasing attenuation = 210.4 dB
- Passband attenuation = 0.00 dB

The value of differential delay M and number of sections used N , corresponding to this aliasing and passband attenuation is given as:

- $M = 1$;
- $N = 5$;

Here the truncating technique has been implemented. B_{\max} , the MSB at the filter output is given by $B_{\max} = [N \log_2 RM + B_m - 1]$

With

- $N = 5$
- $M = 1$
- $R = 400$

Thus, $MSB = B_{\max} = [5 \log_2 400 + 16 - 1] = 58$

The section word lengths are computed by subtracting the LSBs from the maximum word lengths in the filter. Thus the word lengths applied to each filter section with MSB 58 bits is shown as under:

<u>Filter Section</u>	<u>No of LSBs Discarded</u>	<u>Word Length Calculated (MSB-Discarded LSBs)</u>
1	1	57 (58-1)
2	6	52 (58-6)
3	9	49 (58-9)
4	13	45 (58-13)
5	14	44 (58-14)
6	15	43 (58-15)
7	16	42 (58-16)
8	17	41 (58-17)
9	18	40 (58-18)
10	19	39 (58-19)

4.1.6 Design of Chebyshev Filter

There are two types of Chebyshev low pass filters, both are based on polynomials. A type I Chebyshev low pass filter has an all-pole transfer function. It has an equiripple passband and a monotonically decreasing stop band. A type II Chebyshev low pass filter has both poles and zeros; its passband is monotonically decreasing, and it has an equiripple stop band. By allowing some ripple in the pass band or stop band magnitude

response, a Chebyshev filter can achieve a steeper pass-to-stop transition region (i.e., filter roll-off is faster) can be achieved. With a maximally flat response at $\Omega=0$, the type II Chebyshev low-pass filter exhibits a monotonic behaviour in the pass band and an equiripple response in the stop band.

To design a type II filter, we must know passband edge Ω_p , stop band edge Ω_s , a maximum passband attenuation factor δ_1 , and a minimum stop band attenuation factor δ_2 . We use this information to calculate ε , filter order N . First ε must be determined. At $\Omega = \Omega_p$, the filter response must be equal to the known δ_1 , so that

$$\delta_1^2 = \frac{1}{1 + \varepsilon^2} \quad (4.4)$$

This leads to

$$\varepsilon = \frac{\sqrt{1 - \delta_1^2}}{\delta_1} \quad (4.5)$$

Assuming the peak to peak ripple d as 0.04, ε may be calculated as

$$\varepsilon^2 = \frac{1}{(1 - .04)^2} - 1 = 0.29 \quad (4.6)$$

For designing the Chebyshev low pass filter, keeping in view the input signal of 10 KHz, and the output frequency of the decimator i.e. 160 KHz, let's consider the following:

- Pass band frequency = $f_p = 10$ KHz
- Stop band frequency = $f_{sb} = 150$ KHz
- Thus, cut off frequency = 80 KHz [assuming $(f_p + f_{sb})/2 =$ cutoff frequency]

Also, the attenuation in stop band is calculated as

$$-20 \log \delta_2 = 30 \quad (4.7)$$

$$\Rightarrow \delta_2 = G = 0.031$$

$$\text{Also, Passband frequency} = \Omega_p = 2\pi f_p = 2\pi \times 10 \times 10^3 \quad (4.8)$$

$$\omega_p = \Omega_p \times T_d = \frac{20\pi \times 10^3}{64 \times 10^6}$$

$$\Omega_p = \frac{2}{T_d} \tan\left(\frac{\omega_p}{2}\right)$$

$$\text{Similarly, Stopband frequency} = \Omega_s = 2\pi f_{sb} = 2\pi \times 150 \times 10^3 \quad (4.9)$$

$$\omega_s = \Omega_s x T_d = \frac{2\pi \times 150 \times 10^3}{64 \times 10^6}$$

$$\Omega_s = \frac{2}{T_d} \tan\left(\frac{\omega_s}{2}\right)$$

Calculating the normalized value

$$\Omega_s = \frac{\Omega_s}{\Omega_p} = 30 \text{ in radian}$$

In the stop band (i.e., for $\Omega \geq \Omega_s$), the square of the maximum response is the square of the stop band ripple peak value and is given by

$$\delta_2^2 = \frac{1}{1 + \varepsilon^2 T_N^2(\Omega_s / \Omega_p)} \quad (4.10)$$

Where

$$T_N^2(\Omega_s / \Omega_p) = \cosh^2\left(N \cosh^{-1}\left(\Omega_s / \Omega_p\right)\right) = \frac{1 - \delta_2^2}{(\varepsilon \delta_2)^2} \quad (4.11)$$

This yield, the order of the filter as

$$N = \frac{\cosh^{-1}\left(\sqrt{1 - \delta_2^2} / \varepsilon \delta_2\right)}{\cosh^{-1}\left(\Omega_s / \Omega_p\right)} \quad (4.12)$$

With $\Omega / \Omega_p > 1$, the function $T_N(\Omega / \Omega_p)$ increases with increasing N . Since δ_2 is fixed, rounding N upward causes a decrease in the effective value of ε .

Thus calculating the order of the filter

$$N \geq \frac{\cosh^{-1}\left(\frac{\sqrt{1 - 0.031^2}}{0.031 \times 0.29}\right)}{\cosh^{-1}(30)} \geq 1.32$$

Thus, the order of the filter is assumed as two.

4.2 APPLICATION SOFTWARE

The MATLAB application software provides an interface between MATLAB and the ADC/ DAC board windows drivers [25] (written in VC++). This is done by means of a series of MEX-files (examples ICS-652ConfigGet.c, etc); these functions can be called directly from MATLAB M-files. The format of the “C” language MEX-files is defined by MATLAB. Each one provides access to part of the functionality of these boards. Taken as a whole, they provide control over the operation of the board. If the operating system does not support the Dynamic Link Library (DLL) files that come with the MATLAB driver, then either they must be compiled and built to produce a set of DLL files; or else the precompiled and built DLL files, that come with driver can be used. MATLAB M-files have also been included in the software package. These applications (M-files) allow the user to perform a simple data acquisition with the boards (eg ICS-652) and plot the results in the time and frequency domains.

The ICS-652 MATLAB Application software requires a personal computer with the following:

- Microsoft windows 2000/ NT 4.0/ 98/ ME/ XP
- MATLAB 6.0 R12 or later
- ICS-652 Windows software development kit
- Microsoft visual C ++ 6.0 or later

The ICS-652 Application Programmer’s Interface (API) is supplied as a windows DLL file. A reference of APIs, along with the outline of the program written (to include the CIC decimation filter and Chebyshev low pass filter) is covered in detail as under:

- **[error hICS652] = ML652DevOpn(boardName)** - This initializes the board. This MAX-file opens a device handle to the driver, boardName is the device name.
- **Int ics652BoardReset (IN HANDLE hnd)** - This routine reset the ICS-652 to its power up default condition. All interrupts are disabled and the previous configuration is cleared. *hnd* is the ICS-652 board file handle
- **temp=ML652StructureToInt32(allocations.ICS652_CONFIG, CS652_CONFIG)**– MATLAB does not support the C language data type of

structures. Thus, before calling functions that have this data type the parameters must be formatted to

- 32-bit numbers. The M-file `ML652StructureTo Int32.m` provided in the MATLAB driver performs this conversion.
- $e = \text{ics652ConfigSet}(h\text{ICS652}, \text{temp})$ - This routine writes the value contained in the structure to the control register.
- $e = \text{ics652AcquireCountSet}(h\text{ICS652}, \text{config.transmitLength} - 1)$ - This routine sets the Acquisition Count register. The Acquisition Count is programmed as the number of frames of data (less 1) to be acquired following trigger occurrence
- $e = \text{ics652BufferLengthSet}(h\text{ICS652}, \text{config.bufferLength} - 1)$ - This routine sets the value of the buffer length. The buffer length is programmed in terms of the number of 32-bit words to be stored in each side of the swing buffer
- $e = \text{ics652DecimationSet}(h\text{ICS652}, \text{config.decimation} - 1)$ - This routine sets the value of the decimation register.
- $e = \text{ics652Enable}(h\text{ICS652})$ - This routine enables acquisition of ICS-652. A trigger is supplied after this in order for the acquisition to occur.
- $e = \text{ics652ADCReset}(h\text{ICS652})$ - This routine resets all on board memories and loads all configuration information onto the ICS-652. This is done as the last activity before enabling acquisition.
- $[e \text{ bufHnd}] = \text{ML652Allocate}(\text{config.bufferLength})$ - This MEX- file allocates a buffer for the ICS-652 to write too, bufferlength is the length of the buffer programmed on the ICS-652
- $e = \text{ics652Trigger}(h\text{ICS652})$ - This routine initiates storage when internal triggering has been selected. A call to this function causes sampling to begin in continuous mode. Data storage will continue until `ics652Disable()` is called
- $\text{ics652WaitADCInt}(h\text{ICS652}, 10)$ - This function enables ADC interrupts for the specified board and then waits for the interrupt to occur, or for the timeout period to expire, whichever comes first. When this function is called, the current task is suspended during the waiting period, and resumes at the following statement when one of the two events occurs.

- $[e \text{ data}] = ML652RdData(hICS652, bufHnd, config.bufferLength)$ - This file reads data from the buffer, which is written to by the ICS-652, handle is the handle to the device driver, bufferHandle is the handle to a buffer written to by the ICS-652, bufferLength is the number of 32-bit words to read from the buffer
- **CIC-Decimation Filter** is designed using MATLAB. With the specifications discussed below, the cascaded integrator-comb (CIC) decimation filter is designed as:

$hm = mflt.cicdecim(R, M, N, iwl, owl, wbps);$

where

- R is the decimation factor applied to the input signal;
- M is the differential delay.
- N is the number of sections. This is the number either of comb or integrator sections, and not the total section count.
- Iwl is the word length of the input signal.
- Owl is the word length of the output signal.
- $Wbps$ defines the number of bits per word in each filter section while accumulating the data in the integrator sections

CIC filter objects allow only fixed-point arithmetic (the Arithmetic property is always set to fixed) since these filters are inherently fixed-point filters. Various parameters as discussed in section 4.1.5.1 are taken into account for the design. These are given as under:

- FilterStructure: 'Cascaded Integrator-Comb Decimator'
- Arithmetic: 'fixed'
- DifferentialDelay: 1
- NumberOfSections: 5
- DecimationFactor: 400
- PersistentMemory: false
- InputWordLength: 16
- InputFracLength: 15
- FilterInternals: 'SpecifyWordLengths'

- SectionWordLengths: [57 52 49 45 44 43 42 41 40 39]
- OutputWordLength: 16

The Filter Visualization Tool (FVTool) provides graphical access to all analyses. FVTool returns the magnitude response for hm , shown in figure 4.8

$hfvf=fvtool(hm);$

Finally both I and Q components are convolved with the impulse response of the CIC decimation filter. This I and Q components thus obtained are low pass filtered using Chebyshev low pass filter.

- **Chebyshev Low Pass Filter** is designed using MATLAB.

$[b, a] = cheby2(N, R, W_{st}, 's');$ designs an order N lowpass analog Chebyshev Type II filter with angular stopband edge frequency W_{st} rad/s. It returns the filter coefficients in the length $N + 1$ row vectors b and a . Various parameters required for designing the filter are calculated in section 4.1.6. These are summarized as under:

- Order of the filter $N = 2$;
- Stopband ripple $R = 0.5$ dB;
- Angular stopband edge frequency $W_{st} = 30$ radian per second

For cheby2, the angular stopband edge frequency W_{st} must be greater than 0 rad/s.

After this the impulse response of this low pass filter is given by:

$[l1, T11] = impz(b, a, 1);$

Which computes the impulse response of the filter with numerator coefficients $l1$ and denominator coefficients $T11$. This impulse response is then convolved with I and Q component obtained as the output of the CIC decimation filter as:

$I_comp2 = conv(I_comp1, l1);$

$Q_comp1 = conv(Q_signal, l1)$

- $e = ics652Disable(hICS652)$ - In the end, this routine disables acquisition of the ICS-652. Any acquisition is immediately halted.
- $e = ML652DevCls(hICS652)$ - Finally this MEX-file closes the device handle to the driver at the end of the program.

The detailed MATLAB code used for implementing the system shown in figure 4.1 is attached as Appendix.

4.3 RESULTS

4.3.1 PN Sequence Generation

Figure 4.2 shows the circuit of a PN sequence generator. The output of first and third flip-flop is assumed as Inphase (I) and Quadraturephase (Q) components. These I and Q components, as obtained are shown in figure 4.5. This output is used as I and Q inputs to the SMIQ-02B vector signal generator, which generates the vector modulated QPSK signal.

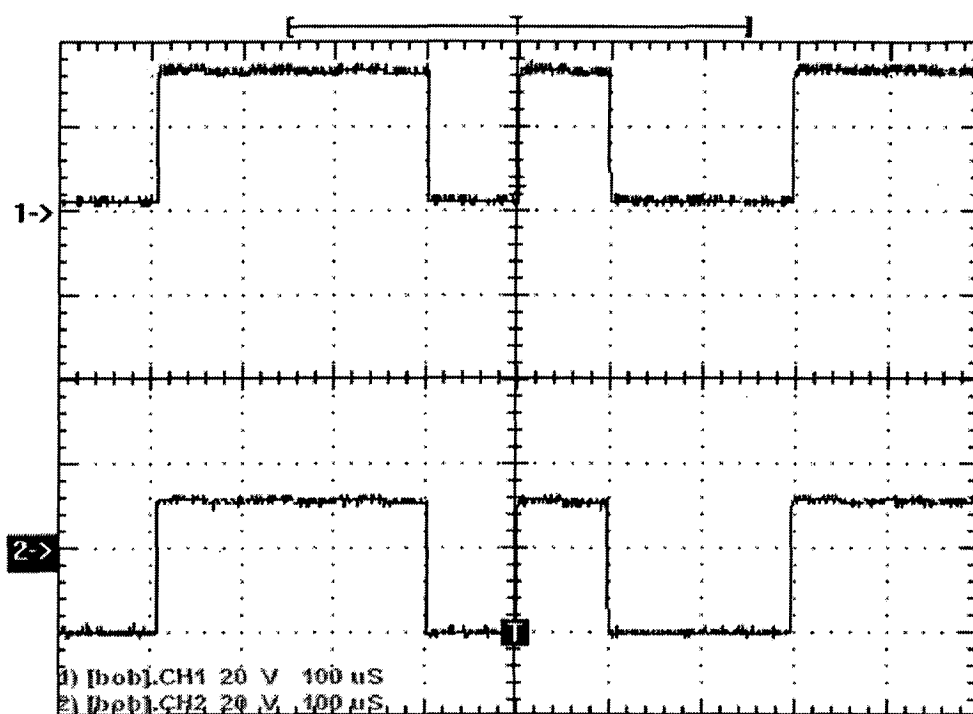


Figure 4.5: Generated PN Sequence

4.3.2 Output of Vector Signal Generator SMIQ-02B

The vector signal generator takes I and Q components and generates a vector modulated QPSK signal as shown in figure 4.6. This output is sampled, quantized using ICS-652 ADC card.

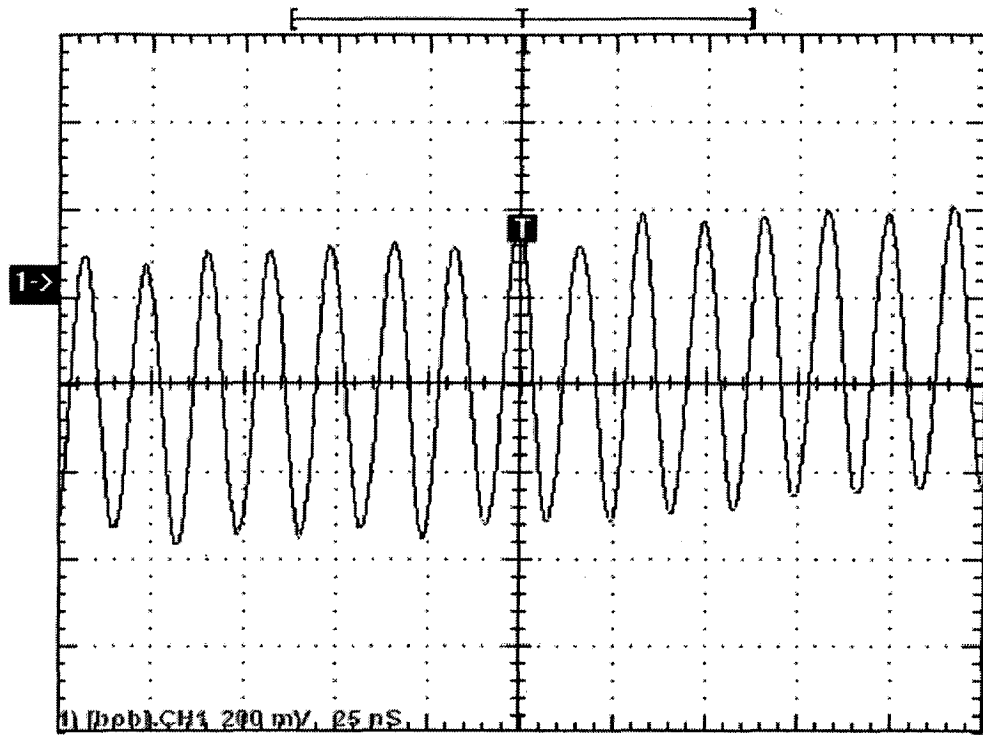


Figure 4.6: Output of Vector Signal Generator

4.3.3 Output of ICS-652 ADC card

The vector-modulated output of vector signal generator is digitized using an ICS_652 ADC card having a sampling frequency of 64 MHz. The output from the ADC card is captured and is stored in an array inside the buffer. This output is shown in figure 4.7

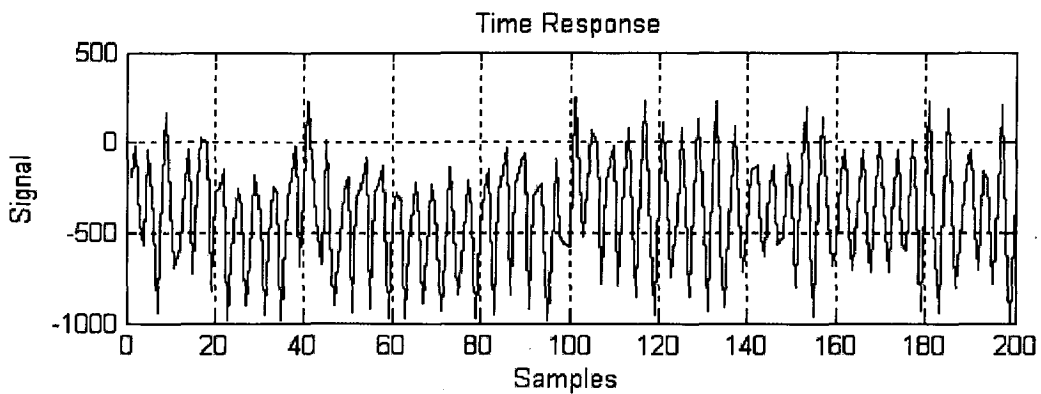


Figure 4.7: Output Stored in Buffer as Array

4.3.4 CIC Decimation Filter

Figure 4.8 shows the frequency response plots for specific value of the design parameters N , R and M . For any CIC filter, there are always RM zeros in the transfer function. The zeros are equally spaced around the unit circle in the z -plane at integer multiples of $1 / RM$ - there is of course no zero at $z = 0$. Increasing N has the effect of increasing the order of the zeros. This, in turn, increases the attenuation at frequencies in the locality of the zero. As N increases, attenuation of the filter sidelobes also increases. Also, as the order of the zeros increase, the passband droop also increases, thus narrowing the filter bandwidth. The increased droop may not be acceptable in some applications. The droop is frequently corrected using an additional (non-CIC-based) stage of filtering after the CIC decimator. The designing of CIC and Chebyshev low pass filter is done using MATLAB code.

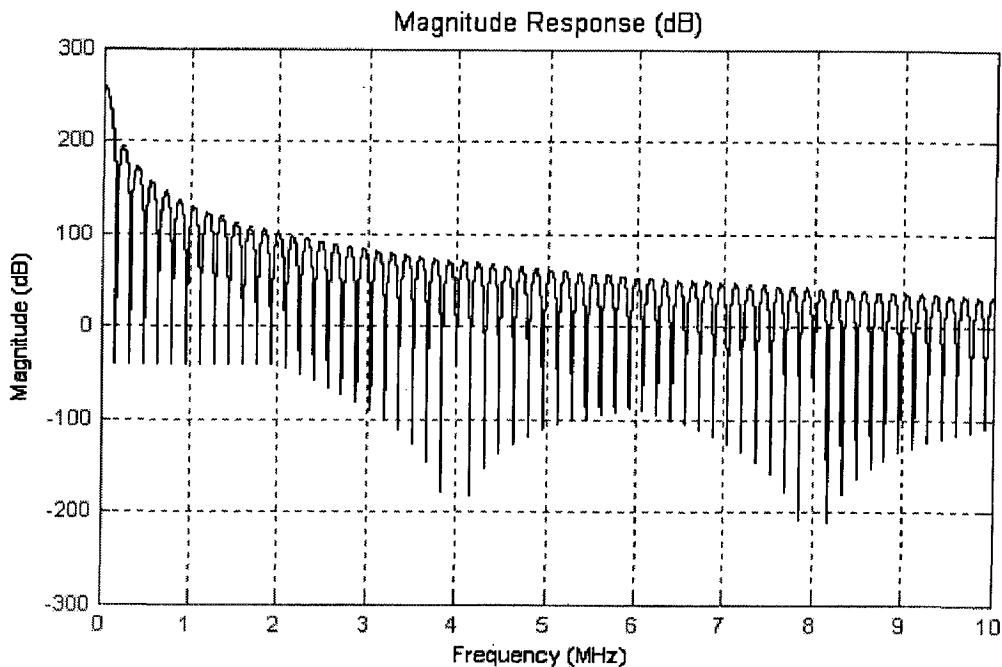


Figure 4.8: Frequency response of CIC Filter

The output signal after decimation is shown in figure 4.9. Figure shows output as 1 and 0 bit. In typical decimation filtering applications we want reasonably flat passband and narrow transition-region filter performance. These desirable properties are not provided by CIC filters alone, with their drooping passband gains and wide transition regions. We alleviate this problem by following the CIC filter with a low pass

filter, to narrow the output bandwidth and flatten the passband gain. The low pass filter's frequency magnitude response is ideally an inverted version of the CIC filter passband response. The low pass filter used for this purpose is a Chebyshev low pass filter.

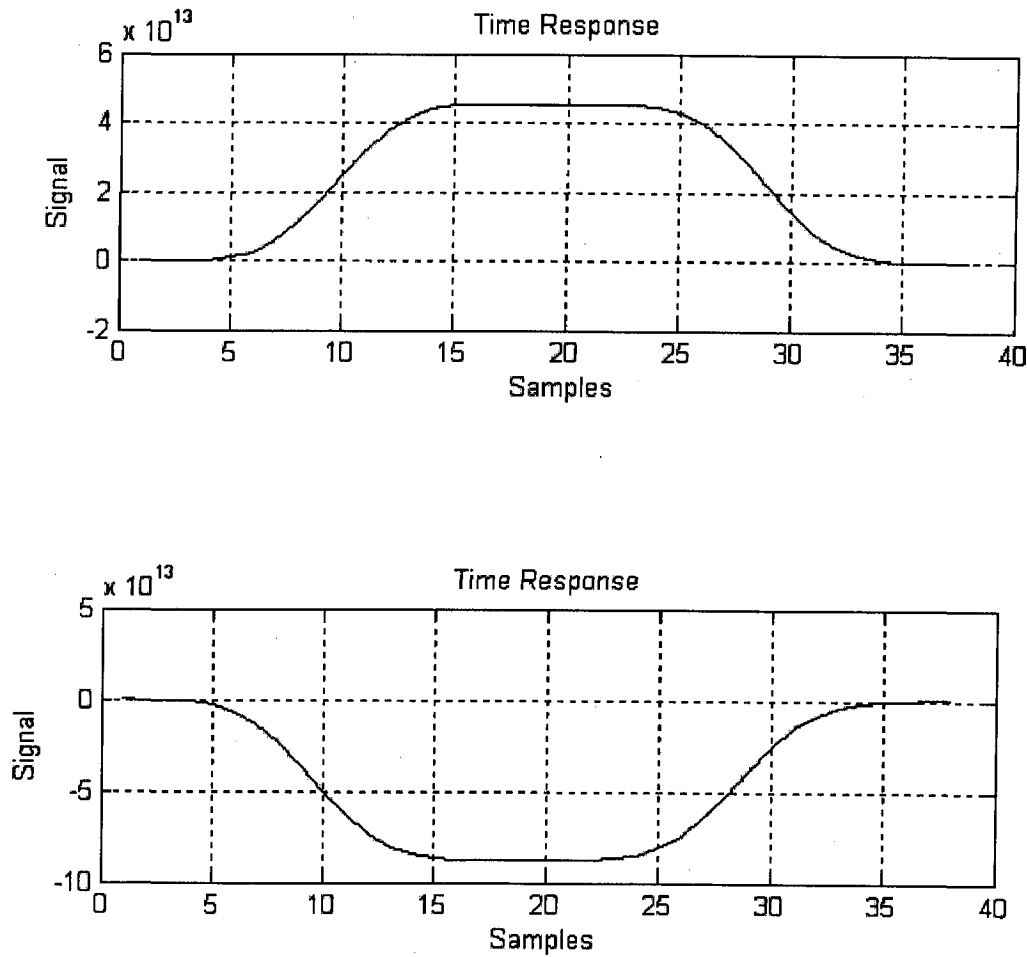


Figure 4.9: CIC Decimation Filter Output

4.3.5 Chebyshev Low Pass Filter

The final demodulated output (I-component here, Q-component being identical as I and Q inputs are same) after decimation and low pass filtering using this Chebyshev lowpass filter is shown in figure 4.10

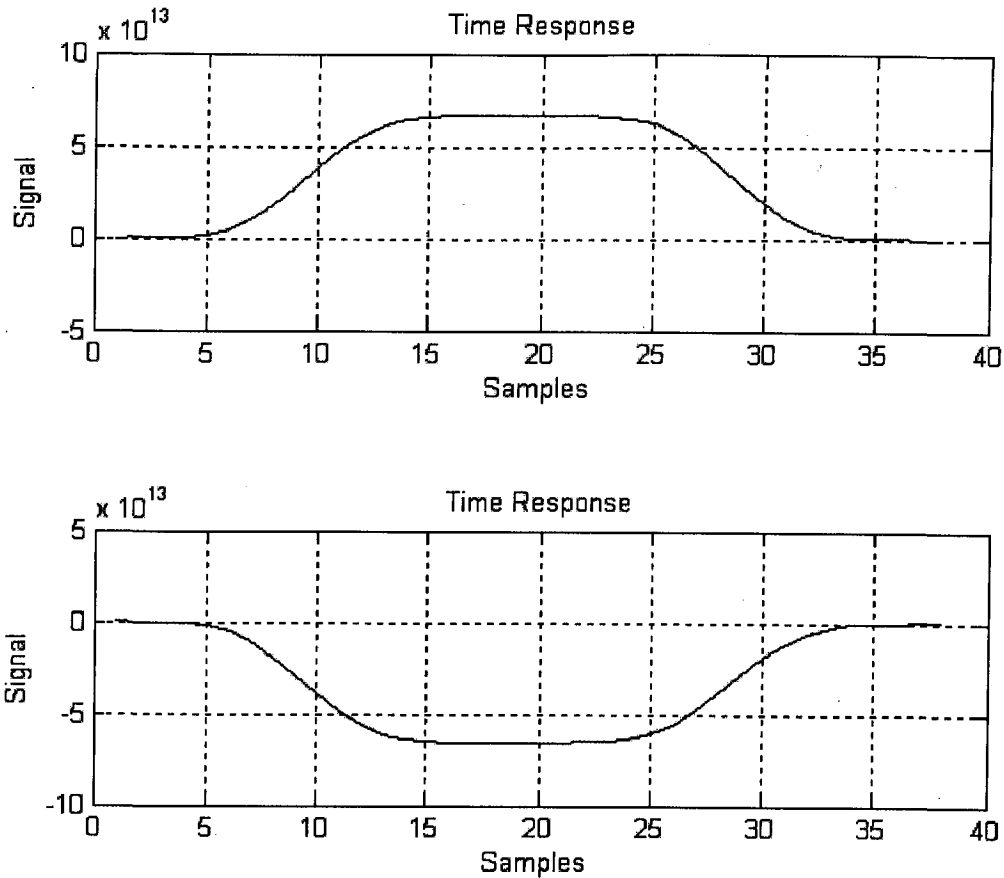


Figure 4.10: Demodulated Data

4.4 CONCLUSION

In this dissertation, a class of digital linear phase finite impulse response (FIR) filter for decimation (sampling rate decrease) is presented [20]. It requires no multipliers their arithmetic is strictly addition and subtraction and use of limited storage makes them an economical alternative to conventional implementations for certain applications. Their performance allows us to state that, technically speaking, CIC filters are lean, mean filtering machines. A digital filter in this class consists of cascaded ideal integrator stages operating at a high sampling rate and an equal number of comb stages operating at a low sampling rate. Together, a single integrator-comb pair produces a uniform FIR. The number of cascaded integrator comb pairs is chosen to meet design requirements for aliasing or imaging error.

The frequency response of CIC filters is fully determined by only three integer parameters resulting in a limited range of filter characteristics. The aliasing/imaging error

in the passband can be held within arbitrary bounds by appropriate choice of these parameters. However, the bandwidth and the frequency response outside the passband are severely limited.

4.4.1 Nonrecursive CIC Decimation Filters

CIC filters are computationally efficient and simple to implement [26]. However, one of the difficulties in using CIC filters is accommodating large data word growth, particularly when implementing integrators in multistage CIC filters. There are various methods available to reduce the complexity, and enhance the usefulness, of CIC filters. Here's a method that eases the word width growth problem using nonrecursive CIC decimation filter structures, obtained by means of *polynomial factoring*. These nonrecursive structures achieve computational simplicity through *polyphase decomposition* if the sample rate reduction factor R is an integer power of two. The transfer function of an N th-order decimation CIC filter can be expressed in either a recursive form or a nonrecursive form as given by (3.8) as

$$H(z) = H_I^N(z)H_C^N(z) = \frac{(1 - z^{-RM})^N}{(1 - z^{-1})^N} = \left[\sum_{k=0}^{RM-1} z^{-k} \right]^N \quad (4.13)$$

$$= (1 + z^{-1} + z^{-2} + \dots + z^{-RM+1})^N \quad (4.14)$$

Considering the differential delay M as 1, if the sample rate change factor R is an integer power of two, then $R = 2^j$ where j is a positive integer and the N th-order nonrecursive polynomial form of $H(z)$ in (4.14) can be factored as

$$H(z) = (1 + z^{-1})^N (1 + z^{-1})^N (1 + z^{-2})^N (1 + z^{-4})^N \dots (1 + z^{-2^{j-1}})^N \quad (4.15)$$

The benefit of the factoring given in (4.15) is that the CIC decimation filter can then be implemented with j nonrecursive stages as shown for the multistage CIC filter in figure 4.11

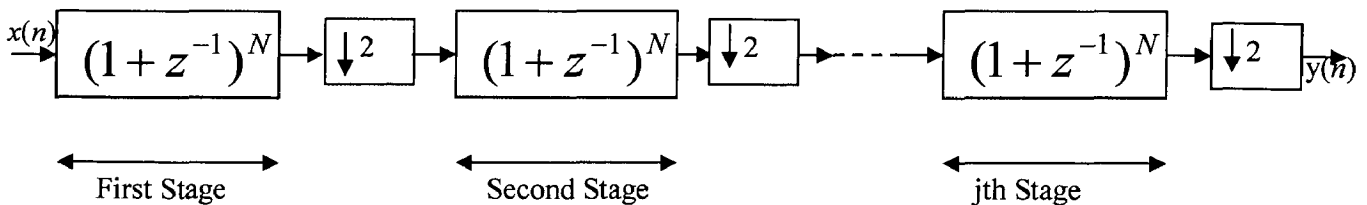


Figure 4.11: Multistage N th order nonrecursive CIC structure

This implementation trick eliminates the integrators with their unpleasant binary word width growth. The data word widths increase by only N bits per stage, while the sampling rate is reduced by a factor of two for each stage. The cascade of nonrecursive subfilters in figure 4.11 are still called CIC filters even though they have no integrators.

REFERENCES

- [1] Jeffrey H. Read, "Software Radio- A Modern Approach to Radio Engineering", Pearson Education, pp 1-10, 2002
- [2] Adam S. Harrington, Chin-Gi Hong, and Anthony L. Piazza, "Software Defined Radio: The Revolution of Wireless Communication", www.bsu.edu/cics/media/pdf/620_sdr_final.pdf, pp 14-18, 2004
- [3] Paul Burns, "Software Defined Radio for 3G", Artech House, 2002
- [4] Selvaraj Seefharaman, Zoran Kotic, "Digital Signal Processors in Cellular Radio Communications", IEEE Communications Magazine, pp 34, December 1997
- [5] Srikathyayani Srikanteswara, Ramesh Chembil Palat, Jeffrey H. Reed, and Peter Athanas, "An Overview of Configurable Computing Machines for Software Radio Handsets", IEEE Communications Magazine, pp 134-141, Jul 2003
- [6] Ronald M. Hickling, "New technology facilitates true software-defined radio", www.rfdesign.com, pp 18-20, April 2005
- [7] Simon Haykin, "Cognitive Radio: Brain-Empowered Wireless Communications", IEEE Journal on Selected Areas in Communications, Vol. 23, No. 2, pp 201-202, February 2005
- [8] Joseph Kennedy and Mark C. Sullivan, "Direction Finding and "Smart Antennas" Using Software Radio Architectures", IEEE Communications Magazine, pp 62-67, May 1995
- [9] Joe Mitola, "The Software Radio Architecture", IEEE Communications Magazine, pp 30, May 1995
- [10] Afshin Haghighat, John Fitton, "Enabling Technologies for Software Defined Radio", HARRIS Corporation, MCD Montreal, Canada - RFCD Rochester, U.S.A
- [11] K El-Sunkay, and M Sawan, "High Resolution Self-Calibrated ADCs for Software Defined Radios" Poly STIM Neurotechnology Laboratory, Ecole Polytechnique de Montreal, Department of Electrical Engineering, Montreal, Canada, IEEE, pp.120, 2004
- [12] Brian Black, "Analog-to-Digital Converter Architectures and Choices for System Design, Analog Dialogue, Volume 33, Number 8, pp. 1-3, September 1999

- [13] Sanjit K. Mitra, "Digital Signal Processing- A Computer-Based Approach", Tata McGraw-Hill Publishing Company Limited, pp. 815-820, 2001
- [14] Akira Fujimaki, Koichi Nakazono, Hiroaki Hasegawa, Takashi Sato, Akira Akahori, Nobuo Takeuchi, Futoshi Furuta, Masaaki Katayama, and Hisao Hayakawa, "Broad Band Software-Defined Radio Receivers Based on Superconductive Devices", pp. 2-3
- [15] Walter Tuttlebee, "Software Defined Radio, Enabling Technologies", John Wiley & Sons, LTD, pp 158-163, 2002
- [16] Mike Petrowski, David B. Chester, and W Ronald Young, "Single Chip Digital Down Converter Architecture", IEEE, pp.349-351, 1993
- [17] G. Girau M. Martina A. Molino A. Terreno F. Vacca, "FPGA Digital Down Converter IP for SDR Terminals", IEEE, pp.1011, 2002
- [18] Sanjit K. Mitra, "Digital Signal Processing- A Computer-Based Approach", Tata McGraw-Hill Publishing Company Limited, pp. 47-689, 2001
- [19] Kambiz C. Zangi, and R. David Koilpillai, "Software Radio Issues in Cellular Base Stations", IEEE Journal on Selected Areas in Communications, Vol. 17, No. 4, pp 561-572, Apr. 1999
- [20] E B Hogenauer, "An Economical Class of Digital Filters for Decimation and Interpolation", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. Assp-29, No. 2, pp. 155-161, April 1981
- [21] <http://www.datasheets.org.uk>, " IC 74174-DatasheetArchive.com"
- [22] Operating Manual, "Vector Signal Generator SMIQ-02B", Rhode and Schwartz, Republic of Germany
- [23] ICS-660B, "Operating Manual", Interactive Circuits and Systems Ltd, www.ics-ltd.com, September 2004
- [24] ICS-652, "Operating Manual", Interactive Circuits and Systems Ltd, www.ics-ltd.com, September 2002
- [25] ICS-652, "MATLAB Application Software manual", Interactive Circuits and Systems Ltd, www.ics-ltd.com, August 2003
- [26] Ricardo A. Losada and Richard Lyons, "Reducing CIC Filter Complexity", IEEE Signal Processing Magazine, pp 124-126, July 2006

Appendix

```
function ICS650Continuous
% calling impulse response of CIC filter
l=cic();

boardName = '\\.\ICS652-1';
% Constants
ICS652_DISABLE      =0;
ICS652_ENABLE       =1;
ICS652_CONTINUOUS   =0;
ICS652_CAPTURE      =1;
ICS652_PCI          =0;
ICS652_FPDP         =1;
ICS652_LOW          =0;
ICS652_HIGH         =1;
ICS652_INTERNAL     =0;
ICS652_EXTERNAL     =1;
ICS652_FALLING      =0;
ICS652_RISING       =1;
ICS652_COMPLEX      =0;
ICS652_REAL         =1;
ICS652_80           =0;
ICS652_90           =1;
ICS652_OK           =1;

%Open handle to the board
[error hICS652] = ML652DevOpn(boardName);

% Defining sample rate
config.sampleRate = 64;
% Configuring buffer length
config.bufferLength = (2048*4);
config.transmitLength = config.bufferLength;

config.decimation = 1;
config.channelCount = 2;
```

Appendix: Code Listing

```
config.numSampsPerChan =  
(config.bufferLength*4)/(config.channelCount*2);  
  
%ICS652_INTERNAL or ICS652_EXTENAL  
ICS652_CONFIG.trigsel    = ICS652_INTERNAL;  
  
%ICS652_INTERNAL or ICS652_EXTENAL  
ICS652_CONFIG.clksel    = ICS652_INTERNAL  
  
%ICS652_FALLING or ICS652_RISING  
ICS652_CONFIG.xclkedge = ICS652_RISING;  
  
%ICS652_HIGH or ICS652_LOW  
ICS652_CONFIG.freqsel   = ICS652_HIGH;  
  
%ICS652_ENABLE or ICS652_DISABLE  
ICS652_CONFIG.modulese1 = ICS652_DISABLE;  
  
%ICS652_CAPTURE or ICS652_CONTINUOUS  
ICS652_CONFIG.mode      = ICS652_CONTINUOUS  
  
%ICS652_ENABLE or ICS652_DISABLE  
ICS652_CONFIG.fpdpenable= ICS652_DISABLE;  
  
%ICS652_ENABLE or ICS652_DISABLE  
ICS652_CONFIG.diagenable= ICS652_DISABLE;  
  
%ICS652_ENABLE or ICS652_DISABLE  
ICS652_CONFIG.enable    = ICS652_ENABLE;  
  
%ICS652_ENABLE or ICS652_DISABLE  
ICS652_CONFIG.inttrig   = ICS652_DISABLE;  
  
%ICS652_PCI or ICS652_FPDP  
ICS652_CONFIG.datapath  = ICS652_PCI;  
ICS652_CONFIG.filler1   = 0;  
allocations.ICS652_CONFIG = [1 1 1 1 1 1 1 1 1 1 2 20];
```

Appendix: Code Listing

```
% Reset everything on the board
e = ics652BoardReset(hICS652);

% Configure the board and choose PLL frequency range
ICS652_CONFIG.freqsel = config.sampleRate > 25;

temp = ML652StructureToInt32(allocations.ICS652_CONFIG, ICS652_CONFIG);
e = ics652ConfigSet(hICS652, temp);
e = ics652AcquireCountSet(hICS652, config.transmitLength - 1);
e = ics652BufferLengthSet(hICS652, config.bufferLength - 1);
e = ics652DecimationSet(hICS652, config.decimation - 1);
e = ics652ADCClkSet(hICS652, config.sampleRate);
e = ics652Enable(hICS652);
e = ics652ADCReset(hICS652);
pause(1e-3);
[e bufHnd] = ML652Allocate(config.bufferLength);

%allocate array
data = zeros(1,config.bufferLength);

%allocate array
plotDataTemp = zeros(1,config.numSampsPerChan*config.channelCount);

% acquire and display data
plotData = zeros(config.numSampsPerChan, config.channelCount);
done = 0;
h = figure('visible','off','name','ICS652 Continuous
Acquisitions','NumberTitle','off','Backingstore','off','DoubleBuffer','
on','renderer','painters');
s1 = subplot(2,1,1);
set(s1,'DrawMode','fast');
s2 = subplot(2,1,2);
set(s2,'DrawMode','fast');
try
    fprintf('Press "Enter" to Trigger (Ctrl C to quit)','s');
    pause
    e = ics652Trigger(hICS652);
    while ~done
```

Appendix: Code Listing

```
        if ics652WaitADCInt(hICS652,10);
            [e data] = ML652RdData(hICS652, bufHnd,
config.bufferLength);
        else
            fprintf('Timed out...\n');
            break;
        end

% retrieve data and convert it to channel data form
dataLSW = floor(mod(data,2^16));

%get LSW of ADC
negLSW = find(dataLSW > 32767);

%convert to 2's complement of negative LSB
dataLSW(negLSW) = -1.*(2^16 - dataLSW(negLSW));
dataMSW = floor(data./2^16);

%Extracts most significant word;
plotDataTemp(1:2:end) = dataMSW;
plotDataTemp(2:2:end) = dataLSW;
plotData =
reshape(plotDataTemp,config.channelCount,config.numSampsPerChan)';

%*****
                CIC decimating filter
%*****

% carrier frequency
f= 500000;
    for q=1:2
        for r=1:(2048*4)
            t1(q,r)=((q-1)*2047)+(r-1);
        end
    end
end
t= (t1)/(64e6);
```

Appendix: Code Listing

```
% I component
J= cos (2*pi*f*t);

% Q component
D= sin (2*pi*f*t);
plotData1=plotData';

% Defining I and Q components
I_signal1=J.*real(plotData1);
Q_signal1=D.*real(plotData1);

count=1;
    for q=1:2
        for r=1:(2048*4)
            I_signal(count)=I_signal1(q,r);
            Q_signal(count)=Q_signal1(q,r);
            count=count+1;
        end
    end

% Decimation factor
R = 400;
% Differential delay
M = 1;
% Number of sections
N = 5;
% Input word length
IWL = 14;
% Input word length
OWL = 14;
hcic = mfilt.cicdecim(R,M,N,IWL,OWL,[33 28 25 21 20 19 18 17 16 15]);

% Set input fractionlength
hcic.InputFracLength = 18;

% Impulse response of CIC decimation filter
l=impz(hcic);
I_comp1=conv(I_signal,l);
```

Appendix: Code Listing

```
%*****  
                                Chebyshev Low pass Filter  
%*****  
  
% Defining parameters n and Wn  
N=2;  
Wst =30;  
[b,a] = cheby2(N,0.5, Wst, 'low');  
  
% Impulse response of the filter  
[l1,Tl1] = impz(b,a,1) ;  
% Convoluting impulse response with I component  
I_comp2=conv(I_comp1,l1);  
% Convoluting impulse response with Q component  
Q_comp1=conv(Q_signal,1);  
N=2;  
Wst =30;  
[b1,a1] = cheby2(N,0.5, Wst, 'low');  
  
[l11,Tl11] = impz(b1,a1,1) ;  
  
Q_comp2=conv(Q_comp1,l11);  
I_comp=I_comp2';  
Q_comp=Q_comp2';  
z=size(I_comp);  
y=size(Q_comp);  
for i=1:z  
    if (I_comp(i)>0)  
        I_comp(i)=5;  
    else  
        I_comp(i)=0;  
    end  
end  
for i=1:y  
    if (Q_comp(i)>0)  
        Q_comp(i)=5;  
    else  
        Q_comp(i)=0;
```

Appendix: Code Listing

```
        end
    end
    count=1;cnt1=0;cnt2=0;
    for i=1:32:z
        for m=1:16
            data(count)=I_comp(m+cnt1);
            count=count+1;
        end
        for n=1:16
            data(count)=Q_comp(n+cnt2);
            count=count+1;
        end
        cnt1=cnt1+32;cnt2=cnt2+32;
    end
    for i=1:z
        data(i)=xor(I_comp(i),Q_comp(i));
    end
    b=size(data);
    for i=1:b
        if (data(i)>2.5)
            data(i)=5;
        else
            data(i)=0;
        end
    end
    end
    set(h,'visible','on');
    subplot(s1);
    plot(real(Q_comp));
    xlabel ('Samples');
    ylabel ('Signal');
    title ('Time Response');
    xlim ([0 size(plotData,1)]);
    grid on;
    % windowing
    coeff = blackman(size(plotData,1));
    mCoeff = repmat(coeff,1,config.channelCount);
    plotData = plotData.*mCoeff;
    len = size(plotData,1)./2; %calculate FFT length
```

Appendix: Code Listing

```
warning off
f=abs((fft(plotData)));
f=20*log10(f/(2^15*len));
warning on
subplot(s2);
plot(f(1:len,:));
xlabel ('Samples [Bin]');
ylabel ('Spectrum Power [dB]');
title ('Spectrum');
ylim([-160 0]);
xlim ([0 len]);
grid on;
pause(1/10000);
end
```

```
%*****
```

```
clear ML652Allocate;
e = ics652Disable(hICS652);
e = ML652DevCls(hICS652);
clear all
close all
catch
clear ML652Allocate;
e = ics652Disable(hICS652);
e = ML652DevCls(hICS652);
clear all
close all
end
```