CODING for Transmission

Professor Izzat Darwazeh
Head of Communications and Information System Group
University College London
i.darwazeh@ee.ucl.ac.uk

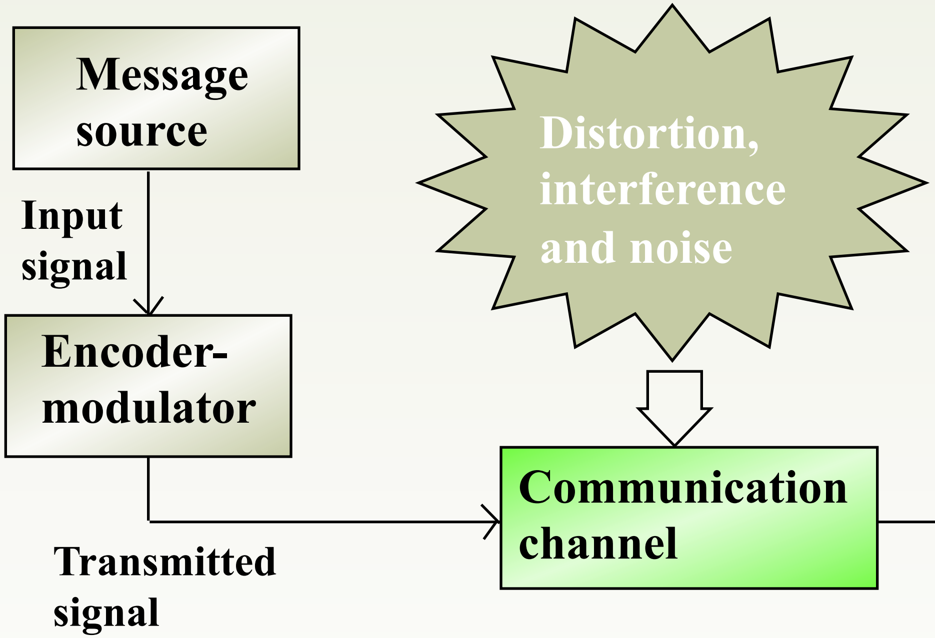
Acknowledgement:
Dr A. Chorti, Princeton University for slides on FEC coding
Dr P.Moreira, CERN, for slides on the CERN Gigabit Transmitter (GBT)
Dr J. Mitchell, UCL, for the sampled music and voice.

June 2011

Coding

- Definitions and basic concepts
- Source coding
- Line coding
- Error control coding

Digital Line System

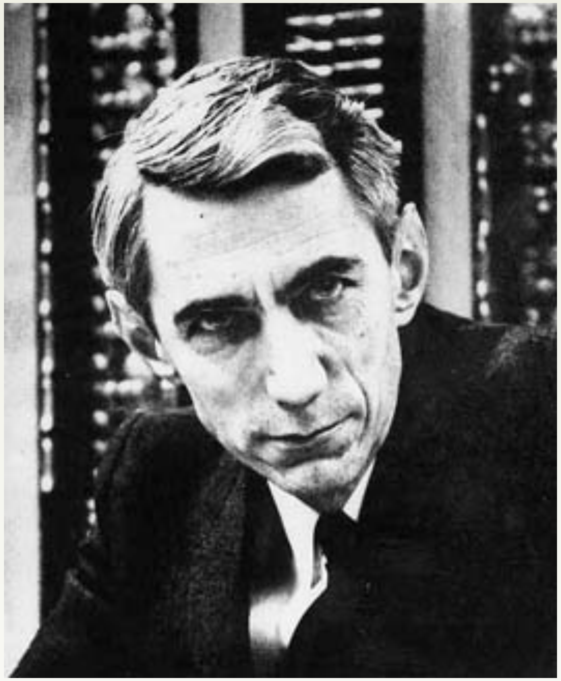


Claude Shannon

- Shannon's Theorem predicts reliable communication in the presence of noise

  "Given a discrete, memoryless channel with capacity C, and a source with a positive rate R (R<C), there exist a code such that the output of the source can be transmitted over the channel with an arbitrarily small probability of error."

- *B* is the channel bandwidth in Hz and S/N is the signal power to noise power ratio

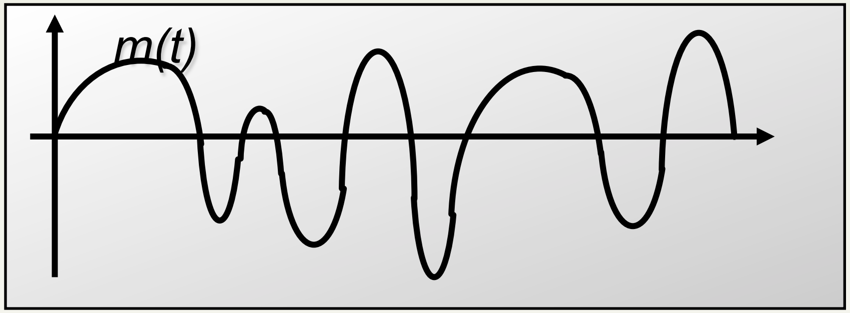


$$C_c = B\log_2\left(1+\frac{S}{N}\right)$$

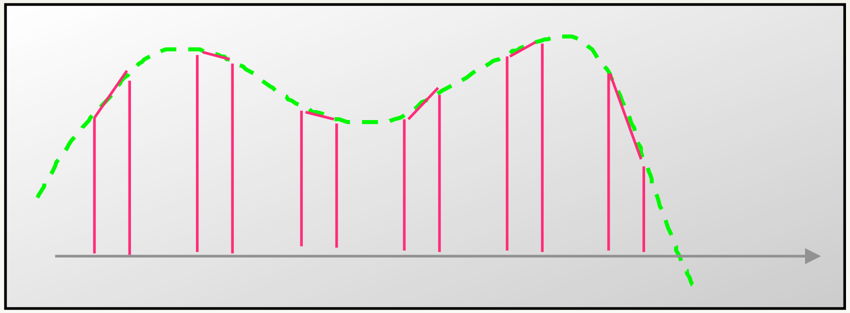
Types of Coding

- Source Coding
  - Encoding the raw data
- Line (or channel) Coding
  - Formatting of the data stream to benefit transmission
- Error Detection Coding
  - Detection of errors in the data sequence
- Error Correction Coding
  - Detection and Correction of Errors
- Spread Spectrum Coding
  - Used for wireless communications

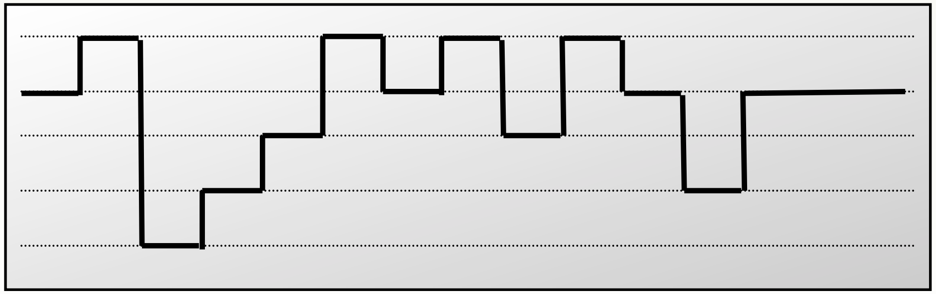
Signals and sources: *Discrete - Continuous*



- Continuous Time and Amplitude



- Discrete Time, continuous Amplitude – PAM signal



- Discrete Time, and Amplitude – Multi-level digital

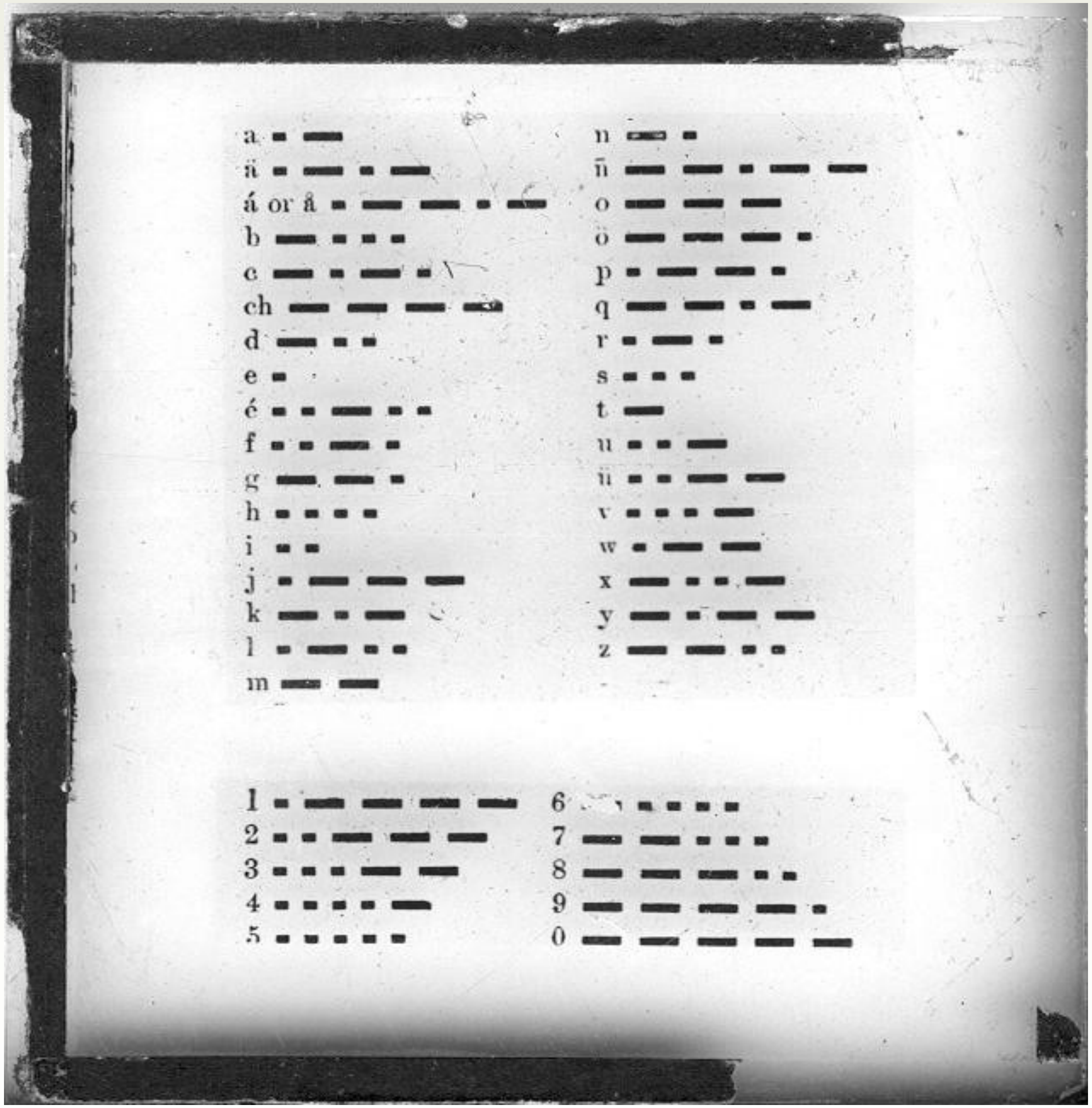
Source Coding

- Coding of information to make it suitable for transmission
- Character and information coding (IATx, ASCII,BCD & EBCDIC, Unicode and many others)
- Examples: image (JPEG), video (MPEG), audio(MP3) and voice (PCM to CELP) and many others
- Lossy and lossless source coding

!	0100001
/	0101111
0	0110000
9	0111001
:	0111010
@	1000000

A	1000001
Z	1011010
a	1100001
z	1111010
~	1111110

Source Coding is not new



a ·—
ä ·—·—
á or å ·——·—
b —···
c —·—·
ch ————
d —··
e ·
é ··—··
f ··—·
g ——·
h ····
i ··
j ·———
k —·—
l ·—··
m ——

n —·
ñ ——·——
o ———
ö ———·
p ·——·
q ——·—
r ·—·
s ···
t —
u ··—
ü ··——
v ···—
w ·——
x —··—
y —·——
z ——··

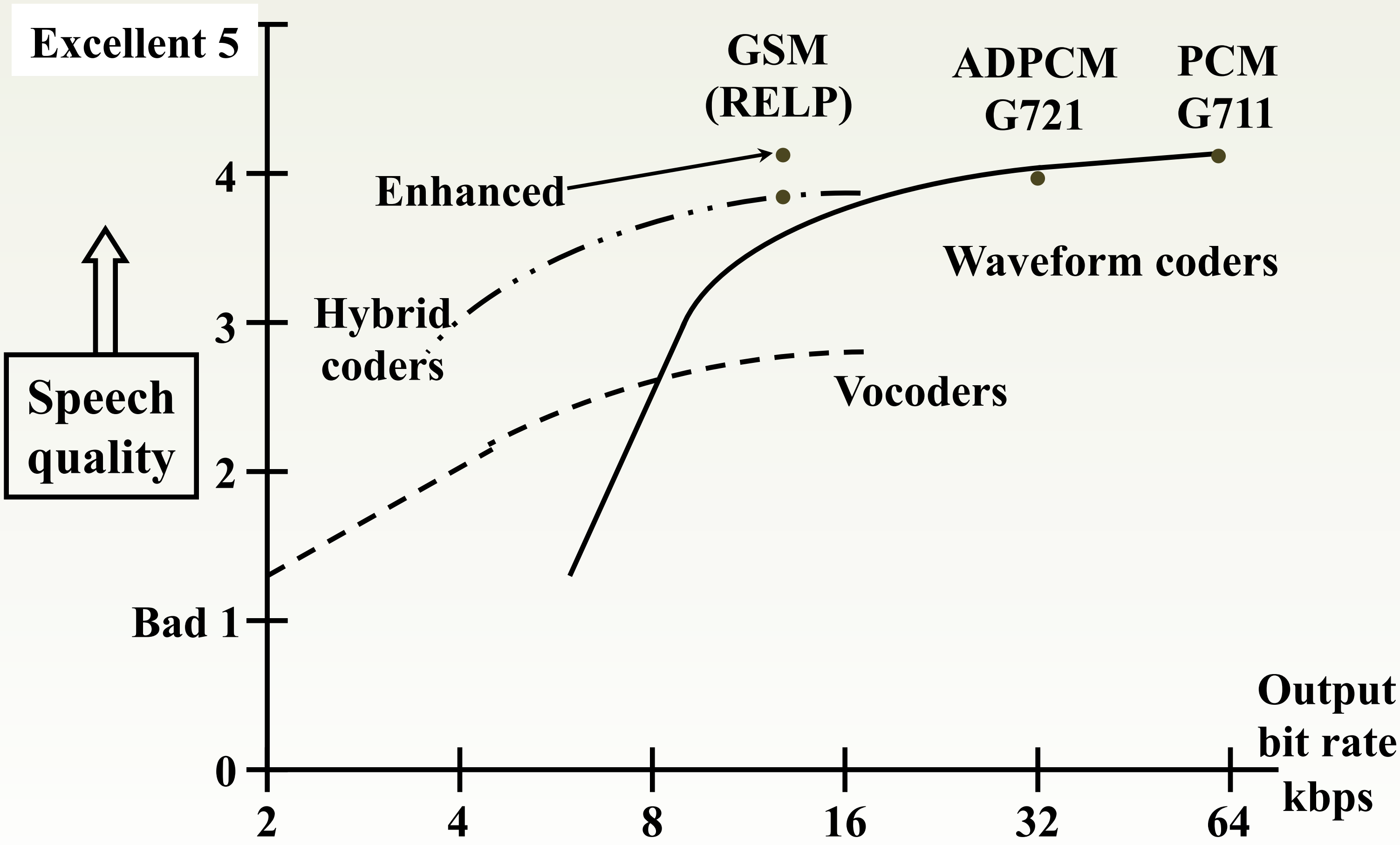
1 ·————
2 ··———
3 ···——
4 ····—
5 ·····

6 [illegible] ····
7 ——···
8 ———··
9 ————·
0 —————

Voice Coding

- Source coding applied to voice in telephony
- Use of electronic or software codec systems
- A codec (Coder/Decoder) converts analogue signals to a digital bitstream, and another identical codec at the far end of the communication converts the digital bitstream back into an analog signal.
- In the VoIP world, codec's are used to encode voice for transmission across IP networks.
- Codec's for GSM and VoIP use are also referred to as *vocoders*, for "voice coders".
- Codecs generally provide a compression capability to save network bandwidth. Some codecs also support silence suppression, where silence is not encoded or transmitted.

Speech Quality versus Bit Rate



Required Bandwidth

- Typically the range of human hearing is from 20Hz up to a maximum of 20kHz
- However, different types of signals have different requirements
- Music

22kHz – 16bit

12kHz – 16bit

5.5kHz – 16bit

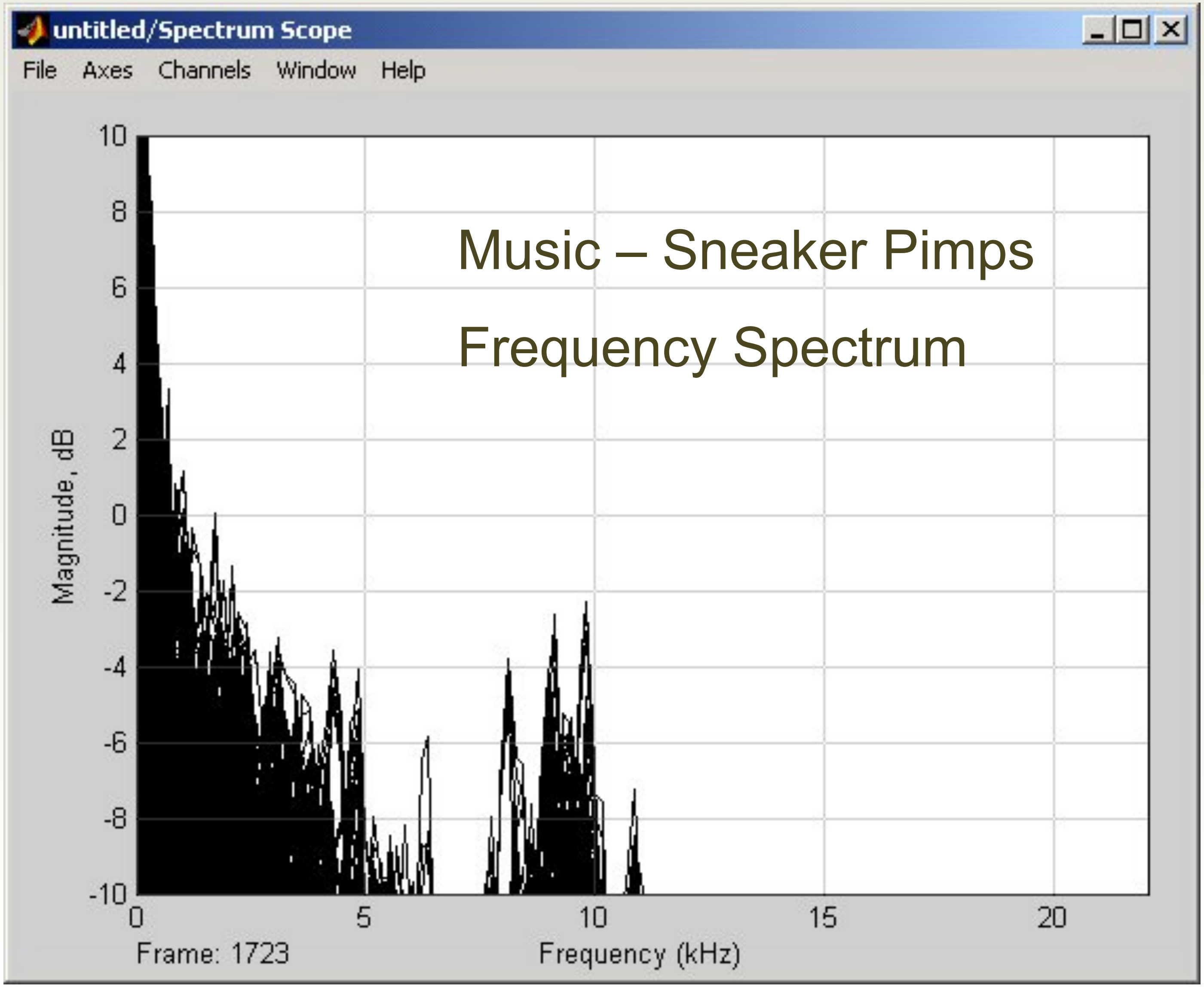
4kHz – 16bit

- Telephony

16kHz – 8bit

4kHz – 8bit

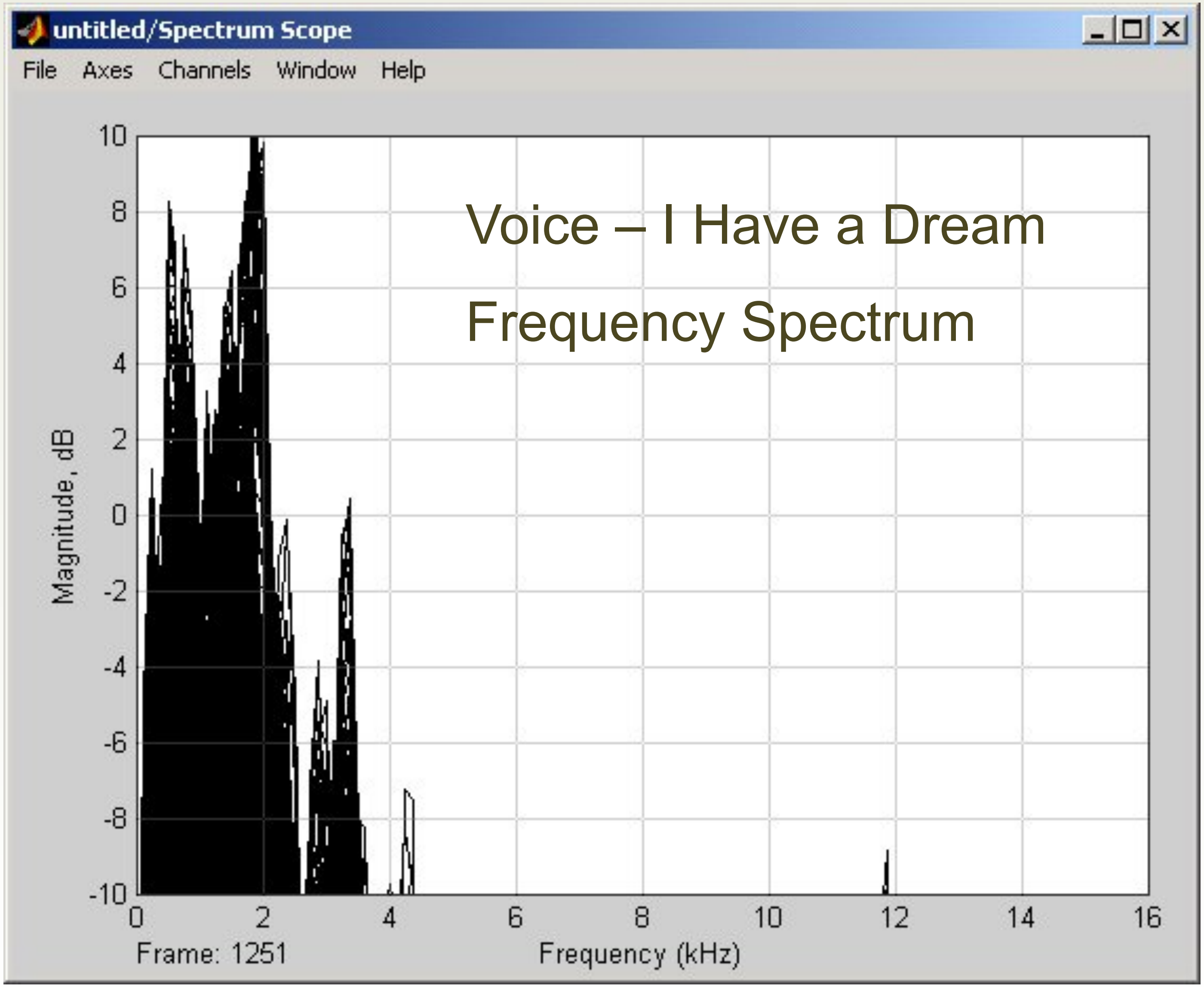
Why?



Music – Sneaker Pimps

Frequency Spectrum

Why?



Voice – I Have a Dream

Frequency Spectrum

The moral of the story..

Serious bandwidth saving can be achieved by source coding if:

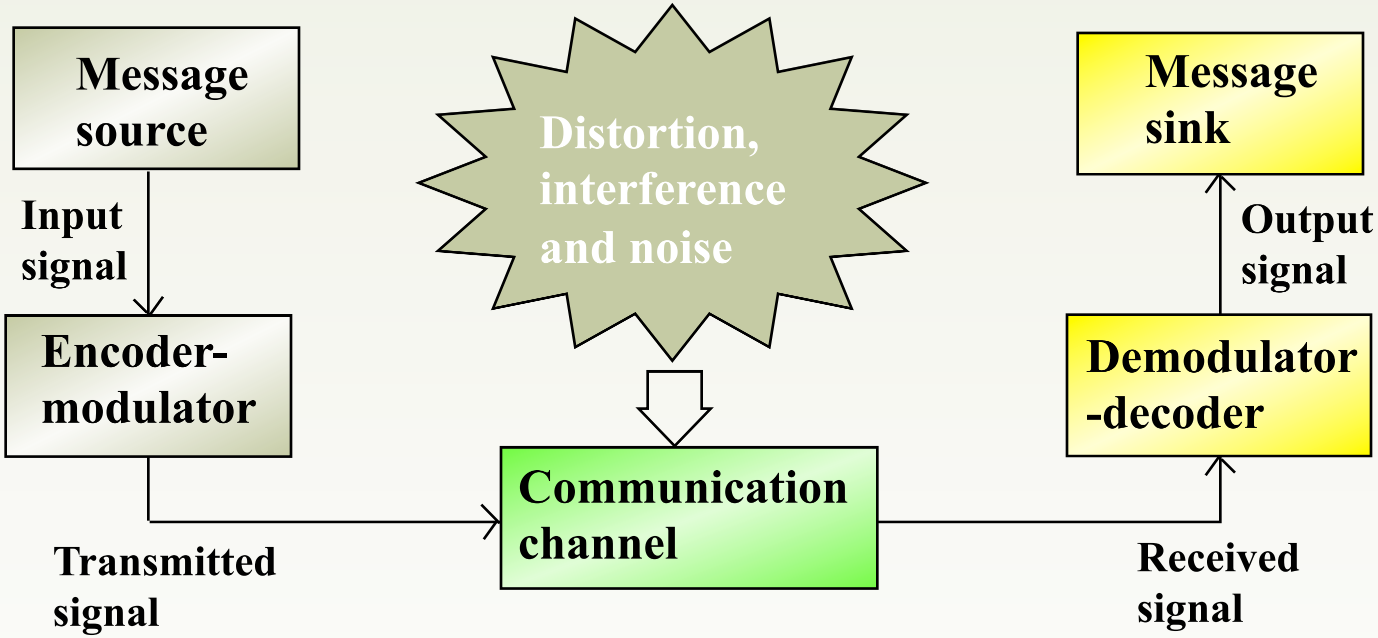
1- Good knowledge of signal characteristics (spectral and time domain behaviour)

And..

2- appreciation of the “nature” and information requirements of the receiver

What else?

Digital Line System



Message source

Input signal

Encoder-modulator

Transmitted signal

Distortion, interference and noise

Communication channel

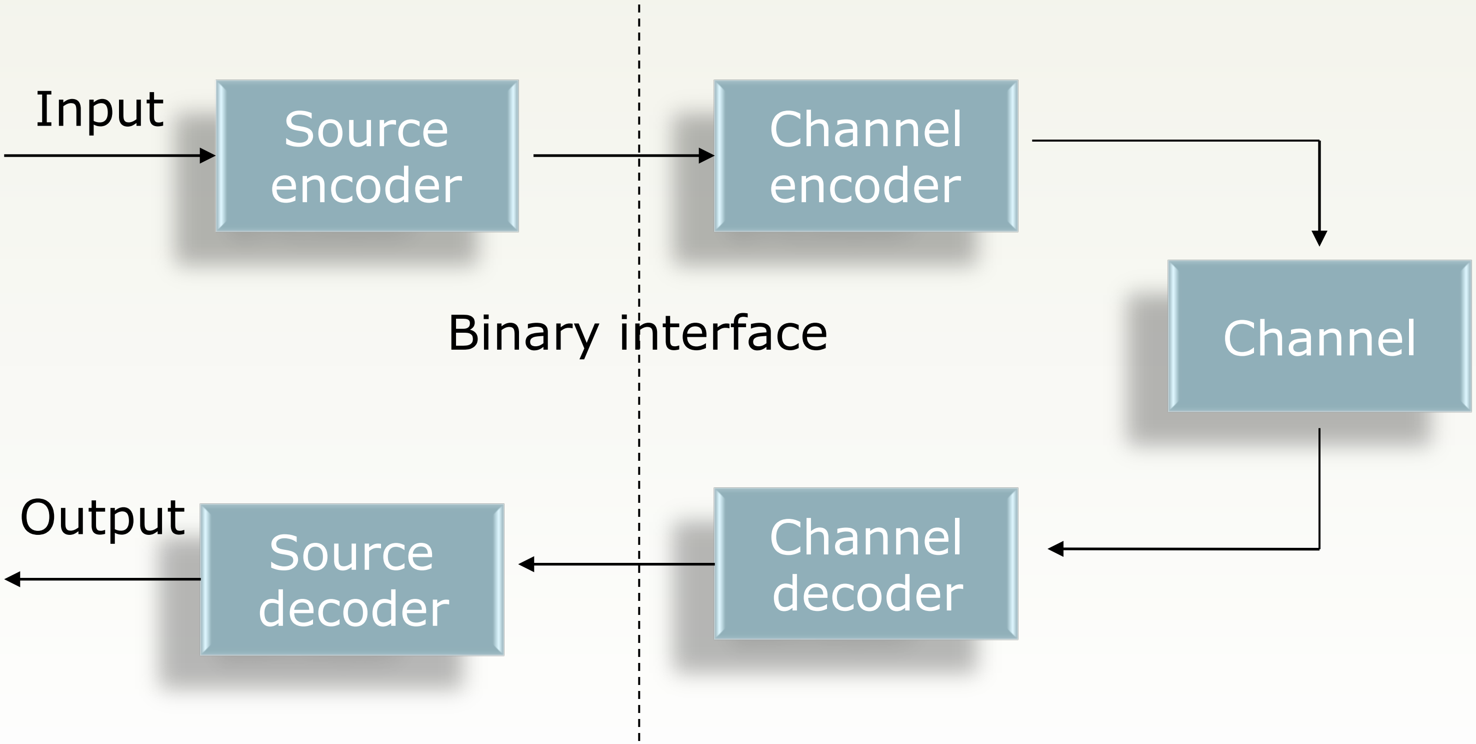
Received signal

Demodulator -decoder

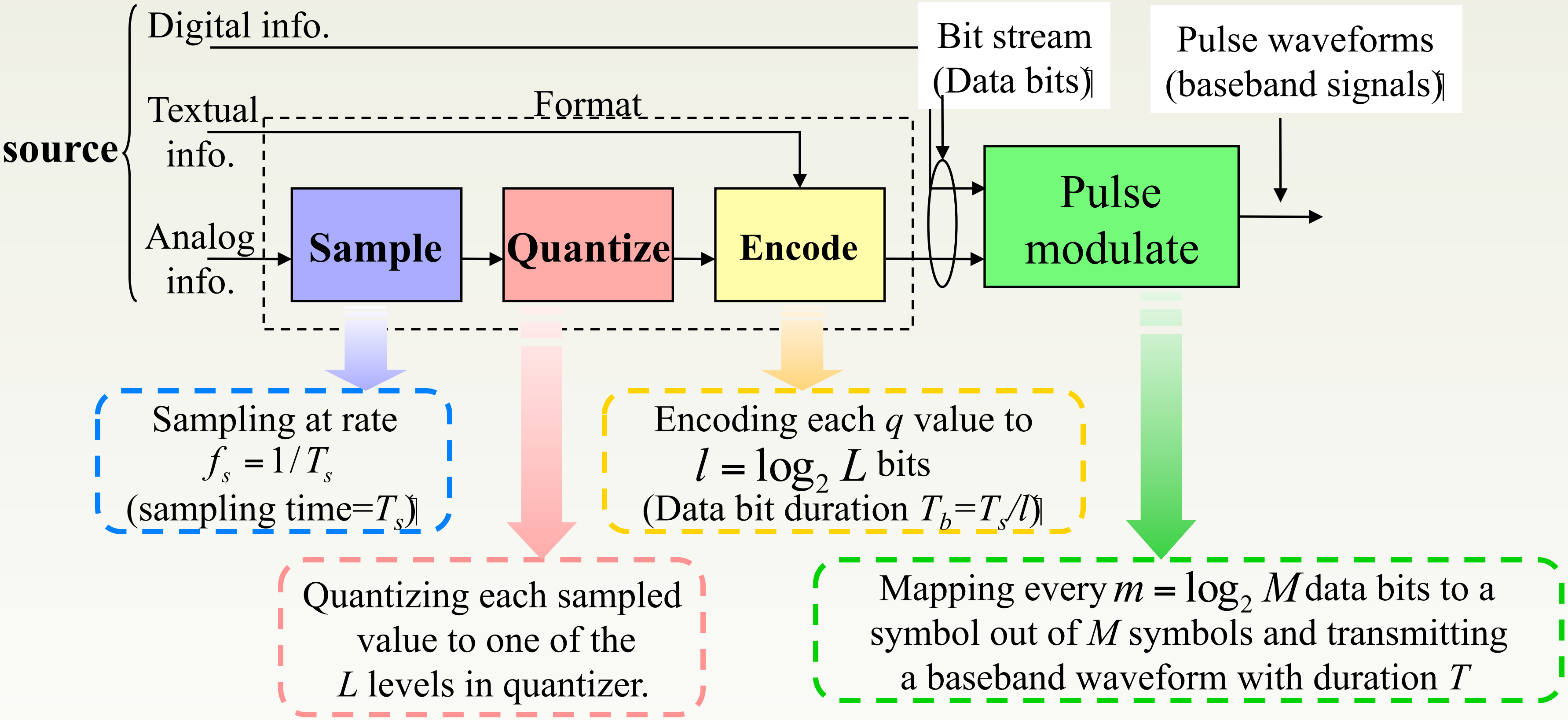
Output signal

Message sink

- Separation of source and channel coding
  - The source encoder needs to understand the probabilistic structure of the source
  - The channel encoder needs to understand the probabilistic nature of the noise



Formatting and transmission of baseband signal



Sampling at rate $f_s = 1/T_s$ (sampling time=T_s)

Quantizing each sampled value to one of the L levels in quantizer.

Encoding each q value to $l = \log_2 L$ bits (Data bit duration T_b=T_s/l)

Mapping every $m = \log_2 M$ data bits to a symbol out of M symbols and transmitting a baseband waveform with duration T

$$R_b = 1/T_b \ \text{[bits/sec]}$$

- Information (data) rate: $R = 1/T \ \text{[symbols/sec]}$
- Symbol rate : $R_b = mR$
  - For real time transmission:

What is channel coding

- Channel coding refers to the class of signal transformations designed to improve communications performance, by enabling the transmitted signals to better withstand the effects of the various **channel** impairments (noise, interference, fading, filtering..etc)
- Desirable system trade-offs (e.g. **error performance vs bandwidth, or power vs bandwidth**)
- **We can achieve as much as 10 dB performance improvement**

- Two main categories of channel coding
  - Waveform coding
  - **Structured sequences**

Waveform vs structured sequences coding

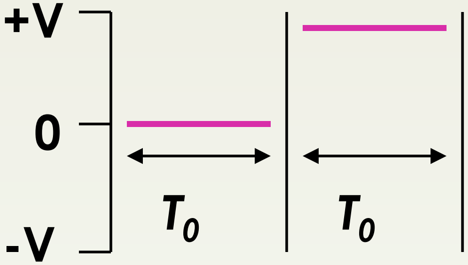
- Transforming signals to improve communications performance by increasing the robustness against channel impairments (noise, interference, fading, ..)
- Waveform coding: Transforming waveforms to "better" waveforms
- Structured sequences: Transforming data sequences into "better" sequences, having structured redundancy.
    - Block codes
    - Convolutional codes
    - Turbo codes

"Better" in the sense of making the decision process less subject to errors *or* that we somehow manage to increase the distance between them

Waveform (Line) Coding

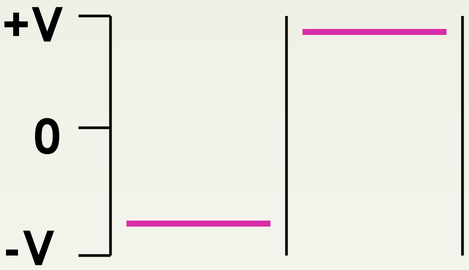
- Basic binary signals are “unipolar” pulses
- Other pulse types may be used to represent data
- Line coding (makes the data suited to the line or channel) changes the pulse format from simple NRZ for one or more of the following reasons:
  - Presence or absence of a DC level,
  - To shape the power spectral density
  - Bandwidth
  - Noise immunity
  - Ease of clock recovery

Line Coding (examples)



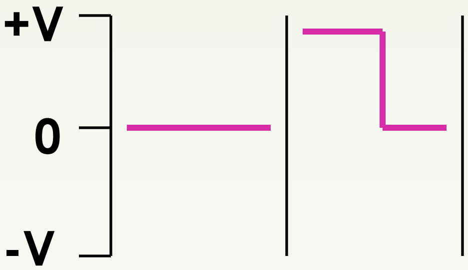
Unipolar

NRZ



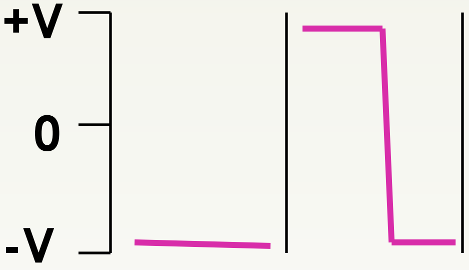
polar

NRZ



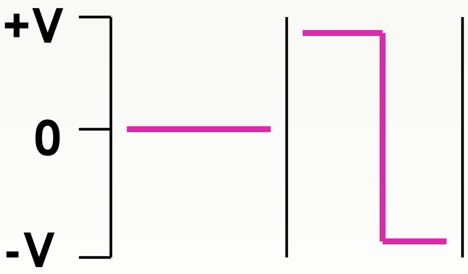
Unipolar

RZ



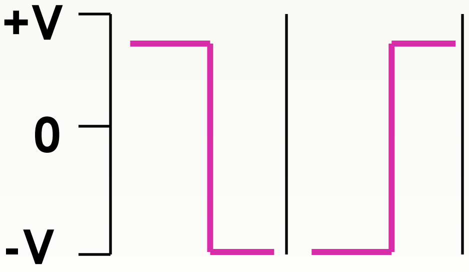
polar

RZ



Dipolar

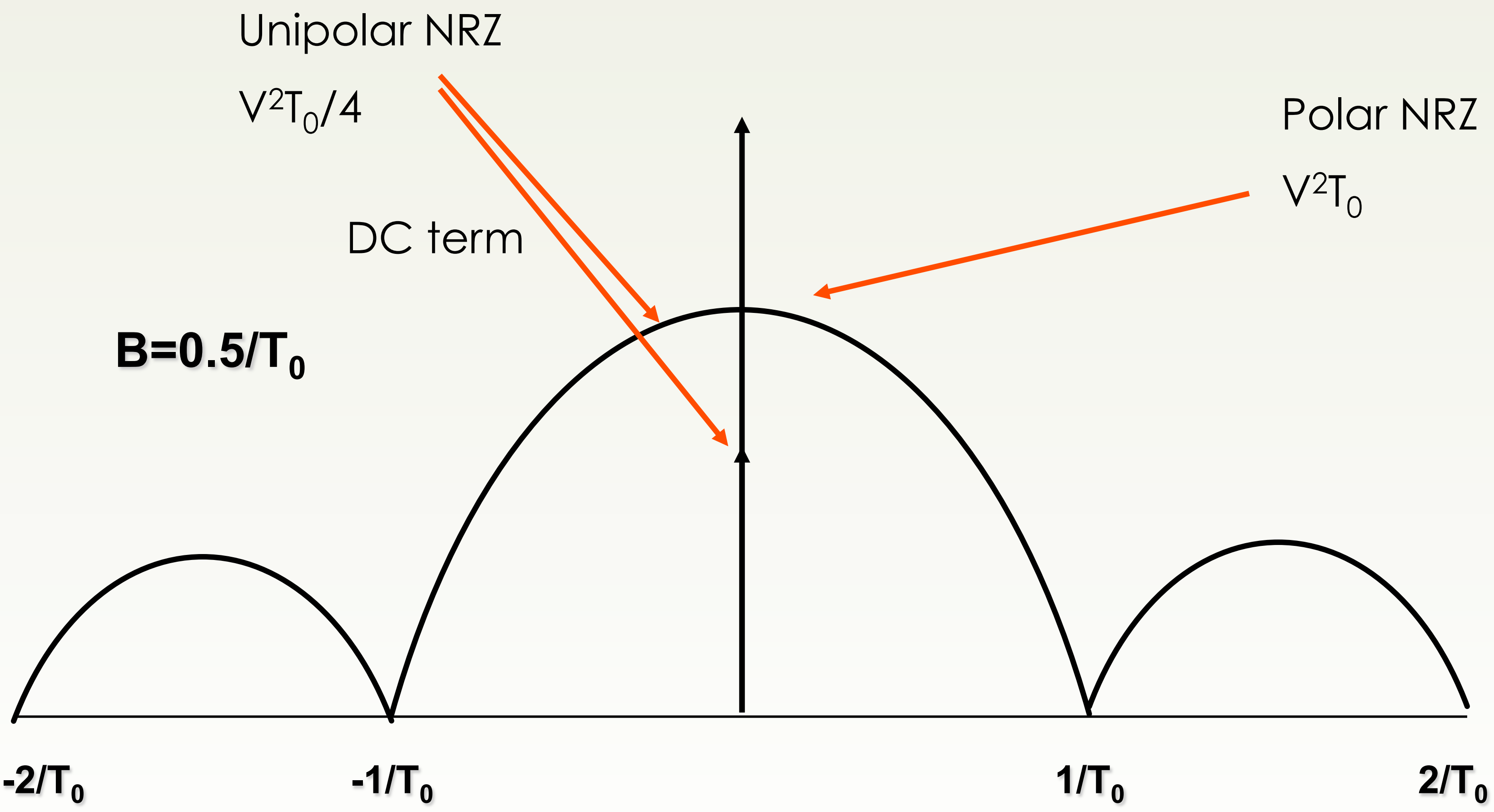
OOK/RZ



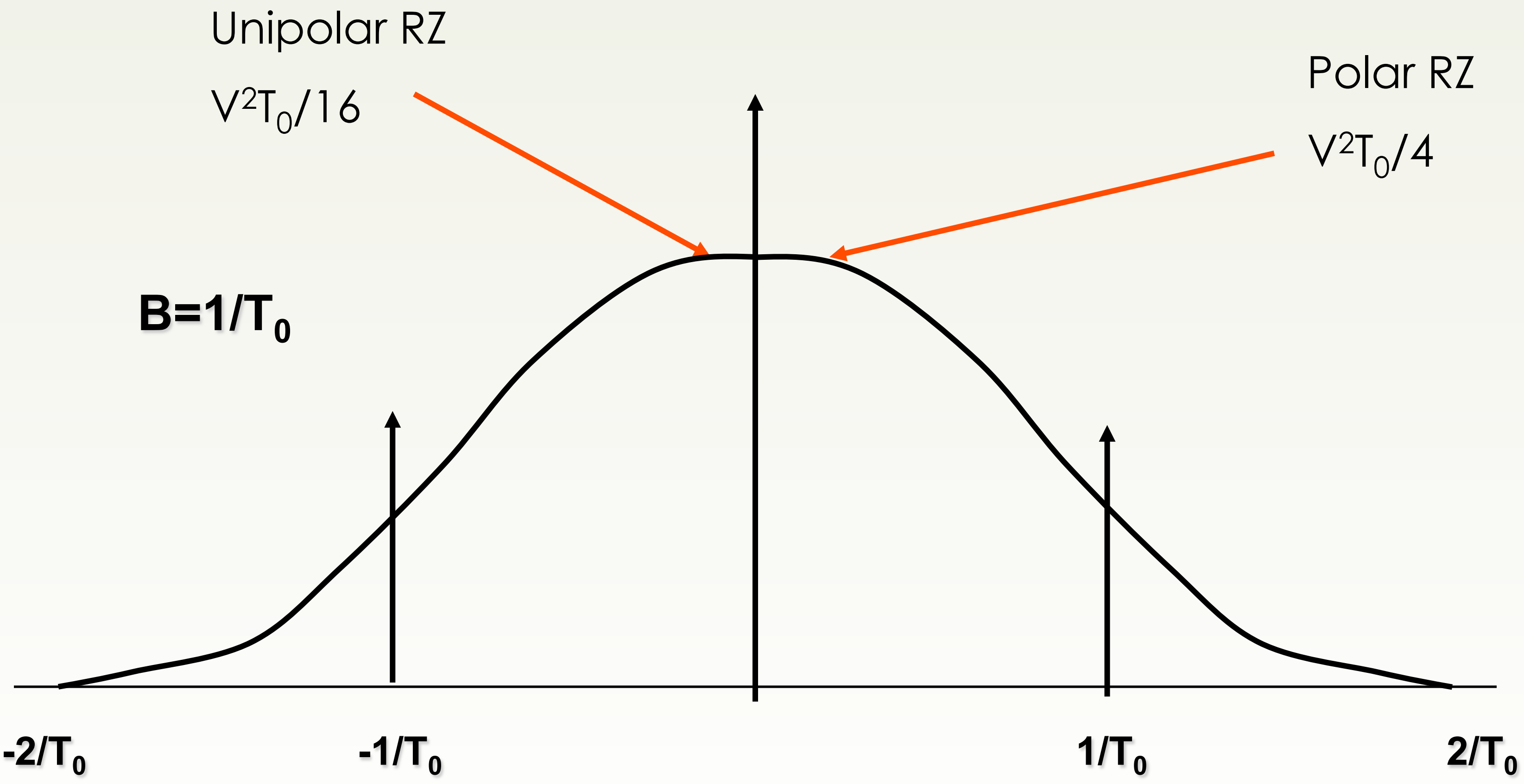
Manchester

polar

Line Code Spectrum



Line Code Spectrum



Unipolar RZ

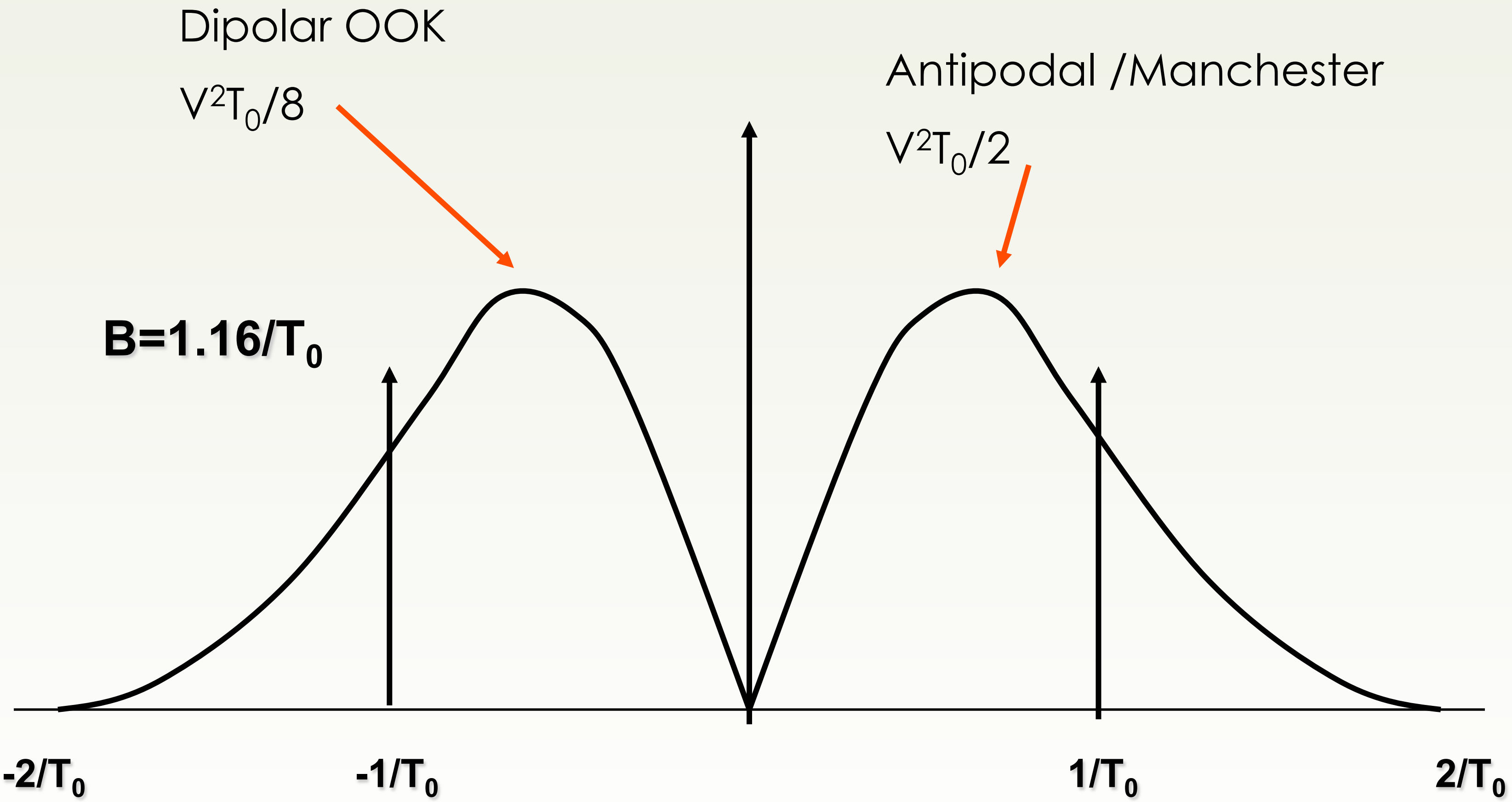
$V^2T_0/16$

Polar RZ

$V^2T_0/4$

$B=1/T_0$

Line Code Spectrum



Dipolar OOK

$V^2T_0/8$

Antipodal /Manchester

$V^2T_0/2$

$B=1.16/T_0$

Example Line Codes

Coding Type	Comment
Unipolar NRZ	Usual format, DC term
Unipolar RZ	Simple timing extraction, DC term
Polar NRZ	Constant envelope, no DC term
Polar RZ	Simpler timing extraction, DC term
Dipolar OOK	Simple timing extraction,
Manchester	simple timing extraction, no DC term and Constant envelope,
*m*B*n*B	Simpler timing extraction, no DC term

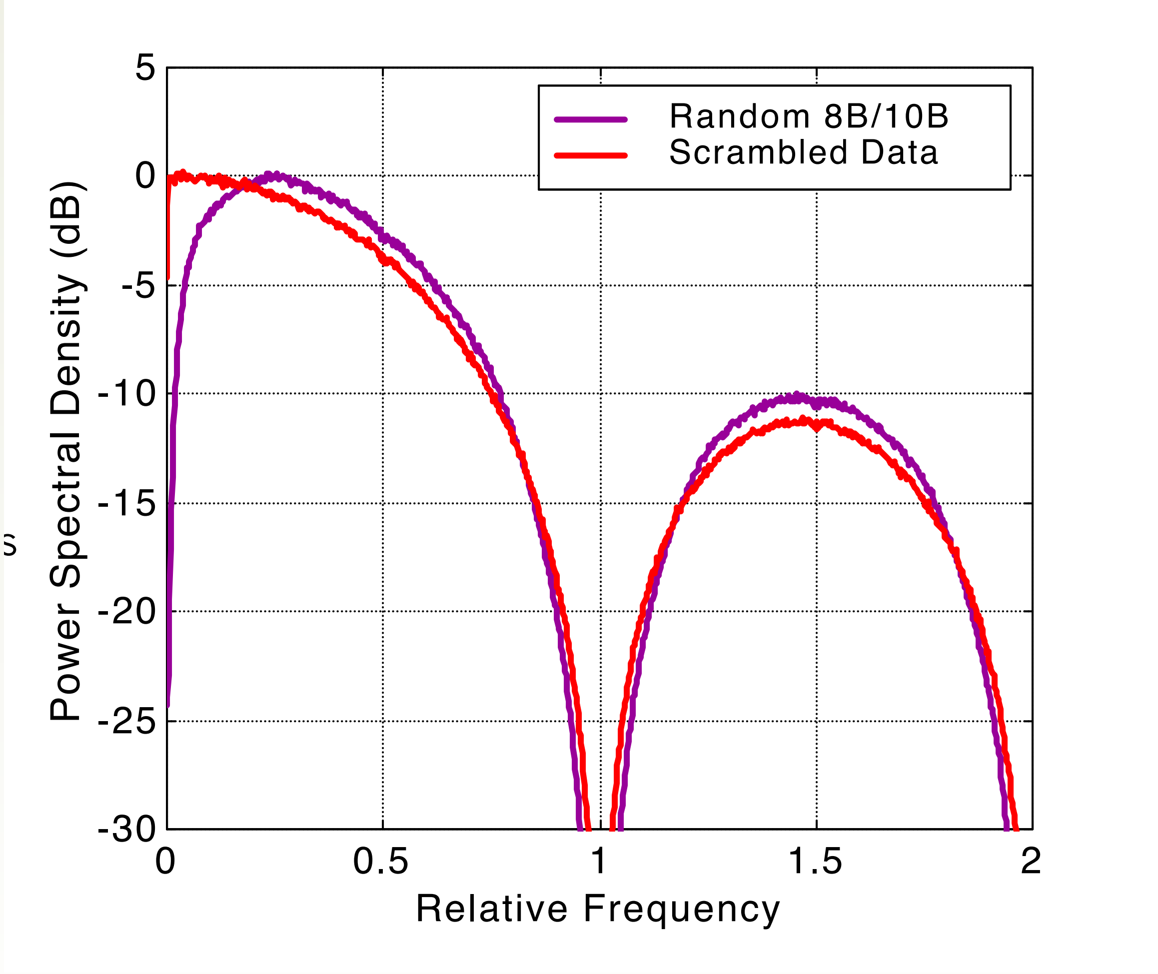
mBnB codes

- Commonly used as line codes. n>m.
- Known as block codes as they encode a block of data (of length m) into another block (of length n).
- Redundancy is equal to $2^n - 2^m$. Redundant bit patterns can be used for transmission of control information.
- Defined in terms of maximum length of one type bit patterns “run length” and ratio of total no. of 1s to 0s “Disparity”.
- Used to adjust the:
  - DC component level,
  - shape the spectrum
  - provide timing information

- Allows error monitoring by simple rule violation detection logic circuits
- Can be constructed as self-framing

- Always result in bandwidth increase by (n/m)

8B10B spectrum (with and without scrambling)



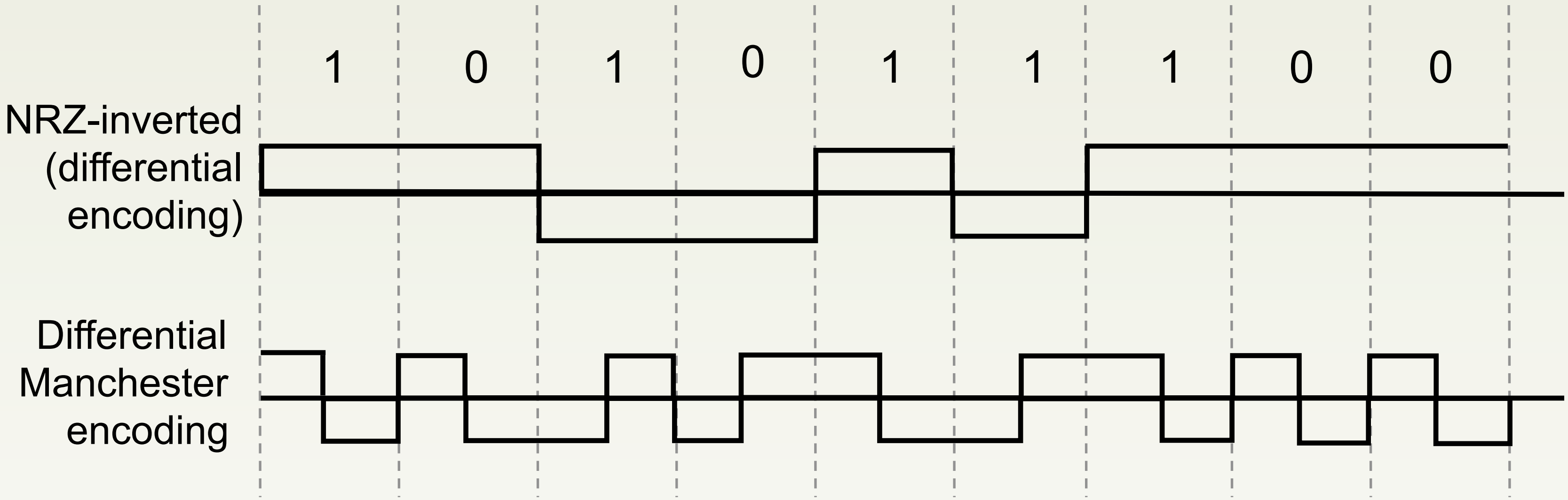
Ref. 802.3ae task force . IBM

mBnB examples

Code	n/m	N_{max}	D	redundancy	example
1B2B	2	2	+/- 1	50%	802.3
3B4B	1.33	4	+/- 3	25%	
4B5B	1.2	5	+/- 4	25%	FDDI, 10M Ethernet
5B6B	1.2	6	+/- 4	28%	
6B8B	1.33	6	+/- 3	75%	
8B10B	1.2	8	+/- 8	75%	1G Ethernet USB3 PCI Express
9B10B	1.11	11	+/- 8	24%	
64B66B	1.03	64			10G Ethernet

n/m, is bandwidth proportional increase
N_{max} run length•
D maximum disparity.
Redundancy, proportion of n-bit words not used
Codes with even “n” have the zero DC advantage

Differential Coding



- To avoid transposition in polarity where all bits may be received in error
  - "1" mapped into transition in signal level
  - "0" mapped into no transition in signal level

- Same bandwidth and spectral characteristics as NRZ
- Errors occur in pairs
- May be used with other encoding (e.g with Manchester code)

Error Detection/Correction

- We require a reliable channel, i.e. BER > x
- What can we do if this is not possible?
  1. Increase the power
  2. Diversity in space/frequency or time
  3. Full Duplex (echo information)
  4. Automatic repeat Request (ARQ)
  5. Forward Error Correction
  6. Predict bit (and error) patterns?

Price to pay…introduction of a certain amount of redundancy

prediction/error correction is also not new…

THE PAOMNNEHAL PWEOR OF THE HMUAN MNID

Aoccdrnig to a rscheearch at Cmabrigde Uinervtisy, it deosn't mttaer in waht oredr the ltteers in a wrod are, the olny iprmoatnt tihng is taht the frist and lsat ltteer be in the rghit pclae. The rset can be a taotl mses and you can sitll raed it wouthit porbelm. Tihs is bcuseae the huamn mnid deos not raed ervey lteter by istlef, but the wrod as a wlohe.

Amzanig huh?

Error control techniques

- **Automatic Repeat reQuest (ARQ)**
  - Full-duplex connection, error detection codes
  - The receiver sends a feedback to the transmitter, notifying if any error is detected in the received packet or not (Not-Acknowledgement (NACK) and Acknowledgement (ACK), respectively).
  - The transmitter retransmits the previously sent packet if it receives NACK.
- **Forward Error Correction (FEC)**
  - Simplex connection, error correction codes
  - The receiver tries to correct some errors
- **Hybrid ARQ (ARQ+FEC)**
  - Full-duplex, error detection and correction codes

ARQ

- Three common ARQ techniques exist
  - **Stop-and-wait**: The sender(Tx) sends a frame and then waits for acknowledgement for the receiver (Rx).
  - **Go-back-n**: Sender sends frames in a sequence with acknowledgement from Rx. On error Rx discards current and further frames and notifies Tx of frame required. Tx starts again from this point
  - **Selective repeat**: Tx only repeats those frames for which negative acknowledgements are received.

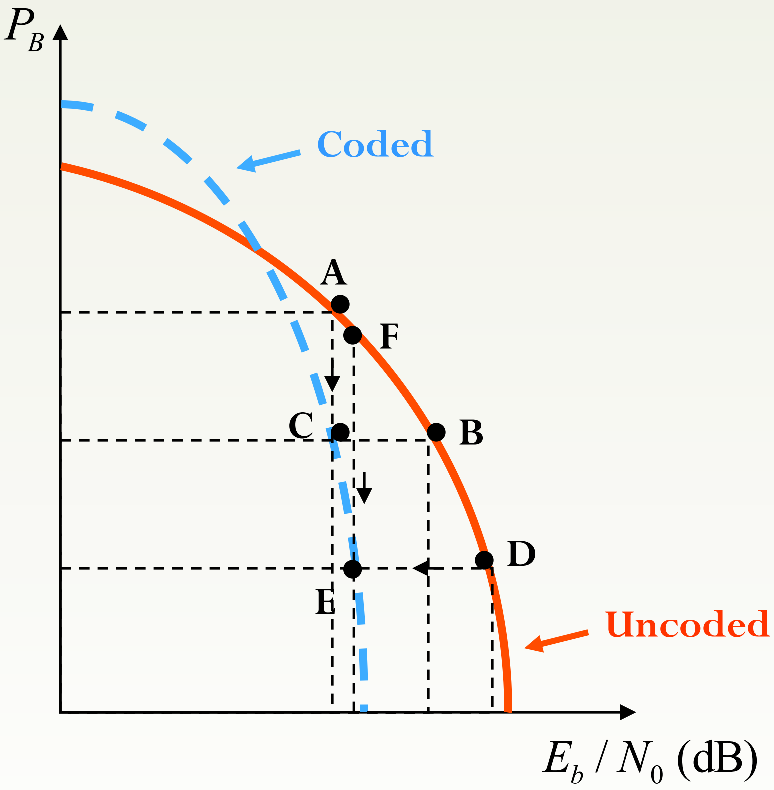
Why use error correction coding?

- Error performance vs. bandwidth
- Power vs. bandwidth
- Data rate vs. bandwidth
- Capacity vs. bandwidth

Coding gain:

For a given bit-error probability, the reduction in the E_b/N_0 that can be realized through the use of code:

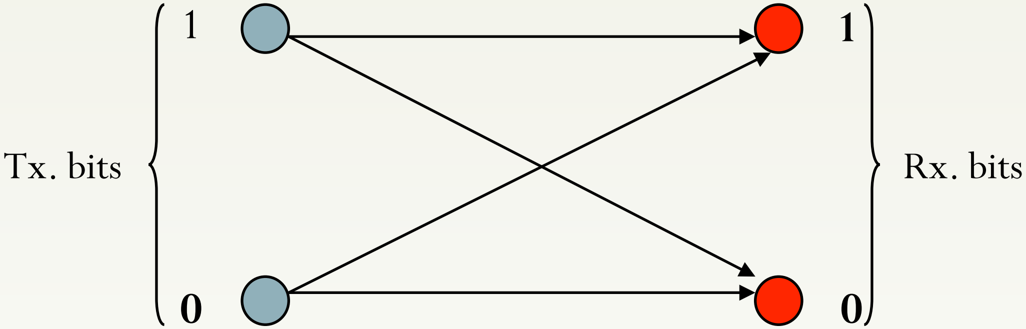
$$G\,[\text{dB}] = \left(\frac{E_b}{N_0}\right)_{\text{u}} [\text{dB}] - \left(\frac{E_b}{N_0}\right)_{\text{c}} [\text{dB}]$$



A=(8, 1e-2), C=(8, 1e-4), D=(14, 1e-6), E=(9, 1e-6)

Channel models

- **Discrete Memoryless Channels**
  - Discrete input, discrete output
  - Each output symbol depends only on a corresponding input symbol according to some predefined pdf
- **Binary Symmetric Channels**
  - Binary input, binary output
  - The conditional probabilities are symmetric
    - P(0|1)=P(1|0)=p
    - P(1|1)=P(0|0)=1-p
- **Gaussian Channels**
  - Discrete input, continuous output (after the addition of continuous noise)
  - We have added continuous noise. The output is a RV centered on the input with mean and variance the respective mean and variance of the noise



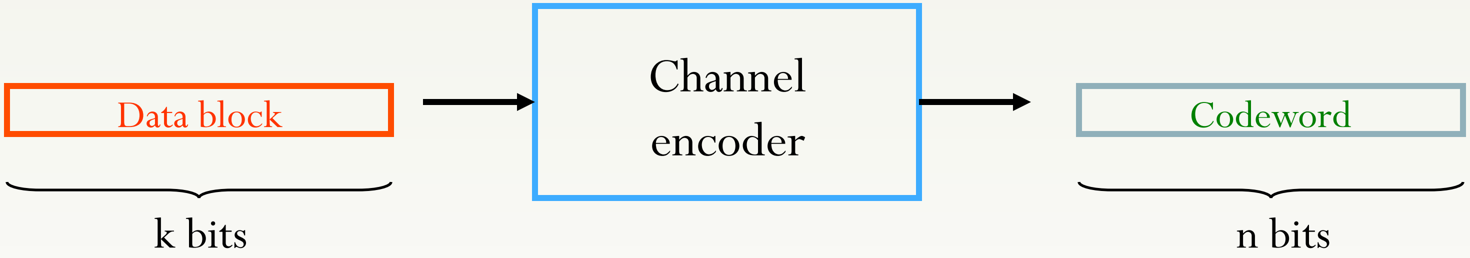
$$p(z \mid u_k) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(- \frac{(z - u_k)^2}{2\sigma^2} \right)$$

Linear block codes

- A class of parity-check codes
- The encoder transforms a block of k message digits into a longer block of n codeword digits
- The original k-tuple is drawn from 2^k distinct sequences. These sequences can be viewed as vectors.
- It is then mapped to 2^k n-tuples drawn from 2^n possible sequences. These sequences also will be viewed as vectors.
- We select those 2^k n-tuples so that we increase the distance between them
- The transformation is linear: it can be represented by a mapping in a vector space

Linear block codes – cont'd

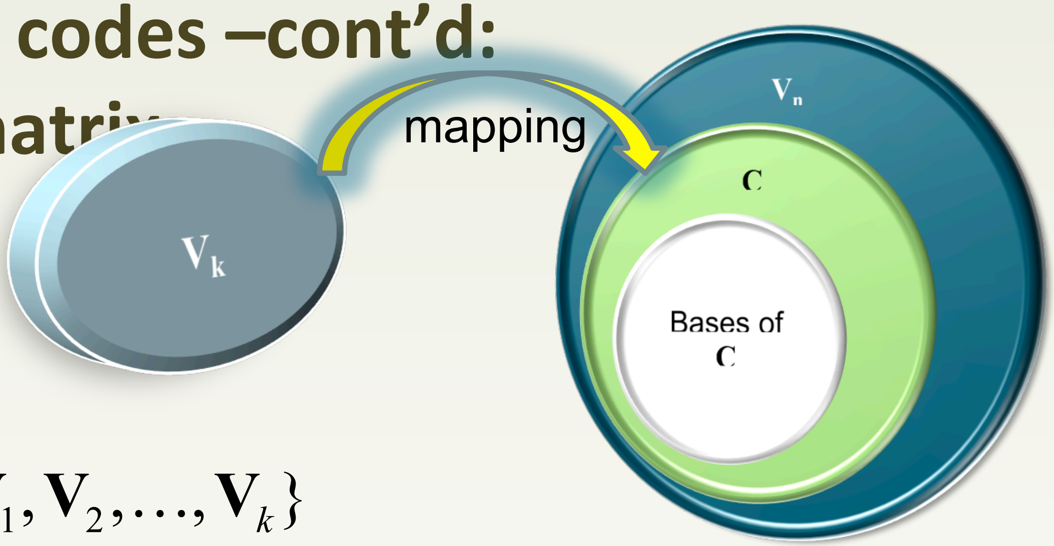
- The information bit stream is chopped into blocks of k bits.
- Each block is encoded to a larger block of n bits.
- The coded bits are modulated and sent over channel.
- The reverse procedure is done at the receiver.



$n-k$ Redundant bits

$$R_c = \frac{k}{n}$$ Code rate

Linear block codes –cont'd: generator matrix



$$\{\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_k\}$$

A matrix $\mathbf{G}$ is constructed by taking its rows as the vectors on a basis, .

$$\mathbf{G} = \begin{bmatrix} \mathbf{V}_1 \\ \vdots \\ \mathbf{V}_k \end{bmatrix} = \begin{bmatrix} v_{11} & v_{12} & \cdots & v_{1n} \\ v_{21} & v_{22} & \cdots & v_{2n} \\ \vdots & & \ddots & \vdots \\ v_{k1} & v_{k2} & \cdots & v_{kn} \end{bmatrix}$$

Linear block codes

- Encoding in *(n,k)* block code

$$\boxed{\mathbf{U} = \mathbf{mG}}$$

$$[u_1, u_2, \ldots, u_n] = [m_1, m_2, \ldots, m_k] \cdot \begin{bmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \\ \vdots \\ \mathbf{V}_k \end{bmatrix}$$

$$[u_1, u_2, \ldots, u_n] = m_1 \cdot \mathbf{V}_1 + m_2 \cdot \mathbf{V}_2 + \ldots + m_2 \cdot \mathbf{V}_k$$

m is the original message

G is the generator matrix

U is the encoded message

  - The rows of $\mathbf{G}$, are linearly independent.

Linear block codes

Example: Block code (6,3)

$$\mathbf{G} = \begin{bmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \\ \mathbf{V}_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Addition	Multiplication
$0 \oplus 0 = 0$	$0 \cdot 0 = 0$
$0 \oplus 1 = 1$	$0 \cdot 1 = 0$
$1 \oplus 0 = 1$	$1 \cdot 0 = 0$
$1 \oplus 1 = 0$	$1 \cdot 1 = 1$

Message vector	Codeword
000	000000
100	110100
010	011010
110	101110
001	101001
101	011101
011	110011
111	000111

Systematic block codes

- Systematic block code *(n,k)*
  - For a systematic code, the first (or last) k elements in the codeword are information bits.

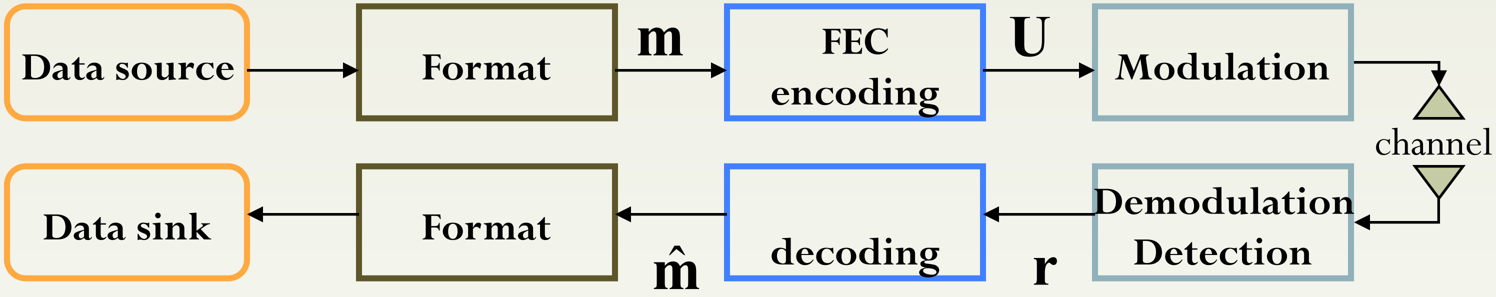
$$\mathbf{G} = [\mathbf{P} \;\vdots\; \mathbf{I}_k]$$

$$\mathbf{I}_k = k \times k \text{ identity matrix}$$

$$\mathbf{P}_k = k \times (n-k) \quad \text{matrix}$$

$$\mathbf{U} = (u_1, u_2, \ldots, u_n) = (\underbrace{p_1, p_2, \ldots, p_{n-k}}_{\text{parity bits}}, \underbrace{m_1, m_2, \ldots, m_k}_{\text{message bits}})$$

Linear block Coding block diagram



$$\mathbf{r} = \mathbf{U} + \mathbf{e}$$

$\mathbf{r} = (r_1, r_2, \ldots., r_n)$ received codeword or vector

$\mathbf{e} = (e_1, e_2, \ldots., e_n)$ error pattern or vector

- Syndrome testing:
  - **S** is syndrome of **r**, corresponding to the error pattern **e**.

$$\mathbf{S} = \mathbf{r}\mathbf{H}^T = \mathbf{e}\mathbf{H}^T$$

Error detecting and correcting capability

- **Hamming distance** between two codewords is the number of elements in which they differ. This equals the number of ones in the sum of the two codewords, known as **Hamming weight.**
- In a vector subspace, the minimum Hamming distance $d_{\min}$ of all codewords can be determined by the weight of the codewords.
- Error correcting capability of a code is the maximum number of errors that are guaranteed to be correctable per codeword

$$t = \left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor$$

Hamming codes

- Hamming codes
  - Hamming codes are a subclass of linear block codes and belong to the category of *perfect codes*.
  - Hamming codes are expressed as a function of a single integer $m \geq 2$.

Code length : $n = 2^m - 1$

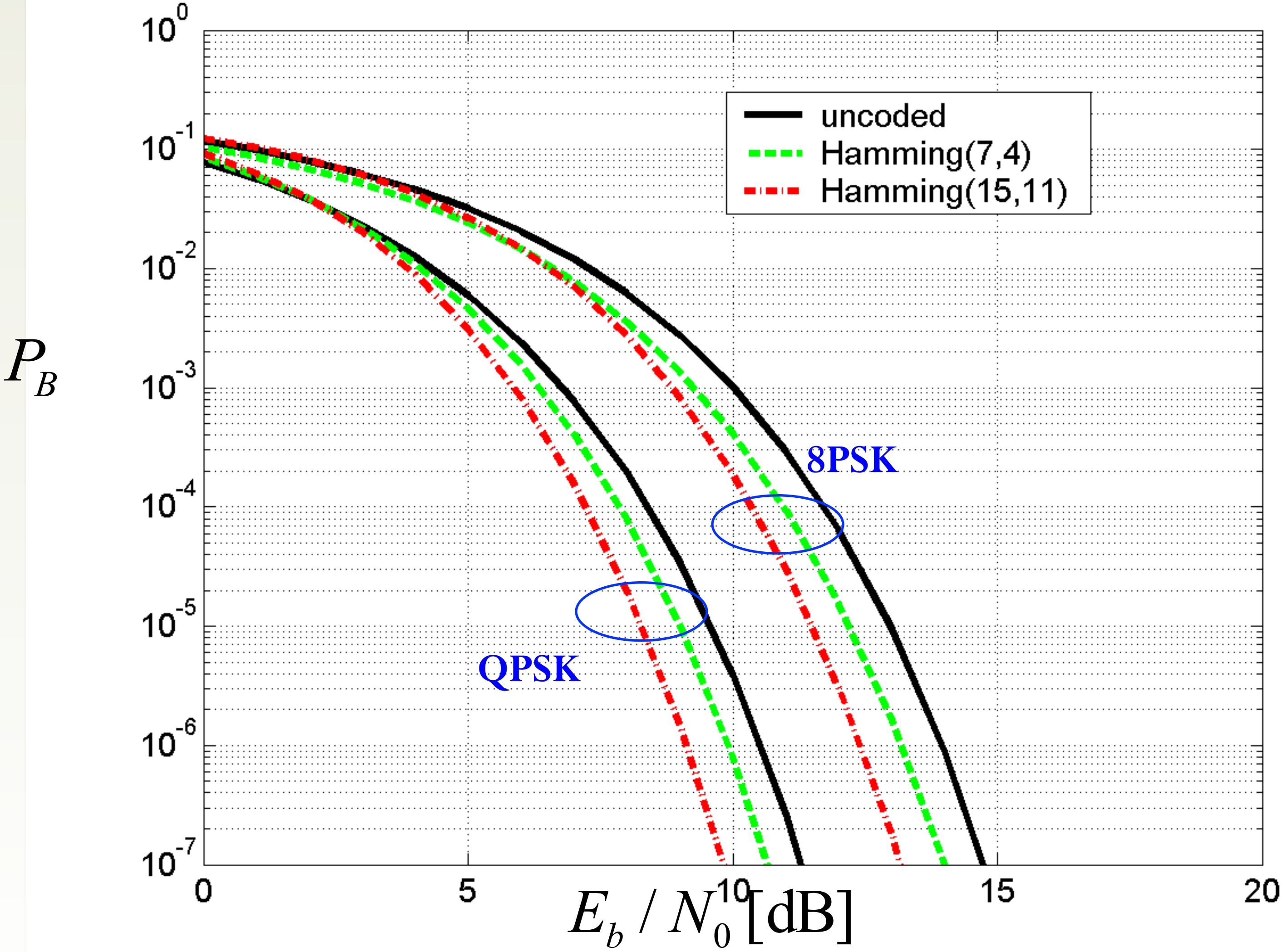
Number of information bits : $k = 2^m - m - 1$

Number of parity bits : $n\text{-}k = m$

Error correction capability : $t_{\min} = 1$

  - The columns of the parity-check matrix, **H**, consist of all non-zero binary m-tuples.
  - Because of the simplicity of Hamming codes, they are widely used in computer memory (RAM). In particular, a single-error-correcting *and* double-error-detecting variant commonly referred to as **SECDED.**

Example of Hamming codes



Cyclic block codes

- Cyclic codes are a subclass of linear block codes
- Better error capabilities than the simple Hamming code
- Encoding and syndrome calculation are easily performed using feedback shift-registers.
  - Hence, relatively long block codes can be implemented with a reasonable complexity.
- BCH and Reed-Solomon codes are cyclic codes.

Cyclic block codes

- A linear *(n,k)* code is called a Cyclic code if all cyclic shifts of a codeword are also a codeword.

$$\boxed{\mathbf{U} = (u_0, u_1, u_2, \ldots, u_{n-1})}$$

"*i*" cyclic shifts of **U**

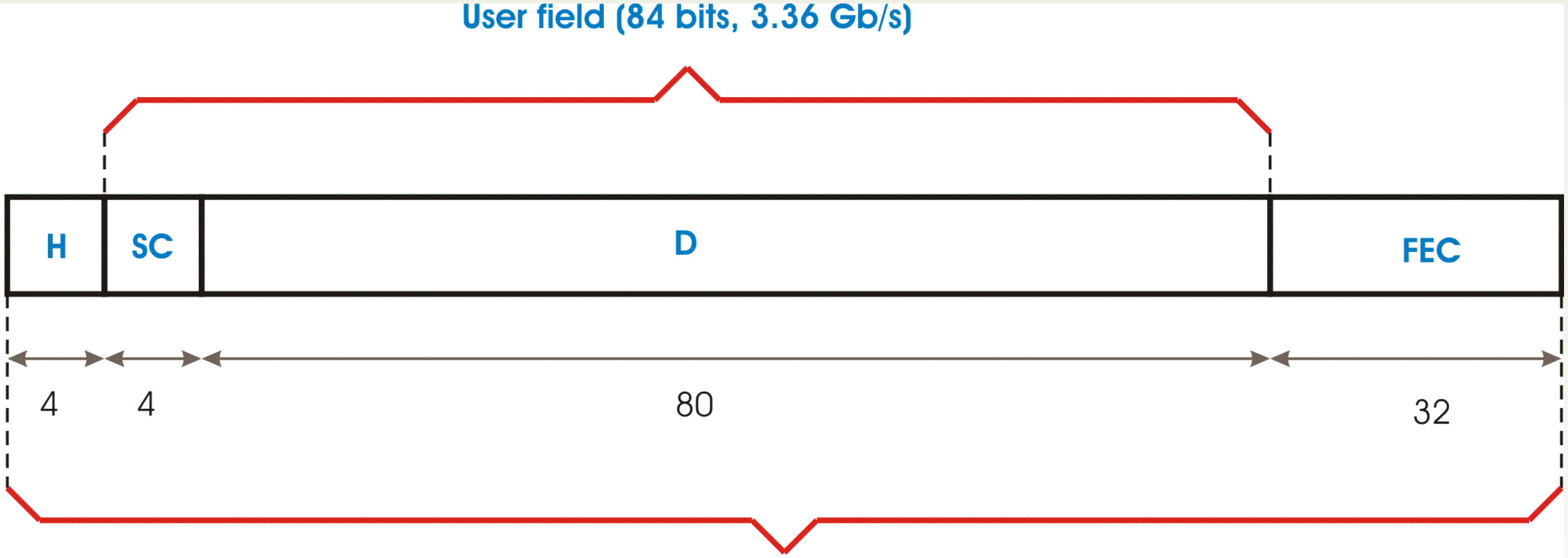
$$\boxed{\mathbf{U}^{(i)} = (u_{n-i}, u_{n-i+1}, \ldots, u_{n-1}, u_0, u_1, u_2, \ldots, u_{n-i-1})}$$

  - Example:

$$\mathbf{U} = (1101)$$

$$\mathbf{U}^{(1)} = (1110) \quad \mathbf{U}^{(2)} = (0111) \quad \mathbf{U}^{(3)} = (1011) \quad \mathbf{U}^{(4)} = (1101) = \mathbf{U}$$

GBT Frame



Frame: 1 SLHC Clock Cycle (120 bits, 4.8 Gb/s)

H: Header, 4 bits

SC:

- GBT control, 2 bits (80 Mb/s)
- Slow control port, 2bits (80 Mb/s)

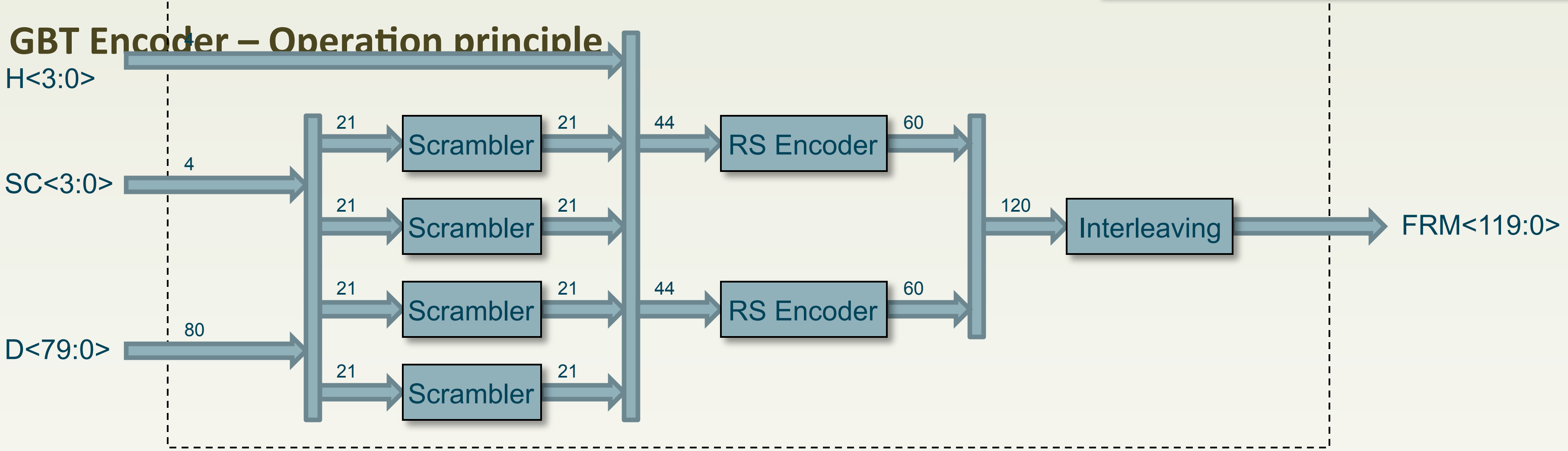
D: DATA/TTC/EC - User data, 80 bits (3.2 Gb/s)

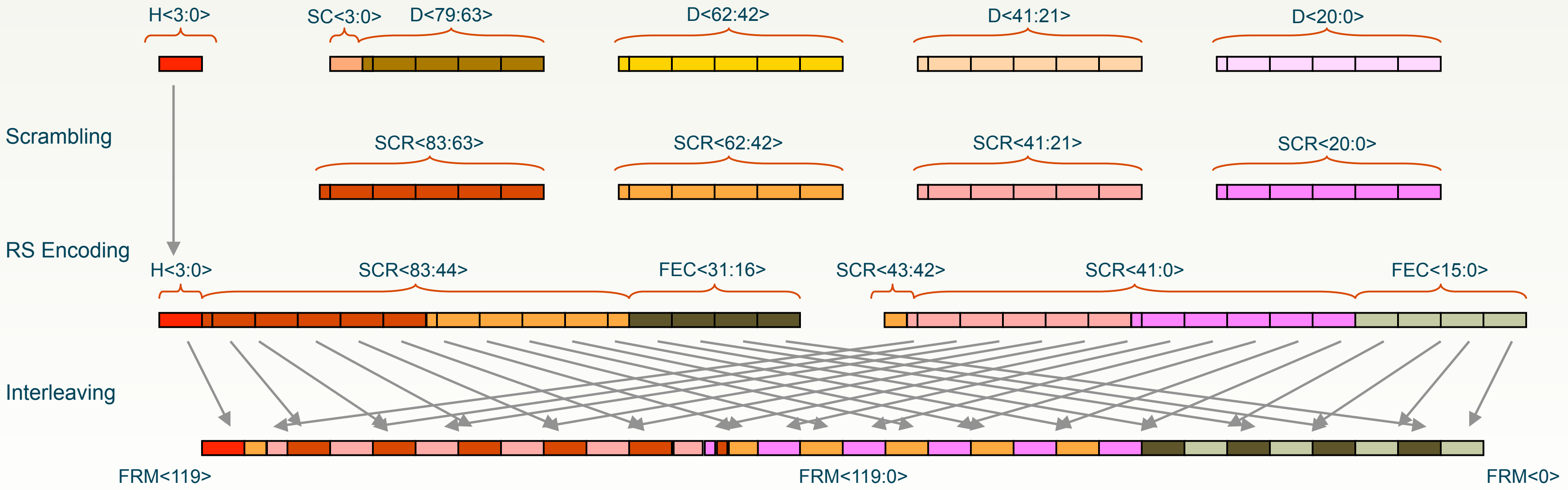
FEC: Forward Error correction, 32 bits

Ref. Paulo Moreira -CERN

Ref. Paulo Moreira -CERN

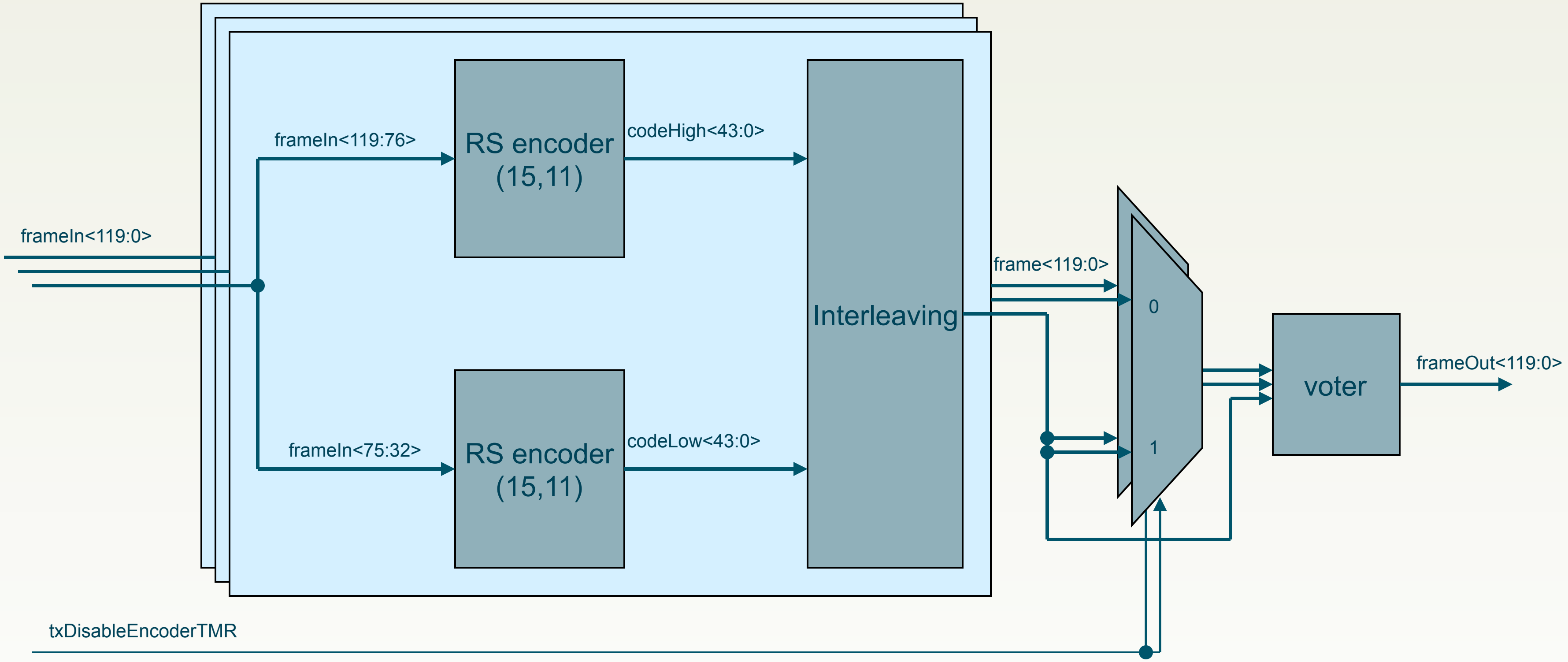
GBT Encoder – Operation principle





The Frame is shifted out MSB first, that is: FRM<119>, FRM<118>, … FRM<0> (The header shifts out first)

RS encoder

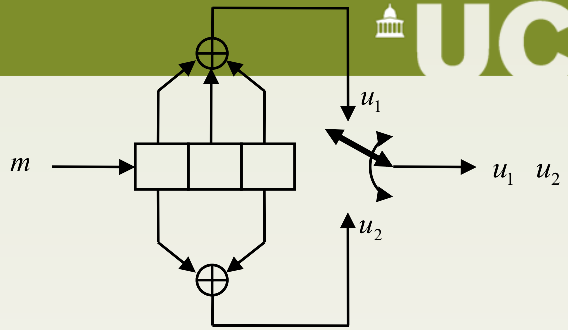


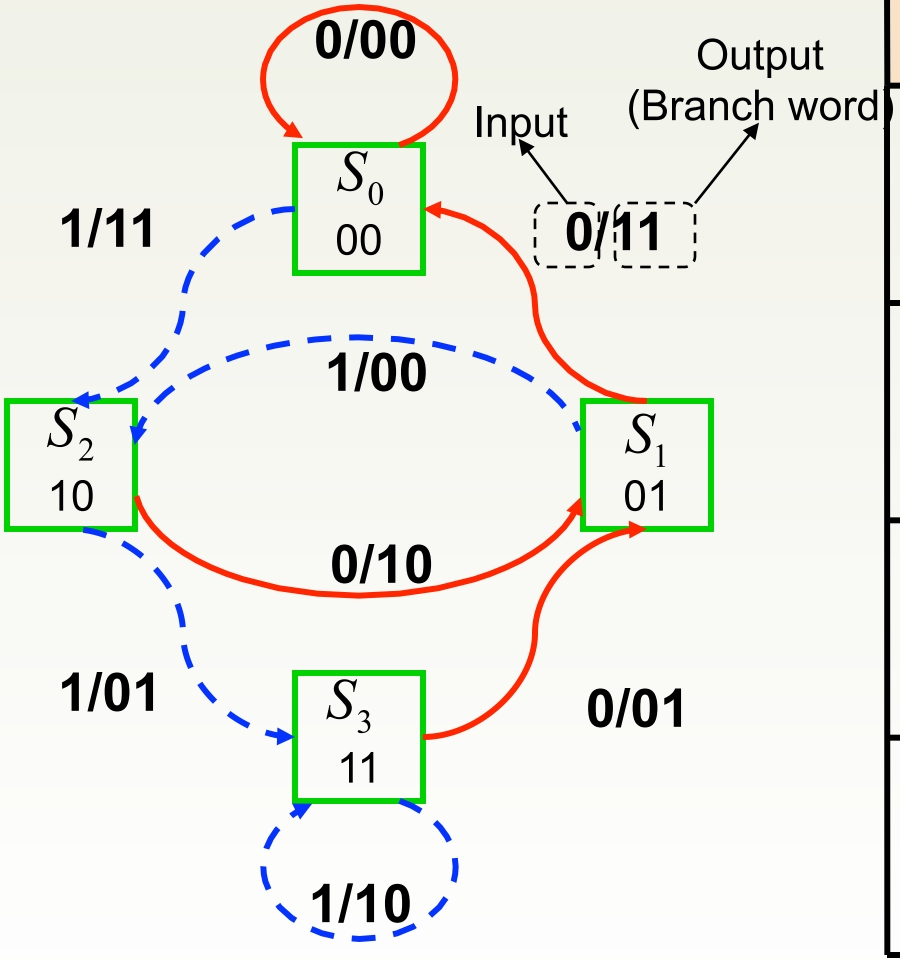
Ref. Paulo Moreira -CERN

Convolutional codes

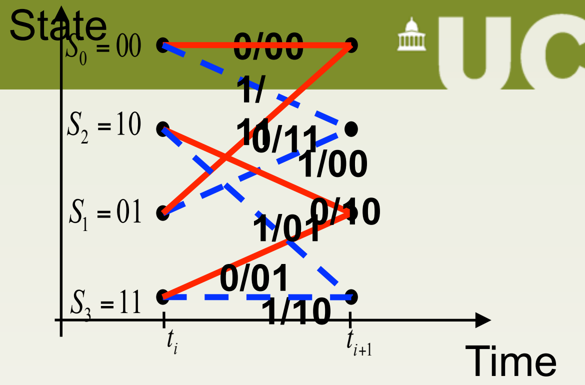
- Convolutional codes offer an approach to error control coding substantially different from that of block codes
    - encodes the entire data stream, into a single codeword
    - does not need to segment the data stream into blocks of fixed size (*Convolutional codes are often forced to block structure by periodic truncation*)
    - Has memory
    - Encodes on bit-by-bit basis
- 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
    - – Require memory and feedback
- Detection and error correction using ML
- Use of Viterbi decoder

State diagram – cont'd

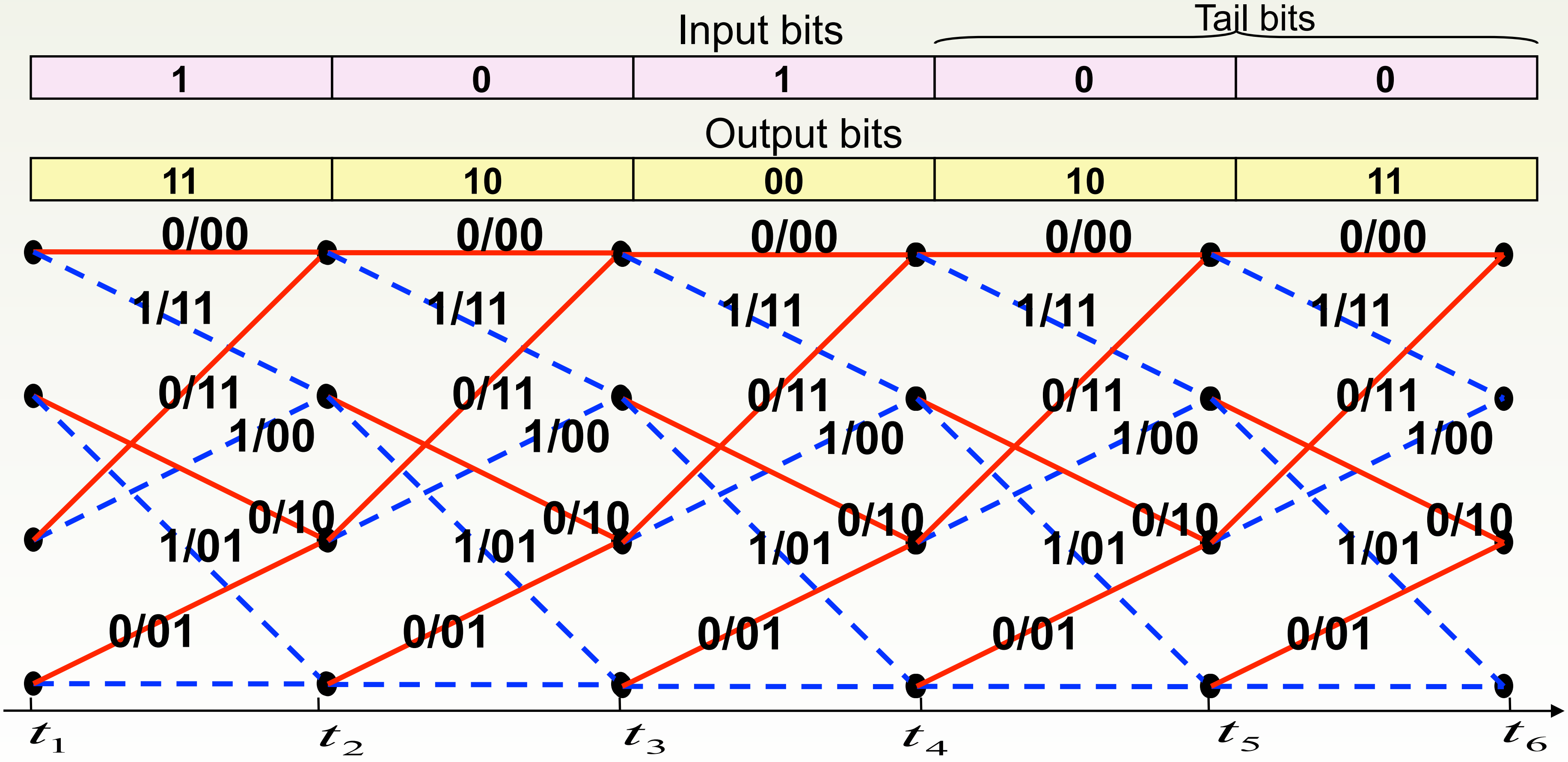




Current state	input	Next state	output
S_0 00	0	S_0	00
	1	S_2	11
S_1 01	0	S_0	11
	1	S_2	00
S_2 10	0	S_1	10
	1	S_3	01
S_3 11	0	S_1	01
	1	S_3	10



- A trellis diagram for the example code



What are Turbo codes

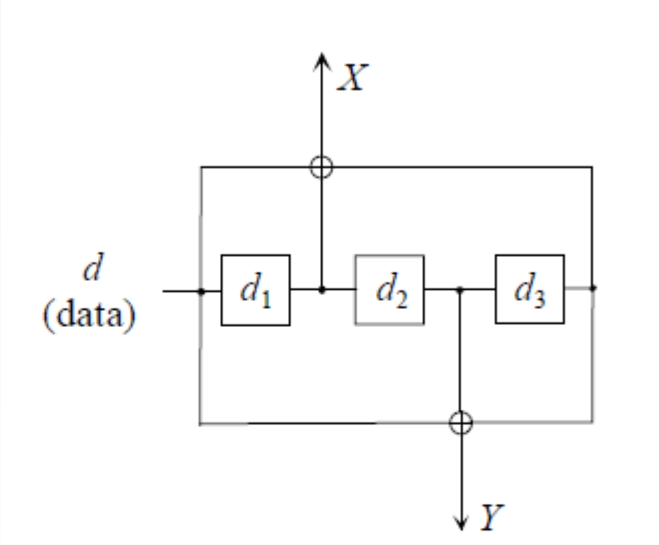
- Introduced by Claude Berrou *et al.* in 1993
- Based on convolutional codes with Viterbi decoding
- They combine two ideas:
  - We have a pair of encoders separated by a pseudorandom interleaver (turbo interleaver)
  - Iterative detection with feedback
  - Soft-input, Soft-output

Before Turbo codes….

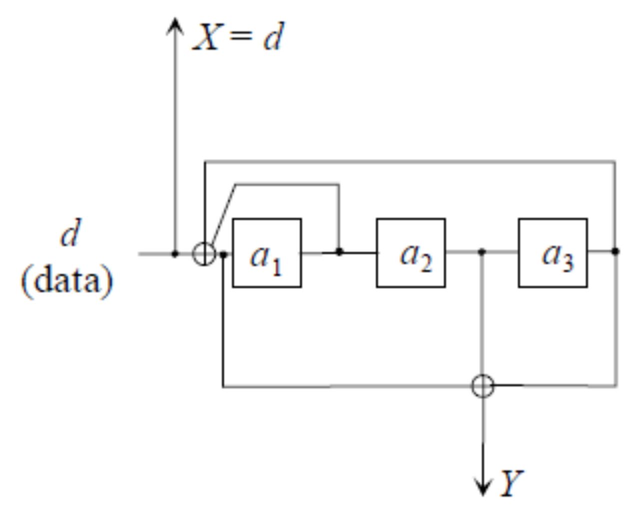
- Prior to turbo codes, the best constructions were serial concatenated codes based on an outer Reed-Solomon error correction code combined with an inner Viterbi-decoded short constraint length convolutional code.
- In the matter of channel coding and spectral efficiency, up to the invention of Turbo codes**, 3 dB or more** stood between what the theory promised and what real systems were able to offer.
- With the introduction of Turbo codes this gap has been removed, allowing communication systems to be designed with quasi-optimum performance

How does it work?

- Classic, non-recursive,



-



Both figures taken from “*The ten year old turbo codes are entering into service*”, Claude Berrou

How does it work cntd'

- Parallel con stematic
  convolution

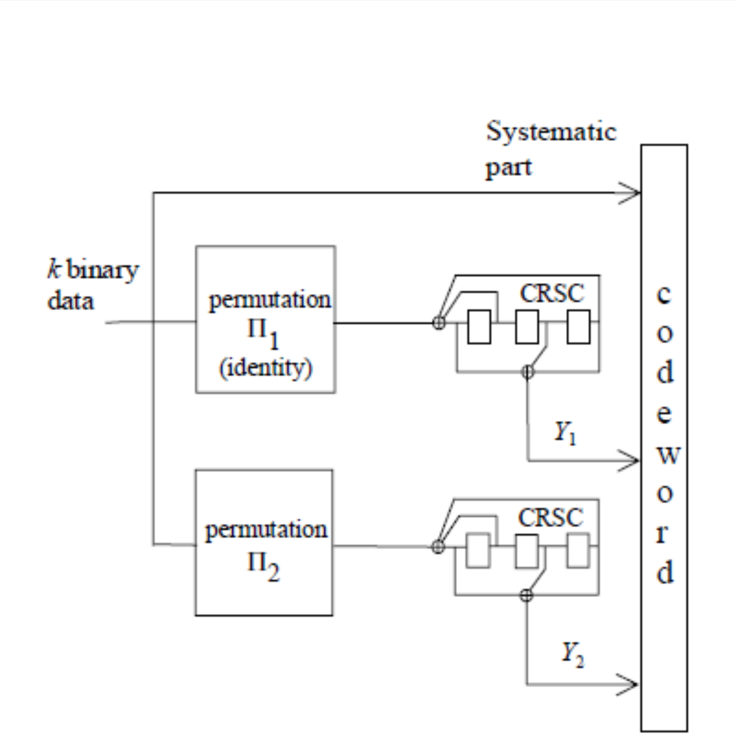
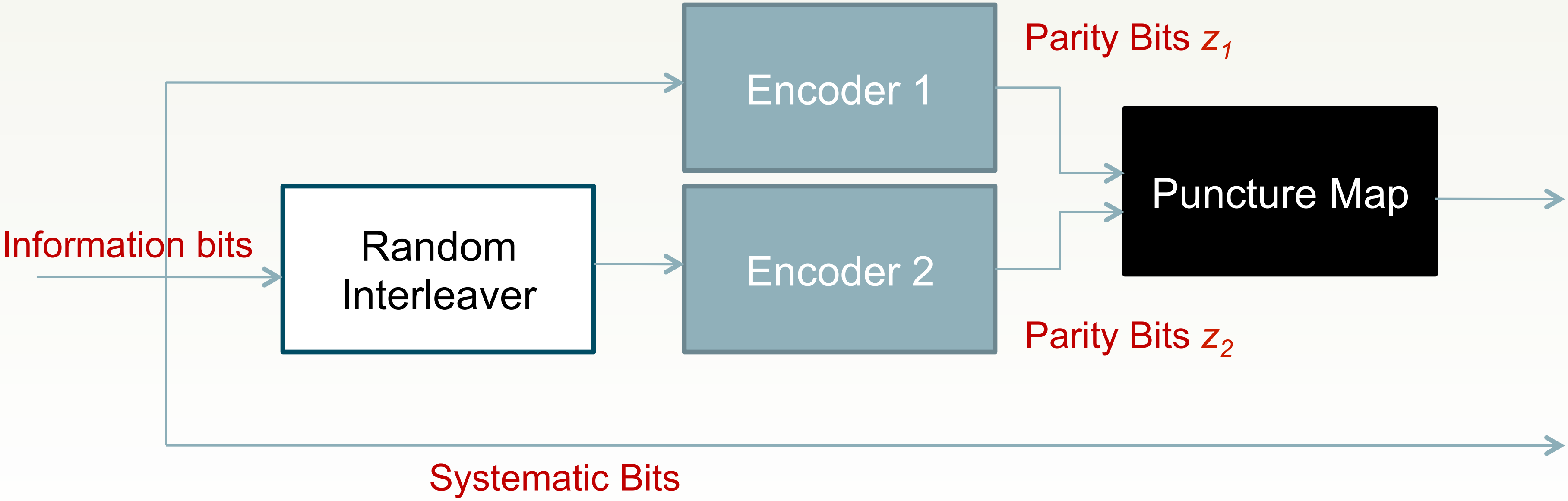


Figure taken from "*The ten year old turbo codes are entering into service*", Claude Berrou

Turbo encoding

- Based on parallel forward error correction encoding
- Two identical encoders



Turbo decoding

- Basic principle: exchange of probabilistic messages between all the processors dealing with the **same** data
- For instance, the decoding of the turbo code in the previous slides involves using two processors, namely two Soft-In/Soft-Out (SISO) decoders
- Each decoder processes its own data, and passes the so-called extrinsic information to the other decoder

Performance, decoding complexity and latency of Turbo codes

- Performance: near-optimum
- Complexity: bypass the exponential complexity of ML decoding by means of an iterative procedure. However, additional complexity compared to simple Viterbi decoding
- Latency: The latency issue is a weak point for turbo codes, which need several repeated calculations at the receiver side

SUMMARY

- Source, line and FEC coding (in very general terms)
- Design compromises
- Some system applications

Questions?