

Presentation in course

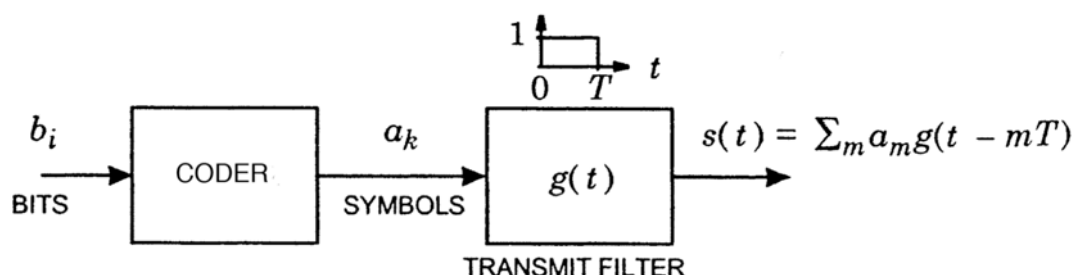
TTT4130 Digital Communication

Line codes and scrambling

Curriculum found in Barry, Lee, Messerschmitt
Chapter 19 (19.1, 19.3 and 19.5)



Line coding



The purpose of a line code is to match the output signal to the channel for baseband transmission. The line coder consists of a mapping of bits into symbols (nonlinear) and a pulseshaping filter (linear).



Properties of line codes

- Power spectrum
- Timing recovery properties
- DC-balance (zero DC component)
- Redundancy
- Linearity
- Polarity independence



Efficiency and redundancy

Maximum bitrate:

$$R = f_b \cdot \text{lb}(L) \quad (\text{lb} = \log_2)$$

L : no of output symbol levels

f_b : symbol rate in symb/s

B : actual bitrate in bit/s

Code efficiency: $\eta = B/R$

Redundancy: $r = 1 - \eta$



Example, 4B3T code

The 4B3T code is coding 4 input bits to 3 three-level symbols ($2^4 = 16$ combinations of input and $3^3 = 27$ potential output combinations).

Maximum bitrate: $R = f_b \cdot \lg 3$

Actual bitrate: $B = f_b \cdot 4/3$

Code efficiency: $\eta = R/B = 4/(3 \cdot \lg 3) = 0.84$

Redundancy: $r = 1 - 0.84 = 0.16$

Linear codes

Input symbols are $a_k = \pm 1$ and pulseshapes are defined in Fig 19.2

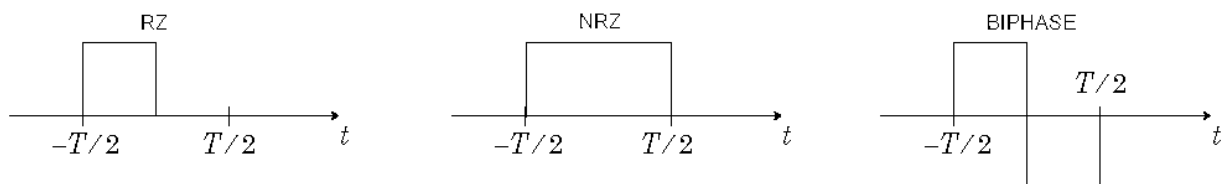


Fig. 19-2. Return-to-zero (RZ), non-return-to-zero (NRZ), and biphase pulses.

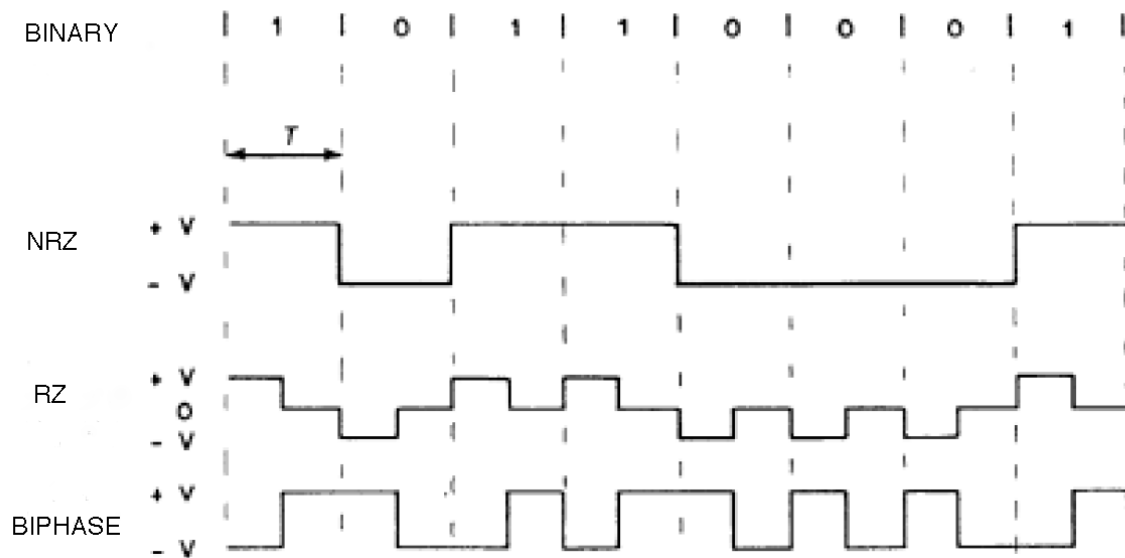
Biphase has zero DC component in each pulse

RZ and NRZ may have a DC component

Biphase and RZ has transitions in each pulse): good timing properties

NRZ may have long sequences of +1 or -1): poor timing properties

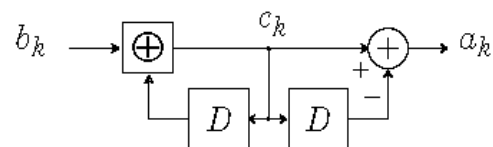
Coded sequences



AMI code (Alternate Mark Inversion)

$$a_k = \begin{cases} 0 & b_k = 0 \\ \pm 1 & b_k = 1 \end{cases} \quad (\text{alternating } -1 \text{ and } +1)$$

The AMI code may be generated as shown in the figure:



$$b_k = 0: \quad c_k = c_{k-1} \quad a_k = 0$$

$$b_k = 1: \quad c_k \neq c_{k-1} \quad a_k = \pm 1$$

c_k changes between 0 and 1 each time $b_k = 1$

+ zero DC component

- may contain long sequences of 0 (poor timing recovery)

DC-balance

Most transmission systems will be a.c. coupled. In metallic cables a transformer is used to avoid overvoltages. A.c. coupling may also be used due to implementation aspects. A.c. coupling means that d.c. is filtered out in a high-pass filter as shown in Fig 19.1a. This causes a slowly varying InterSymbol Interference that is denoted **baseline wander**. A Nyquist channel + a high pass filter in cascade will give a model for ISI as shown in Fig 19.1b.

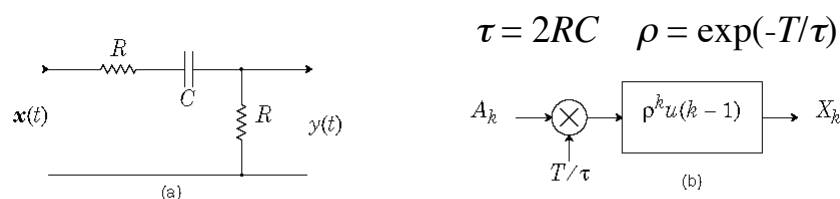


Fig. 19-1. a. An a.c. coupled circuit. b. A discrete-time system characterizing the postcursor ISI introduced by baseline wander in this circuit.

Running digital sum

By assuming $\rho = 1$, the overall time discrete impulse response may be approximated:

$$p_k = p(kT) = \begin{cases} 0 & k < 0 \\ 1 & k = 0 \\ -T/\tau & k > 0 \end{cases}$$

Output signal at time k :
$$y_k = \sum_{m=-\infty}^{\infty} a_m p_{k-m} = a_k - \frac{T}{\tau} \sum_{m=-\infty}^{k-1} a_m = a_k - \frac{T}{\tau} \cdot S_{k-1}$$

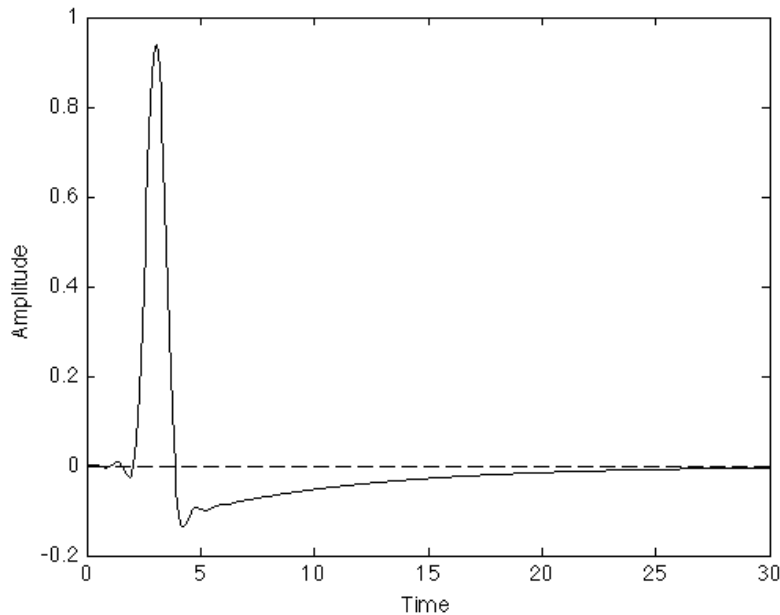
Running digital sum, RDS:
$$S_k = \sum_{m=-\infty}^k a_m$$

Intersymbol interference at time k :
$$ISI(k) = -\frac{T}{\tau} \cdot S_{k-1}$$

$$\max(ISI) = -\frac{T}{\tau} \cdot \min(S_{k-1}), \quad \max(ISI) \rightarrow \infty \text{ for independent } a_k$$

For a balanced line code RDS is bounded, and the DC content is zero.

Impulse response for DC cutoff, simulation example illustrating baseline wander



Cosine rolloff 100% +

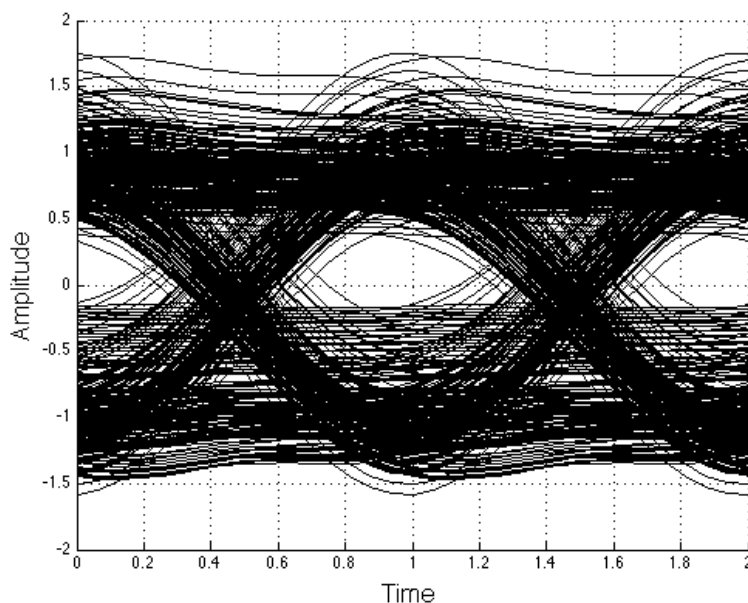
$$H(f) = \frac{j\omega}{j\omega + 2\pi \cdot 0.02}$$

3 dB cutoff frequency:
 $0.02/T$

$$\tau = T/(2\pi \cdot 0.02) = 7.96T$$



Eye curve for NRZ code with DC cutoff



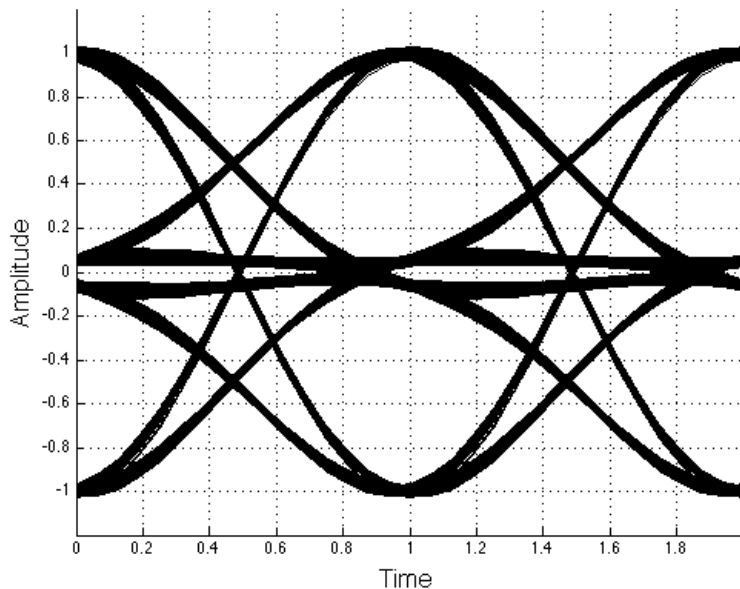
Cosine rolloff 100% +

$$H(f) = \frac{j\omega}{j\omega + 2\pi \cdot 0.02}$$

NRZ is an unbalanced
line code): serious
baseline wander



Eye curve for AMI code with DC cutoff



Cosine rolloff 100% +

$$H(f) = \frac{j\omega}{j\omega + 2\pi \cdot 0.02}$$

AMI is a balanced
line code): small ISI

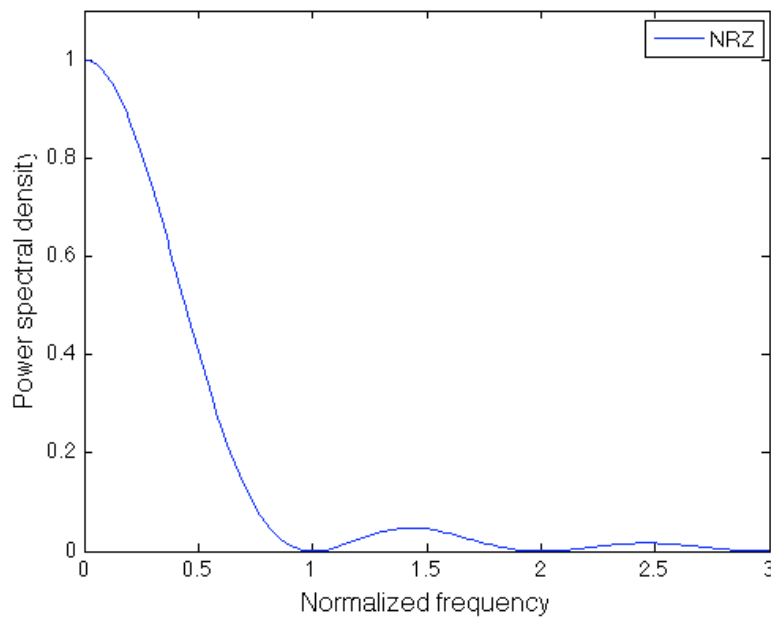
Power spectrum of a line code

Transmitted signal:
$$X(t) = \sum_{k=-\infty}^{\infty} a_k g(t - kT)$$

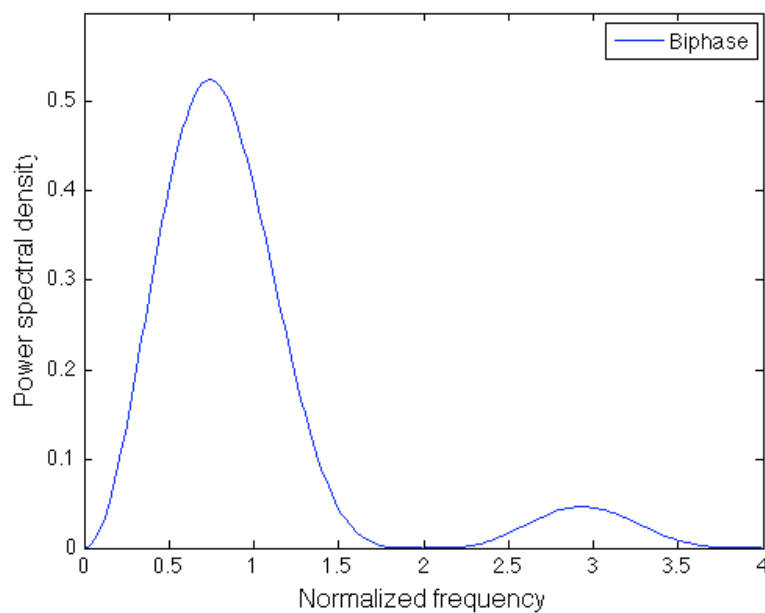
Power spectrum:
$$S_X(f) = \frac{1}{T} S_a(e^{j2\pi fT}) \cdot |G(f)|^2$$

$S_a(z)$: Time discrete power spectrum of sequence $\{a_k\}$

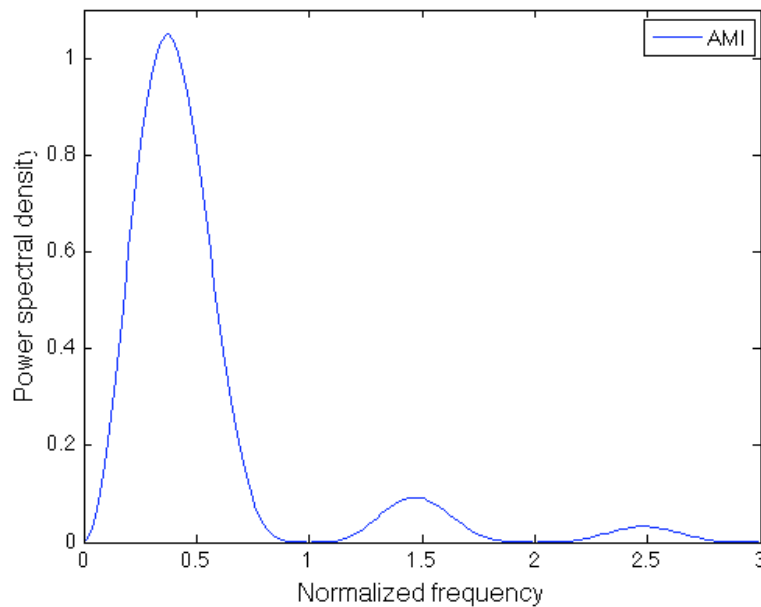
Power spectrum NRZ code



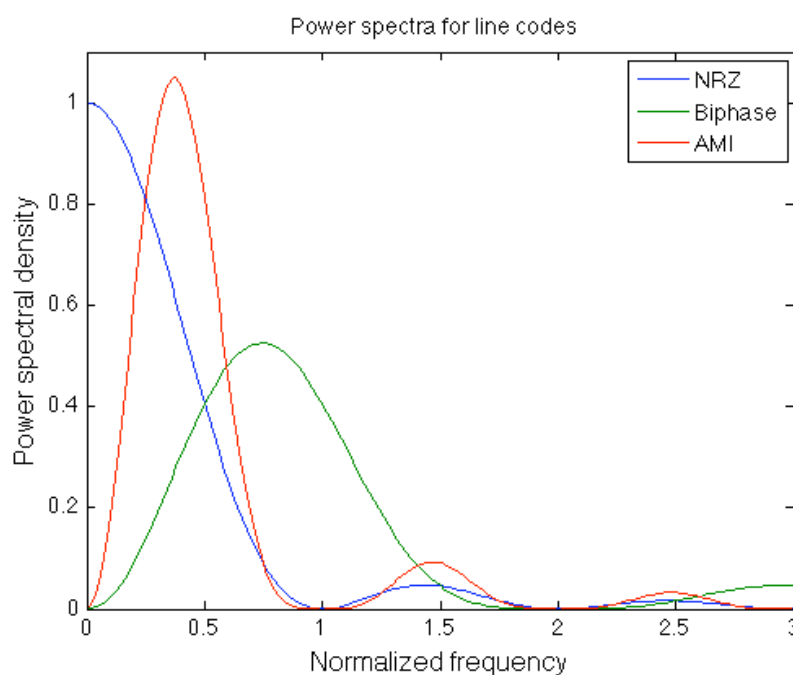
Power spectrum Biphase



Power spectrum AMI code



Power spectrum for three simple codes



**NB: All codes
have equal
average power**

Comparison of simple codes

Code	Balanced	Timing properties	Minimum bandwidth
RZ	no	good	$1/(2T)$
NRZ	no	poor	$1/(2T)$
Biphase	yes	good	$1/T$
AMI	yes	poor	$1/(2T)$



Ternary (3-level) codes

An $kBnT$ block code is coding k bits into n 3-level symbols

k	n	Code	Efficiency
1	1	1B1T	63%
3	2	3B2T	95%
4	3	4B3T	84%
6	4	6B4T	95%
7	5	7B5T	89%



One alternative of a 4B3T code

Table 19-1. An example of a 4B3T code illustrating a bimode block line code.

Input Block	Ternary Output Block		Block
	Mode A	Mode B	Digital Sum
0000	+ 0 -	+ 0 -	0
0001	- + 0	- + 0	0
0010	0 - +	0 - +	0
0011	+ - 0	+ - 0	0
0100	++ 0	-- 0	±2
0101	0 ++	0 --	±2
0110	+ 0 +	- 0 -	±2
0111	+++	-++	±3
1000	++-	--+	±1
1001	-++	+- -	±1
1010	+ - +	- + -	±1
1011	+ 0 0	- 0 0	±1
1100	0 + 0	0 - 0	±1
1101	0 0 +	0 0 -	±1
1110	0 + -	0 + -	0
1111	- 0 +	- 0 +	0



Binary codes

Binary codes are relevant for optical transmission.

The number of balanced codewords of length n is given by:

$$N = \frac{n!}{(n/2)!(n/2)!}$$

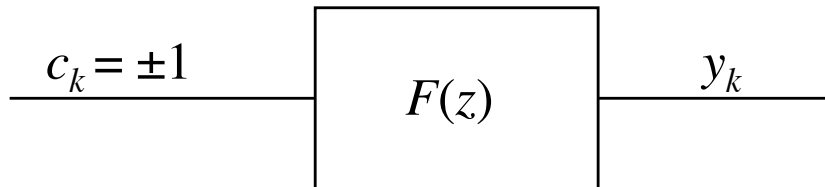
Corresponding codes are shown in the table

n	N	$\log_2 N$	k	Efficiency	
2	2	1	1	50%*	
4	6	2.58	2	50%*	* errors in textbook
6	20	4.32	4	67%*	
8	70	6.13	6	75%*	
10	252	7.97	7	70%*	

Due to low efficiency, nBmB codes are used instead, where $m = n + 1$ (e.g. 5B6B)



Partial response



$$F(z) = (1 - z^{-1})^m \cdot (1 + z^{-1})^n$$

Alternatives of partial response

Type partial response	$n =$	$m =$	$F(z) =$
Dicode	0	1	$1 - z^{-1}$
Duobinary	1	0	$1 + z^{-1}$
Modified duobinary	1	1	$(1 + z^{-1})(1 - z^{-1}) = 1 - z^{-2}$

$$1 - z^{-1} = 1 - \exp(-j2\pi fT) = 2j \cdot \exp(-j\pi fT) \cdot \sin(\pi fT), \quad) : \text{ zero at } f = 0$$

$$1 + z^{-1} = 1 + \exp(-j2\pi fT) = 2 \cdot \exp(-j\pi fT) \cdot \cos(\pi fT), \quad) : \text{ zero at } f = 1/(2T)$$

Duobinary partial response

$$y_k = c_k + c_{k-1}$$

c_k	c_{k-1}	Output y_k
-1	-1	-2
-1	1	0
1	-1	0
1	1	2

For duobinary, a two-level signal is coded into three levels. Modified duobinary and dicode will also have three-level output for binary input. Other alternatives of partial response will have more output levels.



Spectral shaping by partial response

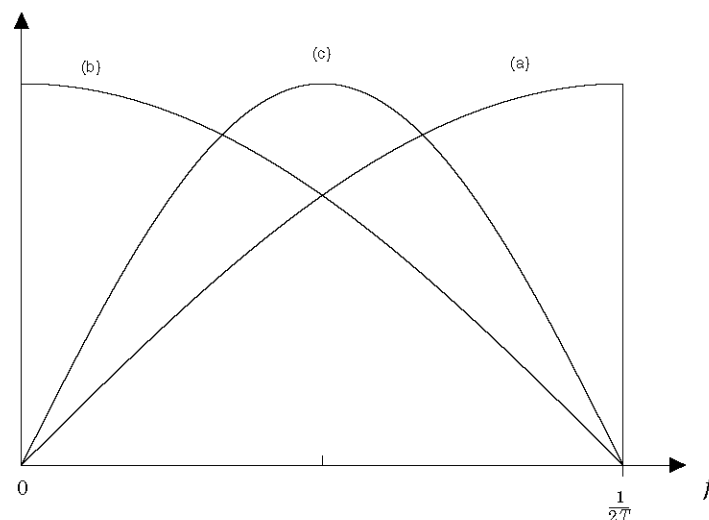


Fig. 19-11. The frequency response of $F(D)$ plotted up to half the symbol rate. (a) Dicod, (b) duobinary, and (c) modified duobinary.



Decoding of partial response

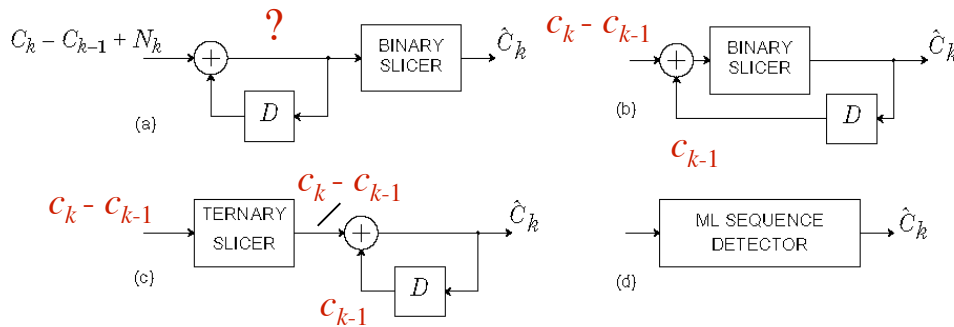
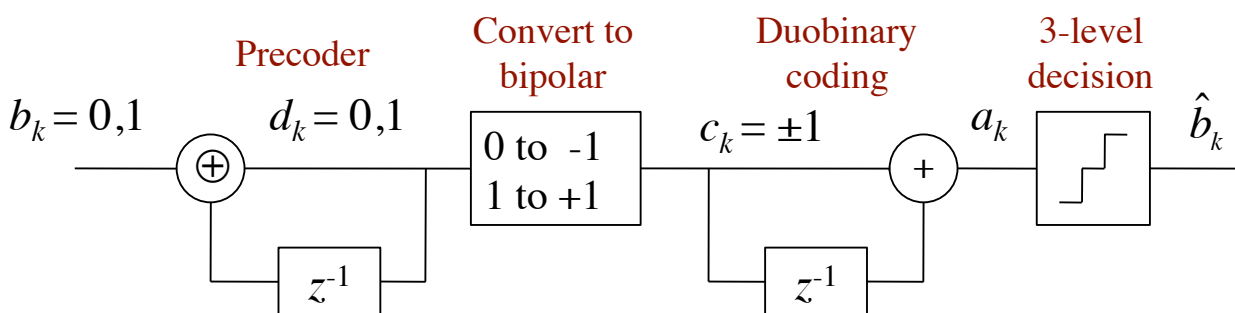


Fig. 19-12. Four options for detecting a duobinary signal. a. A linear equalizer filter $F^{-1}(D)$. b. A decision-feedback filter with binary slicer. c. A filter $F^{-1}(D)$ after a ternary slicer. d. A ML sequence detector (Viterbi algorithm.)

Precoding of partial response (duobinary)



$$\begin{array}{llllll}
 b_k = 0 & d_k = d_{k-1} & c_k = c_{k-1} & a_k = c_k + c_{k-1} = \pm 2 & \hat{b}_k = 0 \\
 b_k = 1 & d_k \neq d_{k-1} & c_k \neq c_{k-1} & a_k = c_k + c_{k-1} = 0 & \hat{b}_k = 1
 \end{array}$$

Decoding depends only on current received signal sample):
no error propagation.

Precoding may be used also for modified duobinary and dicode.

Scrambling

- Most transmission systems are based on an assumption of random input data): statistically independent bits and $p("0") = p("1") = 1/2$
- Practical input data often differ from this assumption and may for instance contain long strings of ones or zeroes
- Scrambling is a way to ensure approximately random data
- Scrambling is based on pseudorandom sequences generated by linear feedback shift registers.

Linear feedback shift register

Number of states in a shift register of length n : 2^n

The output sequence is periodic, and for a

maximum length shift register the period is: $2^n - 1$

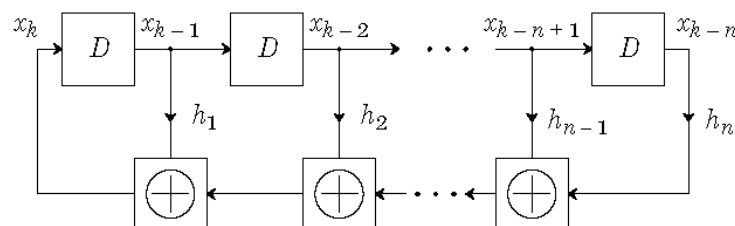


Fig. 19-18. A linear feedback shift register with binary input. The coefficients are binary, and the summation is modulo-two.

Frame-synchronized scrambler

Assuming error free transmission:

$$\hat{b}_k = \hat{c}_k \oplus x_k = c_k \oplus x_k = b_k \oplus x_k \oplus x_k = b_k$$

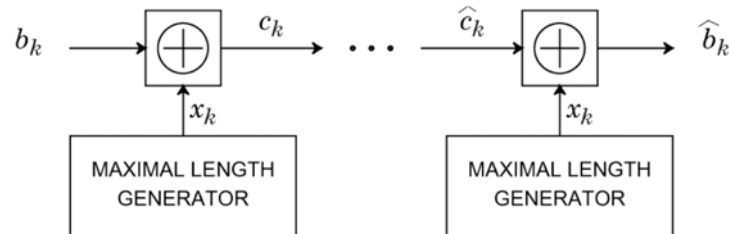


Fig. 19-19. A frame-synchronized scrambler where the maximal-length generator is as in Fig. 19-18.

- The transmitter and receiver shift registers have to be synchronized
- + One bit error per transmission error

Self synchronized scrambler

Assuming error free transmission:

$$\hat{b}_k = \hat{c}_k \oplus \hat{d}_k = c_k \oplus d_k = b_k \oplus d_k \oplus d_k = b_k$$

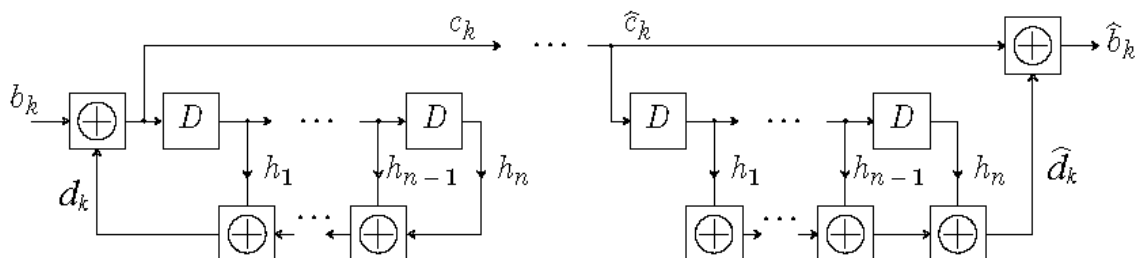


Fig. 19-20. A self-synchronized scrambler.

- + Automatic synchronization
- Minimum three bit errors per transmission error

Applications of scrambling

- Echo cancelling
 - Two different scramblers, one for each direction of transmission
- Synchronization
- Adaptive equalization
- Interference cancellation
- ...



Maximum length shift registers

Length of Shift Register	Feedback Taps	Period of Sequence
3	1, 3	7
4	1, 4	15
5	2, 5	31
6	1, 6	63
7	1, 7	127
8	1, 6, 7, 8	255
9	4, 9	511
10	3, 10	1,023
11	2, 11	2,047
12	2, 10, 11, 12	4,095
13	1, 11, 12, 13	8,191
14	2, 12, 13, 14	16,383
15	14, 15	32,767
16	11, 13, 14, 16	65,535
17	14, 17	131,071
18	11, 18	262,143
19	14, 17, 18, 19	524,287
20	17, 20	1,048,575



Table of maximum length shift registers

Table 19-2. Minimal weight primitive polynomials of orders two through 34 [36]. Each entry in the table is an octal number, which when converted to binary specifies the coefficients of the polynomial $h(D)$. The most significant (left-most) bit is $h_n = 1$ and the least significant (right-most) bit is $h_0 = 1$.

Order	Polynomial	Order	Polynomial
2	7	19	2000047
3	13	20	4000011
4	23	21	10000005
5	45	22	20000003
6	103	23	40000041
7	211	24	100000207
8	435	25	200000011
9	1021	26	400000107
10	2011	27	100000047
11	4005	28	200000011
12	10123	29	400000005
13	20033	30	1004000007
14	42103	31	2000000011
15	100003	32	4002000007
16	210013	33	100000020001
17	400011	34	201000000007
18	1000201		