International Journal of Scientific Engineering and Research (IJSER)
www.ijser.in
ISSN (Online): 2347-3878, Impact Factor (2015): 3.791

Simulation and Implementation of Convolution Encoder and Viterbi Decoder

V. S. Sreekanth[1], Y. Rama Krishna[2], A. V. V. Prasad[3]

[1]Student, PVP Siddhartha Institute of Technology, Vijayawada, India
[2]Associate Professor, ECE, PVP Siddhartha Institute of Technology, Vijayawada, India
[3]Group Director, MRS & GDP, N.R.S.C-I.S.RO, Hyderabad, India

Abstract: ***Convolution encoding and Viterbi decoding are error correction techniques widely used in communication systems to improve the bit error rate (BER) performance. Satellite communication has got many applications such as payload information reception in which the messages transmitted are encoded into the communication channel and then decoding it at the receiver end. Viterbi algorithm enables clear and practically error-free communication over long distances, from moving low power transmitters and receivers. Convolution encoding with Viterbi decoding is a Forward Error Correction (FEC) technique that is well suited for use with channels where the transmitted signal is corrupted mainly by Additive White Gaussian Noise (AWGN). The convolution encoder inserts redundant information bits into the data stream so that the decoder can reduce and correct errors caused by the channel. However, the extra bits increase the data rate (bits/sec) and as a consequence, also increase the bandwidth of the encoded signal. Using a technique known as puncturing, the data rate can be dynamically changed according to the channel quality. The technique works by deleting symbols from the encoded data and thus, reduces the coding rate. FEC is typically used in digital communications systems to enable the receiver to detect and correct errors without having to ask the sender for additional data.***

Keywords: Spread Spectrum, AWGN, FEC, Puncturing, Interference, Channel capacity

1. Introduction

Viterbi decoding has become one of the most widely used algorithms in a wide range of fields and engineering systems. Modern disk drives with “PRML” technology to speed-up accesses, speech recognition systems, natural language systems, and a variety of communication networks use this scheme or its variants. Today the algorithm is used in billions of cell phones, magnetic recording, and most satellite TV receivers, a variety of cable TV systems, voice recognition, and even DNA sequence analysis. The Transfer Control Protocol and the Internet Protocol (TCP/IP), Wi-Fi and Bluetooth are another uses of the Viterbi algorithm and methods based on it. In short, Viterbi's algorithm is enabling today's exploding wireless world.

2. Realisation of Viterbi Decoder

Convolutional coding and Viterbi decoding, along with quadrature phase-shift keyed modulation (QPSK), is presented as an efficient system for reliable communication. The hardware realization consists of add compare and select block(ACS); branch metric unit(BMU) block; clock divided by 2 block; serial-in-parallel-out (SIPO) register block; Memory block; last-in-first-out(LIFO) structure. Performance results, obtained theoretically and through computer simulation, are given for optimum short constraint length codes for a range of code constraint lengths and code rates. The hardware realization is done on ALTERA’S MAX 7000S FAMILY EPM7160SLC84-7 programmable logic device. The hardware implementation is done for a data of four bit length because of limited macro cells in EPM7160SLC84-7.Viterbi algorithm has got many applications due to its error detection and correction nature.. Thousands of low emitting power transmitters can operate in same band range at the same time in small areas without interfering with each other, because their carrier frequencies are coded with different patterns. The important concept to understand is that a trellis is constructed by computing the cost of being in each possible convolution encoder state at every symbol period. The data bit (0 or 1) most likely to have caused entry to each state is stored in a table. Practical realisation of decoder consists of two stages: simulation of decoder; hardware implementation. Simulation is done using ALTERA QUARTUS-II Version 6.0 using VHDL language.

Organisation of Paper

The paper consists of three levels:

- Understanding concept of Spread Spectrum
- Viterbi Algorithm
- Implementation
- Comparison of results

3. Spread Spectrum

Convolution encoding with Viterbi decoding is a powerful method for forward error correction. It has been widely deployed in many wireless communication systems to improve the limited capacity of the communication channels. The algorithm makes it possible to spread a carrier frequency over a wide area of the electromagnetic spectrum defined as spread spectrum.

A. Spread Spectrum Technique

Spectral Spreading is a technique in which a telecommunication signal is transmitted on a bandwidth considerably larger than the frequency content of the original information. Frequency hopping is a basic modulation technique used in spread spectrum signal transmission.

Spread-spectrum telecommunications is a signal structuring technique that employs direct sequence, frequency hopping, or a hybrid of these, which can be used for multiple access and/or multiple functions. This technique decreases the potential interference to other receivers while achieving privacy. Spread spectrum generally makes use of a sequential noise-like signal structure to spread the normally narrowband information signal over a relatively wideband (radio) band of frequencies.

The receiver correlates the received signals to retrieve the original information signal. Originally there were two motivations: either to resist enemy efforts to jam the communications (anti-jam, or AJ), or to hide the fact that communication was even taking place, sometimes called low probability of intercept (LPI).The core principle of spread spectrum is the use of noise-like carrier waves, and, as the name implies, bandwidths much wider than that required for simple point-to-point communication at the same data rate. In digital communications, giving data to a convolutional encoder is a kind of spectral spreading technique.

B. Convolutional Coding

In convolutional encoding n tuple of data is generated for every k tuple of inputs based on both current and K1 previous k tuples where K is called constraint length of the code. A (n, k, m) convolution code can be implemented with a k input, n output linear sequential circuit with input memory 'm'. Typically, 'n' and 'k' are small integers with k<n, but the memory order 'm' must be made large to achieve low error probabilities. The constraint length K of the code represents the number of bits in the encoder memory that effect the generation of the n output bits and is defined as K = m+1. The code rate r of the code is a measure of the code efficiency and is defines as r= k/n. Contents of first K1 shift register stages defines the encoder state. Memory register start with 0 and modulo-2 adders among the registers and input generate the encoded data. Generator polynomial defines how the adders (XOR gates) are placed. In this consider a information data of length four bits that are convolution encoded using encoder whose rate(r)=1/2 and constraint length(K)=3(see figure 1)

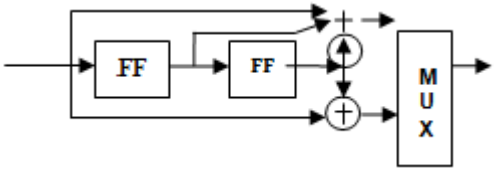

Figure 1: Block diagram view of convolutional encoder for k=3 and rate =1/2

C. Viterbi Algorithm

The decoding algorithm uses two metrics: the branch metric (BM) and the path metric (PM). The branch metric is a measure of the "distance" between what was transmitted and what was received, and is defined for each arc in the trellis. In hard decision decoding, where a sequence of digitized parity bits given, the branch metric is the Hamming distance between the expected parity bits and the received ones.

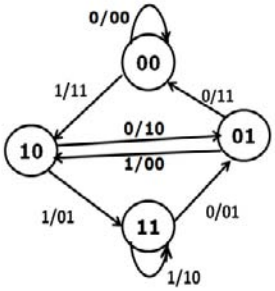

Figure 2: State diagram view of encoder for rate =1/2 and k=3

The branch metric is the Hamming distance between the expected parity bits and the received ones. The non-zero branch metrics correspond to cases when there are bit errors (figure 2).The path metric is a value associated with a state in the trellis (i.e., a value associated with each node). For hard decision decoding, it corresponds to the Hamming distance over the most likely path from the initial state to the current state in the trellis. By "most likely", mean the path with smallest Hamming distance between the initial state and the current state, measured over all possible paths between the two states. The path with the smallest Hamming distance minimizes the total number of bit errors, and is most likely when the BER is low. The key insight in the Viterbi algorithm is that the receiver can compute the path metric for a (state, time) pair incrementally using the path metrics of previously computed states and the branch metrics. All computations are based on state diagram shown in figure.

D. Computing Path Metric

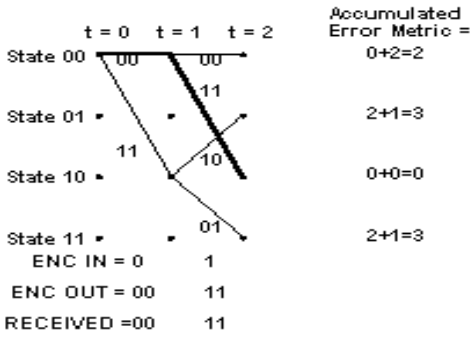

Figure 3: Computation of path metric

Suppose the receiver has computed the path metric PM[s, i] for each state s (of which there are $2k-1$, where k is the constraint length) at time step i. The value of $PM(s,i)$ is the total number of bit errors detected when comparing the received parity bits to the most likely transmitted message, considering all messages that could have been sent by the transmitter until time step i (starting from state "00", which will take by convention to be the starting state always).Among all the possible states at time step i, the most likely state is the one with the smallest path metric. If there is more than one such state, they are all equally good possibilities. Now, how do we determine the path metric at time step $i+1, PM(s,i+1)$ for each state. If the transmitter is at state s at timestep i+1, then it must have been in only one of two possible states at time step i. These two predecessor states, labelled α and β, are always the same for a given state (figure 3). In fact, they depend only

on the constraint length of the code and not on the parity functions. Any message sequence that leaves the transmitter in state$s$at time $i+1$ must have left the transmitter in state α or state β at time i. where α and β are the two predecessor states.

$$PM(s,i+1) = min[PM(\alpha,i) + BM(\alpha \to s), PM(\beta,i) + BM(\beta \to s))$$

The main loop of the algorithm consists of two main steps: calculating the branch metric for the next set of parity bits, and computing the path metric for the next column. The path-metric computation may be thought of as an add-compare-select procedure:

1. Add the branch metric to the path metric for the old state.
2. Compare the sums for paths arriving at the new state (there are only two such paths to compare at each new state because there are only two incoming arcs from the previous column).
3. Select the path with the smallest value, breaking ties arbitrarily. This path corresponds to the one with fewest errors.

E. Performance Issues

There are three important performance metrics for convolution coding and decoding:

1. State and Space needed by encoder
2. Time taken by the decoder to decode
3. Amount of reduction in the bit error rate (BER)

The amount of space is linear in K, the constraint length and the encoder is much easier to implement than the Viterbi decoder. The decoding time depends mainly on K and need to process O(2K) transitions each bit time, so the time complexity is exponential in K. In practice the decoder starts to decode bits once it has reached a time step that is a small multiple of the constraint length; experimental data suggests that is 5 · K message bit times (or thereabouts) is a reasonable decoding window, regardless of how long the parity bit stream corresponding to the message it. The reduction in error probability and comparisons with other codes is a more involved and detailed issue. The answer depends on the constraint length (generally speaking, larger K has better error correction), the number of generators (larger this number, the lower the rate, and the better the error correction), and the amount of noise.

4. Implementation

The implementation phase consists simulation of viterbi decoder using ALTERA QUARTUS-II (figure 4). The branch metric uses the Hamming distance for the four possible paths. The BMU perform simple check and select operations on the decision bits to generate the output. When the 4 possible input distance is ready, the ACS block' butterfly module adds the results and the related distance value stored in the state metric storage to get the each two paths for the 64initial states. The butterfly module is shown in the next figure. Since, each butterfly computes 4possible paths and selects the two smaller distance paths form.

For each nod (state), the ACS module selects a smaller one as the survival path and stores them to the accumulated state metric storage block and the survivor path metric. When trellis diagram is reached its finial state, the survivor path metric is been set up. In this metric, it use one bit to record all the survival states from which one of the previous state (0 is from higher path with 1 from lower path.).When the trellis diagram is finished, the trace-back module will search the ML path from the final state which is state0 to the beginning state which is state0. Each time, the trace-back block just left shifts one bit of the binary state number and add one bit from the survivor path metric to compute the previous state. After FEC block, the most likelihood path is available. The last step is to decode the original data. First step begins with time sample 0, the state0. Each time, by checking the next state in ML path and comparing it with the correspond state in the Next state table ROM, get the input data, one or zero. Thus the Maximum Likely hood path is decoded.

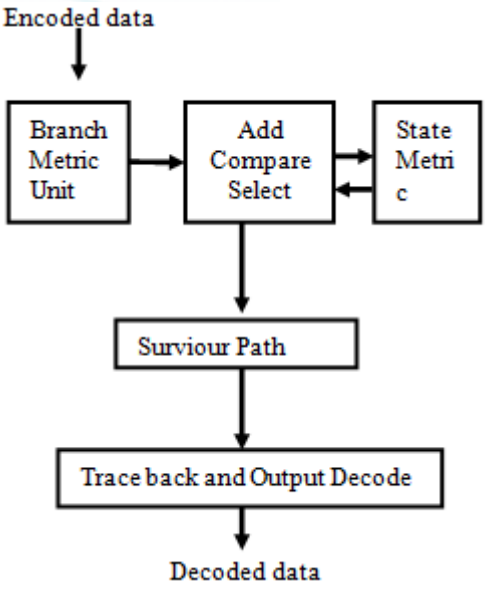

Figure 4: Logical implementation of viterbi decoder using VHDL

MAX 7000 devices offer a power-saving mode that supports low-power operation across user-defined signal paths or the entire device. This feature allows total power dissipation to be reduced by 50% or more, because most logic applications require only a small fraction of all gates to operate at maximum frequency. MAX 7000 devices can be programmed on Windows-based PCs with the Altera Logic Programmer card, the Master Programming Unit (MPU), and the appropriate device adapter. The MPU performs a continuity check to ensure adequate electrical contact between the adapter and the device. For dumping the ".pof" file select appropriate MPU (master programming unit) which is different for EPLD's. The suitable MPU for the used EPLD is ALTERAS MPU 700. After inserting device in MPU, there are three stages: BLANK-CHECK, EXAMINE, VERIFY. Now select program tab in the program window. If everything is right, programming is complete. If is shows that continuity failed at specific pin, then trim the pin at specific location of discontinuity. Repeat the same process until the EPLD gets programmed. The discontinuity has many reasons- Oxidation by environment where EPLD is kept. To decouple other sub circuits from

the effect of the sudden current demand, a decoupling capacitor can be placed between the supply voltage line and its reference (ground) next to the switched load. While the load is switched out, the capacitor charge up to full power supply voltage and otherwise does nothing.

5. Comparison of Results

The Viterbi decoder implemented here will decode up to 4 bits of input data (8 bits of convolutional encoded data).Since the number of microcells in the present EPLD are 160. For complete implementation of Viterbi decoder, a maximum of 2000 macro cells are required for decoding any length of incoming data.

First, simulation results are taken out, for practically implemented Viterbi decoder (figure 5), and obtained simulation results (figure 6) are compared with the results seen in the TEXTRONICS 714 Logical Analyser.

F. Simulation result of Viterbi decoder

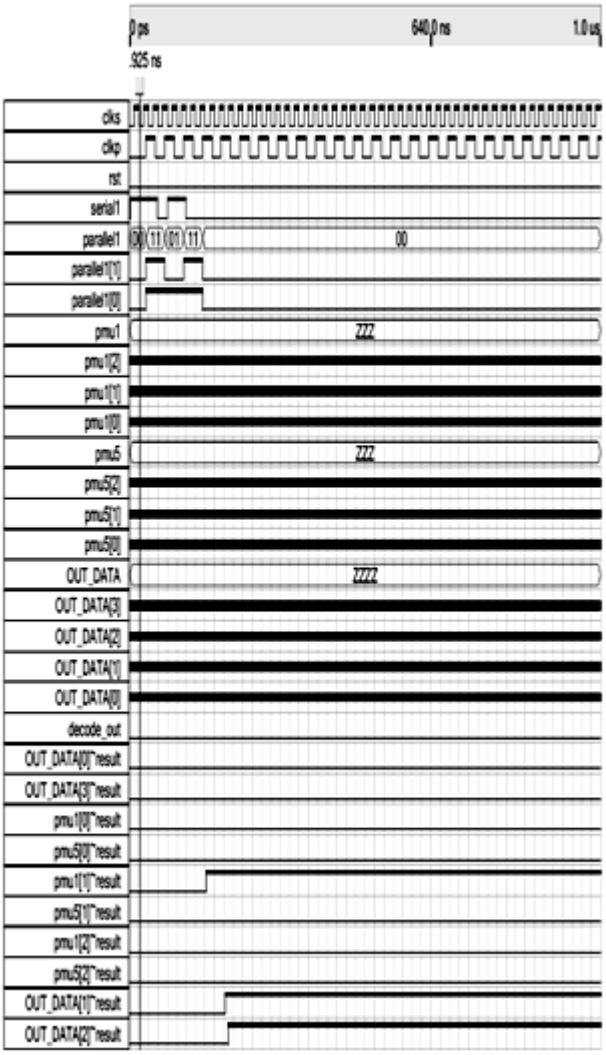

Figure 5: Simulated waveform of practical Viterbi Decoder

G. Wave forms Observed on TLA-714 Logical Analyser

Input data at before encoding: 1 0 0 0

Convolutionally encoded data: 11 10 11 00 11 10 11

Decoded data: 1 0 0 0

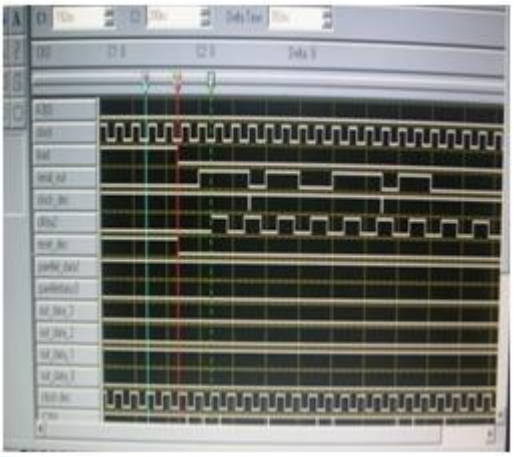

Figure 6: Wave forms observed on logical analyzer after implementation of viterbi decoder on EPM7160SLc84-7

H. Synchronization at the Receiver

A problem with Viterbi decoding occurs when a Viterbi decoder is not synchronized with received convolutional encoded data. There are two methods that can be used for synchronizing the data. A Frame sync sequence can be used to detect the beginning of the transmitted data. Normalization rate may be used to detect.

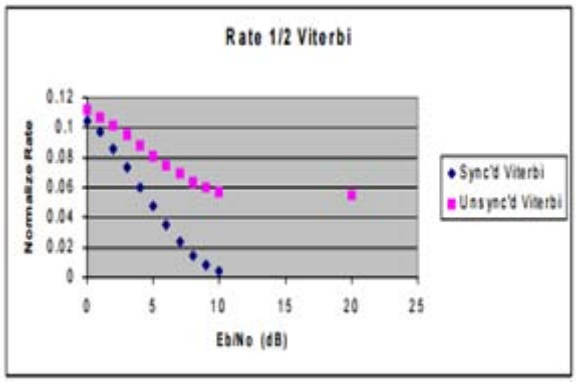

Figure 7: The plot of Normalization rate versus E_b/N_0 for rate ½ convolutional codes

Viterbi decoder synchronization status. A high normalization rate, exceeding a predetermined threshold, indicates a loss of synchronization. When the normalization rate is observed to be very high we shift by one bit and then, check if the normalization limit is high. This process is repeated as long as we observe a small normalization rate. Normalization rate threshold is dependent on Eb/No. In other words, such a normalization rate threshold does not work for all values of Eb/No. figure 8 shows the normalization rate plots versus Eb/No for a code with rate ½.

Normalization rate threshold also depends on code rate. As code rate increases, margin of error decreases, which makes normalization more problematic. Bit Error Rate (BER) may be used to detect Viterbi decoder synchronization status. BER is estimated by comparing Viterbi decoder decisions to channel hard decisions. A BER exceeding a predetermined

threshold is used to indicate loss of synchronization of a Viterbi decoder.BER threshold is dependent on Eb/No as shown in figure 16. In other words, such a BER threshold does not work for all values of Eb/No. It would be useful to provide a method and apparatus to detect when a Viterbi decoder is synchronized that is independent of both Eb/No and code rate. A method was devised that uses both normalization rate and BER to achieve this goal.

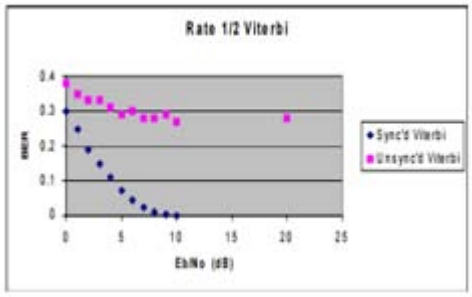

Figure 8: The plot of BER versus E_b/N_0 for rate ½ convolutional codes.

6. Conclusion

Puncturing is a technique used to make a m/n rate code from a "basic" low-rate (e.g., 1/n) code. It is reached by deletion of some bits in the encoder output. Bits are deleted according to a puncturing matrix. Simple Viterbi-decoded convolutional codes are now giving way to turbo codes, a new class of iterated short convolutional codes that closely approach the theoretical limits imposed by Shannon's theorem with much less decoding complexity than the Viterbi algorithm on the long convolutional codes that would be required for the same performance. Concatenation with an outer algebraic code (e.g., Reed-Solomon) addresses the issue of error floors inherent to turbo code designs. The turbo code decoder is based on a modified Viterbi algorithm that incorporates reliability values to improve decoding performance.

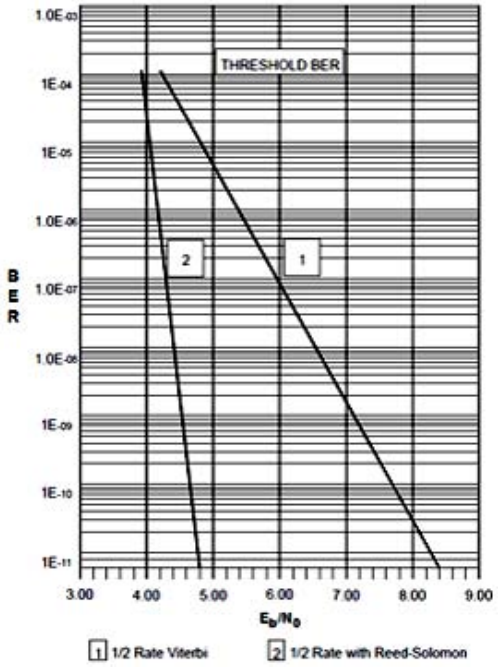

Figure 9: Combination of viterbi and RS codes performance of BER for varying E_b/N_0

Convolutional encoding with Viterbi decoding has been in use for many years in commercial satellite systems. A characteristic of the Viterbi decoding process is that as the Eb/N0 becomes increasingly smaller, the uncorrectable errors that are passed through the system are clumped together. Although the distributions of errors caused on the link are Gaussian, the uncorrectable error distribution looks like one found on a classical bursty channel. A class of codes that are well known to correct bursty errors is the Reed-Solomon code. The obvious conclusion is to concatenate the Viterbi and Reed/Solomon codes (figure 9). The attached graphs (graph 1 and graph 2) show the expected performance improvements for concatenation of 1/2 rate, 2/3 rate, and 3/4 rate Viterbi codes concatenated with the Reed/Solomon code for modems back-to-back with additive Gaussian noise. The 2/3 and 3/4 rates are achieved by puncturing the 1/2 rate convolutional code. The constraint standard K=7.

References

[1] Malepati Hazarathaiah, "Viterbi decoder Implementation using C", Digital Media Processing Algorithms using DSP Algorithms using C, 3rdEdition, Wiley India, 1998, ch.4, sec.4.3, pp.234-286.
[2] Peter.J.Ashenden, "Sequential Statements", The VHDL cook book, 1st Edition, Prentice Hall India, 1990, ch.2.4.3, pp.2.11-2.22.
[3] J.S.Lee and Miller, "Convolutional codes", CDMA Systems Engineering Hand book, 3rd Edition, Wiley India, 2004, ch.6, pp.543-659.
[4] Peter Sweeny, "Viterbi decoder", Error Control Coding from Theory to Practice, 3rd Edition, John Wiley and Sons, 2002, ch.2, pp.23-38.
[5] Robert C.Dixon, "Spread Spectrum", Spread Spectrum Systems, with commercial Applications, 3rd Edition, John Wiley and sons, 2010, ch.2, pp.54-90.
[6] Tektronix Logical Analyzer Family Usermanual, Tektronix, Beaverton. Oregon, 1998, pp.78-120.
[7] AlteraUsers.(2007).AlteraForums[online].Available:http://www.alteraforum.com/forum/forumdisplay.php?f=16
[8] Altera.(2007).MAX7000Altera[online].Available:https://www.altera.com/literature/ds/m7000.pdf

Author Profile

Mr. V. Satya. Sreekanth obtained B.Tech in Electronics and Communication Engineering from Potti Sri Ramulu College of Engineering and Technology in 2013. Currently pursuing M.Tech in Microwave and Communication Engineering from PVP Siddhartha Institute of Technology, Vijayawada and he is an intern student in NRSC Hyderabad and his interests are in periodic loading of dielectric substrates, Ultra wide band Antennas, Wireless Communications, Mathematical background of coding theory necessary for digital communications.

Dr. Y. Ramakrishna obtained Ph.D. in 2015 from JNTU Kakinada in the field of Smart Antennas for Mobile Communications.He received M.Tech Degree in Microwave Engineering from Acharya Nagarjuna University, India in

2005. Currently he is working as Associate Professor in the Department of ECE, Prasad V. Potluri Siddhartha Institute of Technology, India. He is also a Member of ISTE. His Research interest includes Smart Antennas, Antennas and Wave Propagation, Mobile Communications and Microwave Engineering.

Dr. A. V. V. Prasad did his M.Sc, Physics (Electronics) from Andhra University, in 1985. Since then he is working in NRSA in different capacities. He is actively involved in the H/W development of Frame Synchronizer units, Ultra SCSI interface units and PCI interfaces. He is involved in the installation of Remote Sensing satellite data reception systems and test systems like Advanced Front End Hardware units (AFEH), Serializer systems, Data logging systems etc., for different satellites like IRS-1C, IRS-P4, IRS-P5, IRS-P6, TES etc.,. Designed and developed a high speed Serializer used for testing satellite data reception systems and S/W. Established a data capturing facility at Arctic station Svalbard, Norway and OBSSR data processing facility at NRSC for having more global coverage of Remote sensing data for IRS-P5 and P6. Presently working on the parallel processing Software development using GP-GPUs and development of data processing S/W modules required for remote sensing. Present Heading the Microwave Remote Sensing and Global Data Processing Department (MRS & GDP) at N.R.S.C-I.S.R.O.