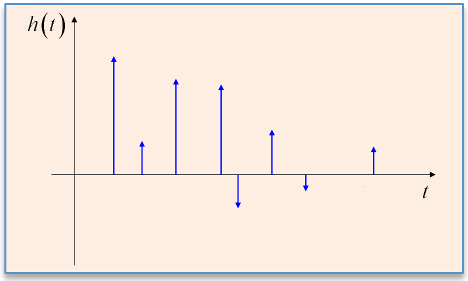
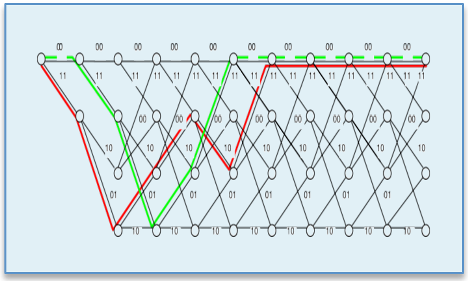
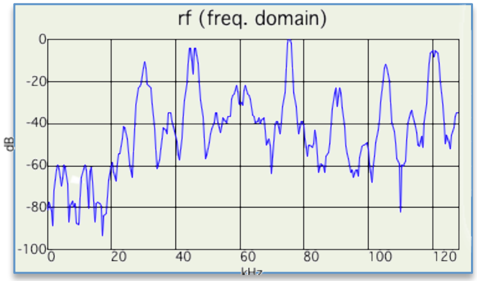
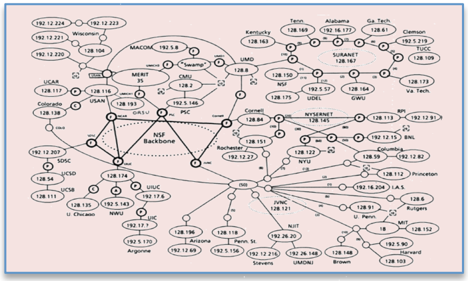








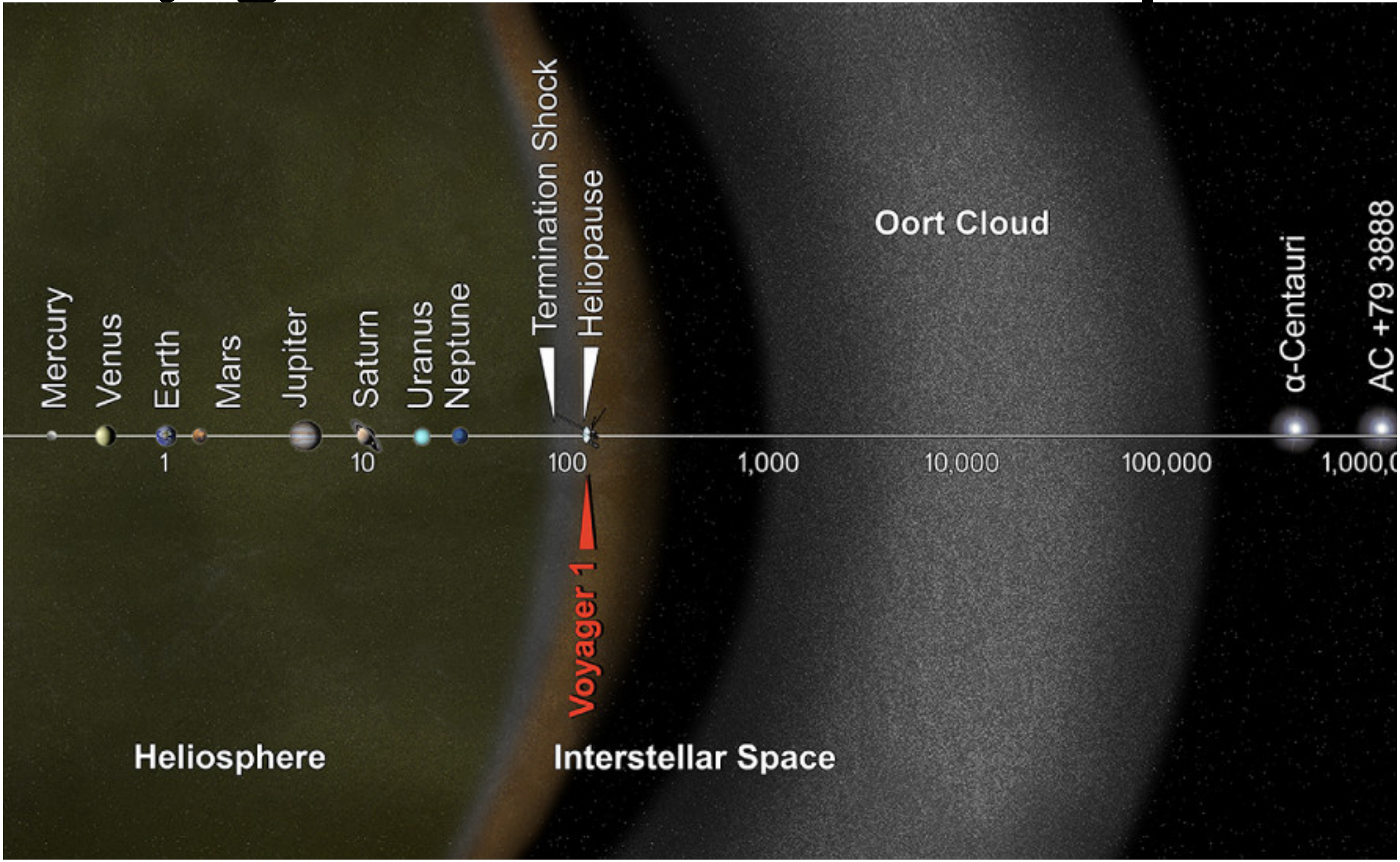
INTRODUCTION TO EECS II

DIGITAL COMMUNICATION SYSTEMS

6.02 Fall 2013
Lecture #6

- Convolutional codes
- State-machine view & trellis

September 2013: Voyager 1 "enters interstellar space"

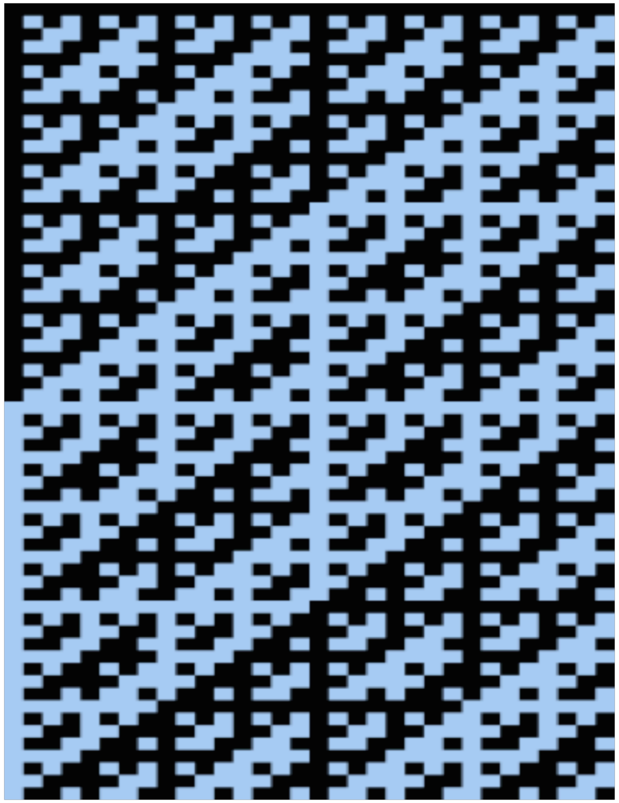


NASA/JPL-Caltech

Error Control Codes for Interplanetary Space Probes

- Early Mariner probes, 1962-1967 (Mars, Venus) – no ECC
- Later Mariner and Viking probes, 1969-1976 (Mars, Venus) – linear block codes, e.g.,

(32,6,**16**) bi-orthogonal or Hadamard linear code: the all-0 word, the all-1 word, and the other codewords all have sixteen 0’s, sixteen 1’s. The complement of each codeword is a codeword.



Bi-orthogonal/Hadamard Codes

- e.g., used on Mariner 9 (1971, Mars orbit) to correct picture transmission errors.
  - Data word length: k=6 bits, for 64 grayscale values.
  - Usable block length n around 30 bits. Could have done 5-repetition code, but comparable rate with better error correction from a

    [32, 6, 16] Hadamard code.
  - Used through the 1980's.
- The efficient decoding algorithm was an important factor in the decision to use this code.
- More generally for such codes,

  n=2^(k-1), d=2^(k-2)

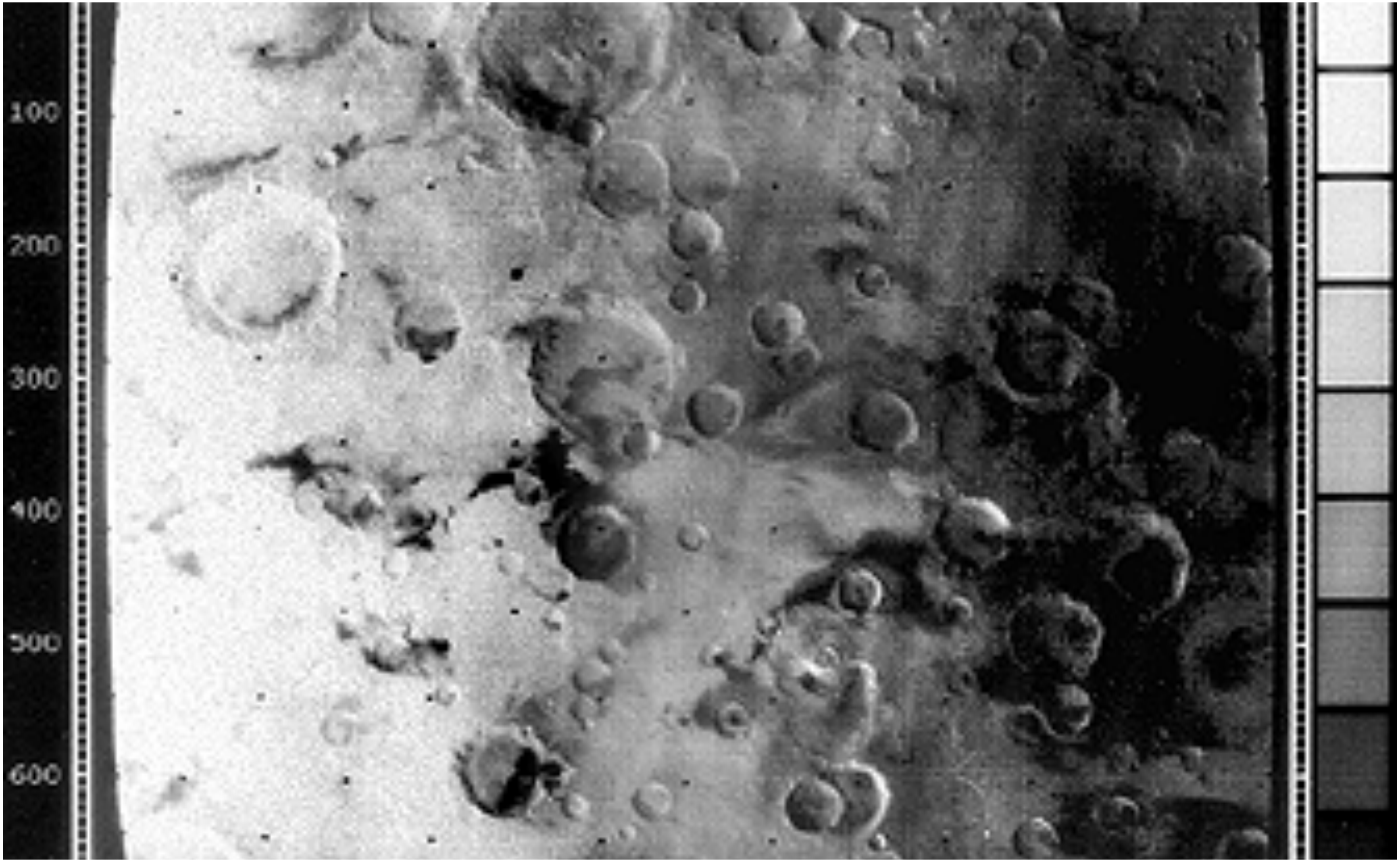
Mariner 9 (400 million km trip)

- “Spacecraft control was through the central computer and sequencer which had an onboard memory of 512 words. The command system was programmed with 86 direct commands, 4 quantitative commands, and 5 control commands. Data was stored on a digital reel-to-reel tape recorder. The 168 meter 8-track tape could store 180 million bits recorded at 132 kbits/s. Playback could be done at 16, 8, 4, 2, and 1 kbit/s using two tracks at a time.

- Telecommunications were via dual S-band 10 W/20 W transmitters and a single receiver through the high gain parabolic antenna, the medium gain horn antenna, or the low gain omnidirectional antenna.”

(NASA)

7329 images, e.g.:

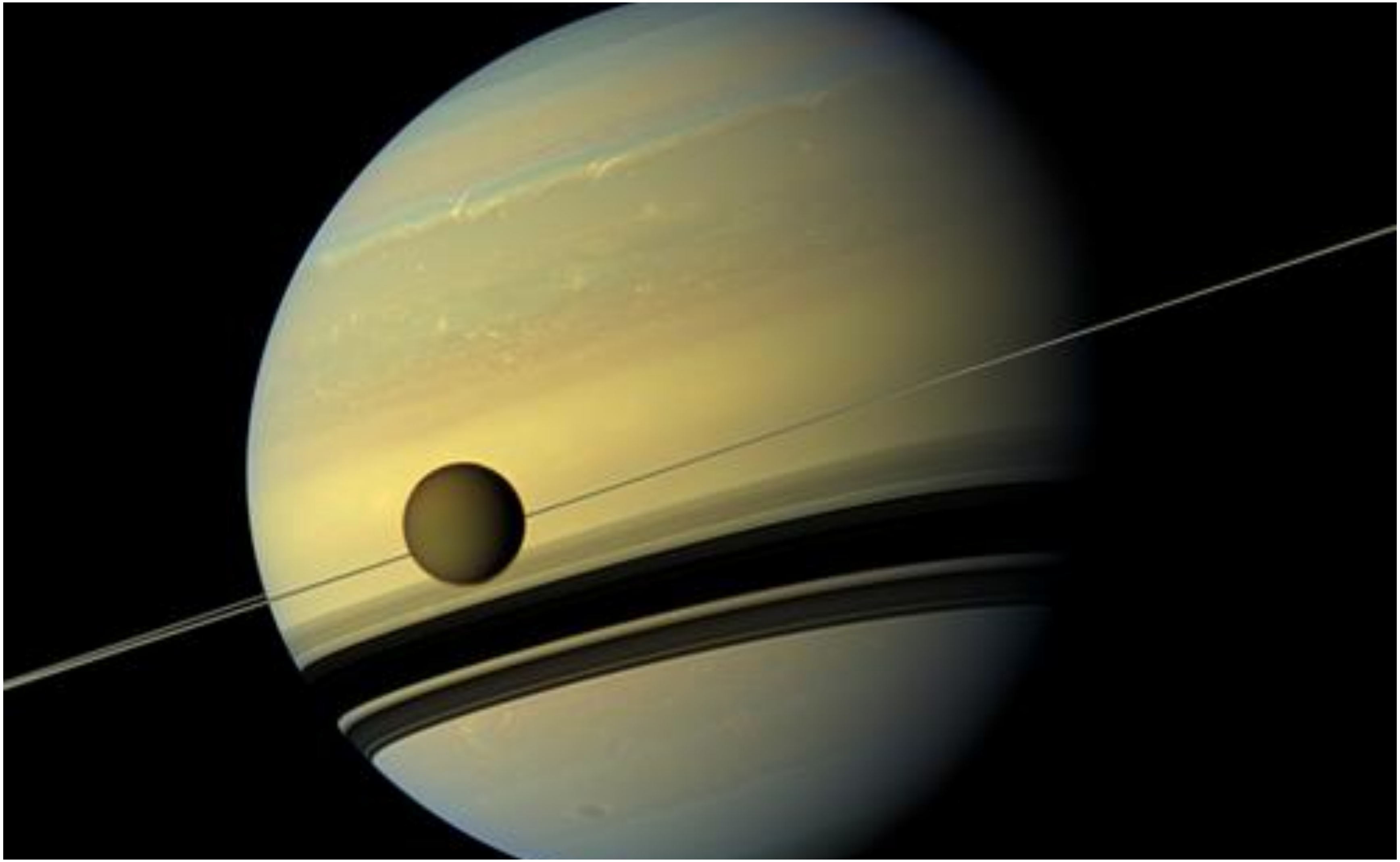


More powerful codes needed for higher data rates with limited transmitter power

- Space probe may have a 20W transmitter to cover tens of billions of kilometers (Voyager 1 is now nearly 19 billion km from sun!)
  - Part of the secret is the antenna --- directs the beam to produce the same received intensity as an omnidirectional antenna radiating in the megawatts
  - Also "cryogenically-cooled low-noise amplifiers, sophisticated receivers, and data coding and error-correction schemes. These systems can collect, detect, lock onto, and amplify a vanishingly small signal that reaches Earth from the spacecraft, and can extract data from the signal virtually without errors." (JPL quote)
- Convolutional codes with Viterbi* decoding – Voyager (1977) onwards, Cassini, Mars Exploration Rover, …

*Andrew Viterbi, MIT VI-A, USC professor, Qualcomm cofounder, ...

Saturn and Titan from Cassini, August 29, 2012



Cassini ECC

- QUESTION: What kind of error-correcting code(s) will be used in the data transmission of the Cassini orbiter to Earth?

- ANSWER from FAQ on June 3, 1999:

1) A convolutional code, either a (k=7,r=1/2) or (k=15,r=1/6) code. Compared to an uncoded channel, the k=7,r=1/2 code is 4.5 dB better; and the k=15,1/6 code is 2 dB better than the k=7, r=1/2. The convolutional code typically provides a BER (bit error rate) of 1 per 200.

d=33; all in 8-bit bytes, not bits!

2) A Reed-Solomon code, which is a block code (255,223), is also used in combination with the convolutional code (that is, the spacecraft first does the Reed-Solomon encoding of science data, and then does the convolutional encoding of Reed-Solomon symbols; we call this arrangement a concatenated coding scheme). The BER of the concatenated code is 1 per million or better, which is what the Cassini project needs.

(NASA)

… but the huge application now is in terrestrial communication

- Even back in 2005, David Forney* writes that

"VA** decoders are currently used in about one billion cellphones, which is probably the largest number in any application. However, the largest current consumer of VA processor cycles is probably digital video broadcasting. A recent estimate at Qualcomm is that approximately 10^15 bits per second are now being decoded by the VA in digital TV sets around the world, every second of every day."

*Forney (MIT PhD and MIT professor), "The Viterbi Algorithm: A personal history"

**Viterbi Algorithm

Convolutional Codes (Peter Elias, 1955 – MIT EECS faculty)

- Like the block codes discussed earlier, but act on message bits streaming into the encoder.
- Send parity bits computed from sliding window of message bits
  - Unlike block codes, generally don't send message bits, send only the parity bits! (i.e., "non-systematic")
  - The code rate of a convolutional code tells you how many parity bits are sent for each message bit. We'll mostly be talking about rate 1/*r* codes, i.e., r parity bits/message bit.
  - Use a sliding window to select which message bits are participating in the parity calculations. The width of the window (in bits) is called the code's constraint length **K**.

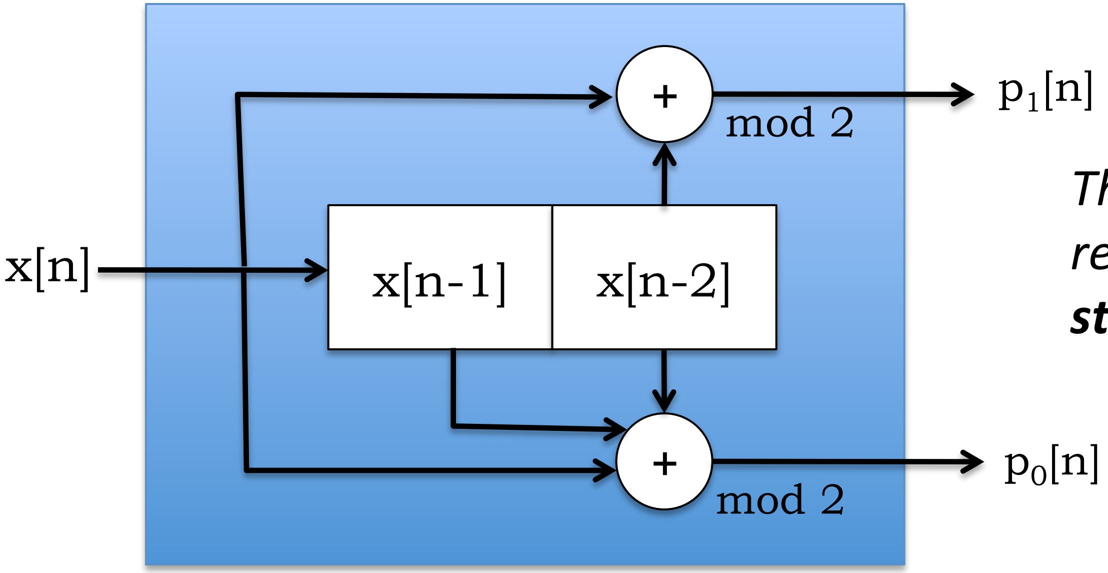
$$p_0[n] = x[n] + x[n-1] + x[n-2]$$

$$p_1[n] = x[n] + x[n-2]$$

Addition mod 2
(aka XOR)

Shift-Register View

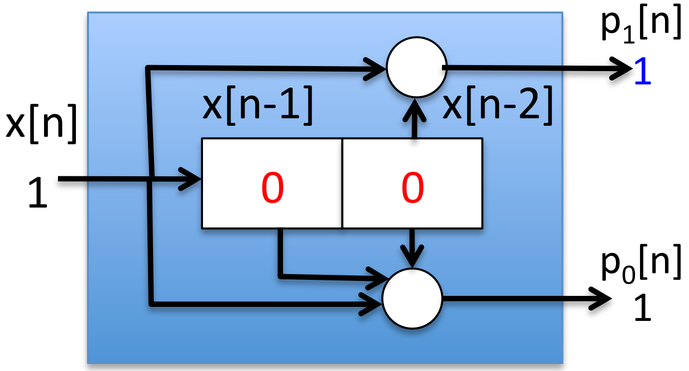
- One often sees convolutional encoders described with a block diagram like the following:



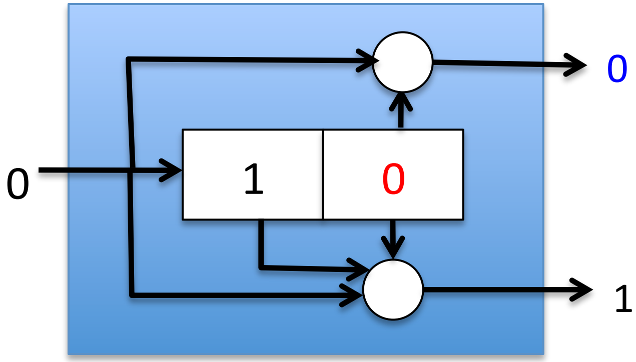
The values in the registers define the ***state*** *of the encoder*

- Message bit in, parity bits out
  - Input bits arrive one-at-a-time from the left
  - The box computes the parity bits using the incoming bit and the *K*-1 previous message bits
  - At the end of the bit interval, the saved message bits are *shifted right* by one, and the incoming bit moves into the left position.

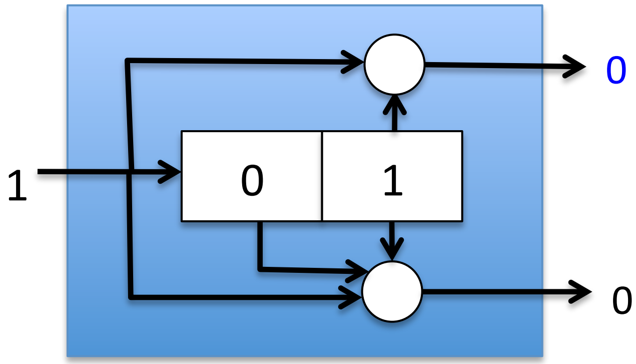
Example: Transmit message 1011



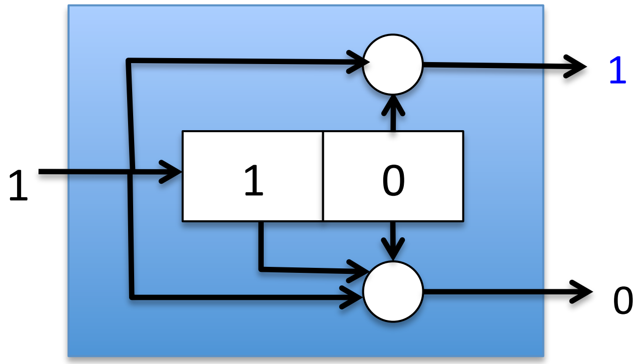
Processing x[0]



Processing x[1]



Processing x[2]



Processing x[3]

$p_0[n] = x[n] + x[n-1] + x[n-2]$

$p_1[n] = x[n] + x[n-2]$

Xmit seq: 1, 1, 1, 0, 0, 0, 0, 1, …
(codeword)

Parity Bit Equations

- A convolutional code generates sequences of parity bits from sequences of message bits by a convolution operation:

$$p_i[n] = \left(\sum_{j=0}^{K-1} g_i[j]x[n-j] \right) \bmod 2$$

- *K* is the constraint length of the code
  - The larger *K* is, the more times a particular message bit is used when calculating parity bits
    → greater redundancy
    → *better error correction possibilities (usually, though not always)*

- g_i is the *K*-element generator for parity bit p_i.
  - Each element $g_i[j]$ is either 0 or 1
  - More than one parity sequence can be generated from the same message; the simplest choice is to use 2 generator polynomials

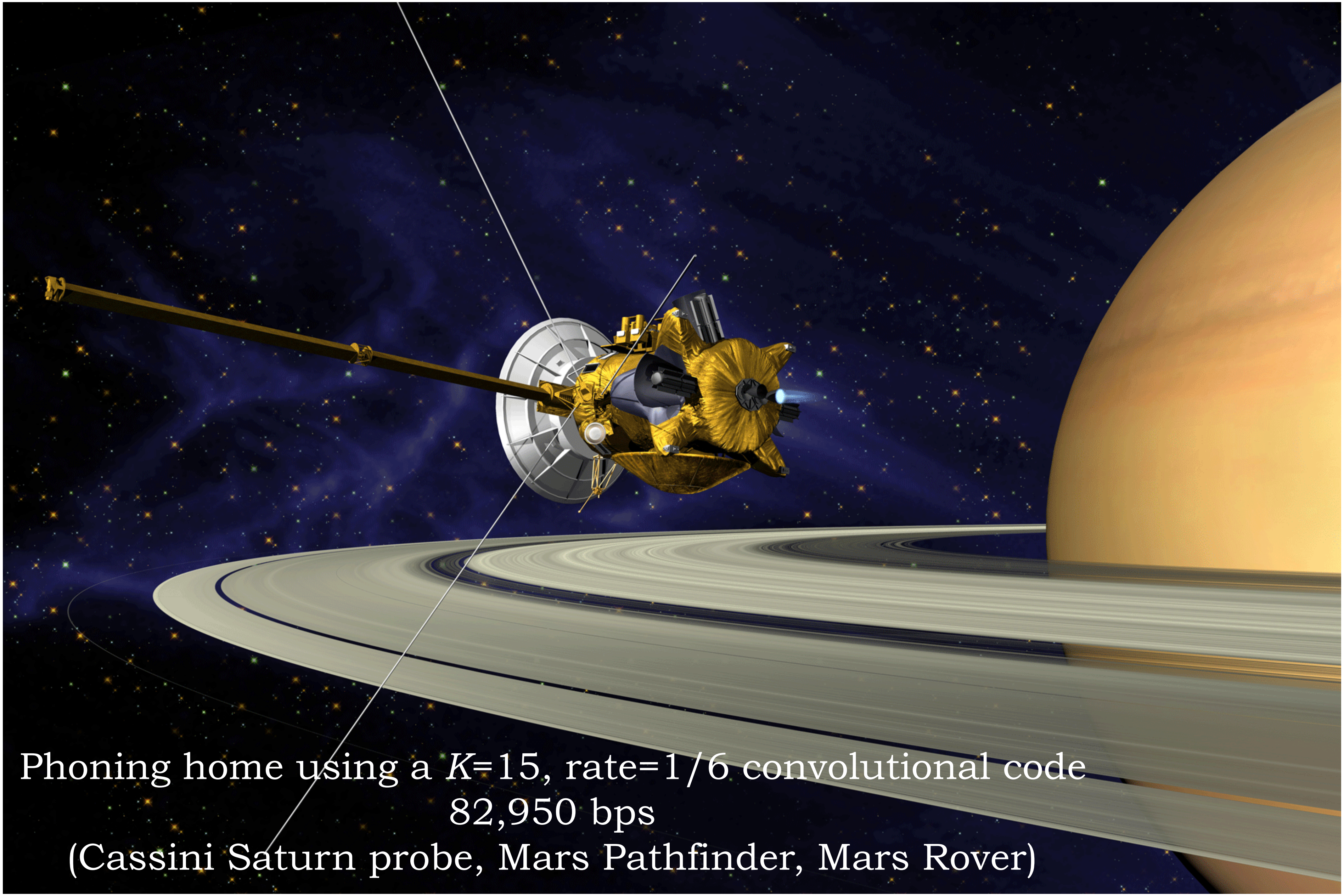
Transmitting Parity Bits

- We transmit the parity sequences, not the message itself
  - As we'll see, we can recover the message sequences from the parity sequences
  - Each message bit is "spread across" *K* elements of each parity sequence, so the parity sequences are better protection against bit errors than the message sequence itself

- If we're using multiple generators, construct the transmit sequence by interleaving the bits of the parity sequences:

$$xmit = p_0[0], p_1[0], p_0[1], p_1[1], p_0[2], p_1[2], \ldots$$

- Code rate is 1/number_of_generators
  - 2 generators → rate = ½
  - Engineering tradeoff: using more generators improves bit-error correction but decreases rate of the code (the number of message bits/s that can be transmitted)

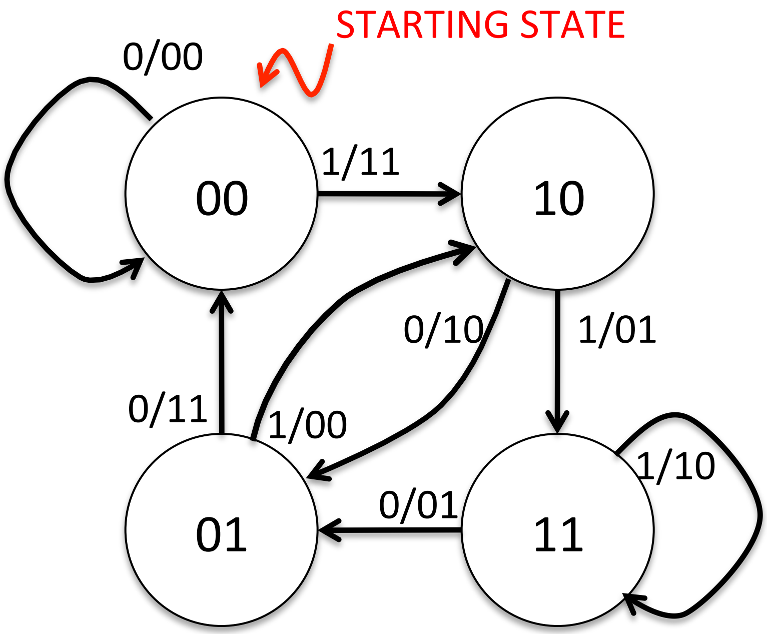


Phoning home using a K=15, rate=1/6 convolutional code

82,950 bps

(Cassini Saturn probe, Mars Pathfinder, Mars Rover)

State-Machine View



$p_0[n] = x[n] + x[n-1] + x[n-2]$

$p_1[n] = x[n] + x[n-2]$

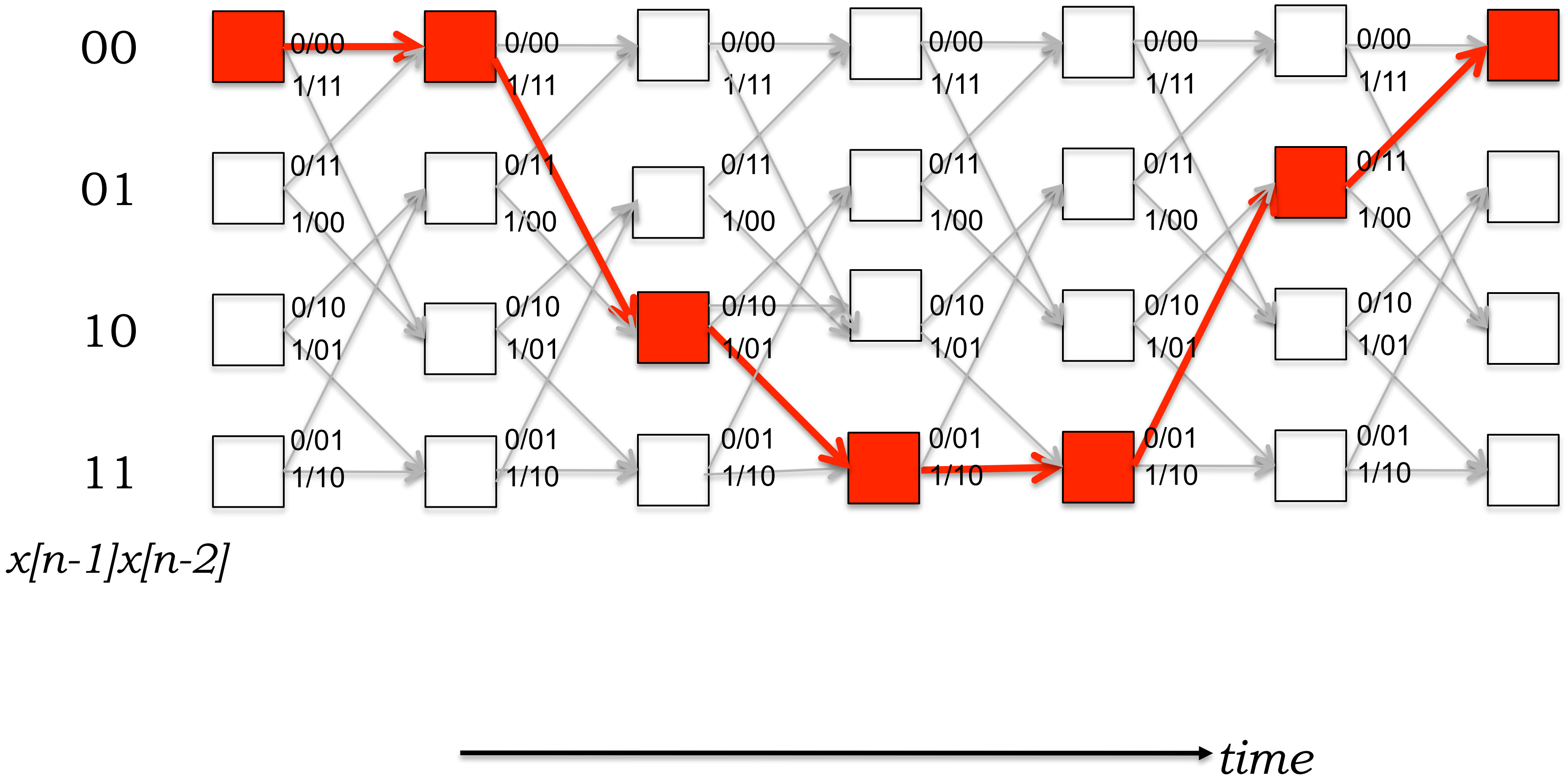
(Generators: $g_0 = 111$, $g_1 = 101$)

The state machine is the *same* for all *K*=3 codes. Only the p_i labels change depending on number and values for the generator polynomials.

- Example: *K*=3, rate-½ convolutional code
- There are 2^{K-1} states
- States labeled with (x[n-1], x[n-2]) value
- Arcs labeled with $x[n]/p_0[n]p_1[n]$
- msg=101100; xmit = 11 10 00 01 01 11

Trellis View at Transmitter

x[n]	0	1	1	1	0	0
Codeword	00	11	01	10	01	11



The Parity Stream forms a Linear Code

- Smallest-weight nonzero codeword has a weight that (locally in time) plays a role analogous to d, the minimum Hamming distance. It's called the free distance (fd) of the convolutional code.

- What is fd for our example?

Encoding & Decoding Convolutional Codes

- Transmitter (aka Encoder)
  - Beginning at starting state, processes message bit-by-bit
  - For each message bit: makes a state transition, sends $p_0p_{1\ldots}$
  - End message with *K*-1 zeros to ensure return to starting state

- Receiver (aka Decoder)
  - Doesn't have direct knowledge of transmitter's state transitions; only knows (possibly corrupted) received parity bits, p_i
  - Must find most likely sequence of transmitter states that could have generated the received parity bits, p_i
  - If BER < ½, *P*(more errors) < *P*(fewer errors)
  - *When BER < ½, maximum-likelihood message sequence is the one that generated the codeword (here, sequence of parity bits) with the smallest Hamming distance from the received codeword (here, parity bits)*
  - I.e., find nearest valid codeword *closest* to the received codeword – Maximum-likelihood (ML) decoding

In the absence of noise …

- Decoding is trivial:

$$p_0[n] = x[n] + x[n-1] + x[n-2]$$

$$p_1[n] = x[n] + x[n-2]$$

- Can you see how to recover the input x[.] from the parity bits p[.] ?

- In the presence of errors in the parity stream, message bits will get corrupted at about the same rate as parity bits, with this simple-minded recovery.

Spot Quiz!

Consider the convolutional code given by

$p_0[n] = x[n] + x[n-2] + x[n-3]$

$p_1[n] = x[n] + x[n-1] + x[n-2]$

$p_2[n] = x[n] + x[n-1] + x[n-2] + x[n-3]$

1. Constraint length, K, of this code = ______

2. Code rate = _____

3. Coefficients of the generators = _____, _____, _____

4. No. of states in state machine of this code = _____