Convolutional Codes

Encoder

Isha Chaudhary

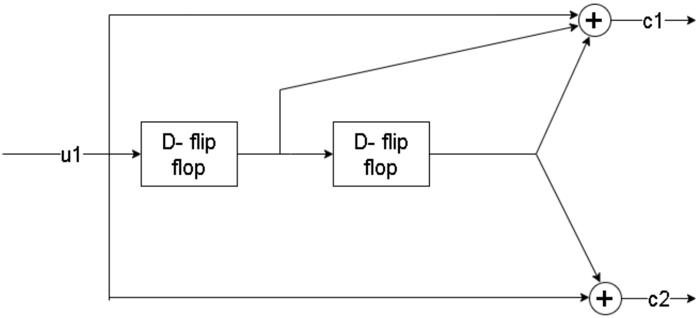
July 2020

Outline I

Basic Convolutional Encoder Design

- It takes in an input stream of message bits and outputs a stream of codeword bits.
- *Encoder State:* The encoder has a memory for some of the past input bits, which is called the encoder state.
- The output bit at a given instance t depends on the input bit and the encoder state at that instant.
- This is unlike block code encoders, where the encoder must wait for the input of the entire message before computing the codeword.
- Initial state of the encoder is set to be all-zero state.

Encoder Design using Shift registers



A rate 1/2 non-systematic non-recursive binary convolutional encoder

Encoder Design using Shift Registers

- Shift registers (with D flip flops) are used to store encoder state.
- The design rate of the encoder is $\frac{1}{m}$, where m is the number of output streams, with one input stream.
- In order to form a codeword, ending with zero termination, the Zero-terminated rate becomes:

$$R = \frac{k}{mk + m\mu}$$

  where k is the number of input bits, m is the number of output bits and μ is the number of encoder states. Although the number of inputs required to cause 0 termination are k + μ, having μ 0s appended, k message containing bits are considered for the rate.
- Generally, we use convolutional codes for small codes. So the zero-termination requirement (to provide for the initial condition for the next encoding), deteriorates the rate of encoding.

Impulse response of the Convolutional Encoder

- The results of impulse input are evaluated as impulse response at each output stream.
- For any other input, impulse response is convolved with the input, to get the output.
- Impulse input is: u = 100000..0 (k bits)
- For the above figure on slide 4, considering k = 3, u = 100, $g^{(1)} = 111 \textit{ and } g^{(2)} = 101$ are the impulse responses at the first and second output streams respectively.
- For any other input, u, the outputs at the two output streams are $v^{(1)} = u \circledast g^{(1)}$ and $v^{(2)} = u \circledast g^{(2)}$
- The codeword is formed by interleaving the two/more output streams as:

$$C = [v_1^{(1)} v_1^{(2)} v_2^{(1)} v_2^{(2)} \ldots v_{mk}^{(1)} v_{mk}^{(2)} \ldots v_{mk+m\mu}^{(1)} v_{mk+m\mu}^{(2)}]$$

+

D Transform

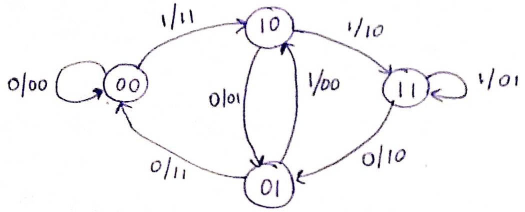
- By performing the D tranform, we can convert convolution to polynomial multiplication.
- For the example on page 4,

$$u(D) = 1$$

$$G^{(1)}(D) = 1 + D + D^2$$
$$G^{(2)}(D) = 1 + D^2$$
$$V^{(1)}(D) = u(D)G^{(1)}(D)$$
$$V^{(2)}(D) = u(D)G^{(2)}(D)$$

- Zero-termination enables the polynomial representation.

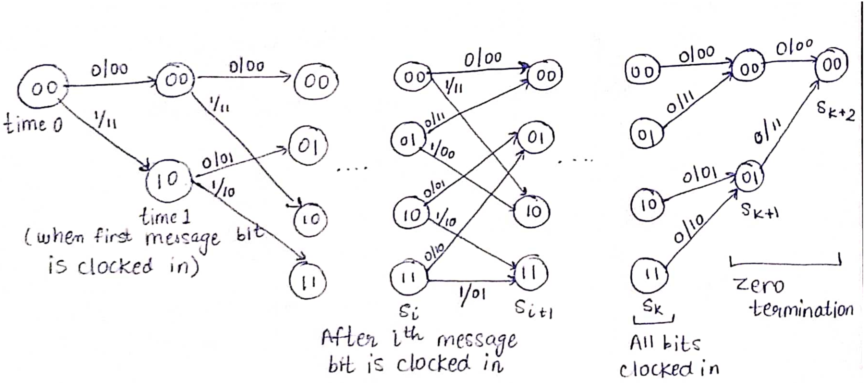
State Diagram



- The above state diagram is for the encoder on page 4.
- Such a state diagram has 2^{μ} states, with 2 branches in and out each. (μ is the number of encoder states.)
- Paths on the state diagram correspond to message/codeword (denoted as the output).
- Here, the number of message bits or the time of their occurence are not accounted for.

Trellis Diagram

For the encoder on page 4



Trellis Diagram

- This representation accounts for the time at which message bits clock in.
- Number of instants on the trellis are $k + \mu + 1$.
- μ bits are needed to reach the full 2^{μ} state list in the beginning and to converge the full state list during zero termination in the end.
- Each path on the trellis corresponds to a message/codeword. So there are 2^{k} paths on the trellis.
- Rate $= \frac{k}{m(k+\mu)} \approx \frac{1}{m}$, for large k.
- Complexity of trellis $= 2^{\mu}$.