Experiment No.: **Date:**

Title: **To study convolutional code encoding operation and verify its operation by time domain and transform domain approach.**

Learning Objectives: At the end of this experiment, students will be able to:

- generate code words for the given convolutional encoder using time domain and transform domain approach.
- apply tree diagram and state diagram approach for developing code words.

Pre-requisite: Concept of error control coding, FEC codes

Apparatus: MATLAB, Computer

Theory:

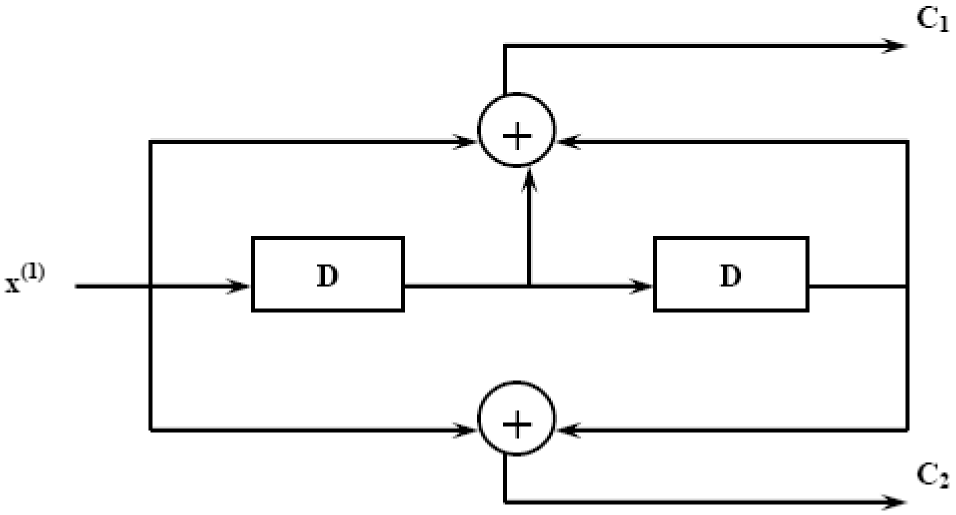
When transmitting digital information over a noisy channel, it is possible that the data bits are received in errors. There are two options open to the receiver in dealing with possible errors:
1. **Backward Error Correction** or **Automatic Repeat Request (ARQ):** Detecting the fact that the received data is erroneous and asking the transmitter to retransmit it
2. **Forward Error Correction (FEC):** Correcting the error itself. For both detection and correction of error, redundant information must be sent along with the data bits. In general, parity bits are added to the message bits to form Error Correcting/Detecting
Codes. Such techniques are known as Error Control Coding.

Convolutional codes are commonly described using two parameters: the code rate and the constraint length. The code rate, k/n, is expressed as a ratio of the number of bits into the convolutional encoder (k) to the number of channel symbols output by the convolutional encoder (n) in a given encoder cycle. The constraint length parameter, K, denotes the "length" of the convolutional encoder, i.e. how many k-bit stages are available to feed the combinatorial logic that produces the output symbols. Closely related to K is the parameter m, which indicates how many encoder cycles an input bit is retained and used for encoding after it first appears at the input to the convolutional encoder. The m parameter can be thought of as the memory length of the encoder.

Convolutional codes are widely used as channel codes in practical communication systems for error correction. The encoded bits depend on the current k input bits and a few past input bits. The main decoding strategy for convolutional codes is based on the widely used Viterbi algorithm. As a result of the wide acceptance of convolutional codes, there have been several approaches to modify and extend this basic coding scheme. Trellis coded modulation (TCM) and turbo codes are two such examples. In TCM, redundancy is added by combining coding and modulation into a single operation. This is achieved without any reduction in data rate or expansion in bandwidth as required by only error correcting coding schemes.

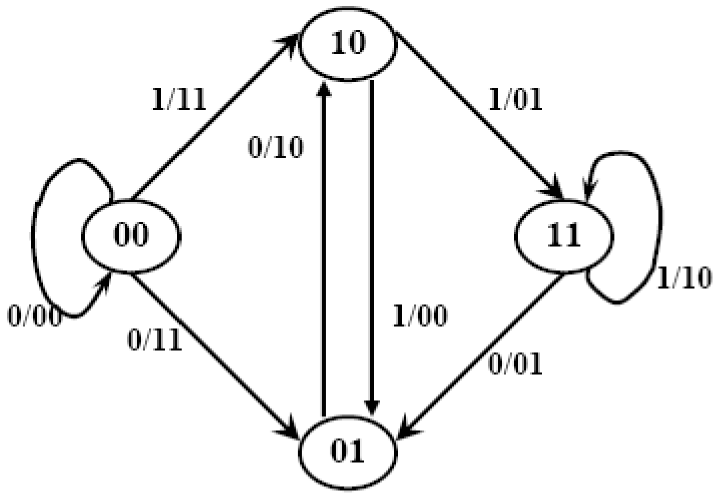
A simple convolutional encoder is shown in Figure. The information bits are fed in small groups of k-bits at a time to a shift register. The output encoded bits are obtained by modulo-2 addition

(EXCLUSIVE-OR operation) of the input information bits and the contents of the shift registers which are a few previous information bits.

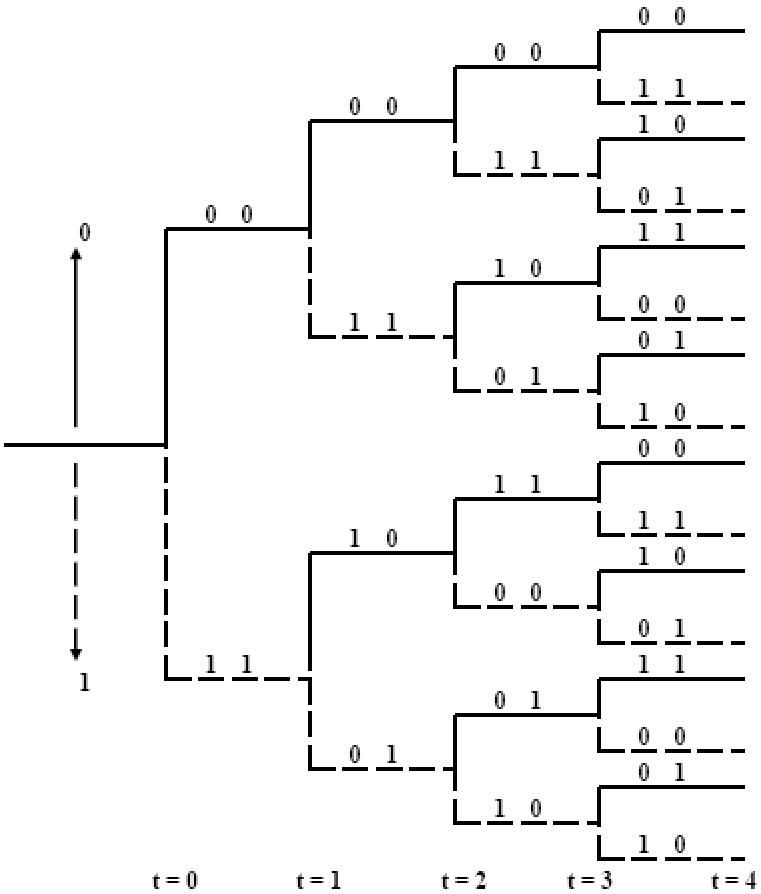


A convolutional encoder with k=1, n=2 and r=1/2

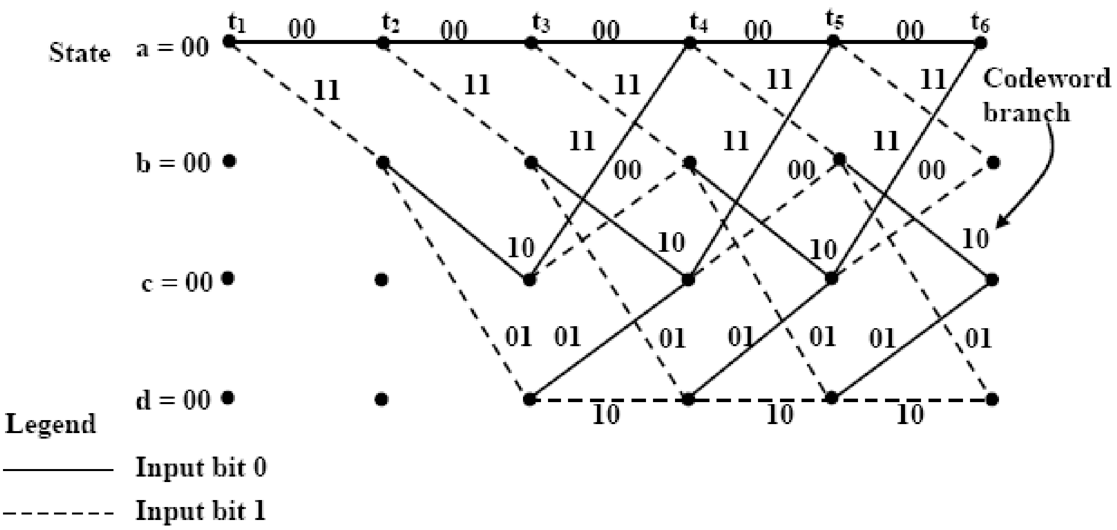
If the encoder generates a group of 'n' encoded bits per group of 'k' information bits, the code rate R is commonly defined as R = k/n. In figure, k = 1 and n = 2. The number, K of elements in the shift register which decides for how many codewords one information bit will affect the encoder output, is known as the constraint length of the code. For the present example, K = 3. The shift register of the encoder is initialized to all-zero-state before encoding operation starts. It is easy to verify that encoded sequence is 00 11 10 00 01 ….for an input message sequence of 01011…. The operation of a convolutional encoder can be explained in several but equivalent ways such as, by a) state diagram representation, b) tree diagram representation and c) trellis diagram representation.



State diagram representation for the encoder



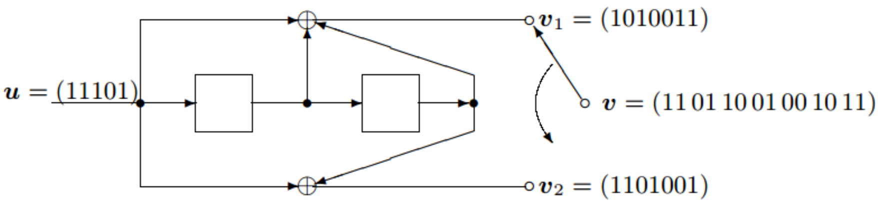
A tree diagram for the encoder



Trellis diagram for the encoder :

Time Domain approach:
The encoders of convolutional codes can be represented by linear time-invariant (LTI) systems. 3. $g^{(i)}_j$ can be found by stimulating the encoder with the discrete impulse (1, 0, 0, . . .) at the ith input and by observing the jth output when all other inputs are fed the zero sequence (0, 0, 0, . . .). The impulse responses are called generator sequences of the encoder.

Impulse Response for the Binary (2, 1, 2) Convolutional Code



$$
\begin{aligned}
g_1 &= (1,1,1,0,\ldots) = (1,1,1),\ g_2 = (1,0,1,0,\ldots) = (1,0,1)\\
v_1 &= (1,1,1,0,1) * (1,1,1) = (1,0,1,0,0,1,1)\\
v_2 &= (1,1,1,0,1) * (1,0,1) = (1,1,0,1,0,0,1)
\end{aligned}
$$

Transform Domain Approach: In a linear system, time-domain operations involving convolution can be replaced by more convenient transform-domain operations involving polynomial multiplication. Since a convolutional encoder is a linear system, each sequence in the encoding equations can be replaced by corresponding polynomial, and the convolution operation replaced by polynomial multiplication. In the polynomial representation of a binary sequence, the sequence itself is represent by the coefficients of the polynomial.

Procedure:

1. Develop MATLAB code for generating code words using time domain and transform domain approach.
2. Verify the code words generated using tree diagram and trellis diagram approach. Also prepare state diagram for the same.

Result and Discussion:

Conclusion: