

<b>1. Introduction .....</b>	
<b>1.1 Aim and objective of the project .....</b>	
<b>2. Code Division Multiple Access .....</b>	
<b>2.1 Introduction to CDMA .....</b>	
<b>2.2 Channels &amp; Sub Channels.....</b>	
<b>2.3 IS 95 CDMA Cellular system .....</b>	
<b>2.4 Block diagram of Reverse CDMA Access Channel .....</b>	
<b>2.5 Requirement and Functioning of different blocks .....</b>	
<b>2.6 Advantages of CDMA Technology.....</b>	
<b>3. Digital Design process.....</b>	
<b>3.1 Process of design .....</b>	
<b>3.2 Simulation .....</b>	
<b>3.3 Synthesis Optimization &amp; Device Fitting .....</b>	
<b>3.4 Study of Various EDA tools.....</b>	
<b>4. Digital Specifications &amp; Proposed Digital Architecture .....</b>	
<b>4.1 Frame Quality Indicator.....</b>	
<b>4.2 Tail Encoder .....</b>	
<b>4.3 Convolutional Encoder .....</b>	
<b>4.4 Block Interleaver.....</b>	
<b>4.5 Orthogonal Modulator.....</b>	
<b>4.6 Long Code Generator.....</b>	
<b>5. Hardware Implementation.....</b>	
<b>6. Conclusion .....</b>	
<b>7. References.....</b>	
<b>8. Appendix .....</b>	
<b>8.1 IS 95 Specifications for Reverse CDMA Access Channel.....</b>	

<b>8.2 VHDL codes.....</b>	<b>53</b>
<b>8.3 Simulation Results.....</b>	<b>73</b>
<b>8.4 Walsh Matrix.....</b>	<b>86</b>
<b>8.5 Important terms.....</b>	<b>87</b>

## **1. Introduction**

### **1.1 Aim and Objective of the Project**

The Project is aimed at designing a digital architecture for “**Reverse CDMA Access Channel**” according to the **IS 95 CDMA** specifications and to simulate and test the circuit using a Hardware Description Language. The HDL used for describing the digital architecture is Very High Speed Integrated Circuit Hardware Description Language .i.e. **VHDL**.

Our Project also involves the use of industry standard **EDA** tools for VHDL Simulation as well as Synthesis. The hardware simulation has been done using tools such as **MODELSIM, ACTIVE HDL** and the device fitter files and the Netlists were generated using tools such as **ALTERA MAXPLUS** and **WARP**.

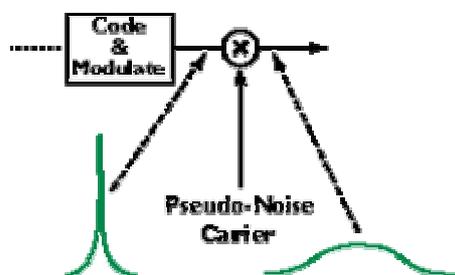
The last step in the implementation of a digital circuit is its hardware realization using Complex Programmable Logic Devices (CPLDs) or Field Programmable Gate Arrays (FPGAs). Many Synthesis tools like Warp provide an easy way of doing this using its device fitters. We have successfully implemented one of the blocks (long code generator) into an **Ultra 37256V** CPLD from Cypress. Rest of the circuit after testing successfully will be kept as a file which includes all the essential details of fitting which can be used for hardware fitting.

## 2. CODE DIVISION MULTIPLE ACCESS

### 2.1 Introduction to CDMA

Code Division Multiple Access (CDMA) is the fastest growing digital wireless technology. It is a form of *spread-spectrum*, a family of digital communication techniques that have been used in military applications for many years. CDMA allows simultaneous digital transmission by multiple devices over the same channel and is most commonly used in mobile communications. CDMA assigns each subscriber a unique code to put multiple users on the same wide band channel at the same time. The codes called pseudo random code sequences are used by both the mobile station & the base station to distinguish between conversations. There is no time division, and all users use the entire carrier, all of the time. The signals are separated at the receiver by using a correlator that accepts only signal energy from the desired channel. Undesired signals contribute only to the noise.

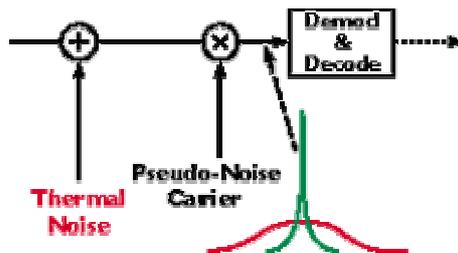
CDMA uses a form of direct sequence. Direct sequence is, in essence, multiplication of a more conventional communication waveform by a pseudonoise (PN)  $\pm 1$  binary sequence in the transmitter.



**Spreading the Data**

In reality spreading takes place prior to any modulation, entirely in the binary domain, and the transmitted signals are carefully bandlimited.

A second multiplication by a replica of the same  $\pm 1$  sequence in the receiver recovers the original signal.



### Despreading the Data

A CDMA call starts with a standard rate of 9600 bits per second (9.6 kilobits per second). This is then spread to a transmitted rate of about 1.23 Mbps. Spreading means that digital codes are applied to the data bits associated with users in a cell. These data bits are transmitted along with the signals of all the other users in that cell. When the signal is received, the codes are removed from the desired signal, separating the users and returning the call to a rate of 9600 bps.

The use of CDMA for civilian mobile radio applications is novel. It was proposed theoretically in the late 1940's, but the practical application in the civilian marketplace did not take place until 40 years later. Commercial applications became possible because of two evolutionary developments. One was the availability of very low cost, high density digital integrated circuits, which reduce the size, weight, and cost of the subscriber

stations to an acceptably low level. The other was the realization that optimal multiple access communication requires that all user stations regulate their *transmitter* powers to the lowest that will achieve adequate signal quality. It is an advanced digital technology that can offer about 7 to 10 times the capacity of analog technologies and up to 6 times the capacity of digital technologies such as Time Division Multiple Access (TDMA).

CDMA changes the nature of the subscriber station from a predominately analog device to a predominately digital device. Old-fashioned radio receivers separate stations or channels by filtering in the frequency domain. CDMA receivers do not eliminate analog processing entirely, but they separate communication channels by means of a pseudo-random modulation that is applied and removed in the digital domain, not on the basis of frequency. Multiple users occupy the same frequency band. This universal frequency reuse is not fortuitous. On the contrary, it is crucial to the very high spectral efficiency that is the hallmark of CDMA

## **2.2 Channels and Subchannels**

CDMA based wireless communication system has basically two different channels. They are the 1) Forward Channel 2) Reverse Channel

### **Forward Channel**

Forward Channel is used for transmission of data as well as control information from the Base Station of the digital cellular network to mobile station or the mobile handset. A frequency spectrum from 869 to 894 MHz is allocated for the forward channel. Forward channel is again divided into 4 subchannels. They are the following

#### **1. Pilot channel**

The pilot channel is used for the transmission of timing information, phase reference information for coherent demodulation etc. from the base station to the mobile station. It also provides a means for signal strength comparisons between base stations for determining when to handoff

#### **2. Synchronization channel**

Synchronization channel is used for broadcasting future state of the long code register from the base station to the mobile station.

#### **3. Paging channel**

Paging channel transmits the call control information from the base station to the mobile station.

#### **4. Traffic channel**

Forward Traffic channel is used for the transmission of Voice data or speech information from the base station to the mobile station.

#### **Reverse Channel**

Reverse Channel is used for transmission of data as well as control information from the mobile station or mobile handset to the Base Station. A frequency band between 824 and 849 MHz is allocated for the reverse channel. The reverse CDMA Channel is made up of access channels and reverse traffic channels. Both share the same frequency assignment, and each Traffic/Access channel is identified by a distinct user long code. Data transmitted on the reverse channel is grouped into 20 ms frames. All data on the reverse channel is convolutionally encoded, block interleaved, and modulated by 64-ary orthogonal modulation, and spread prior to transmission.

#### **Reverse Access channel**

Reverse Access channel is used for the transmission of control information like signaling and call initiation information from the mobile station to the base station. Also reverse access channel is used to respond to paging channel messages. The reverse channel may contain a maximum of 32 access channels per supported paging channel. The access channel works at a fixed data rate of 4800 bps.

### **Reverse Traffic channel**

Reverse Traffic channel is used for the transmission of voice or speech information from the base station to the mobile station. The reverse traffic channel is used to transmit the user information. The reverse traffic channel may use either 9600, 4800, 2400, 1200 bps data rates for transmission.

### **2.3 IS 95 CDMA Cellular System**

The IS-95 CDMA standard was adopted by the TIA (Telecommunications Industry Association) and became a digital cellular standard in 1992. CDMA is the first digital technology which meets the exacting standards of the CTIA (Cellular Telecommunications Industry Association). Depending on the level of mobility of the system, it provides 10 to 20 times the capacity of AMPS, and 4 to 7 times the capacity of TDMA.

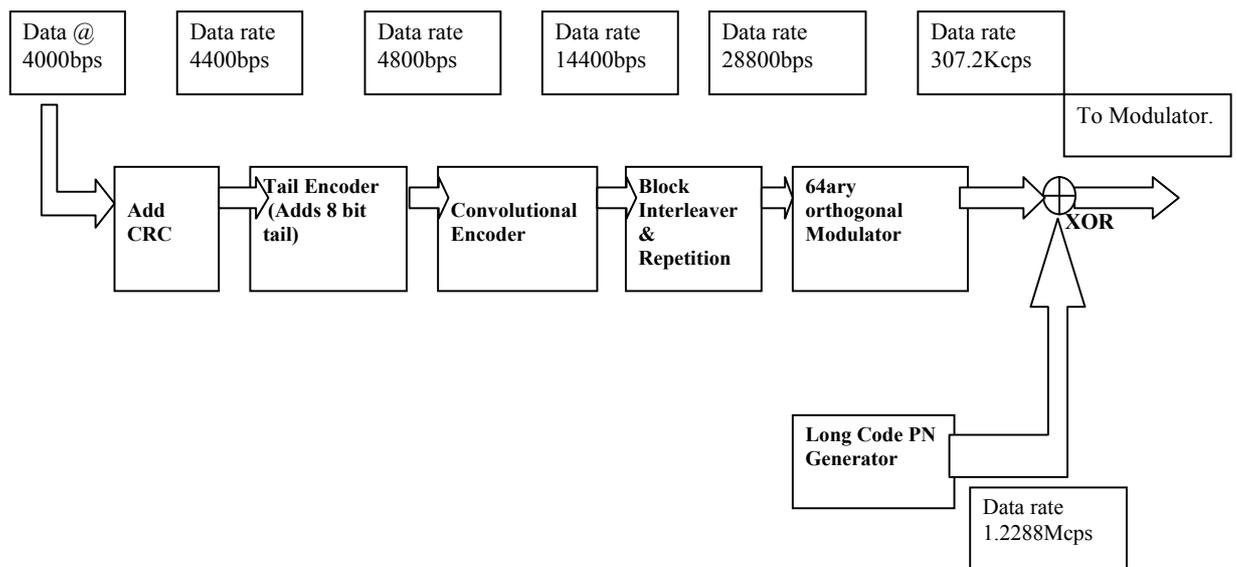
The IS-95 standard describes a Code Division Multiple Access (CDMA) system in which the audio band data signal is added to a high rate spreading signal. This spreading signal is formed from a pseudo-noise code sequence, which is then multiplied by a Walsh code for maximum orthogonality to (i.e. to have low cross-correlation with) the other codes in use in that cell. Typically, CDMA pseudo-noise sequences are very long, thereby giving excellent crosscorrelation characteristics. (IS-95 uses a  $2^{42}-1$  chip period, derived from a 42 bit mask.). The IS-95 system can be thought of as having many layers of protection against interference. It allows many users to co-exist, with minimal mutual interference. They can be

described by the signal conditioning sequence that occurs on forward and reverse channels. The forward channel carries information from the base station to the mobile unit; the reverse channel carries information from the mobile unit to the base station. The forward channels are between 869 and 894 MHz, while the reverse channels are between 824 and 849 MHz. The IS-95 channel occupies 1.25 MHz within this band, the rest of which is occupied by other cellular services such as AMPS (Advanced Mobile Phone System). A forward and reverse channel pair is separated by 45 MHz. The Maximum user data rate is 9.6 kbps. User data in IS-95 is spread to a channel chip rate of 1.2288 Mchip/s (a total spreading factor 128) using a combination technique. All cells in the same area can employ the same spectral band, because the various signals are sorted out by the spread spectrum process rather than by frequency discrimination.

Voice privacy of IS 95 CDMA is provided by means of the long code mask. The long code mask is not transmitted through any channel; it is constructed by the base station and the mobile station.

## 2.4 Block Diagram of Reverse CDMA Access Channel

Reverse Access channel is used for the transmission of control information like signaling and call initiation information from the mobile station to the base station. Also reverse access channel is used to respond to paging channel messages. The reverse channel may contain a maximum of 32 access channels per supported paging channel. The access channel works at a fixed data rate of 4800 bps.



**Reverse CDMA Access Channel – Block Diagram**

Access channels enable the mobile to communicate nontraffic information. The data rate is fixed at 4.8 kbps. These are identified by a distinct access channel long-code sequence offset; a paging channel number is associated with an access channel. Frequency re-use factor is 1. All base stations are on the same frequency, and use the same set of 64 Walsh Codes. Each base station is distinguished from other base stations by the offset of the short Pseudo Noise Codes.

The access channel is used by the mobile to initiate communication with the base station and to respond to paging channel messages. The reverse channel may contain a maximum of 32 access channels per supported paging channel. The access channel works at a fixed data rate of 4800 bps. Data transmitted on the reverse channel is grouped into 20 ms frames. All data on the reverse channel is convolutionally encoded, block interleaved, and modulated by 64-ary orthogonal modulation, and spread prior to transmission. For the Reverse Access channel, the CRC will be calculated on information bits within the frame, except the encoder tail bits. The generator polynomial is  $g(x) = 1 + x + x^3 + x^4 + x^7 + x^8$ . CRC will add 8 bits to the end of each frame. 8 bit Encoder Tail add 8 zero bits to the end of each frame to set the convolutional encoder back to zero state. In reverse channel, (3, 1, 8) Convolutional Encoder is used (R=1/3, K=9). The generator sequences for this encoder are  $g_0 = (101101111)$ ,  $g_1 = (110110011)$ ,  $g_2 = (111001001)$ . The initial state of the convolutional encoder is assumed to be all zeros. The convolutional encoder output symbols must be repeated before being interleaved. The reverse traffic channel symbol repetition varies with data rate.. For the Reverse

Access channel, each code symbol is repeated 1 time. The Block interleaver spanning 20 ms interleaves all code symbols on the reverse channel prior to orthogonal modulation. The interleaver forms a cell array with 16 rows and 18 columns. Repeated code symbols at data rates lower than 9600 bps are written into the interleaver by columns filling the complete 16x18 matrix. Code symbols will be output from the interleaver by rows.

Modulation for reverse channel is 64-ary orthogonal modulation. One of 64 possible modulation symbols generated by Walsh functions is transmitted for each six symbols. Modulation symbols are selected according to the modulation symbol index (MSI).  $MSI = c_0 + 2c_1 + 4c_2 + 8c_3 + 16c_4 + 32c_5$  where each  $C_i$  represents the binary valued code symbol. Long code generator generates long code to perform direct sequence spreading (DSS). The long code is periodic with period  $2^{42}-1$  chips. Long code generator polynomial is  $p(x) = 1 + x + x^2 + x^3 + x^5 + x^6 + x^7 + x^{10} + x^{16} + x^{17} + x^{18} + x^{19} + x^{21} + x^{22} + x^{25} + x^{26} + x^{27} + x^{31} + x^{33} + x^{35} + x^{42}$ . Each PN chip of the long code is generated by the modulo-2 sum of the results through the AND gates inputting a 42-bit mask with the 42-stage LFSR of the sequence generator. The long code provides limited privacy. The long code mask varies depending on the channel type on which the mobile station is transmitting. The long code mask consists of a 42-bit binary sequence that creates the unique identity of the long code. There are two types of long code mask such as public long code mask and private long code mask. Direct sequence spreading is done by XORing the orthogonal modulation symbols and the long code PN chips (PN chips/Walsh chip is 4).

## 2.4 Requirement and Functioning of Different Blocks

### Cyclic Redundancy Check (CRC)

Error detection techniques allow a receiver to determine when a message has been corrupted during transmission through a noisy channel. This is typically done by having the transmitter calculate a value (often known as a checksum) based on the message to be transmitted. This checksum is then appended to the message to be transmitted. The receiver can then recalculate this checksum as the message is received and compare it to the transmitted checksum to determine if there were any errors during transmission. Traditional checksum calculations used simple summing formulas. The traditional formulas are not random enough to detect transmission errors that occur so that the checksum is internally consistent (i.e. multiple bit errors resulting in the same checksum). An inherent problem with using simple summing formulas for calculating checksums is that the probability of not detecting an error is only dependent on the width of the incoming data words, and not on the width of the checksum itself.

Cyclic Redundancy Codes (CRCs) are a popular type of redundant encoding. Cyclic redundancy code checkers (CRCCs) test for the differences between the transmitted and original data. CRCCs are very effective for a variety of reasons. They provide good protection against many common errors. The CRC is one of the most used techniques for error detection in data communications. The technique gained its popularity because it combines three advantages:

- Extreme error detection capabilities.
- Little overhead.
- Ease of implementation.

CRCCs are used to verify a frame of data by treating the entire frame as a very large binary number. This number is divided by a generator number and a remainder is produced. This remainder is then transmitted along with the data. At the receiver end, the incoming data is again divided by the same generator number, and the remainder is compared against the transmitted checksum. If the two remainders are different, a transmission error has occurred. Because the implementation of division operations is not practical for CPLDs, the incoming information must be converted to a form that is not as computationally intensive. The incoming bit stream is represented as the coefficients of a large polynomial. This polynomial is then divided by a generator polynomial that is chosen for different error detection capabilities. Optimum error detection is achieved by choosing a generator polynomial that can detect the expected transmission errors.

In general CRC codes are able to detect:

- All single- and double-bit errors.
- All odd numbers of errors.
- All burst errors less than or equal to the degree of the polynomial used.
- Most burst errors greater than the degree of the polynomial used.

The commonly used generator polynomials for the CRC calculation are the following:

16-bit frame quality indicator:  $x^{16} + x^{15} + x^{14} + x^{11} + x^6 + x^5 + x^2 + x + 1$

12-bit frame quality indicator:  $x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^4 + x + 1$

10-bit frame quality indicator:  $x^{10} + x^9 + x^8 + x^7 + x^6 + x^4 + x^3 + 1$

8-bit frame quality indicator:  $x^8 + x^7 + x^4 + x^3 + x + 1$

6-bit frame quality indicator:  $x^6 + x^5 + x^2 + x + 1$

### **CRC creation process:**

1. Get the raw frame
2. Left shift the raw frame by n bits and the divide it by Polynomial.
3. The remainder of the last action is the FCS.
4. Append the FCS to the raw frame. The result is the frame to transmit

### **CRC check process:**

1. Receive the frame.
2. Divide it by Polynomial.
3. Check the remainder. If not zero then there is an error in the frame.

## Tail Encoder

All Reverse Fundamental and Supplemental Channels use 8 tail bits for code termination of the forward error protection (FEC). The Tail encoder block is used for appending tail bits to the end of data stream from the frame quality indicator block. Once the tail bits are appended to the incoming data stream the convolutional encoder can be terminated in a known state, and thus it can be used for encoding the future bit streams otherwise after the encoding of one block of data, the convolutional encoder will be in some unknown state depending upon the last data frame. According to the IS 95 CDMA specifications the incoming data frames have 88 bits in a time frame of 20 ms. The tail encoder adds 8 zeros as tail bits; thus the data coming out of the tail encoder has 96 bits in 20 ms. The incoming data rate is 4.4 kbps and the outgoing data rate is 4.8Kbps.

## Convolutional Encoder

**Convolutional codes** are widely used to encode digital data before transmission through noisy or error-prone channels. Convolutional encoding serves to recover the corrupted received input data. As current information bit has algorithmic relations with some past information bits in convolutional code, it shows excellent capability in protecting data, though complexity in its decoding makes code analysis difficult. During encoding,  $k$  input bits are mapped to  $n$  output bits to give a rate  $k/n$  coded bit stream. The encoder consists of a shift register of  $kL$  stages, where  $L$  is described as the constraint length of the code.

At the receiver, the bitstream can be decoded to recover the original data, correcting errors in the process. The optimum decoding method is maximum-likelihood decoding where the decoder attempts to find the closest "valid" sequence to the received bitstream. The most popular algorithm for maximum-likelihood decoding is the **Viterbi Algorithm**. The possible received bit sequences form a "trellis" structure and the Viterbi Algorithm tracks likely paths through the trellis before choosing the most likely path.

## Block Interleaver & Repetition

### Block Interleaver

Interleaving is a standard DSP function used in many communication systems. Applications that store or transmit digital data require error correction to reduce the effect of spurious noise that corrupt data. Digital communication systems designers can choose many types of error correction codes (EECs) to reduce the effect of errors in stored or transmitted data. For example, Reed-Solomon encoders/decoders which are block encoding algorithms are used frequently to perform forward error correction (FEC). But for the correction of burst errors (which are very large amplitude errors occurring at a very small area) these error correction codes seems to be not enough. So for the correction of burst errors we go for interleaving techniques.

Interleaving is the process of reordering the symbols in a group of code words such that adjacent symbols in the data stream are not from the same codeword. The receiver reassembles the codeword when it

processes the data stream. The interleaving process helps reduce the effects of burst errors (i.e. multiple errors in a row), because rearranging the codeword symbols spread the errors among multiple code words.

### **Symbol Repetition**

Code symbol output from the convolutional coder are repeated, in order to increase the redundancy of transmission. The repetition factor represents the number of times a symbol appears. For instance, a repetition factor 1 means no repetition; a factor of 2 means a symbol is repeated once. Symbol repetition plays a role in adjusting the transmission rate to the desired chip rate.

### **Orthogonal Modulator**

Orthogonality is critical to any CDMA system. If each chip sequence is not perfectly orthogonal to all others, it is impossible to accurately decode the data streams under all circumstances. However, generating orthogonal codes is relatively straightforward. A very commonly used set is the Walsh functions, which has 64 such codes. Since the single base station generates all transmissions on the forward link, it can ensure that the chip sequences are correctly synchronized. However, this cannot be said for the reverse link (mobiles  $\rightarrow$  base station). In this direction, each mobile transmits completely asynchronously. Thus the base station must be able to simultaneously track, despread and demodulate each transmission. So

in the reverse direction Walsh chips are used for orthogonal data modulation.

The orthogonal modulator is used to spread the incoming data over a wide spectrum and also to orthogonally modulate the data. In CDMA, the entire 1.25-MHz transmission bandwidth is occupied by every station. The Walsh functions are chosen so that the set of functions are all orthogonal to each other. All base stations in the system are on the same frequency and use the same set of time-shifted Walsh functions. Every base station in the system is synchronized to every other base station in the system. Different base stations use time-shifted versions of the PN sequence to permit mobile stations to select transmissions from different base stations. Thus, for CDMA, the frequency reuse factor  $N$  is 1. The PN sequences used by the MS are found by computer simulation and are chosen to have low autocorrelation and cross-correlation properties.

### **Walsh Chip**

Hadamard-Walsh functions are binary orthogonal sequences, with power-of-two lengths. Within the function, Walsh chips are transmitted serially from the  $n^{\text{th}}$  row from left to right. Hadamard matrices are formed using the following recursion process:

$$\mathbf{H}_M = \begin{bmatrix} \mathbf{H}_{M/2} & \mathbf{H}_{M/2} \\ \mathbf{H}_{M/2} & -\mathbf{H}_{M/2} \end{bmatrix}$$

Where

$$\mathbf{H}_2 = \begin{bmatrix} +1 & +1 \\ +1 & -1 \end{bmatrix}$$

and  $M$  is a power of two. The rows of any instance form a mutually orthogonal set over the inner product.

The Walsh functions of order 64 are used as orthogonal cover on the forward link channels. They are also used as orthogonal modulation symbols in the reverse link. Although the functions are the same, they are used for entirely different purposes in the forward and reverse links.

### Long Code Generator

Long code generator generates long code to perform direct sequence spreading. In Reverse channel spreading the operation involves modulo 2 addition of the orthogonal modulator output stream and the long code. The long code is periodic with period  $2^{42}-1$  chips.

There are two different criteria that apply to the reverse link spreading. When a mobile is engaged in user traffic, i.e., in a conversation, it is desirable that that mobile use a unique code that is distinct from all others. A mobile-unique code, rather than a base-station-associated code, facilitates handoff. With a mobile-unique code, nothing needs to change about the mobile's modulation or coding when handoff occurs. Again, *phase* of the long code is used to distinguish stations. All  $2^{42}-1$  possible phases are available as logical addresses.

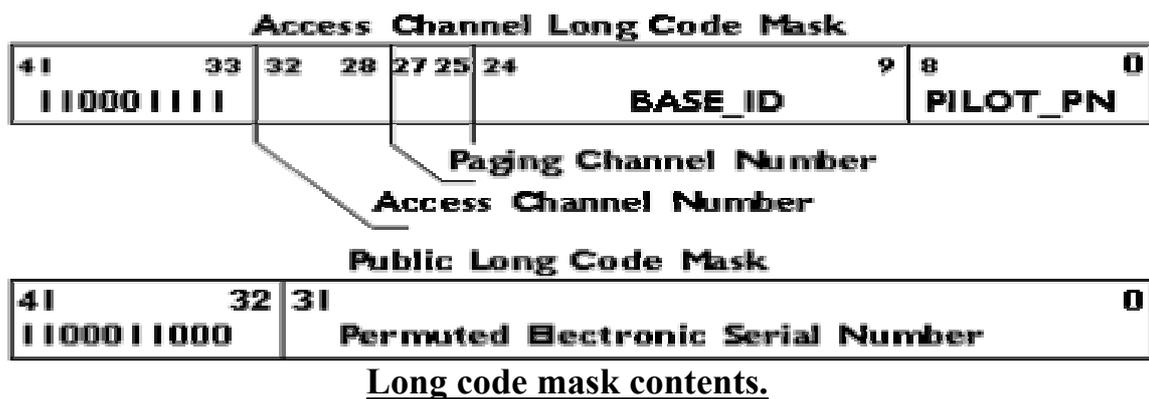
The second situation occurs when a mobile is attempting to gain the attention of a base station. Initially the base station has no knowledge that any particular mobile is in its service area. It is wildly impractical for each base station to search simultaneously for millions of potential subscriber codes. For these initial accesses, or any other non-traffic

uses of the air interface, it is desirable to have some Reverse Spreading codes that are base station-associated. If there are only a few associated codes for each base station, then it is practical for the base station to search for them continuously and simultaneously, awaiting the arrival of any user who wants service.

### **Long Code Mask**

The long code mask is a 42-bit number that serves as a logical address for Reverse CDMA Channel spreading codes. It is used to select specific bits from the long code linear feedback shift register to be added, modulo-two, in order to produce the actual long code, at the proper phase.

When transmitting on an access channel, a mobile station constructs the mask from the base station identity, the paging channel number, and the access channel number. When transmitting traffic, a mobile station uses a mask constructed from its Electronic Serial Number.



## 2.6 Advantages of CDMA Technology

When implemented in a cellular telephone system, CDMA technology offers numerous benefits to the cellular operators and their subscribers. The following is an overview of the benefits of CDMA.

1. Capacity increases of 8 to 10 times that of an AMPS analog system and 4 to 5 times that of a GSM system
2. Improved call quality, with better and more consistent sound as compared to AMPS systems
3. Simplified system planning through the use of the same frequency in every sector of every cell
4. Enhanced privacy and security
5. Improved coverage characteristics, allowing for the possibility of fewer cell sites
6. Increased talk time for portables
7. Bandwidth on demand
8. Reduced average transmitted power, thus increasing talk time for portable devices.
9. Reduced interference to other electronic devices.
10. Reduction in the number of calls dropped due to handoff failures .
11. Development of a reliable transport mechanism for wireless data communications ...

### **3. Digital Design Process**

#### **3.1 Process of Design**

Initially a designer starts with a design idea. An initial design idea goes through several transformations before its hardware implementation is obtained. At each step of transformation, the designer checks the result of last transformation adds more information to it and passes through to the next step of transformation. A more complete definition of the intended hardware must than be developed from the initial design ideas. Therefore it is necessary for the designer to generate a behavioral definition of the system under design.

The next process of the design is the design of the systems data path. In this phase, the designer specifies the registers and logic units necessary for implementation of the system. These components may be interconnected using either bi-directional or unidirectional buses .Based on the intended behavior of the system, the procedure for controlling the data between the registers and logic units through buses is then developed.

Logic design is the next step in the design process, and involves the use of primitive gates and flip-flops for the implementation of data registers buses, logic units and their controlling hardware. The result of this stage is a netlist of gates and flip-flops .The next design stage transforms the netlist of the previous stage into a transistor list or layout. This involves the replacement of gates and flip-flops by their transistor equivalents or

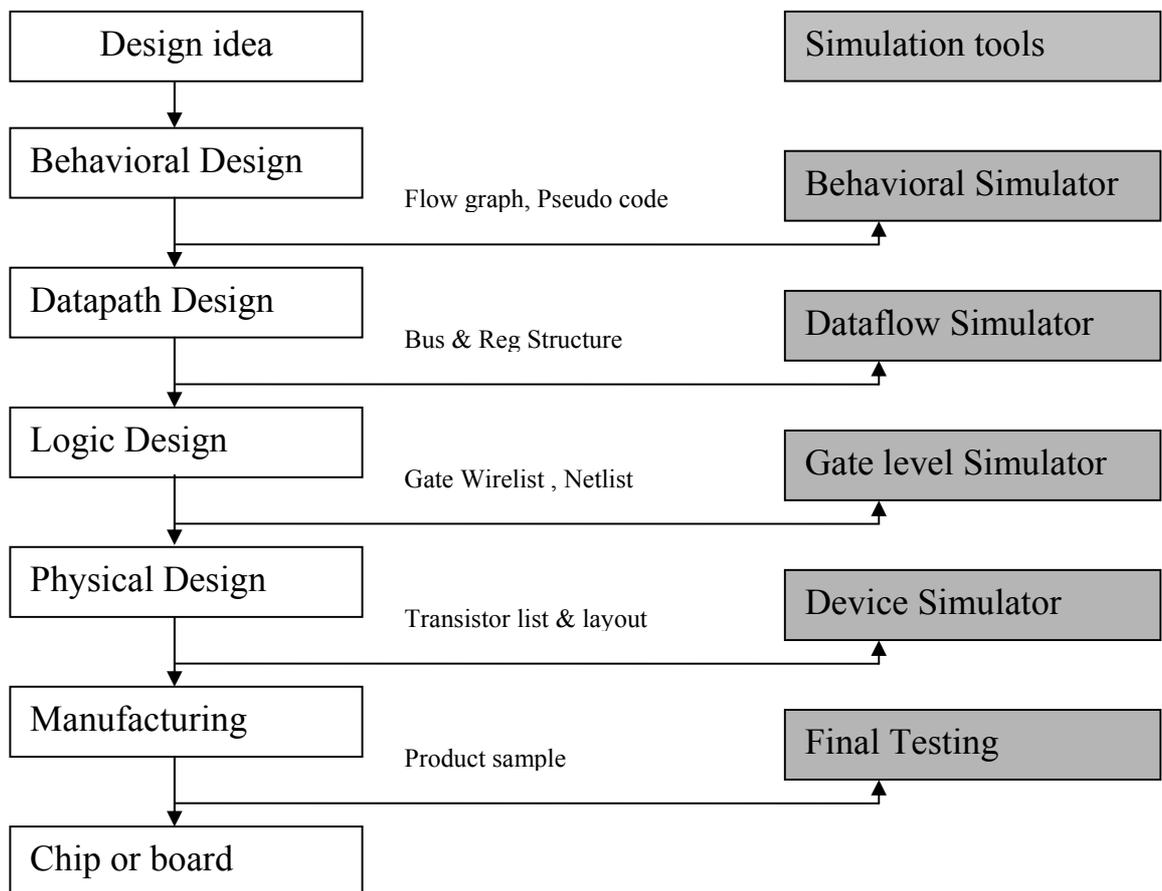
library cells. This stage considers loading and timing requirements in its cell or transistor selection process.

In the design process, much of the work of transforming a design from one form to another is tedious and repetitive. These activities for verification of a design stage output can be done at least in part by the computers. This process is defined as design automation. Design Automation tools can help the designer with the design entry, hardware generation, text sequence generation, documentation verification and design management. Hardware Description Languages provide formats for representing the outputs of various design stages. An HDL based DA tool for the analysis of a circuit uses this format for its input description, and a synthesis tool transforms its HDL input into an HDL which contains more hardware information.

### **3.2 Simulation**

In a design automation environment, HDL descriptions of systems can be used for the input of simulation programs. Simulators may be used to verify the results of any of the design stages in the previous figure. In addition to the circuit description the simulator needs a set of stimulation data or stimuli. The simulation program applies this data to the input description at the specified times and generates responses of the circuit. The results of a simulation program may be illustrated by waveforms, timing diagrams or time value tabular listings. These results are interpreted by the designer who determines whether to repeat a design stage if simulation results are not satisfactory.

As shown in figure below simulators can be used at any design stage. At the upper levels of design process, simulation provides information regarding the functionality of the system under design. These simulation runs are usually very quick. Simulation at lower levels of design process, for example gate level or device simulation runs much more slowly, but provides more detailed information about the timing and functionality of the circuit. In our project, we had done the logic design of the Reverse CDMA channel and then simulated and tested it using various EDA tools and One of the blocks was physically implemented.



**Verifying each design stage by simulating its output**

### 3.3 Synthesis, Optimization and device fitting

**Synthesis** is the process by which abstract design descriptions are reduced into a lower level circuit representation, such as netlists or equations. HDLs provide the input and output of hardware synthesizers.

**Optimization process** depends on 3 things, the form of the Boolean expressions, the type of resources available and automatic or user applied synthesis directives (sometimes called constraints). These constraints may be to limit the number of literals in an expression (to reduce fan in ) or limit number of terms in an expression . Optimizing for CPLDs usually involves reducing the logic to minimal sum of products which in turn reduces the product term utilization and number of logic inputs before any given expression .Optimizing for FPGAs involve factoring of systems of equations based on device specific resources and directive driven optimization goals .

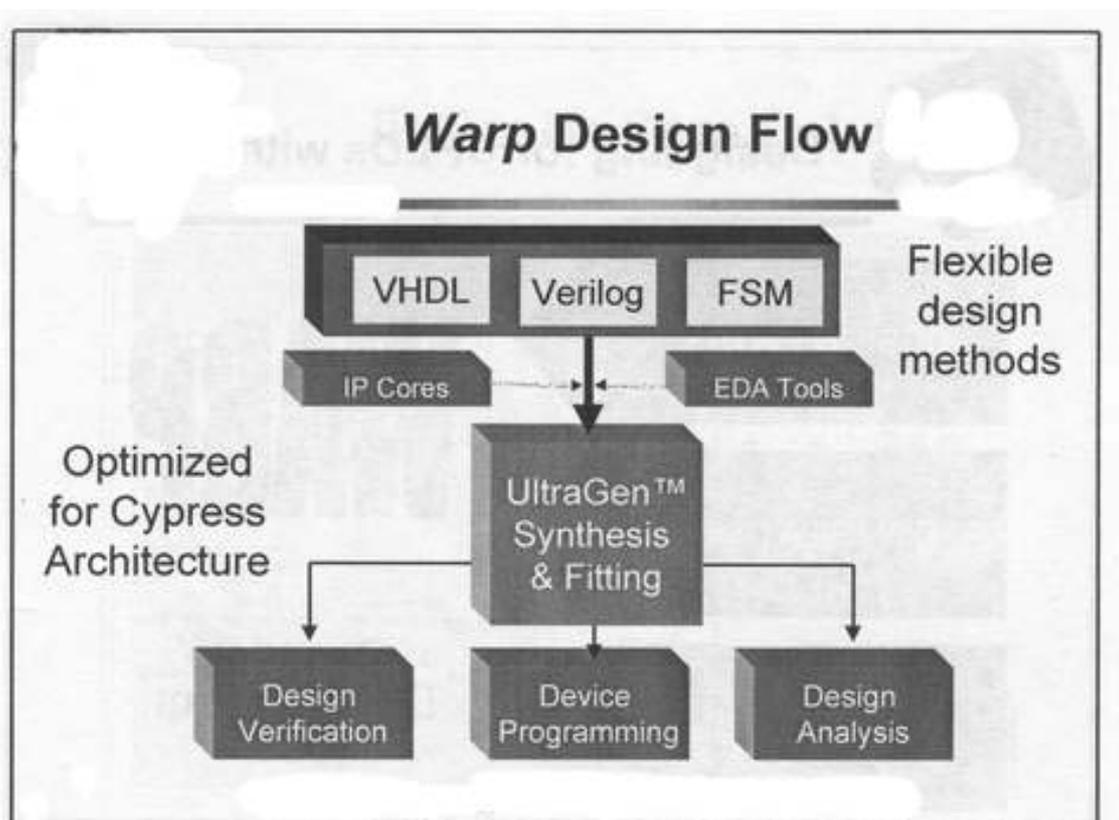
**Fitting** is the process of taking the logic produced by the synthesis and optimization processes and placing it into a logic device to obtain the best fit.

**Placing and Routing** is the process of taking logic produced by synthesis and optimization , transforming it if necessary , packing it into the FPGA logic structures (cells), Placing the logic cells in optimal locations and routing signals from logic cell to logic cell or I/O.

### 3.4 Study of Various EDA tools

#### 1. Warp

**Warp**, the EDA tool from **Cypress Semiconductor** provides complete solution for the design and development of CPLDs. Warp facilitates design entry, compilation and verification and device programming for CPLDs. The major features of *Warp Release 6* includes. VHDL and Verilog synthesis, dynamic timing simulator, static timing analyzer, Architecture Explorer, FSM Graphical Editor, HDL text editor, Graphical Pin Editor, HDL to Graphics conversion tool, HDL debugger, Test bench generator etc.



## 2. Active HDL

*Active HDL*, the EDA tool from *Aldec* provides a complete solution for the design and verification of FPGAs. We used Active HDL version 5.1 for the simulation purposes. Active-HDL is an integrated environment designed for development of VHDL, Verilog, EDIF and mixed VHDL-Verilog-EDIF designs. It comprises three different design entry tools, VHDL'93 compiler, Verilog compiler, single simulation kernel, several debugging tools, graphical and textual simulation output viewers, and auxiliary utilities designed for easy management of resource files, designs, and libraries. The figure below shows graphically all the Active-HDL components.

## 3. ModelSim

*ModelSim* is a industry standard tool from *Model Technology Incorporated*, used for the design and verification of CPLDs and FPGAs. *ModelSim* VHDL supports both IEEE 1076-1987 and 1076-1993 VHDL, the 1164-1993 *Standard Multivalue Logic System for VHDL Interoperability*, and the 1076.2-1996 *Standard VHDL Mathematical Packages* standards. Any design developed with *ModelSim* will be compatible with any other VHDL system that is compliant with either IEEE Standard 1076-1987 or 1076-1993. *ModelSim* Verilog is based on IEEE Std 1364-1995 and a partial implementation of 1364-2001, *Standard Hardware Description Language Based on the Verilog Hardware Description Language*. The Open Verilog International *Verilog LRM version 2.0* is also applicable to a large extent. Both PLI (Programming Language Interface)

and VCD (Value Change Dump) are supported for ModelSim. In addition, all products support SDF 1.0 through 3.0, VITAL 2.2b, VITAL'95 – IEEE 1076.4-1995, and VITAL 2000.

ModelSim also includes various performance tools such as

- Performance Analyzer

Identifies areas in the simulation where performance can be improved.

- Code Coverage

Gives a graphical and report file feedback on how the source code is being executed.

#### 4. Max Plus II

*Maxplus II* is a tool from *Altera*. It is a complete tool with VHDL\Verilog simulation and synthesis options for Max CPLDs and also for all other industry leading CPLDs. The devices that are supported by Max Plus includes Max5000 , Max7000, Max 3000, Flex 6000 series of CPLDs. The various design entry features in Max Plus includes schematic design entry, Text based design entry using Altera HDL, Interface to popular EDA tools, Floor plan editing, Hierarchical design management, library of parameterized modules etc. The design compilation features include Logic Synthesis and automatic fitting, Automatic Error correction etc. The various design verification features include timing analysis, functional simulation, timing simulation, Waveform analysis and also create output files to other simulation tools. Also online help is available for MAX PLUS II

## **4.Digital Specifications & Proposed Digital Architecture**

### **4.1 Frame Quality Indicator \ CRC Encoder**

Cyclic Redundancy Codes (CRCs) are a popular type of Forward error correcting code (FEC). Here the incoming data bit stream is divided by a generator number and a remainder is produced. This remainder is then transmitted along with the data. At the receiver end, the incoming data is again divided by the same generator number, and the remainder is compared against the transmitted checksum. If the two remainders are different, a transmission error has occurred. Because the implementation of division operations is not practical for CPLDs, the incoming information must be converted to a form that is not as computationally intensive. The incoming bit stream is represented as the coefficients of a large polynomial. This polynomial is then divided by a generator polynomial that is chosen for different error detection capabilities.

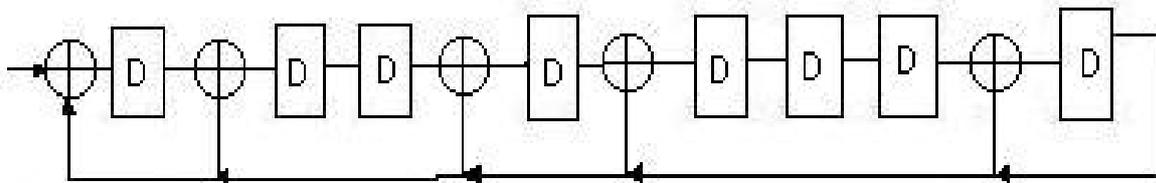
### **Digital Specifications**

- **Input: 80 bits in 20 ms frame i.e. @ 4Kbps**
- **Output : 88 bits in 20 ms time frame i.e. @ 4.4 Kbps**
- **Initial Contents of the register : All ones**

**CRC Polynomial Used :  $1 + X + X^3 + X^4 + X^7 + X^8$**

## CRC Generation

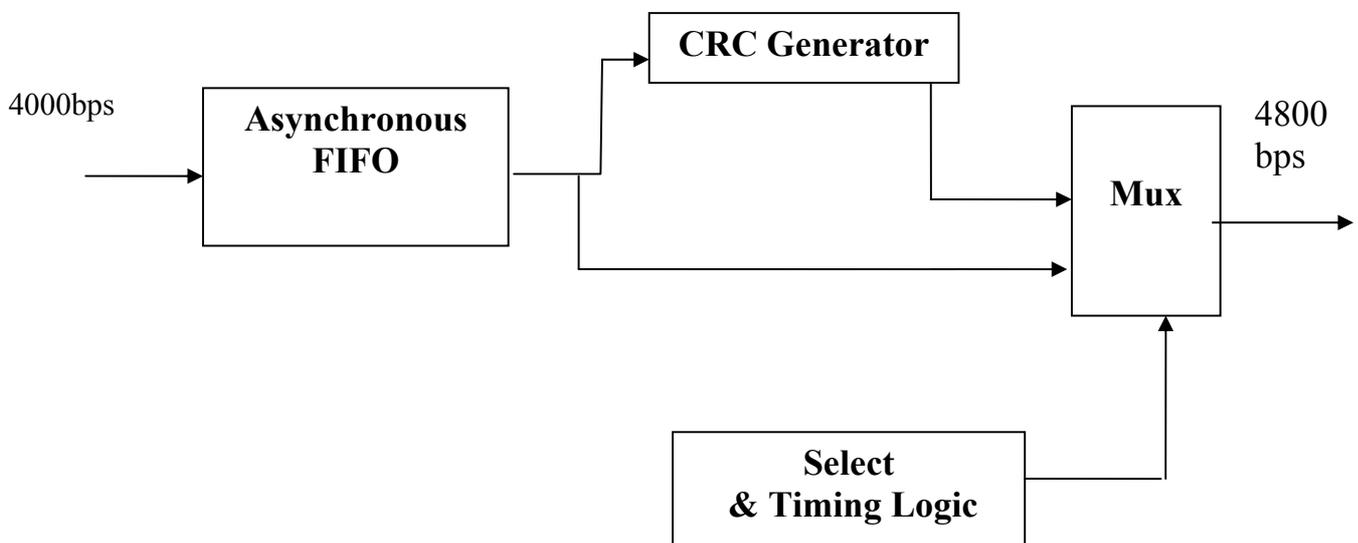
CRC codes make use of a Linear Feedback Shift Register (LFSR) to generate a signature based on the contents of any data passed through it. This signature can be used to detect the modification or corruption of bits in a serial stream. These CRC codes are traditionally calculated on the serial data stream using a Linear Feedback Shift Registers (LFSR) built from flip-flops and XOR gates. In the equation of the CRC polynomial, the superscripts identify the tap location in the shift register. The order of the polynomial is identified by the highest order term, and specifies the number of flip-flops in the shift register. Since these polynomials are for modulo-2 arithmetic, each bit-shift is equivalent to a multiply by 2. Figure shows the CRC-8 generator which is used in our case.



### Digital architecture of CRC GENERATOR

## Changing the Datarate

Since the output datarate is higher than the input datarate, we have to use an asynchronous FIFO which uses a DPRAM (dual port RAM) for changing the datarate. Initially the data is written into the FIFO at a lower datarate and it is read from the FIFO at a higher datarate i.e. using a higher clock. Once the datarate is changed the high speed data stream is passed to the CRC generator to calculate the CRC for the Block. While the CRC generator is calculating the CRC for the particular Block the Multiplexer logic transmits the same data, which is at a higher speed than the input data rate. Once the 80 bits of input data block are transmitted at a higher speed, then multiplexer logic selects the output of the CRC generator block and next 8 bits transmitted is the calculated checksum. Thus CRC gets appended to the data stream.



## Digital architecture of Frame quality indicator

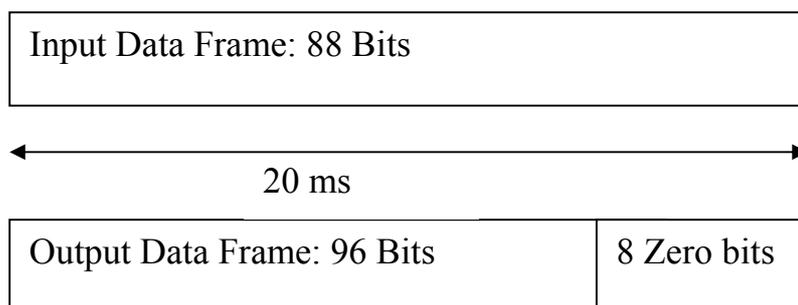
## 4.2 Tail Encoder

The Tail encoder block is used for appending tail bits to the end of data stream from the frame quality indicator block and thus to terminate the convolutional encoder in a known state.

### Digital Specifications

- **Input: 88 bits in 20ms time frame i.e. @ 4.4 Kbps**
- **Output : 96 bits in 20 ms time frame i.e. @ 4.8 Kbps**

**Note:- Here 8 zeroes are appended to 88 bits , thus getting an output of 96bits in the same 20 ms time frame**



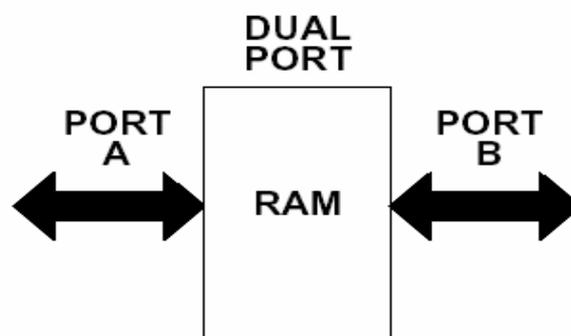
### Input\Output Data frame Structure

From the specifications it is clear that, incoming and outgoing data rates are different. Therefore we used an asynchronous FIFO for implementing the tail encoder. Data can be read from and written into the asynchronous FIFO at different rates at the same time. An asynchronous FIFO is realized using the DPRAM (Dual Port RAM). It has independent read and write clocks which perform the read and write operations at the

required rates. The main feature of the DPRAM is that data can be written into and read from it at the same time but not from the same location. For read and write operations the corresponding addresses are to be provided by using a digital counter logic.

### DPRAM (Dual Port Ram)

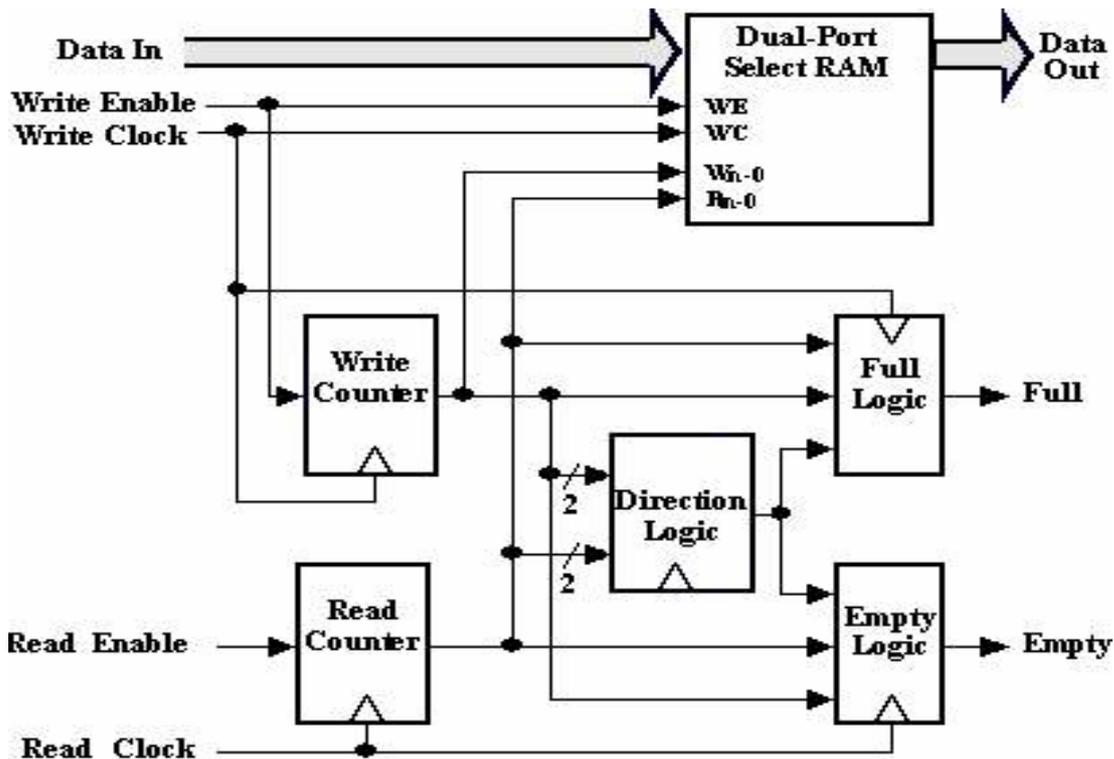
A Dual Port RAM is simply a random access memory having two different ports. One port is used for writing data into the RAM and the other one is used for reading data from the RAM. Since a dual port memory has two ports, we can simultaneously read as well as write into the memory. But if we try to read and write into the same memory location it will result in undesirable results like data bit error etc...



### Basic Dual Port RAM

For the read operation, a read address is given and when the read clock comes the data is read from the address specified. For the write operation, the write address is given and when the write clock comes

the data is written into the address specified. Both these operations occur at the same time and at different locations.

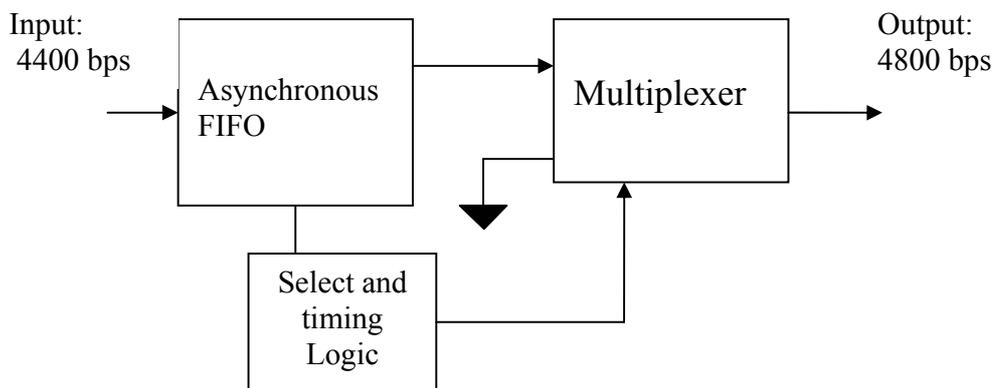


FIFO Block Diagram

In the asynchronous FIFO, there will be a digital logic that produces the read and write addresses according to read pointer and write pointer values. Data are written into the FIFO at one rate and read out of it at another rate. Thus tail encoder function is realized using the asynchronous FIFO using the DPRAM architecture. The input data frame is written into the FIFO at the rate of 88 bits per 20 ms time period and read at a high clock rate (4.8kbps). During the read out process, in the 20 ms

interval, after 88 bits have been read, 8 bits come to the output from a zero generator; thus 96 bits come at the output at a rate of 4.8 kbps. So the tail encoder's requirements of an input data rate of 4.4 kbps and output data rate of 4.8 kbps are satisfied. To append 8 zero bits at the end of the data stream a data multiplexer and timing logic is used. Once the data at high speed is transmitted the timing logic selects the 2<sup>nd</sup> input of the multiplexer which is permanently grounded for a time which corresponds to 8 bit of data at 4.8 Kbps. Thus the zeros get appended to the data stream.

### **Complete Digital Architecture of Tail Encoder**



### 4.3 Convolutional Encoder

**Convolutional codes** are widely used to encode digital data before transmission through noisy or error-prone channels. Convolutional encoding serves to recover the corrupted received input data.

#### Digital Specifications

- **Input : 96 bits in 20ms time frame i.e. 4.8Kbps**
- **Out put: 288 bits in 20ms after Convolutional encoding  
i.e. 14.4 Kbps**

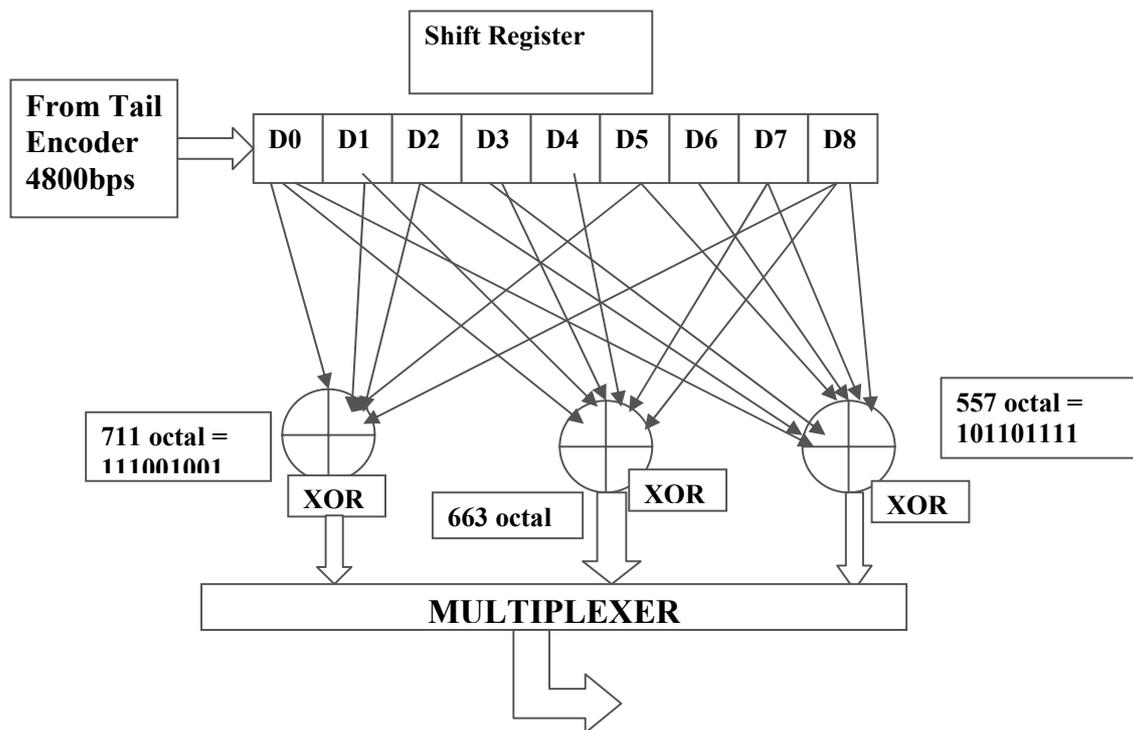
**Generator Polynomials:  $P1=1 +X^2 +X^3 +X^4+X^6+X^7+X^9$**

**$P2=1 +X^2 +X^5+X^6+X^8+X^9$**

**$P3=1 +X^4+X^7+X^8+X^9$**

In the reverse link, we use a rate 1/3, constraint length 9 convolutional code with generator function  $g_0$  equals 557(octal) and  $g_1$  equals 663 (octal) and  $g_2$  equals 711(octal) as specified by a IS 95 CDMA standard for wireless communications. The rate 1/3 convolutional encoder takes every information bit and outputs three coded bits, based on the 9 previous input information bits that are stored in the 9 memory elements. Since there are 9 elements in the encoder, the encoder can be in 512 different states depending on the contents of the memory elements.

The code symbols are sent out such that the code symbol (c0) encoded with generator function g0 goes out first, the code symbol (c1) encoded with generator function g1 is output second, and so forth if necessary. i.e. the output of the Xor gates are multiplexed using a clock which is three times the system clock . The state of the convolutional encoder, upon initialization, is the all-zero state. The first code symbol that is output after initialization is a code symbol encoded with generator function g0.



### Digital Architecture - Convolutional Encoder

## 4.4 Block Interleaver & Repetition

Interleaving is the process of reordering the symbols in a group of code words such that adjacent symbols in the data stream are not from the same codeword. The receiver reassembles the codeword when it processes the data stream. This is done to correct the burst errors that may occur during transmission. The output is repeated in order to increase redundancy of transmission.

### Digital Specifications

**Input: 14.4Kbps**

**Output: After repetition and interleaving 28.8Kbps**

**Output From Block inter leaver is in the following order 1, 17, 33, 49, ---**

**The output from the block Interleaver is read at a higher clock (twice the input clock) in order to repeat the signal for achieving redundancy**

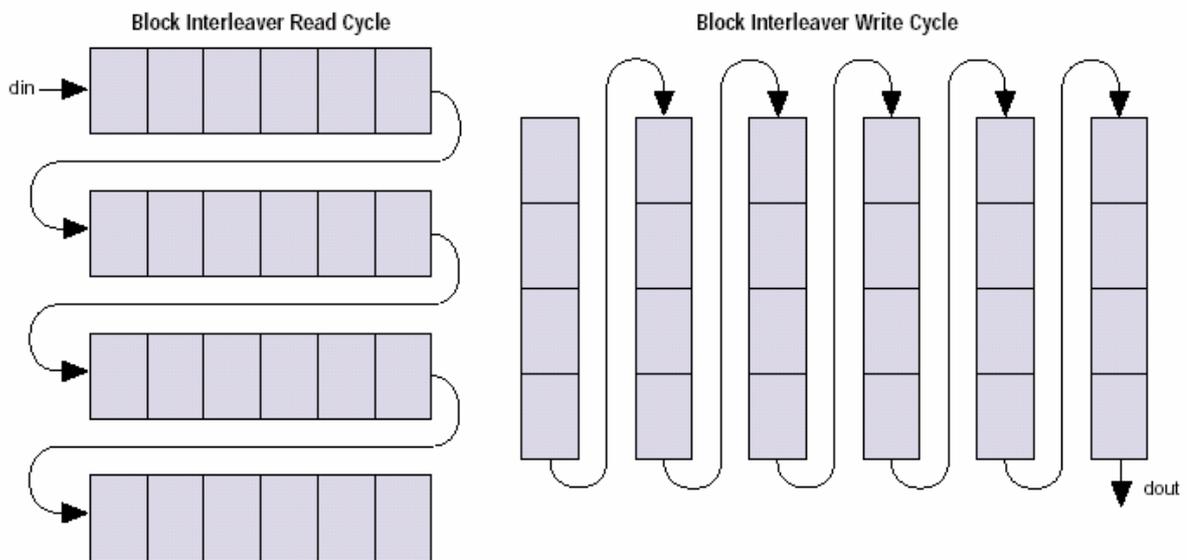
The block interleaver/de-interleaver uses SRAM memory configured as a matrix of  $n$  rows by  $n$  columns to perform interleaving. During the write cycle, the input symbols are written column by column, and during a read cycle, the output symbols are read row by row.

The block interleaver/de-interleaver operates in discrete mode with a single-port memory used as a buffer. The symbol transmission consists of an alternating sequence of write and read cycles.

1	17	.....	273
2	18	.....	274
3	.	.....	275
4	.	.....	276
.	.	.....	.
15	.	-	.
16	.	.....	- 288

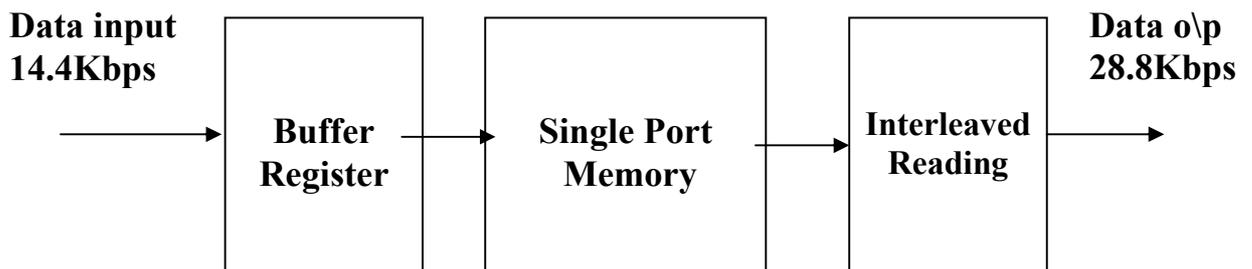
**Ordering of bits in the memory array**

Output from Block inter lever is in the following order 1, 17, 33, 48, ----- .The output from the block Interleaver is read at a higher clock (twice the input clock) in order to repeat the signal for achieving redundancy



**Read and Write Process**

## **Digital Architecture**



The digital architecture of the Block interleaver consists of a buffer register, a single port memory and an interleaved reading mechanism. Initially data will be written into the buffer register column by column and once the buffer register is full , the entire data in the buffer is transferred to the single port memory . Once the data is transferred to the single port memory, it is read in a row wise fashion. Thus the data automatically gets interleaved. While the data from the single port memory is being read the buffer register collects the next data frame.

## 4.5 Orthogonal Modulator

The data modulation in the Reverse link is 64-ary orthogonal, and is applied prior to the spreading. For orthogonal data modulation a Walsh function is used, since all the 64 codes in the Walsh function is orthogonal to each other. Groups of six code symbols is used to calculate a modulation Symbol index and according to the value of the modulation symbol index, we select one of 64 orthogonal sequences from the Walsh function. Each period of the Walsh sequence (a *Walsh Chip*) is four PN chips in duration. The modulation symbol rate is thus always 4,800Sps.

The orthogonal modulator spreads the spectrum of the original data for increase the noise robustness , increase the security etc , The 64 bit Walsh chips are orthogonal to each other , i.e. There will be minimum cross correlation between these chips . Since the chips produced by the block are orthogonal to each other, it is called an orthogonal modulator

### Digital Specifications

- **Input : Data @ 28.8 Kbps**
- **One Walsh chip is transmitted for six coded bits**
- **Modulated symbol rate  $28.8 \text{ kbps} * 64 \text{ chips} / 6 \text{ coded bits} = 307.2$**

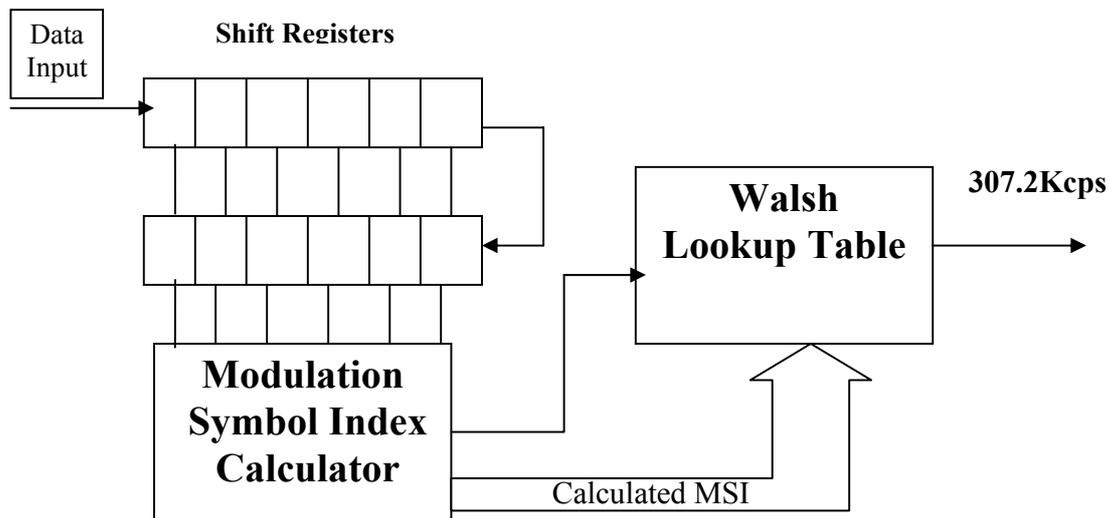
**Kcps**

- **Modulation Symbol Index**

$$\text{MSI} = c_0 + 2c_1 + 4c_2 + 8c_3 + 16c_4 + 32c_5$$

- **According to the value of MSI , corresponding row from the Walsh lookup table is transmitted , instead of the 6 bits**

The incoming data are read at the rate of 6 bits at a time into a buffer. When the buffer is full the bits are transferred into another buffer and their modulation symbol index is calculated. Then the corresponding Walsh chip is sent which is 64 bit array of bits. The index is calculated as follows.



### Orthogonal Modulator-Digital Architecture

## 4.6 Long Code Generator

The long code is a period  $2^{42}-1$  LFSR sequence that is used for spreading the reverse link. There is only one long code *sequence*. Different stations are distinguished not by the sequence itself but by its *relative phase*.

### Digital Specifications

**Input: 42 bit Long Code Mask**

**Output:  $(2^{42}) - 1 = 1.2288\text{Mcps}$  (No: of FFs=42)**

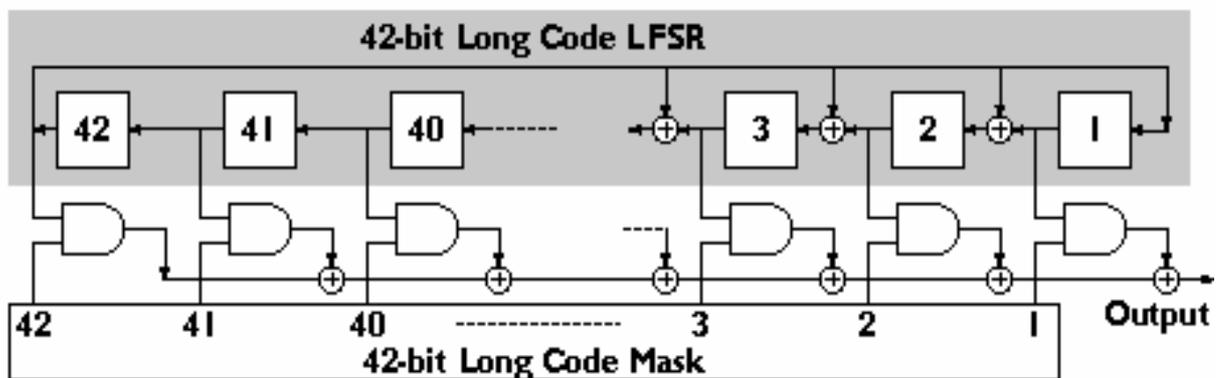
**LFSR Polynomial**

$$P(x) = 1 + X + X^2 + X^3 + X^5 + X^6 + X^7 + X^{10} + X^{16} + X^{17} + X^{18} + X^{19} + X^{21} + X^{22} + X^{25} + X^{26} + X^{27} + X^{31} + X^{33} + X^{35} + X^{42}$$

The different phases of the long code are generated by use of one of the well-known properties of LFSR sequences. Any modulo-2 sum of different phases of a LFSR sequence gives a third phase of that same sequence. A corollary of this property is the fact that all internal nodes of any LFSR generator also run through the same sequence as the generator output, but with different phases. This addition property of LFSR sequences is exploited in the long code generation process for the reverse link spreading. A 42-bit number, the Long Code Mask, is used to select particular bits of the 42-bit long code generator register. The selected nodes are

summed, modulo 2. The resultant of the sum, that is, the modulo-2 inner product of the generator state with the mask, is the generator output corresponding to that mask. The period of the long code, for what it's worth, is 3.6 million seconds, or about 41 days.

Each PN chip of the long code is generated by the modulo 2 sum of the results through the AND gates inputting a 42 bit mask with the 42 stage LFSR of the sequence generator. The Long code mask varies depending on the channel type on which the mobile station is transmitting. The long code mask consists of a 42 bit binary sequence that creates the unique identity of the long code.



### Long Code generator Architecture

## 6. Hardware Implementation on an Ultra 37256V CPLD

The last step in the implementation of a digital circuit is its hardware realization using Complex Programmable Logic Devices (CPLDs) or Field Programmable Gate Arrays (FPGAs). We have successfully implemented one of the blocks (long code generator) into an **Ultra 37256VP160-66AC** CPLD from Cypress. *Warp Release6* the EDA tool from Cypress Semiconductor along with *ISR Release 3.0* (programming software) was used for the synthesis of the CPLD. After implementation, the chip has been tested by giving inputs and the outputs were observed in a CRO. The output signal was identical to that observed in the simulation.

### General Description of Ultra 37000 CPLDs

The Ultra37000™ family of CMOS CPLDs provides a range of high-density programmable logic solutions with unparalleled system performance. The Ultra37000 family is designed to bring the flexibility, ease of use, and performance to high-density CPLDs. The architecture is based on a number of logic blocks that are connected by a Programmable Interconnect Matrix (PIM). Each logic block features its own product term array, product term allocator, and 16 macrocells. The PIM distributes signals from the logic block outputs and all input pins to the logic block inputs. All of the Ultra37000 devices are electrically erasable and In-System Reprogrammable (ISR), which simplifies both design and manufacturing flows, thereby reducing costs. The ISR feature provides the ability to reconfigure the devices without having design changes cause pinout or timing changes. The Cypress ISR function is implemented through a JTAG-compliant serial interface. Data is shifted in and out through the TDI and TDO pins, respectively. Because of the superior routability and

simple timing model of the Ultra37000 devices, ISR allows users to change existing logic designs while simultaneously fixing pinout assignments and maintaining system performance. The entire family features JTAG for ISR and boundary scan, and is compatible with the PCI Local Bus specification, meeting the electrical and timing requirements. The Ultra37000 family features user programmable bus-hold capabilities on all I/Os.

### Features of Ultra 37256VP160-66AC

- In-System Reprogrammable™ (ISR™) CMOS CPLDs
  - ✓ JTAG interface for reconfigurability
- High density
  - ✓ 256 macrocells
  - ✓ 160 I/O pins
  - ✓ 5 dedicated inputs including 4 clock pins
- Simple timing model
  - ✓ No fanout delays
  - ✓ No expander delays
  - ✓ No dedicated vs. I/O pin delays
  - ✓ No additional delay through PIM
- PCI Compatible
- Programmable Bus-Hold capabilities on all I/Os
- Intelligent product term allocator provides:
  - ✓ 0 to 16 product terms to any macrocell
  - ✓ Product term steering on an individual basis
- Flexible clocking
  - ✓ 4 synchronous clocks per device
  - ✓ Product Term clocking

## **6. Conclusion**

A digital architecture was developed for the **Reverse CDMA access channel** according to the IS 95 CDMA specifications. The circuit was then coded in a Very High Speed Integrated Circuit Hardware Description Language (VHDL). The next step involved was the testing and simulation of the circuit. And finally one of the blocks was successfully implemented in a **CPLD (Ultra 37256VP160-66AC)** from Cypress Semiconductor. The EDA tools used for the simulation were *ModelSim* and *Active HDL* and for the synthesis *WARP* and *ISR Programming software* from Cypress were used.

In addition to this we can develop the digital architecture for the reverse traffic channel and other various forward channels in CDMA according to IS 95 standards for the completeness of the project. By successfully completing this project, we believe we are better equipped for taking up the challenges of developing Integrated circuits for other advanced communication systems like CDMA2000, WCDMA etc.

## **6. References**

1. C. Y. Lin and J. Shieh, "IS-95 North American standard-a CDMA based digital cellular system". IEEE Website
2. [www.itu.int\imt](http://www.itu.int/imt)
3. [www.cdg.org](http://www.cdg.org)
4. Mobile cellular Telecommunications, William C Y Lee
5. Digital System Design with VHDL and Synthesis, K C Chang
6. The Designer's Guide to VHDL. Peter J Ashenden

## 7. Appendix

7.1) <u>IS 95 Specifications for Reverse CDMA Access Channel</u>	
Frame Quality Indicator	Input : 80 bits in 20 ms frame Output : 88 bits in 20 ms frame (last 8 bits are calculated crc) CRC Polynomial : $1 + X + X^3 + X^4 + X^7 + X^8$
Tail Encoder	Input : 88 bits in 20 ms frame Output : 96 bits in 20 ms frame Last 8 bits are zeros appended
Convolutional Encoder	Input : 96 bits in 20 ms frame Output : 288 bits in 20 ms frame Generator Polynomials: $P1=1 + X^2 + X^3 + X^4 + X^6 + X^7 + X^9$ $P2=1 + X^2 + X^5 + X^6 + X^8 + X^9$ $P3=1 + X^4 + X^7 + X^8 + X^9$
Block Interleaver & Repetitor	Input : 288 bits in 20 ms frame Output : 576 bits in 20 ms frame <ul style="list-style-type: none"> <li>• Bits are repeated once for achieving redundancy.</li> <li>• Bits are interleaved in the order 1, 17, 33, 49.....</li> </ul>
Orthogonal Modulator	<ul style="list-style-type: none"> <li>• Input : Data @ 28.8 Kbps</li> <li>• One Walsh chip is transmitted for six coded bits</li> <li>• Modulated symbol rate = 307.2 Kcps</li> <li>• Modulation Symbol Index</li> </ul> $MSI = c_0 + 2c_1 + 4c_2 + 8c_3 + 16c_4 + 32c_5$

## 7.5 Important Terms

### Cell

Loosely, one or more collocated base stations. They can service different angular sectors, different frequencies, or both.

### Chip

Informal term used to refer to either a binary element of a spreading sequence, or to the time interval that it occupies, or  $1/1.2288 \text{ MHz} = 813.8 \text{ ns}$ , or, at the speed of light, a distance of  $c/f = (0.3 \text{ m/ns})/1.2288 \text{ MHz} = 244.1 \text{ meters}$ .

### Erlang

A dimensionless unit of telephone traffic intensity. It is numerically equal to the calling rate times the average holding time. It is named for the Norwegian telephone engineer who first popularized the concept.

### Handoff

The act of transferring control of a mobile station from one base station to another is Handoff. There are basically two types of handoffs, Hard Handoff and Soft Handoff: A handoff characterized by a temporary disconnection of the Traffic Channel is called hard handoff. In a Soft Handoff when a mobile nears the boundary of the neighboring cell, it receives signals from both the base station. As a result during handoff call won't be disconnected or no glitches occur.

### Mcps

Mega chips per second ( $10^6$  chips per second).

### Pilot PN Sequence

A pair of modified maximal length PN sequences with period  $2^{15}$  used to spread the Forward CDMA Channel and the Reverse CDMA Channel. Different base stations are identified by different pilot PN sequence offsets.