I. INTRODUCTION

Trellis coded modulation schemes are being used in communication systems including deep space communications. The TCM system employs convolution encoder to encode the message sequence which is being transmitted through a channel. The convolution encoding offers an alternative to block codes for transmission over a noisy channel. An advantage is that it convolution codes are widely used as error correction codes. In the present scenarios, as the numbers of users are simultaneously increasing day by day, the data transferring between the systems plays a vital role. This wide usage leads to major issues like data corruptions. So it is very necessary to reduce the data corruption by providing a suitable solution to the errors in the communication process, one such method that decodes and simultaneously correcting the codes effectively is Viterbi Algorithm. The highest recognizable algorithm for decoding the convolutional codes is Viterbi decoder.[1] The Viterbi algorithm mainly performs maximum likelihood decoding for the correction of errors caused due to the channel noise. The computational load can be reduced by taking an advantage of the special structures in the code trellis. The Viterbi is a recursive optimal solution to the problem of estimation of the state sequence of a Markov process. The Viterbi algorithm is given in Figure 1. The Viterbi algorithm was proposed by Andrew J. Viterbi in 1967 is computationally efficient technique for determining the metric of transition due to the event and selects the best for occurred event. Viterbi algorithm uses the convolutional codes. This algorithm is usually found in Information theory, Speech theory, Keyword spotting, computational linguistics and bio informatics.

II. LITERATURE REVIEW

Viterbi algorithm is an approach to find the most common sequence of hidden states. This dynamic algorithm finds the probability of the observed sequence of each combination. The algorithm works on the state machine assumption for the convolution codes. Initially both the observed and hidden events must be within the same sequence and it must resemble the time, while the next two sequences must be put together and the known even resembles the accurate one. That is the next coming step computes to most probable hidden sequence up to certain point “t” depends on the absorbed point within the sequence of “t1”. The algorithm examines forward by moving the new set of states by combining the metric of possible previous states with incremental most probable path taken through the Markov graph. The decoding process is explained by a Trellis diagram[2].

III. CONVOLUTIONAL ENCODER

Convolutional codes were first introduced by Elias in 1955. Convolution codes are a type of error correcting codes that generates parity symbols through the sliding application of the Boolean polynomial function to a data stream. The Convolution is represented by the sliding application of the encoder over the data, this gives rise to the term “convolution
coding” and this sliding nature facilitates “Trellis” decoding.

To convolutionally encode the data, we start with ‘m’ memory registers each holding 1-bit data. Initially all the memory registers are initialized with the value ‘0’. The encoder has “n” modulo-2 adders where “n” represents the number of output bits. Convolutional codes are commonly specified by three parameters those are ‘n’, ‘k’, ‘m’ where, n, k, m are the number of output bits, input bits and memory registers respectively. The code rate of the convolutional encoder is given as k/n. Often the manufacturers of convolutional code chips specify the code by parameter (n, k,L). Where “L” is the constraint length[5] and is given by L = k(m-1). The constraint length is the number of bits in the encoder memory that affect the generation of the ‘n’ output bits. The convolutional encoder with ½ code rate is shown in Figure 2.

![Figure2. Convolution Encoder.](image)

The output sequence ‘v’ can be computed by convolving the input sequence ‘u’ with the impulse response ‘g’.

\[ v_i^j = \sum_{i=0}^{m} u_{i-i} g_i^j \]

Where \( v_i^j \) is output bit from encoder ‘j’, \( u_{i-i} \) is the input bit and \( g_i^j \) is the \( i^{th} \) term in the polynomial ‘j’. From the Figure 2 \( v_1 = u_1 \ XOR \ u_0 \ XOR \ u_1 \ XOR \ u_2 \), \( v_2 = u_1 \ XOR \ u_0 \ XOR \ u_2 \).

**IV. VITERBI DECODER**

The Viterbi decoder is the most commonly used for the purpose of decoding the convolutionally encoded sequence. A typical block diagram of Viterbi decoder is given in the Figure 3. It mainly consists of a branch metric unit (BMU), add compare select unit (ACSU), path metric unit (PMU) and survivor path management unit (SMU)[3]. From the above architecture first the branch metrics are calculated from the BMU and are fed to ASCU which recursively computes the path metrics and generates the decision bits for each possible state transition and then the decision bits are stored in and retrieved from SMU[4] in order to decode the encoded bits along with the final survivor path.

![Figure3. Block Diagram of Viterbi Decoder](image)

**A. Branch Metric Unit (BMU)**

BMU calculates the entire branch Metrics (BM) from the received input symbols and it is a transition metric unit. The function of BMU is to generate corresponding merit of skip branch according to the input sequence. The branch values are nothing but the coordinate values. The distance can be calculated by the bit difference between the received and expected bits to be calculated. In another words it is the Hamming distance metric.[6]

**B. Add Compare Select Unit (ACSU)**

The branch metrics from BMU are the inputs for ACSU. It recursively computes the path metrics. The ACSU requires two adders for the addition of branch metrics with path metrics and a selector for selecting the optimum path metric. For the received sequence of ‘l’ with the code rate k/n, and total memory of ‘m’, the number of ACSU required to decode the received sequence is \( l \times 2^m \).

**C. Path Metric Unit (PMU)**

The path metrics are computed by the ACSU. The path metric unit maintains the complete track of previous computed path metrics and current path metrics to choose the path with the least merit. The butterfly structure shown in Figure 4 reveals the process of computing the Path Metrics.[6] This procedure is repeated for every encoder state.

![Figure4. Butterfly Structure of PMU.](image)
PM\(^{(i)}\) and PM\(^{(i+1)}\) are the previous computed path metrics corresponding to the states ‘i’ and ‘j’ respectively and PM\(^{p}\) and PM\(^{q}\) represents the current computed path metrics for the states p and q respectively. BM\(^{(i,p)}\) and BM\(^{(i,q)}\) are the correspond branch metrics of \((i,p)\) and \((i,q)\) states and BM\(^{(j,p)}\) and BM\(^{(j,q)}\) are the branch metrics of \((j,p)\) and \((j,q)\) states. Thus the addition of branch metrics and path metrics is done and the decision bits [8] are taken to select the survivor path.

D. Survivor Management Unit

Survivor management unit is responsible for keeping track of the information bits associated with the surviving paths. These are nothing but the decision bits from the ACSU. The SMU makes use of these decision bits to find the final survivor path and decode the source bits. These arbitral bits that are generated from ACSU will be saved in the survival path memory and will be used to search the surviving path. There are mainly two approaches for searching the survival path. They are

1. Register Exchange Method
2. Trace Back Method

V. REGISTER EXCHANGE METHOD

The register exchange method (RE) is the most significant and straightforward method to extract the information bits from the encoded bit stream. The RE method has the lowest latency and simple control circuit. This method is implemented by the connection of multiplexers and registers. The memory requirement of RE is NL bits registers. Here the ‘N’ represents the number of states where ‘L’ represents the length of the bit sequence. For the process of decoding all the NL bits are read and written, and it requires high memory access bandwidth. It consists of a two dimensional register array with multiplexers in between each two columns. The RE method is best suited for high speed and low latency applications with the constraint length less than 7. The circuit structure of RE[9] method is shown in the Figure 5. The decoding process in register exchange method is shown in Figure6

![Figure 5. Structure of RE Method.](image1)

VI. TRACE BACK METHOD

The trace back method (TB) opts the maximum likelihood algorithm. This traces back the maximum likelihood path starting from the best state. The survivor path gets traced back after the entire codeword is received and generates the decoded output sequence. In this method only ‘n’ decision bits are written in each clock cycle. When compared with the RE method, the power consumption of TB is much efficient even with the large constraint lengths. Therefore the TB method is best suited for low power applications. The TB memory stores the history of decision bits from ACSU for the purpose of tracing back. The TB method makes use of a two dimensional circular buffer, with rows and columns. Here the number of rows are equal to the number of states \(N = 2(L-1)\), ‘L’ is the constraint length, the columns store the results related to ‘N’ comparisons corresponding to the incoming coded bits at each interval. The number of columns is considered as the “Trace back depth”. Figure 7 shows the decoding in Trace back method.

![Figure 6. Decoding in RE Method.](image2)

![Figure 7. Decoding in Trace-Back Method.](image3)

VII. PROPOSED METHOD

By comparing the results of above mentioned existing techniques, the trace back technique holds good performance when compare with the register exchange method. But for larger codes the TB method has the drawback of larger latency. On the other hand the RE method with less latency requires frequent switching activity while exchanging the decoded data from one register to other register. Therefore in order to optimize the above aid parameters a Hybrid Register Exchange Method (HREM) has been proposed.
VIII. REGISTER EXCHANGE METHOD

The Hybrid register exchange method is a combination of both trace back and register exchange methods. In this method instead of saving the best single step, we are saving the best two step path to a given state. Thus we save two clock cycles for each state. In this method, the initial state is first traced back through a ‘m’ cycle and the contents in the initial state are transferred to the current state. Figure 8 shows the general working scheme of the HREM, where ‘T’ represents the stage number of the decoding algorithm and the blocks represents the registers which contains the decoded data represents the registers which contains the decoded data.

IX. SIMULATION RESULTS

To observe the speed and resource utilization, RTL is generated and verified and synthesized using Xilinx Synthesis Tool (XST) and implemented on Xilinx Virtex 6 Low-Power based XC6VLX75TL FPGA device. Figure 9 and Figure 10 shows the simulated output waveform of convolution encoder and Viterbi decoder respectively. Figure 11, Figure 12, and Figure 13 shows the RTL schematics of Viterbi decoder with trace back method, register exchange method and hybrid register exchange method. The comparison various parameters in the above mentioned three methods is given in the Table 1.

TABLE I. Comparison using XC6VLX75TLFF484-1L

<table>
<thead>
<tr>
<th>PARAMETERS</th>
<th>REGISTER EXCHANGE</th>
<th>TRACE BACK</th>
<th>HYBRID REGISTER EXCHANGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO. OF LUTS</td>
<td>7247</td>
<td>5892</td>
<td>2079</td>
</tr>
<tr>
<td>DELAY (ns)</td>
<td>12.256</td>
<td>17.03</td>
<td>18.718</td>
</tr>
<tr>
<td>POWER (mW)</td>
<td>121.56</td>
<td>98.8</td>
<td>38.8</td>
</tr>
</tbody>
</table>
Low-Power Hybrid Trace-Back Register Exchange Method of SMU for Viterbi Decoder

X. CONCLUSION

In this project a low power Viterbi decoder has been proposed and is designed by trace back and register exchange approach. The designed Viterbi decoder has been simulated using Xilinx. Synthesized with XST and implemented on Virtex 6 low power based XC6VLX75TL. The results show that in the proposed method the power has decreased to 59% of the TB technique and 82% of RE method. Hence the proposed hybrid Viterbi decoder is efficient in terms of power.

XI. REFERENCES

[14] V. S. Gierenz, O.Weiss, T. G. Noll, "A 550 Mb/s radix-4 bit-level pipelined 16-state 0.25- m CMOS Viterbi decoder," in Proc. IEEE International Conference Application-