Scientific Journal of Impact Factor (SJIF): 5.71
e-ISSN (O): 2348-4470
p-ISSN (P): 2348-6406

International Journal of Advance Engineering and Research Development

Volume 5, Issue 02, February -2018

A Study on Prevention of Noises in Mobile Communication

Dr P.J.Arul Leena Rose

Department of Computer Application, SRM IST (Deemed to be University) Chennai-603203

Abstract:- *In the recent trend of Cellular Mobile electronic is to fabricate the IC by using Soft wares like the VHDL and VERILOG HDL. The noise preventer circuit is a small part in the mobile phone, which is implemented by using the VHDL software for making the mobile handset as compact size. There are many techniques, such as data scramblers, check codes, Block codes and convolution interleaver. Those are available in electronics market. These are used to arrest or eliminate the noise in the digital communication field. In this paper we use the Convolution Interleaver method to eliminate the noises due to the causes of fading. Thus the convolution interleaver is designed by using the shift registers and delay elements such as flip-flops. The circuit is coded by using the VHDL software. After simulating, process of the coding, then this software codings, are dumped into the Xilinx processor by using Xilinx project navigator package. Future scope of our paper is to reduce the complexity of Circuit as much as possible and also to minimize the size of the equipment. Thus this paper achieve a high efficient data transmission and reception in our mobile communication field*

Key Words: *Mobile Communication, Convolution Interleaver*

1. INTRODUCTION

Mobile is used to describe any radio terminal that could be moved during operation. Mobile communication supports the simple, half duplex and full duplex communication modes.

In basic cellular system, which consists of mobile stations, base stations And a Mobile Switching Center MSC or mobile telephone switching office? In this cellular communication high capacity is achieved by limiting the coverage of each base station transmitter to a small geographical area called. A switching technique called hand off enables a call to proceed uninterrupted when the user moves from one cell to another.

A standard Common Air Interface (CAI) defines communication between the base station and the mobiles. That specifies four different channels. The channel is used for voice transmission from the base station to mobiles are called FVC (Forward Voice Channels). The channel is used for voice transmission from the mobile to base stations are called RVC (Reverse Voice Channel). The two channels responsible for initiating mobile calls are the FCC (Forward Control Channel) and RCC (Reverse Control Channel) Those two channels are called as setup channels.

The base station serves as a bridge between all mobile users in the cell and connects the simultaneous mobile calls or microwave links to the MSC. The MSC co-ordinates the activities of the base stations and connects the entire cellular system to the PSTN. A Typical MSC handles 100.00 cellular subscribers and 5000 simultaneous conversations antenna time Accommodates all billing system maintenance functions.

1.1. Mobile Handset Unit

1.1.1. Transmitter

The transmitter circuit consists of phase modulator mixer class c amplifier, final amplifier, pre emphasis circuit, dc amplifier, audio amplifiers, power output detector, speech coder, channel coder, interleaver, modulator, and Automatic Power Control circuit (APC). A special control signal (which is picked up by the receiver) is given to APC that sets the transmitter to one of eight power output levels. APC feature permits optimum cell cite to reception with minimal power.

1.1.2. Receiver

The receiver circuit consists of mixer. RF amplifier de amplifier, audio amplifier audio filters band pass filter, deinterleaver, decoder, and demodulator and received signal strength indicator. RSSI is main part, which is used to send the

received signal to the cell cite antenna again and again. Then MTSO can monitor the received signal from the cell and make decision about switching to another cell. Then the replied it also sends the signal to the transmitter circuit.

1.1.3. Frequency synthesizer

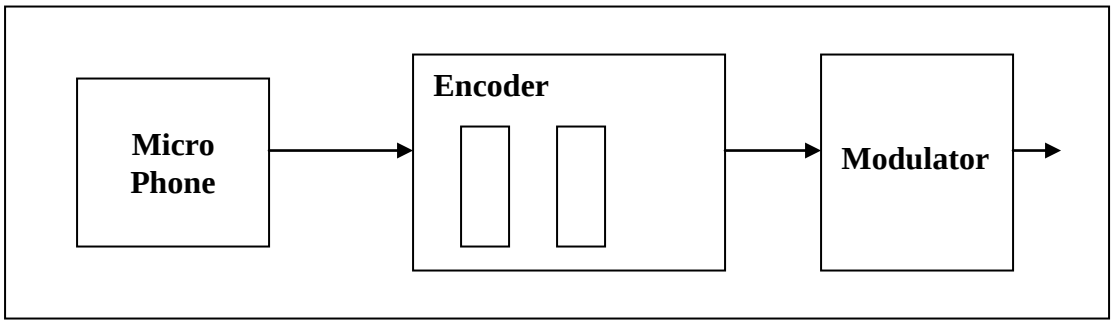
This circuit has standard PLL circuits and mixer. It develops the all the signal used by the transmitter and receiver. Frequency division ratio is supplied by MTSO. Then it makes the transmitting and receiving signal frequency. Signal is given to the transmitter and receiver.

1.1.4. Logic unit

This circuit is made by microprocessor with RAM and RAM plus additional circuitry used for interpreting signal from the MTSO. Then it generates control signal to the transmitter and receiver. PROM or NAM (Number Assignment Module) chip allows the radio signal to identity itself when a call is initiated or when the radio signal is interrogated by the MTSO.

1.1.5. Control unit

It contains handset with speakers and microphone. The main control unit contains a complete tough-tone dialing unit. It is operated by separate microprocessor that drivers the LCD and other indicators. It provides to store the often-called numbers and an auto dial feature



2. Noises and their types

2.1. Noise

Noise is defined, in an electrical sense, as any unwanted form of energy tending to interfere with the proper and easy reception and reproduction of wanted signals.

The noises are classified in to two types.

- Internal noise
- External noise

2.1.1. Internal Noise

If the noise is created by any of the active or passive devices found I receivers are called internal noise.

2.1.2 External noise

The various forms of noise created outside the receiver are called External noise. This is occurred in channels of communications systems. Atmospheric conditions and Extraterrestrial may produce the external noise.

2.2 Noise Figure

The ratio of SNR at the input to that of SNR at its output in order to measure the noisiness of the circuit. This is known as noise figure or noise factor.

2.3. Noises in Mobile Communications

In wireless communication systems, the mobile radio channel is particularly dynamic due to multi path propagation and Doppler spread. These effects have a strong negative impact on the bit error rate of any modulation technique. Mobile radio channel impairments cause the signal at the receiver to distort of fade significantly as compared generally. The following types of noises are occurred in mobile communication channels

- Adjacent channel interference (ACI)
- Co channel interference (CCI)
- Inter symbol interference 9ISI)
- Burst noise (due to fading)

2.3.1 Adjacent channel interference

Interference resulting from signals, which are adjacent in frequency to desired signal, is called adjacent channel interference.

This is minimized through careful filtering and channel assignments.

2.3.2 Co channel Interference

The interference between the signals from these co channel cells such are uses the same set of frequencies in a given area. This is limited by proper frequency reuse technique.

2.3.3 Inter symbol interference

Interference occurs due to modulation pulses are spread in time into adjacent symbols. Using equalizers eliminates this interference.

2.3.4. Burst noise

This noise are occurred due to the effect of Doppler spread and multi path propagation and also external atmospheric conditions (lightning, thunders and etc.,) This is limited by using convolutional interleaving technique.

3. Codings and other Techniques

There are two main techniques are used to prevent the noise completely. Such techniques are Equalization.

- Diversity
- Channel coding

Those are used independently or in tandem to improve received signal quality and link performance over small-scale times and distances. Diversity is another technique used to compensate for fading channel impairments. The various method of diversity is used in wireless field such as Space diversity, antenna polarization diversity, frequency diversity and time diversity

These two techniques are used to improve the quality of a wireless communication link without altering common air interface and without increasing the transmitted power or bandwidth.

3.1 Channel coding and their techniques

Channel coding protects digital data errors by selectively introducing redundancies in the transmitted data [1]. Channel codes that are used to detect errors are called error detection codes while codes that can detect and correct errors are called error correction codes. There are three basic types of error correction and detection codes.

- BLOCK CODES
- CONVOLUTIONAL CODES
- TURBO CODES

Other Techniques.

TRELLIS CODED MODULATION

3.1.1 Block codes

Block codes are forward error correction codes (FEC) that enable a limited number of errors to be detected and corrected without retransmission. The ability of a block code to correct errors is a function of the code distance.

3.2 Turbo codes

Berrou, Glavieux and Thitimajshima first introduced Turbo codes in 1993. This is described that achieves a bit error probability of 10-5 using a rate ½ codes over an AWGN channel. Concatenated coding scheme (by Forney) is a method for achieving larger coding gains by combining two or more relatively simple building block or component codes (constituent codes) on different interleaves versions of the same information sequences. The resulting codes had the error correction of much larger codes and they were encoded with structure that permitted relatively easy to moderately complex decoding.

Turbo codes combine the capabilities of convolutional codes with channel estimation theory and can be though of as nested or parallel convolutional codes. A turbo code can be though of as a refinement of the concatenated encoding structure plus an iterative algorithm for decoding the associated code sequence.

3.3. (Trellis Coded Modulation)

Modulations and coding are must be treated jointly. Coding gain as large as 6db can be achieved without sacrificing data rate or without increasing either bandwidth or power. TCM schemes employ redundant non-binary modulation. In conventional systems involving modulation and coding, it is common to separately describe and implement the detector and the decoder. But in the TCM Combinations with a finite state encoder, which decides the selection of modulation signals to generate coded signal sequences. TCM uses signals set expansion to provide redundancy for coding and to design coding and signal mapping functions jointly, so as to maximize the free distance (Euclidean distance) between the coded signals in receiver section. Then the signals are decoded by a soft decision maximum likelihood sequence

4. Convolutional Encoder

4.1 Convolutional codes

A convolutional code is described by three integers n, k and K. The integer n is the total number of bits in the associated codeword out of the encoder. The integer k is the number by the convolution encoding procedure is not r of data bits that from an input to a block encoder. K=n/k, it is called as code rate. It is integer parameter known as the constrain length. It represents the number of k tuple stages in the encoding shift registers. An important characteristic of the convolution codes, different from block codes, is that the encoder has memory the n tuple emitted only a function of an input k tuple, but is also a function of the previous k – 1 input k tuples.

4.2 General Convolutional Encoder

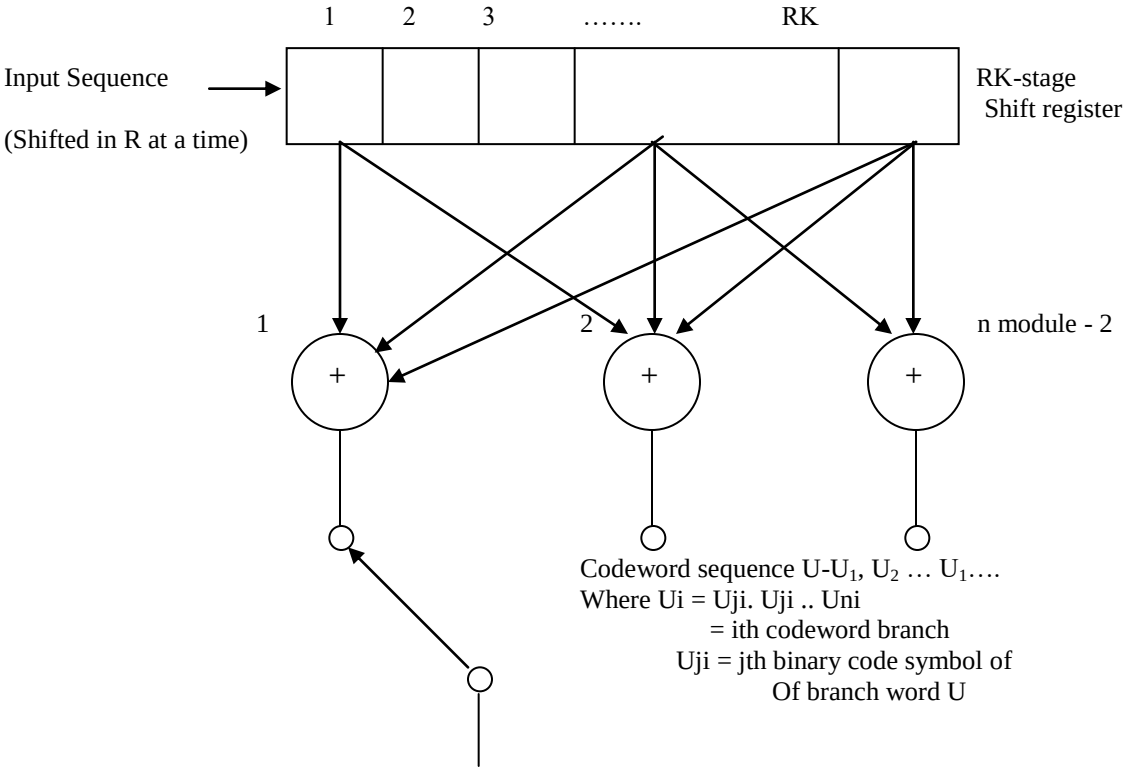
General convolutional encoder is mechanized with a kK stage shift registers and n modulo – 2 adders, where K is the constrain length. The constraint length represents the number of k bit shifts over which a single information bit can influence the encoder output. At each unit of time, k bits are shifted into the first k stages of the register, all bits in the reshifter are shifted k stages to the right, and the outputs of the n adders are sequentially sampled to yield the binary code symbols or code bits. The modulator to specify the waveforms to be transmitted over the channel then uses these code

symbols. Since there are n code bits for each input group of message bits, the code rate is k/n message bit per code bit, where k<n.

We shall consider only the most commonly used binary convolutions encoders which k=1- that is, those encoders in which the message bits are shifted into the encoder one bit at time, although generalization to higher order alphabets is straightforward (1,2). For the K = 1 encoder, at the i^{th} unit of time, message bit mi is shifter into the first shift register stage; all previous bits in the register are shifted one stage to right, and as in the more general case, the outputs of n adders are sequentially sampled and transmitted. Since there are n code bits for each message bit, the code rate is 1/n. The n code symbols occurring at time t, comprise the i^{th} branch word,

Ui = Uil, Ui2, Ui3,……Uin, Where U_{ij} (I = 1, 2,3,……n) is the j^{th} code belonging to the i^{th} branch word. Note that for the rate 1/n encoder, the kK stage shift register can be referred to simply as a K – stage register, and the constraint length K, which was expressed in units pf k – tuple stages, can be referred to as constraint length in units of bits.

Viterbi FEC core consists of a convolution encoder and a Viterbi decoder. It supports error correction for both a burst and a continuous input data stream. The constraint length of the convolution code is 7. The convolution encoder and Viterbi decoder supports all the selectable code rates of 1/3, ½, 2/1, ¾ , 5/6 or 7/8 using industry standard Puncturing Algorithms.



4.3. Convolutional Encoder Parameters

K Constraint Length Number of input bits over which convolutional code is completed 7 Codif Convolutional Code Rate (Ratio of output to input bits for the convolutional encoder). Assigning a value to the parameter codif can choose it. See table 3. 1/2 , 1/3. Rate Puncture Rate Ratio of output to input bits for the convolutional encoder using the puncturing process.

Convolutional Encoder Parameters

Table: 1

Term	Name	Definition	Range
K	Constraint Length	Number of input bits over which convolutional code is computed	7
Codif	Convolutional Code Rate	Ration of output to input bits for the convolutional encoder. Assigning a value to the parameter codif can choose it.	½, 1/3
Rate	Puncture Rate	Ratio of output to input bits for the convolutional encoder using the puncturing process. The puncturing code rates are derived from the ½ mother rate.	½, 1/3, 2/3, ¾, 5/6, and 7/8
Widout	Output data Width	The output data width is configurable to avoid using unnecessary glue logic between blocks depending on the applications.	Configurable depending on the rates

4.4 Encoder + Puncturrer block

The convolutional encoder has a constraint length of 7 and therefore takes form of a shift register with 6 elements. One bit can be input every clock cycle. The output bits are defined as a combination of the shift register elements, according to the standard generator code, and are concatenated to form the encoded output sequence. There is a option for a serial and a byte wide parallel data interface at the input. The output width is programmable depending on the punctured code rate of the application in order to avoid unnecessary logic in between blocks

Table 2: Convolutional Encoder + Puncturrer Block

Signal Name	I/O	Description
Clk	In	Master clock used to clock all register in the encoder design
Reset	In	Asynchronous reset
Nelr	In	Synchronous reset signal supplied to all DFF s in the encoder
EnCeDtin	In	Input data enable indication valid data in the data bus
CeDtin	In	Input data
EnCeDtout	Out	Output data enable indication valid data at the output of the convolutional encoder + puncture block
CeDtout	Out	Output data

5. Interleaving Technique

Here I have considered codes that are designed to constant random independent errors. A channel that has memory is one that exhibits mutually dependent signal transmission impairments. A channel that exhibits multipath fading where signals arrive at the receiver over two or mote paths of different lengths is an example of a channel with memory. The effect is that the signals can arrive out of phase with each other, and the cumulative received signal is distorted. Wireless mobile communication channels, as well as ionospheric and tropopherics propagation channels, suffer from such phenomena. Also, some channels suffer from switching noise and other burst noise (e.g., telephone channels or channels disturbed by pulse jamming). All of these time-correlated impairments result in statistical dependence among successive symbol transmissions that is, the disturbance tend to cause errors that occur in burst instead of as isolated events.

A1	A2	A3	A4	A5	A6	A7	B1	B2	B3	B4	B5	B6	B7	C1	C2	C3	C4	C5	C6	C7

Under the assumption that the channel has memory, the errors no longer can be characterized as single randomly distributed bit errors whose occurrence is independent from bit to bit. Most block or convolutions codes as are designed to combat random independent errors. The result of a channel having decided to take the plunge, the next thing to would be to prepare a memory on such coded signals is to cause degradation in error performance. Coding techniques for channels with memory have been proposed, but the greatest problem with such coding is the difficulty in obtaining accurate models of the often time-varying of such channels. One technique, which only requires knowledge of the duration or span of the channel memory, not it's exact statistical characterization, is the use of time diversity or interleaving.

Interleaving the coded message before transmission and de-interleaving after reception causes bursts of channel errors to be spread out in time and thus to be handled by the decoder as if they were random errors. Since, in all practical causes, the channel memory decreases with time separation, the idea behind interleaving is to separate the codeword symbols in time. The intervening times are similarly filled by the symbols of other code words. Separating the symbols in time effectively transforms a channel with memory to a memory less one, and thereby enables the random-error-correcting codes to be useful in a burst-noise channel.

The interleaver shuffles the code symbols over a span of several block lengths (for block codes) or several constraint lengths (for convolutional codes). The span required is determined by the burst duration. The details of the bit redistribution pattern must be known to the receiver in order for the symbol stream to be de interleaved before being decoded. Let us assume that the code has single error-correcting capability within each seven-symbol sequence. If the memory span of the channel is one codeword in duration, such a seven-symbol-time noise burst could destroy the information in one or two code words, however, suppose that, after having decided to take the plunge, the next thing to would be to proper a encoded the data, the code symbols were then interleaved or shuffled. That is each code symbol of each codeword is separated from its pre interleaved neighbors by a span of seven symbol times. The Interleaved stream is then used to modulate a waveform that is transmitted over the channel. A contiguous channel noise burst occupying seven symbol times, to affect one code symbol from each of the original seven code words. Upon reception, the stream is first de interleaved so that it resembles the original coded sequence. Then the stream is decoded since each codeword possesses a single-error-correcting capability, the burst noise has no degrading effect on the final sequence. Interleaving techniques have proven useful for all the block and convolutional codes described here and in earlier chapters. Tow types of interleavers are commonly used, block interleavers and convolutional interleavers.

5.1 Convolutional Interleaving

Convolutional interleavers have proposed by Ramsey and Fomey. The code symbols are a sequentially shifted into the bank of N registers, each successive register provided j symbols more storage than did the preceding one. The zeroth register provides no storage (the symbol is transmitted immediately). With each new code symbol the commutator switches to a new register is, and the new code symbol is shifted in while the oldest code symbol in then register is shifted out to the modulator transmitter. After the N-1) th register, the commutator returns to the zeroth register and starts again. The deinterleaver performs the inverse operation and the input and output commutators for both interleaving and deinterleaving must by synchronize.

The performance of a convolutional interleaver is very similar to that of a block interleaver. The important advantage of convolutional over block interleaving is that with convolutional interleaving the end-to-end delay is M (N-1) symbols, where M = NJ, and the memory required is M(N-1) at both ends of the channel. Therefore, there is a reduction of one-half in delay and memory over the block interleaving requirements.

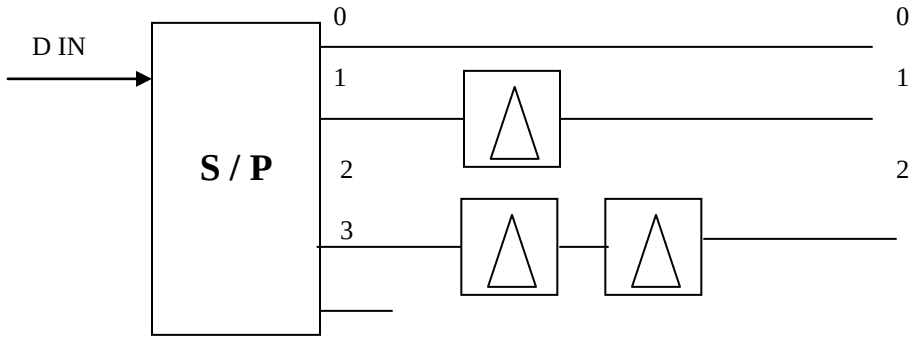
5.2 Designing Convolutional Interleavers

The convolutional interleaver technique is used is telecommunication applications such as SDH and PDH radio systems, GSM and UMTS mobile communication systems, and point to multipoint radio systems to protect transmission channels from noise. On the transmit side, the convolutional interleaver parallelises serial input data into N-bit words and shifts the data word through N delay lines. The delayed data is then shifter out through a PISO shift register for transmission. At the receiver, the incoming data stream is reconstructed with dual delay lines and shift register. Data transmission systems must be protected from stochastic noise events on communication channels. Data scramblers (synchronous and asynchronous), check codes, error correction codes (Read-Solomon, Viterbi, BCH), and interleavers (block or convolutional) are popular techniques for protecting data from noise. Interleavers are a popular choice for controlling impulse noise, which is characterized by high-power, short-duration burses. Convolutional interleavers are often used in conjunction with Viterbi or Reed-Solomon codecs, since the lead is dramatically reduced after the deinterleaver stage.

5.3. Convolutional Interleaver

The heart of a convolutional interleaver is a set of progressive delay lines. The delay lines constitute two parts, one interleaver in the transmitter (TX) and one de-interleaver in the receiver (RX). Consider a data stream, Data – IN, feeding into the transmitter (TX). At a certain rate, this flow will be structured by a serial-in parallel – out (SIPO) register in a B-bit-wide bus (e.g., B=32). The parallel flow then enters a progressive delay line where M is the delay unit. The first signal is not delayed, the second is delayed by M, 2M delay the third, and so on until the B^{th} signal is delayed by (B-1) M. All B-bit-wide signals are reserialized by a parallel –in, serial – out (PISO) register, and the serial output data flow, D- OUT, is ready for transmission.

On the receiving end, the receiver circuits are synchronized with the transmitter to reconstruct the data stream. In TX and RX circuits, a counter provides the correct timing and synchronization of events. Convolutional interleavers are conceptually similar to block intereleavers, although they are more complex to implement. Convolutional interleavers are area efficient, using only half the density needed by block interleavers are area efficient, using only half the density needed by block interleavers. In addition, the synchronization circuitry is simplified at the receiver, because the phases are B instead of B x N, where B is the number of block interleaver rows and N is the number of block interleavers columns



5.4 System Synchronization

In a radio system, a convolutional de-interleaver is generally put before a Viterbi decoder whose output provides the necessary synchronization. Consider a certain period of time, when the error rate is observed after the decoder processing. If in this period the error rate falls to zero, then synchronization is achieved[2]. On the other hand, if this rate is not zero, a stop of one clock to the RX synchronization counter must be provided. At most, after B steps, the synchronization phase is caught up.

5.5 System Architecture

In this example, the TX of the interleaver circuit has B=32 phases, M=1 delay units, and contains three elements one 32-tap SIPO, one 32 – bit progressive delay line from 0 to 31 delay elements), and one 32-tap PISO to stream back serially formatted data for transmission. One 5-bit counter (B = 32) provides the correct delay adjustments, synchronization the SIPO at the input of the delay line with acquisition by the PISO at the output of the delay line. An equation for the number (ND) of the delay elements is then calculated as ND = B (B-1)/2 = 496

The de-interleaver receiver is similar to the interleaver architecture with only the delay elements reversed (Figure 2). The receiver needs an external synchronization signal to enable the 5-bit counter (B=32) to provide the correct circuit delay (similar to the transmitter). If this signal is not available, it is possible to use the output of the Viterbi decoder, as mentioned in the previous paragraph. This circuitry is useful for a one – transmission data channel. With an 8-bit digital transmission, there are eight serial channels where an interleaver/ de-interleaver protection system must be implemented.

5.6. Progressive Delay Line

Traditionally in FPGAs, the unit delay elements M = 1 (Figure 2) of the progressive delay line are implemented with flip-flops, making it a very register-intensive application. The additional resources needed to implement a real transmission system (for example, an 8-bit TX/RX system, 496 x 8x 2 = 7936 flip-flops) traditionally favored ASIC solutions, but produced significant performance degradation.

The Viterx architecture redefines the suitability of FPGAs in transmission systems by implementing the delay elements in an exclusive SRL16 feature. The built-in programmable shift register enables one LUT to hold sixteen delay elements. In a system where B = 32, one LUT can implement from 1 to 16 delay lines. From 17 to 31 delay lines can be implemented with two LUTs per line, saving logic resources and maximizing overall performance. The only limiting factor is the physical LUT access time. In an 8-bit TX/RX system, for instance, the logic resources used are calculated
(16 + (2x15)) x 8x 2 = 736 LUTs

6. Convolutional Decoder

6.1 The Viterbi decoding convolutional decoding Algorithm

The Viterbi decoding algorithm was discovered and analyzed by Viterbi in 1967. The Viterbi algorithm essentially performs maximum likelihood decoding; however, it reduces the computational load by taking advantage of the special structure in code trellis. The advantage of Viterbi decoding, compared with the brute-force decoding, is that the complexity of a Viterbi involves is not a function of the number of symbols in the codeword sequence. The algorithm involves calculating a measure of similarity, or distance, between the received signals, at time it, and all the trellis paths entering each state at time it. The Viterbi algorithm removes from consideration those trellis paths that could not possibly are candidates for the maximum likelihood choice.

When two paths enter the same state, the one having decided to take the plunge, the next thing to would bee to prepare a the best metric is chosen; this path is called the surviving path. This selection of surviving paths is performed for all the states. The decoder continues in this way to advance deeper into the trellis, making decisions by eliminating the lest likely paths reduces the decoding complexity. In 1969, Omura demonstrated that the Viterbi algorithm is in fact, maximum likelihood. Note that the goal of selecting the optimum path can be expressed, equivalently, as choosing the codeword with the maximum likelihood metric, or as choosing the codeword with the minimum distance metric.

6.2 Viterbi Decoder Parameters

Table 3 : Viterbi Decoder Parameters

Term	Name	Definition	Range
Codif	Convolutional Code Rate	Ration of output to input for the convolutional encoder. Assigning a value to the parameter codif can choose it. See table 2.	1/2 , 1/3.
Rate	Puncture Rate	Ratio of output to input bits for the convolutional encoder using the puncturing process. The puncturing code rates are derived form the ½ mother rate. See table 2.	½, 1/3, 2/3, ¾. 5/6 and 7/8
Bitin	Input Bits	Number of processing bits for the decoder. See table 5.	Configurable depending on the application.
Hosd	Hard of Soft Decision	This parameter indicates if the Viterbi decoder is working in hard of soft decision and is related to the former parameter. See details inn table 5.	0, 1
Lmetr	Matrices Register Length	The width of the registers storing the metrices plays an important role in the decoding processing.	Configurable depending on the application
Trb	Trace back Size	Length of the path needed by the Viterbi decoder to compute the most likely decoded bit value.	Minimum = 6 (64)
Memsize	Memory Size	Size of the memory storing the path metrics information for the decoding processing	Minimum = 7 (128)
Lfif	FIFO size	Size of the FIFO between the depuncture and Viterbi blocks	Minimum =2(4)
Widoubt	Viterbi output Width	Width of input data to Viterbi decoder	Configurable

Table 4 : Code Rate Values		
Codif	**Rate**	**Value**
0	0	½
0	1	2/3
0	2	¾
0	3	5/6
0	4	7/8
1	5	1/3

Table 5: Output Data Width Values

Rate	Widout	Widin
½	2n integer	2n integer x bitin
2/3	3n	3nxbitin
¾	4n	4nxbitin
5/6	6n	6nxbitin
7/8	8n	8nxbitin
1/3	3n	3nxbitin

Table 6: Input bits in hard and soft decision

Bitin	Hosd
1(hard)	1
>3(soft)	0

6.3 Encoder + Puncturer block

The convolutional encoder has a constraint length of 7 and therefore takes form of a shift register with 6 elements. One bit can be input every clock cycle. The output bits are defined as a combination of the shift register elements, according to the standard generator code, and are concatenated to form the encoded output sequence. There is the option of a serial and a byte wide parallel data interface at the input[3]. The output width is programmable depending on the punctured code rate of the application in order to avoid unnecessary glue logic between blocks.

The Viterbi block can decode data form the channel using either hard or soft decision decoding. The Viterbi decoder is based on ½ and 1/3 rate code and supports punctured code rates such as 2/3, 3/4, 5/6 and 7/8.

For the ½ rate code the polynomials used are g0 = 171o g1 = 133

And for the 1/3 rate code

g0= 171o g1=165o g2=133o

The minimum trace back depth is 64

6.4 Depuncturer

The depuncturer block replaces the deleted symbols in the puncturing process recovering the original data rate. Following is a list of the puncture patterns.

Table 7: Puncture Patterns

½	X Y	1 1
2/3	X Y	1 0 1 1
¾	X Y	1 0 1 1 1 0
5/6	X Y	1 0 1 0 1 1 1 0 1 0
7/8	X Y	1 0 0 0 1 0 1 1 1 1 1 0 1 0

At the encoder, after the encoding process some symbols are deleted according to the respective puncturing pattern. The decoding process replaces the deleted data, Introducing dummy symbols that are chosen in order to increase the distance the correct and wrong trellis paths.

6.5 Viterbi decoder

The Viterbi decoder divides the input data stream into blocks, estimating the most likely data sequence and outputting each decoded data sequence in a burst. The input and the calculations are continuous and require 4 clocks for every 2 bits of data (3 bits when working with 1/3 code).

The size of the input FIFO Ifif is dependent on the depuncturer input data rate. The average maximum bit rate at the output of the depuncturer is 1 bit (n bits in soft decision) per 2 clock cycles. If the input data for the Viterbi decoder occurs in high-speed data bursts, then the size of the FIFO must be enlarged. The Viterbi algorithm performs the trace operation in parallel with the path calculations. The width of the metric registers must be enlarged when using high rate punctured codes, when the number of bits used in soft decision is large and when the Bit Error Rate at the input of the Viterbi decoder is high. The trace depth should be sufficiently long to avoid loss of accuracy and can set with the programmable parameter trb.

6.6 Data interface

The width of the input data may be varied by setting the parameter widen to the vavlues shown in table 4. The Viterbi block extracts a number of bits (depending on the hosd, codif and bit n parameters) to calculate the path meter ices though the Trellis.

6.7 Interface Description

The interfaces to the convolutional encoder + puncturer block and Viterbi decoder + depuncture block are given below.

Table: 8 Convolutional Encoder + Puncturer Block

Signal Name	I/O	Description
Clk	In	Master clock used to clock all registers in the encoder design
Reset	In	Asynchronous reset
Nclr	In	Synchronous reset signal supplied to all DFFs in the encoder
EnCeDtin	In	Input data enable indicating valid data in the data bus.
CceDtin	In	Input data
EnCeDout	Out	Output data enable indicating valid data at the output of the convolutional encoder + puncture block
CeDout	Out	Output data.

Table: 9 Viterbi decoder + Depuncturer block

Signal Name	I/O	Description
Clk	In	Master clock used to clock all registers in the encoder design
Reset	In	Asynchronous reset
Nclr	In	Synchronous reset signal supplied to all DFFs in the encoder
EnVeDtin	In	Input data enable indicating valid data in the data bus.
ViDtin	In	Input data to the Viterbi decoder + depuncture cblock.
EnViDtout	Out	Output data enable indicating valid data at the output of the convolutional encoder + puncture block
ViDtout	Out	Output data from the Viterbi decoder.

6.8 Testing

A test bench is included which instantiates a random generates to create binary signals as an input to the convolutional encoder. The output of the encoder is provided to the puncturing block, then a memory delay, the Viterbi decoder and finally the depuncturing block. The encoded burst is then stored in a memory in order to introduce a delay and achieve the correct average input data rate of 2 data bits every 4 cycles. The first output burst of the estimated data sequence is output from the decoder circa (trb* input data average rate) clock cycles after the first burst is input.

Thereafter bursts of data are output while there is input data to process. There is the option of testing the pausing of the input to the encoder or the decoder. If the input of the decoder is paused, it will that all the path metric calculations, which are resumed when there is enough received input data to continue processing.

6.9 Coding Gain

Coding gain is defined as the reduction, usually expressed in decibels, in the required Eb/No to achieve a specified error probability of the coded system over an uncoded system with the same modulation and channel characteristics.

The coding gain is given as formula

G(Db) = (Eb/No) u (dB)-(Eb?No)c(dB)

7. XILINX PROCESSORS

7.1 XiLINX 3000 SERIES FPGAS

As an example of a FPGA we will describe thnne Xilinx XC3020 Logic Cell Array (LC.A. It is a part of the basic structure, which consists of an interior array of 64 configurable logic blocks (CLBs)[5] surrounded by a ring of 64 input interaface blocks. The interconnections between these blocks can be programmed by storing data in internal configuration memory cells. Each configurable logic block contains some combinational logic and two D flip-flops and cells are programmed to perform a variety of logic functions.

The configuration memory cells are programmed after power has been applied to the LCA and the programmed logic functions and interconnections are retained until the power is turned off. During configuration each memory cells. Is selected in turn. When a WRITE signal is applied to the pass transistor. DATA is stored in cell. Each connection point in the LCA has an associated memory cell and data stored in that cell determines whether the connection is mad or not.

The block has five inputs (A.B.C.D.E.) a data input (DI) a clock input (K) a clock enable (EC) a direct reset (RD) and to outputs (X and Y). The trapezoidal blocks on the diagram represent multiplexes, which can be programmed to select one of the inputs. For example the X output can some from the up[per flip-flop (OX) or from the output of the Combinatiional Function block. Similarly the Y output can come either from the lower flip (OY) or from G. Each (M) represents a configuration memory cell and the data in the cell determines which MUX input is selected.

The Combinatorial Function block contains RAM memory cell and can be programmed to realize any Function of four variables. The functions are used stored in truth table from, so the number of gates required to realize the block. As before each trapezoidal represents a multiplier, which can be programmed to select one of its inputs. The FG mode generates two functions for four variables (A) must be common to both functions. The next two variables can be chosen from B,C, OX and OY. The remaining variable can be either D or E. For example we could generate F+AB+OX E and G=AC+D. If OX OY are not used then the two four variable functions must have A,B and C in common and the fourth variable can be D or E.

The F mode can generate one function of five variables (A, D, E and two variable chosen from B, C, OX, and OY). We can realize function ranging in complexity from a simple AND gate.

F=G=ABCDE

To a parity function,

F=G=A B C D E

Which has 16 term when expanded to a sum of products. The FGM mode uses a multiplexer with E as a control input to select one of two four – variable functions. Each function uses inputs A, D, and two of the inputs B, C, QX and QY.

7.2 Programmable Interconnects

The programmable interconnections between the configurable logic blocks and I/O blocks can be made in several ways-general purpose interconnects, direct interconnects, and long lines. The general purpose interconnects system. Signals between CLBs or between CLBs and IOBs can be routed through switch matrices as they travel along the horizontal and vertical interconnect lines. Direct interconnection of adjacent CLBs is possible. Long lines are provided to connect CLBs that are far apart. All the interconnections are programmed by storing bits in internal configuration memory cells within the LCA.

Long lines provide for high fan-out,-skew distribution of signals that must travel a relatively long distance. There are four vertical long lines between each pair of adjacent columns of CLBs, and two of these can be used for clocks.

There are two horizontal lines between each pair of adjacent rows of CLBs. The long lines span entire length or width of the interconnection.

7.3. XILINX 4000 SERIES FPGAS

The Xilinx 4000 series FPGAs are similar to the 3000 series but more inputs and outputs and many other features have been added. Shows a simplified block diagram of an XC 4000 configurable logic block. It has nine input (F1, F2,F3, F4,G1,G2,G3.G4, and H1). It can generate two independent functions of four variables.

G(G1,G2,G3,G4) and F(F1,F2,F3,F4)

This is contract to the 300 series where the inputs to the 4-variable function generators must overlap. The 4000 series CLB can also generate a function H, which depends on F.G, and H). by setting F1=G1, F2=G2, F3=G3, and F4=G4. it can generate any function of five variables in the form H=F(F1,F2,F3,F4) H1+G(F1,F2,F3,F4)H1. It can also generate some function of 6.7.8,and 9 variables outputs. Two from the flip-flops and two from the combinational logic function generators. Unlike the 300 CLB. The 4000 CLB has to internal feedback. SO flip – flop outputs must be routed externally back to the logic inputs when feedback is required. The CLB has an S/R routed externally back to the independently configured to connect to the SD or RD input on each flip-flop. Thus one flip-flop could be set to connect to the 0 using the same rising edge

or falling edge of the K (clock) input. The EC input on each flip-flop either be always enables ot it can be enabled by the EC input to the CLB. The D input to each flip-flop can be connected to DIN F, G, or H.

The 4000 logic cell contains dedicated carry logic. Each cell contains carry logic for two bits, and the F and function generators can be used to generate the sum bits. Thus each cell can be programmed to implemented both the carry and um for two bits of a full adder. The carry lines between cells are hardwired (not programmable) too provide for fast propagation of carries. The can propagate up or down between cells and a programmable multiplexed (labeled M1 in Figure 6-12) select the direction of propagation. Multiplexers M2, and M4 allow some of the F and G function generator input to come from carries instead of from the normal F and G cell inputs. In addition to address the carry logic and programmed to implemented subtracts. Adder / subtracts, increments / decrements, 2's complementers and counters.

7.4 Alters complex Programmable Logic Devices (CPLADS)

CPLDs are an extension of pal concept. In general a CPLD is an IC that consists of a number of pal-like logic blocks together with a programmable inter correct matrix. Each PAL-like logic block has a programmable NAD array that feeds macro cells. And the outputs of these macro cells can be routed to the inputs of logic blocks within the same IC many CPLDs are electrically erasable and reprogram able and as

Such as sometimes referred to as EPLDs (Erasable PLDs).

The Altera MAX 7000 series is a family of high performance COMS CPLDs. In contrast to the Xilinx FPGAs the Altera 7000 series uses EEPROM-based configurations memory cells, so that once the configuration is programmed, it is retained until it is erased. The basic 7000 series architecture which consists of number of Logic Array Blocked (LABs). I/O Control Blocks and a Programmable interconnect Matrix (PIX). Each LAB contains 16 macrocells, each of which contains combinational logic and a flip-flop. Each LAB has 36 inputs from the PIA and 16 outputs to the PIA. From 8 to 16 outputs from the I/Q pins can be routed through the I/Q control block to the PIA. The global clock input (GCLLK) and global clear input (GCLRn) connect to all macro cells. Two output enable signals (OE1n and OE2n) connect to all I/O control blocks.

Each macro cells includes a logic array, a product-term select matrix that feeds an OR gate, and a programmable flip-flop. The vertical lines in the logic array. Which are common to all of the macro cells in a Lab are driven with the programmable interconnect signals from the PLA and from shared logic expanders. Product terms are framed in the logic just as they are in a PAL. Five product terms are provided in each macro cell, and the product term matrix allocates these product terms. A product term may be used as an OR gate input, an XOR gate input, a logic expander, or as a flip-flop preset, clear, clock, or enable input.

7.5 SIPOs, PISOs, AND Further Area Reduction

Generally, SIPOs and PISo are implemented with flip-flop. Another technique, depending on logic resources and memory priorities, is to use block RAM.

The available block RAM, primitives are shown in Table 10. With the true dual-port capability of Virtex block RAM, A and B are used effectively as two independent ports, without any additional logic, thus allowing data flow width conversions. For instance, a 16-tap SIPO/PISO is a conversion 1 16 or 16 1.

Table 10: Available Primitives

Primitive	Port A Width	Port B Width
RAMB4-S1-S2	1	1
RAMB4-S1_S2	1	2
RAMB4-S1_S4	1	4
RAMB4-S1_S8	1	8
RAMB4-S1_S16	1	16

7.6 Traditional (Flip-Flop) Vs. Virtex Exclusive (SRL16) Solutions

Table 10 shows the efficiency of implementing this application in the Virtex architecture by comparing an eight-channel interleaver/ de-interleaver system. The interleaver and deinterleaver are implementer with B=32 delay stages. This design was implemented for three Xiliinx families XC5210 (flip-flops), XC406XL (delay elements by Core Gen), and XCV 200 (SRL16s) devices.

Table No. 11: Convolutional Interleaver across Xilinx Families

Device	CLB Array (RxCxLC)	LUTs	Flip-Flop	Used Area	SRL16
XC5210	18x18x4	514	8972	8 FPGAs	0
XC406XL	48x48x2	18114	2040	1090CLBs	0
XCV200	28x42x4	509	1036	1352Slices	736

Additionally, the below table 11 shows the final results of the same interleaver implemented on an XCV400V device, the first with flip-flops and the second with SRL 16s.

Table 12 : SRL16 Vs Flip-Flop Implementation: Resource Usage in the Progressive Delay Line

	CLB Array (RxCxLC)	**LUTs**	**Flip-flop**	**SRL 16**	**Slice Usage**	**Max Frequency (-6)**	**Max Frequency (-8)**
Flip-Flop	40x60x4	509	8972	0	4800 (100%)	102 MHz	131 MHz
SRL16	40x60x4	509	1036	736	1352 (28%)	128 MHz	160 MHz

The Virtex architecture allows the designer to implement a high-performance convolutional interleaver / de-interleaver with a small amount of FPGA resources by utilizing the SRL 16 technique.

The architecture is well suited to implemented easy and efficient convolutional interleavers. With the SRL 16 feature available in this device, very efficient shift registers can be implemented. The flexible shift registers vary in length from one to sixteen bits as determined by the address lines. SRL 16 is used to implement a progressive delay line, thereby saving logic resources and producing the highest performance.

7.7 Synthesizable HDL Code Reference Design

The section of the application note describes a hierarchical synthesizable implementation convolution interleaver/ de-interleaver protection system for an eight-channel TX/RX radio system in Virtex devices with SRL 16s[6]. The following design modules are included

- TOP vhd Top file; example of instantiation of an 8-channel /de-interleaver system.
- MULTI_CH.vhd parametric VHDL for N-bit-wide interleaver /de-interleaver system.
- TX_INTER convolutional interleaver with B=32 delay stages
- RX_INTER convolutional de-interleaver with B=32 delay stages

7.8 Foundation 3.1i ISE Convolutional Interleaver Test Circuit

This archived project can be loaded into the Xilinx Foundations 3.1i ISE front-end tools. The circuit shows an example of how to control a convolution interleaver under ideal conditions. The delay element M is equal to the system clock in this case, giving a relative delay value of 1. This simplifies the synchronization circuitry for this example; in a real communication system, the Viterbi decoder or other control mechanism handles the synchronization. Using this example, the designer has the ability to use the convolutional interleaver on specific designs.

MASTER.sch top-level schematic showing the connectivity between all the components. In simulation the user to supply as serial data vector and an active high GO signal.

t C5c3.sch 5 bit counter

t CONTROLBOX.sch a simple state machine to control the synchronization of the convolutional interleaver. The FSM implements the state transition diagram (Figure 4).

- **SIPO.v** serial-in parallel-out shift register. Provided for the designer to see the output of the receiver as a parallel word instead of as a serial data stream.
- **RX_INTER.vhd** VHDL module describing the 32-delay taps convolutional de-interleaver.
- **TX_INTER.vhd** VHDL module describing the 32-delay tap convolutional interleaver

8. Conclusion

This application note and reference design clearly demonstrates the efficiency of the SRL 16 techniques. This exclusive Virtex architectural feature is the key advantages when designing a convolutional interleaver/ de-interleaver for TX/RX radio systems. Using the SRL 16 feature instead of flip-flop to make variable-delay elements saves resources and maximizes performance. Additionally, the true dual-port capability of Virtex block RAM allows further area reduction by making SPIOs and PISOs using the capability to set widths for port A and port B. This easy fast and efficient solution is feasible only with the Virtex architecture. Other FPGAs utilize traditional approaches.

9. Future Enhancements

In this paper, have made a study with the convolutional interleaving technique for completely eliminate the burst noise in Mobile communication

Advantage

- Interleaving technique provides the noise free data transmission.
- By using interleaver, this paper achieves high speed of data transmission in digital communication.
- Using this technique along with VHDL, the cell phones can be made compact.
- Time diversity is easily obtained by using this technique. So it protects the data from severe fading.

Applications

- This is applied to the compact disc digital audio (CDDA) systems for achieving high fidelity of sound reproduction

This is also used in the satellite communication systems for reduce the cost and size of the payload.

- This techniques is used in GSM, UMTS, and etc.,

10. REFERENCES

[1] Charles H. Roth, Jr "Digital System Design using VHDL".

[2] Bernard Sklar LPE "Digital Communications Fundamentals and Applications".

[3] Kennedy "Electronics Communication Systems" (Second Edition).

[4] Theodore Rappaport "Wireless Communications Principles and practice".

[5] Charles G.Sodini Series editor "VLSI for Wireless Communication".

[6] Lee "Mobile Cellular Tele Communication".

