Digital Communications I: Modulation and Coding Course

Spring- 2015

Jeffrey N. Denenberg

Lecture 7: Convolutional codes

Last time, we talked about:

- Channel coding
- Linear block codes
  - The error detection and correction capability
  - Encoding and decoding
  - Hamming codes
  - Cyclic codes

Today, we are going to talk about:

- Another class of linear codes, known as Convolutional codes.
- We study the structure of the encoder.
- We study different ways for representing the encoder.

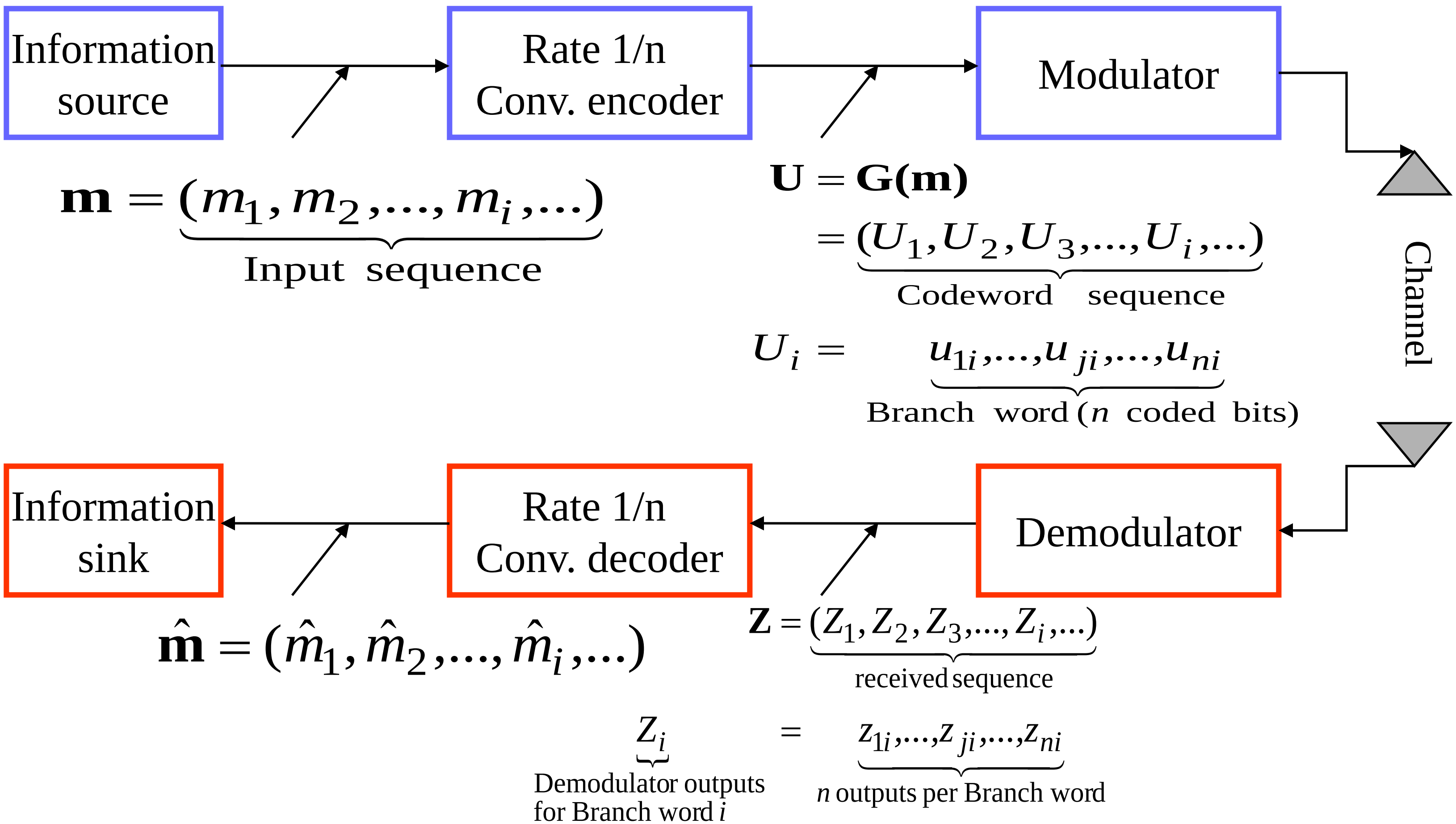
Convolutional codes

- Convolutional codes offer an approach to error control coding substantially different from that of block codes.
  - A convolutional encoder:
    - encodes the entire data stream, into a single codeword.
    - does not need to segment the data stream into blocks of fixed size (*Convolutional codes are often forced to block structure by periodic truncation*).
    - is a machine with memory.
- This fundamental difference in approach imparts a different nature to the design and evaluation of the code.
  - Block codes are based on algebraic/combinatorial techniques.
  - Convolutional codes are based on construction techniques.

Convolutional codes-cont'd

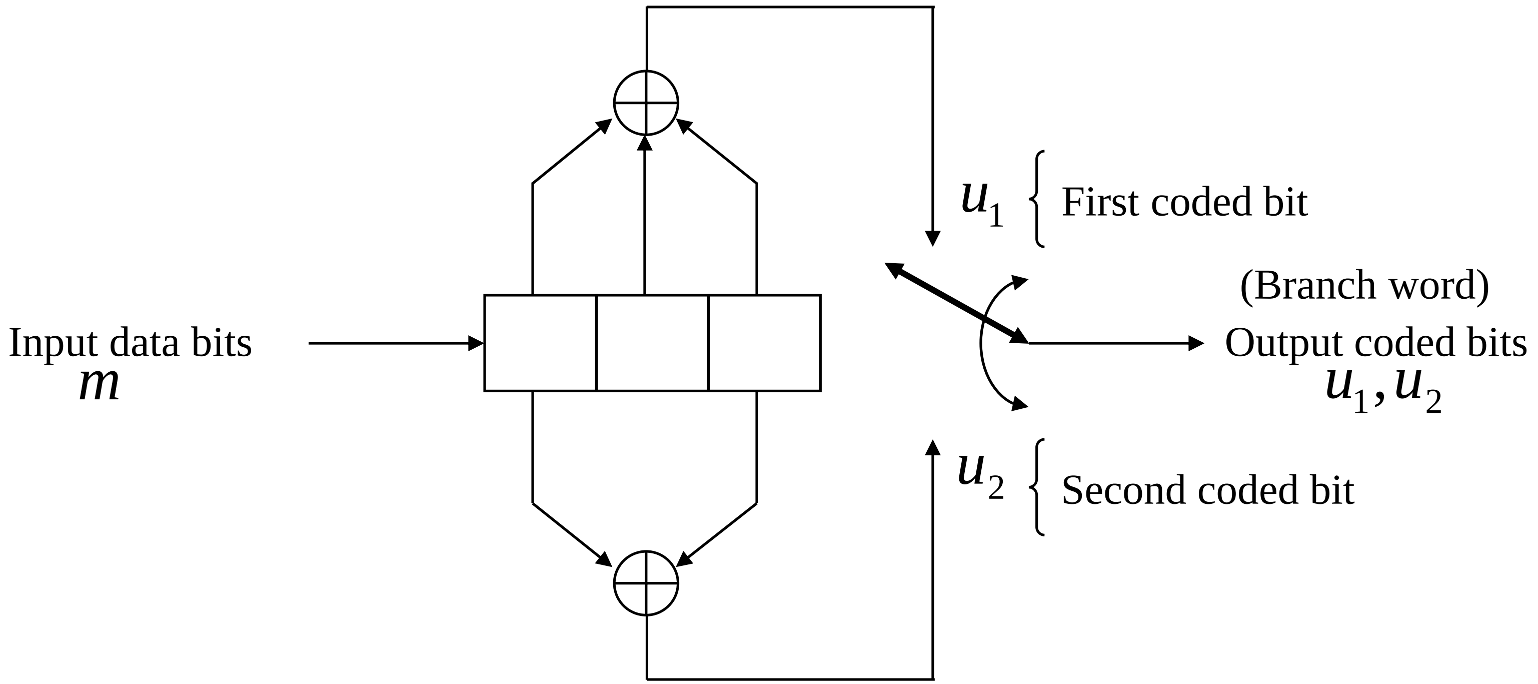
- A Convolutional code is specified by three parameters (n,k,K) or $(k/n,K)$ where
  - $R_c = k/n$ is the coding rate, determining the number of data bits per coded bit.
    - In practice, usually $k=1$ is chosen and we assume that from now on.
  - K is the constraint length of the encoder a where the encoder has K-1 memory elements.
    - There is different definitions in literatures for constraint length.

Block diagram of the DCS



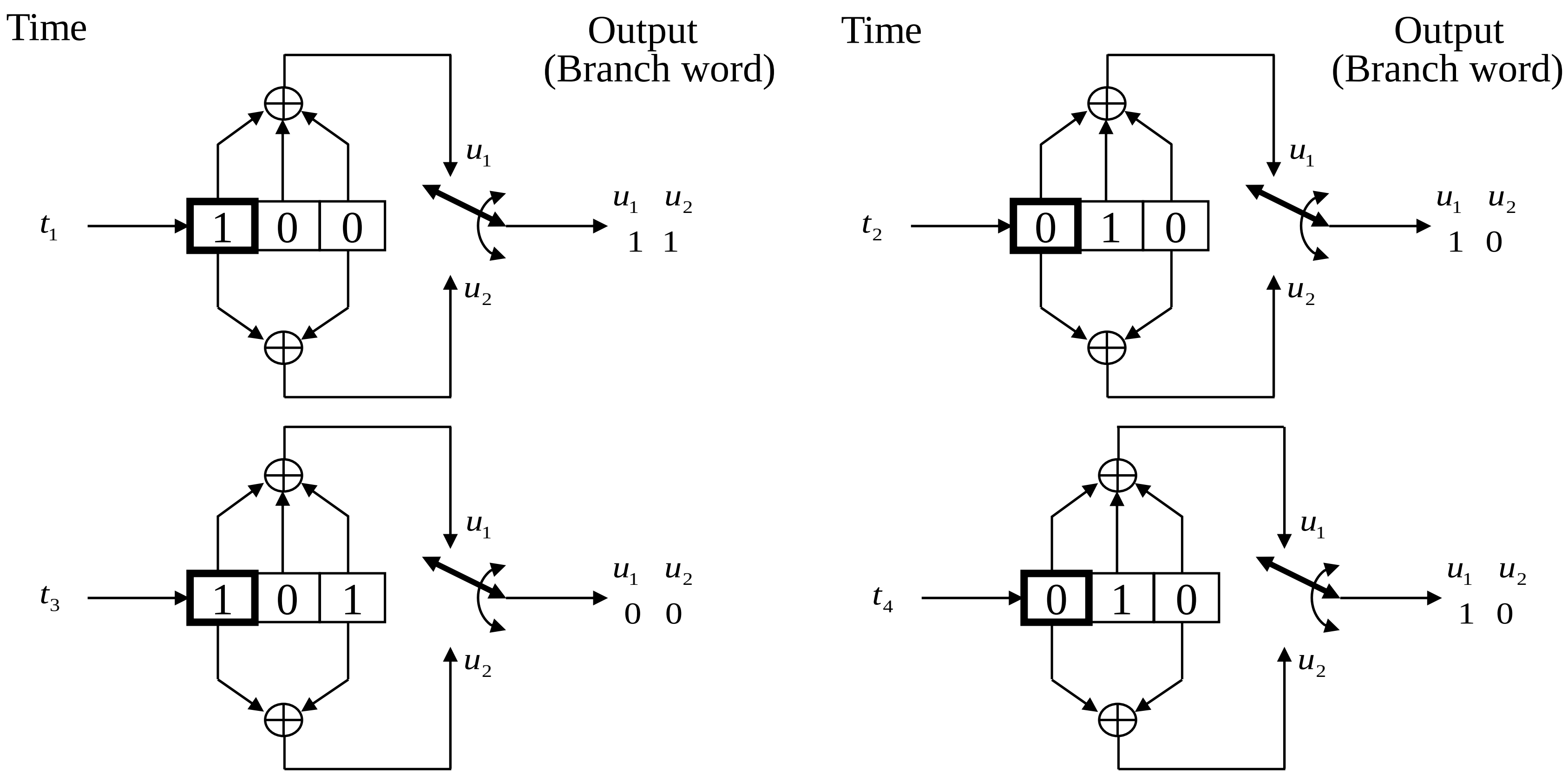
A Rate ½ Convolutional encoder

- Convolutional encoder (rate ½, K=3)
  - 3 shift-registers where the first one takes the incoming data bit and the rest, form the memory of the encoder.

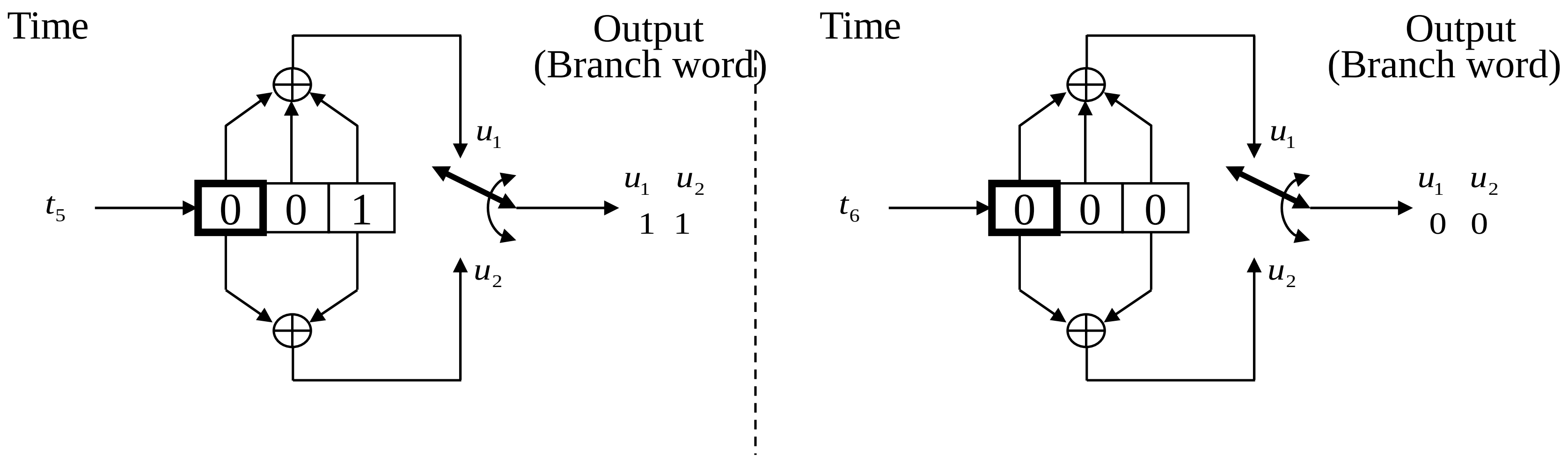


A Rate ½ Convolutional encoder

Message sequence: $\mathbf{m} = (101)$



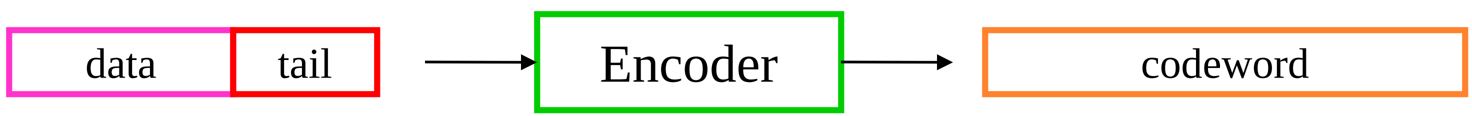
A Rate ½ Convolutional encoder



$\mathbf{m} = (101)$ → Encoder → $\mathbf{U} = (11\ \ 10\ \ 00\ \ 10\ \ 11)$

Effective code rate

- Initialize the memory before encoding the first bit (all-zero)
- Clear out the memory after encoding the last bit (all-zero)
  - Hence, a tail of zero-bits is appended to data bits.



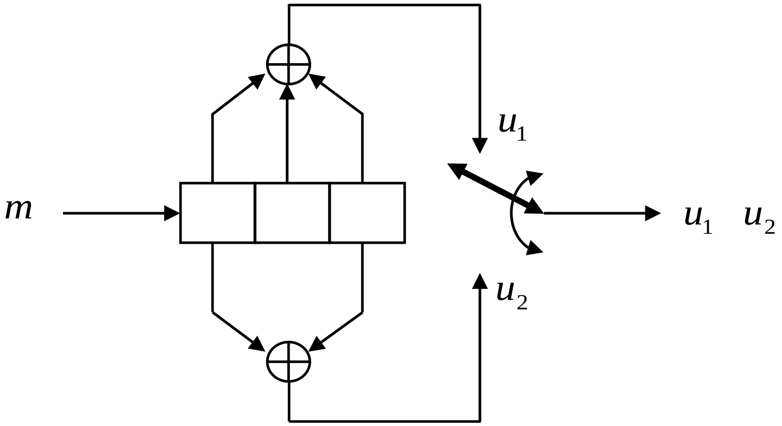
- Effective code rate :
  - L is the number of data bits and $k=1$ is assumed:

$$R_{eff} = \frac{L}{n(L+K-1)} < R_c$$

Encoder representation

- Vector representation:
  - We define n binary vector with K elements (one vector for each modulo-2 adder). The i:th element in each vector, is “1” if the i:th stage in the shift register is connected to the corresponding modulo-2 adder, and “0” otherwise.
    - Example:

$$\mathbf{g}_1 = (111)$$
$$\mathbf{g}_2 = (101)$$



Encoder representation – cont'd

- Impulse response representaiton:
  - The response of encoder to a single "one" bit that goes through it.
    - Example:

Register contents	Branch word u_1	u_2
100	1	1
010	1	0
001	1	1

Input sequence : 1 0 0

Output sequence : 11 10 11

Input **m**	Output				
1	11	10	11		
0		00	00	00	
1			11	10	11
Modulo-2 sum:	11	10	00	10	11

Encoder representation – cont'd

- Polynomial representation:
  - We define n generator polynomials, one for each modulo-2 adder. Each polynomial is of degree *K-1* or less and describes the connection of the shift registers to the corresponding modulo-2 adder.
    - Example:

$$\mathbf{g}_1(X) = g_0^{(1)} + g_1^{(1)}.X + g_2^{(1)}.X^2 = 1 + X + X^2$$

$$\mathbf{g}_2(X) = g_0^{(2)} + g_1^{(2)}.X + g_2^{(2)}.X^2 = 1 + X^2$$

The output sequence is found as follows:

$$\mathbf{U}(X) = \mathbf{m}(X)\mathbf{g}_1(X) \text{ interlaced with } \mathbf{m}(X)\mathbf{g}_2(X)$$

Encoder representation –cont'd

In more details:

$$\mathbf{m}(X)\mathbf{g}_1(X) = (1+X^2)(1+X+X^2) = 1+X+X^3+X^4$$

$$\mathbf{m}(X)\mathbf{g}_2(X) = (1+X^2)(1+X^2) = 1+X^4$$

$$\mathbf{m}(X)\mathbf{g}_1(X) = 1+X+0.X^2+X^3+X^4$$

$$\mathbf{m}(X)\mathbf{g}_2(X) = 1+0.X+0.X^2+0.X^3+X^4$$

$$\mathbf{U}(X) = (1,1)+(1,0)X+(0,0)X^2+(1,0)X^3+(1,1)X^4$$

$$\mathbf{U} = 11 \quad 10 \quad 00 \quad 10 \quad 11$$

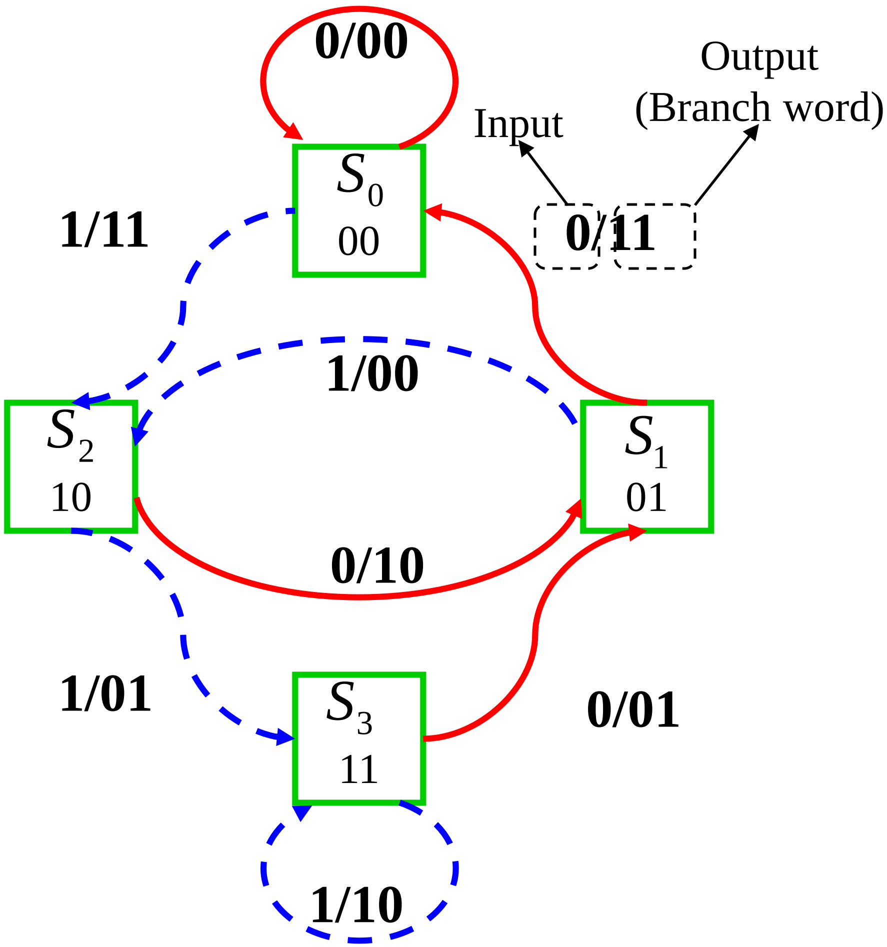
State diagram

- A finite-state machine only encounters a finite number of states.
- State of a machine: the smallest amount of information that, together with a current input to the machine, can predict the output of the machine.
- In a Convolutional encoder, the state is represented by the content of the memory.
- Hence, there are 2^{K-1} states.

State diagram – cont'd

- A state diagram is a way to represent the encoder.
- A state diagram contains all the states and all possible transitions between them.
- Only two transitions initiating from a state
- Only two transitions ending up in a state

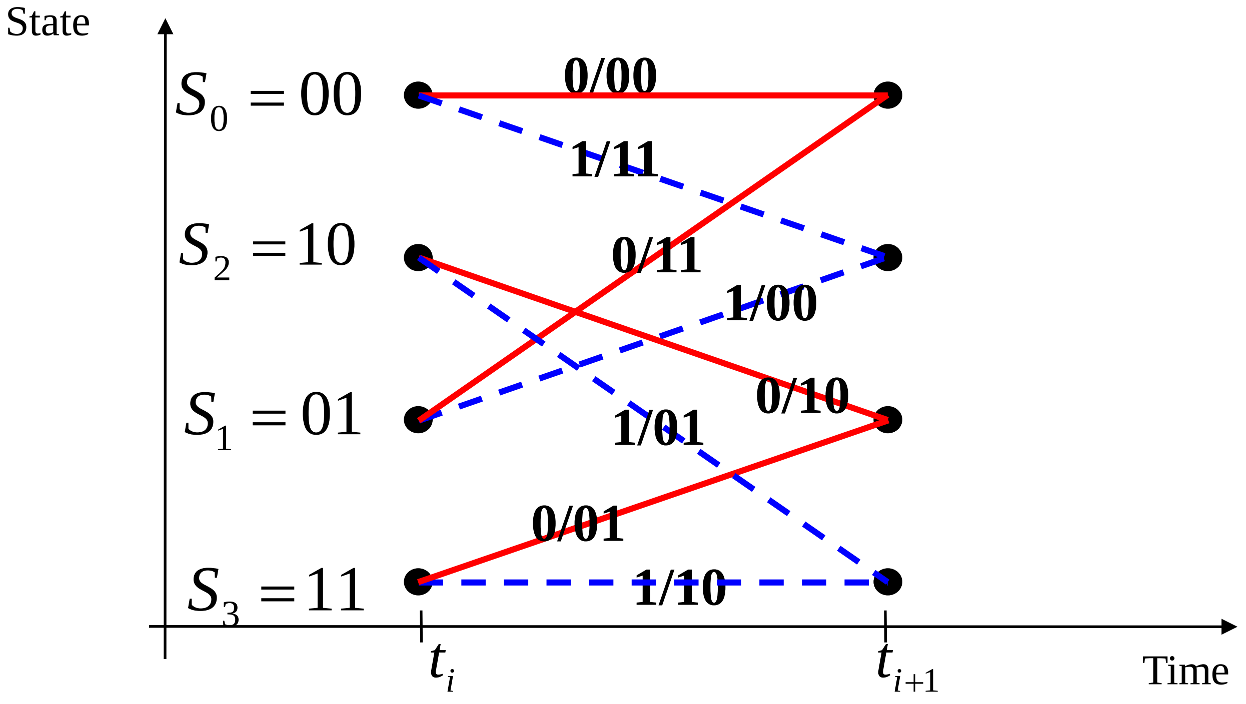
State diagram – cont'd



Current state	input	Next state	output
S_0 00	0	S_0	00
	1	S_2	11
S_1 01	0	S_0	11
	1	S_2	00
S_2 10	0	S_1	10
	1	S_3	01
S_3 11	0	S_1	01
	1	S_3	10

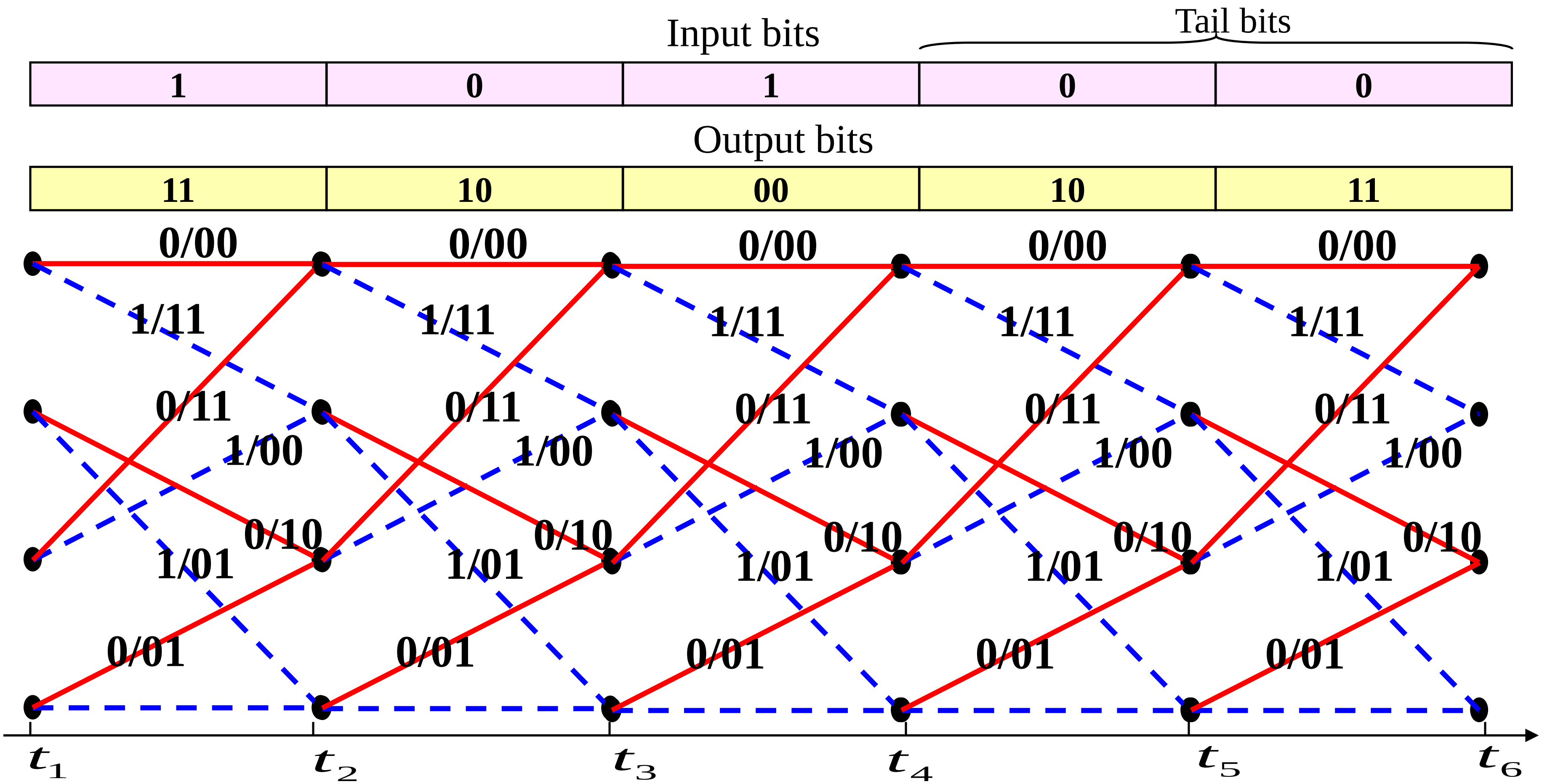
Trellis – cont'd

- Trellis diagram is an extension of the state diagram that shows the passage of time.
  - Example of a section of trellis for the rate ½ code



Trellis –cont'd

- A trellis diagram for the example code



Trellis – cont'd