Line Coding for Digital Communication

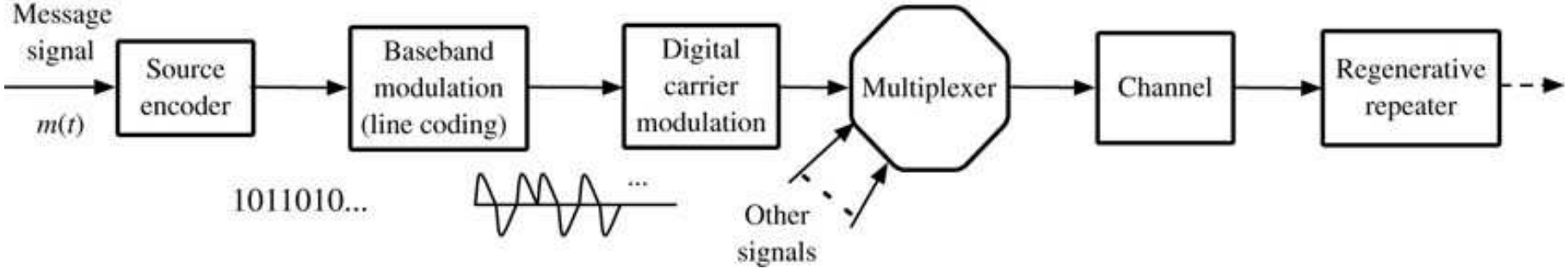
How do we transmit bits over a wire, RF, fiber?

- Line codes, many options
- Power spectrum of line codes, how much bandwidth do they take
- Clock signal and synchronization
- Common serial communications systems

Line Coding

- Goal is to transmit binary data (e.g., PCM encoded voice, MPEG encoded video, financial information)
- Transmission distance is large enough that communication link bandwidth is comparable to signal bandwidth.
- Connections between nearby logic gates have bandwidth greater than switching speed, so no line coding is needed.

  But longer connections use pulse shaping.
- Multiple links may be used, with regenerative repeaters
- First consider baseband communication (e.g., single twisted pair)



Line Coding Requirements

- Small transmission bandwidth
- Power efficiency: as small as possible for required data rate and error probability
- Error detection/correction
- Suitable power spectral density, e.g., little low frequency content
- Timing information: clock must be extracted from data
- Transparency: all possible binary sequences can be transmitted

Line Code Examples

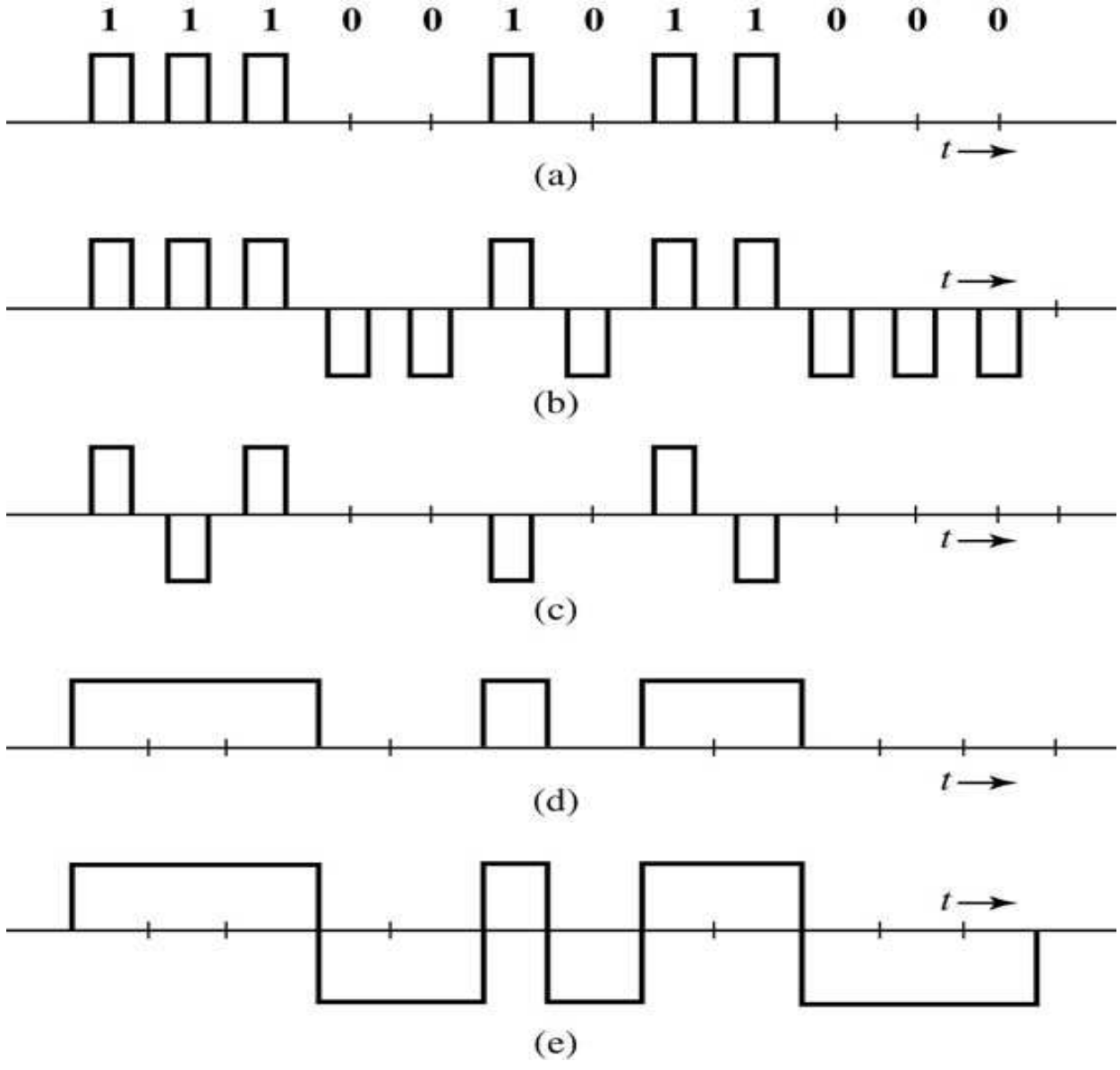
on-off (RZ)

polar (RZ)

bipolar/AMI (RZ)

on-off (NRZ)

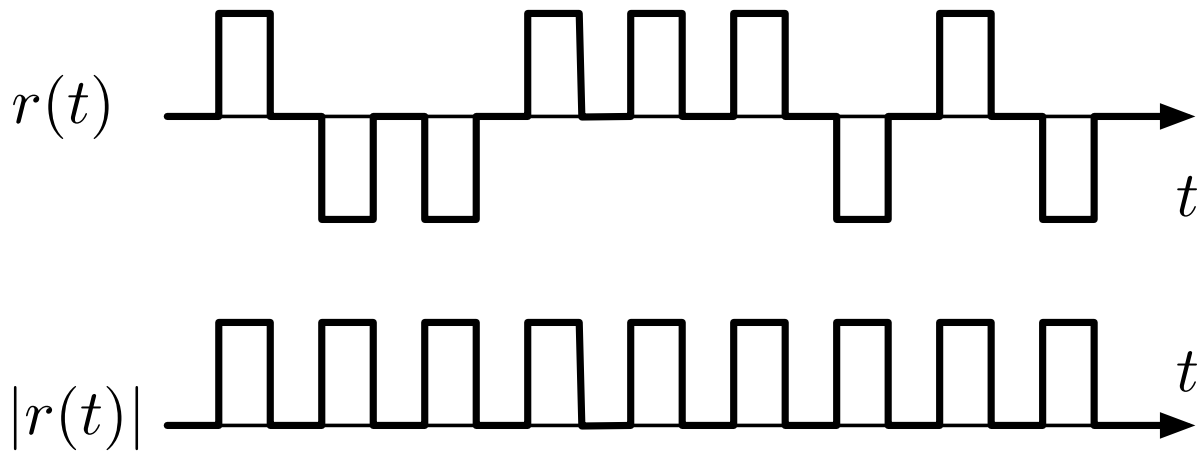
polar (NRZ)



RZ = return to zero, NRZ = non return to zero

Timing Signal

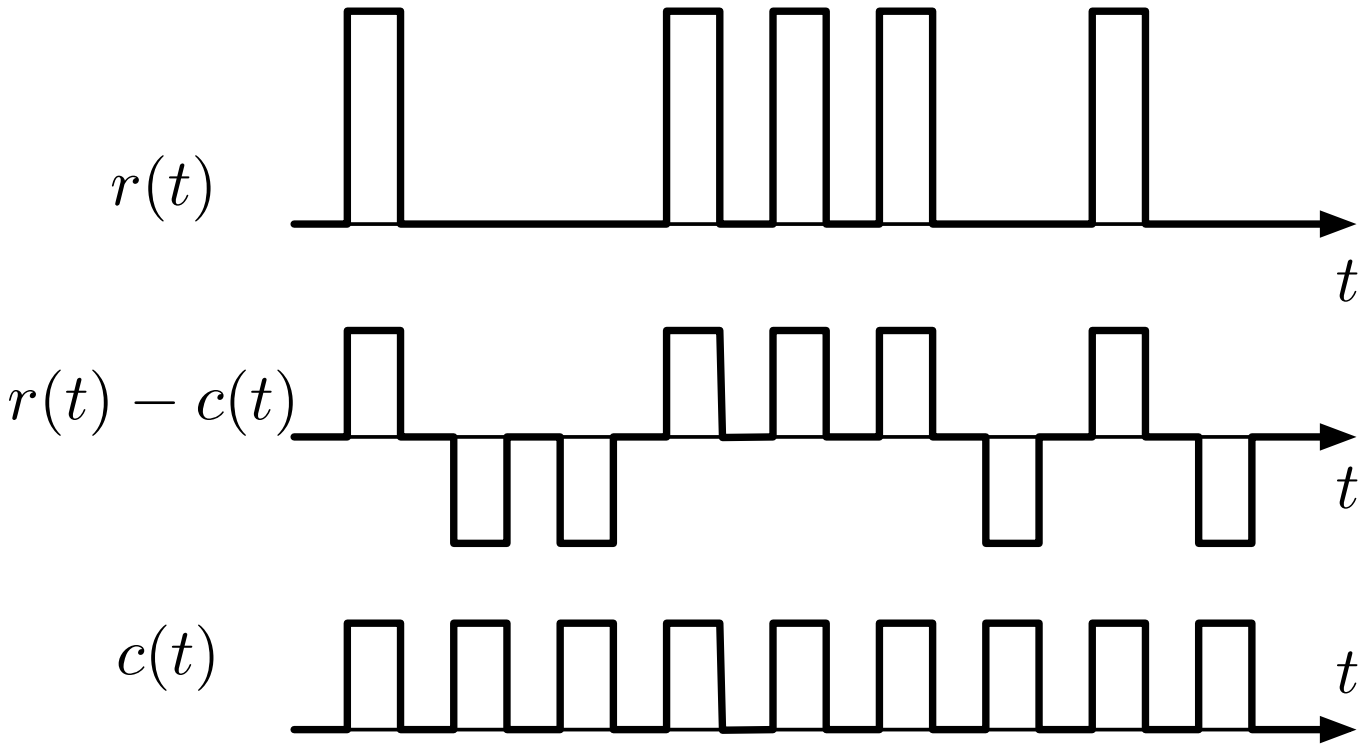
- We want to be able to easily extract the timing information from the signal.
- Consider the polar RZ $r(t)$ shown below. If we take the absolute value we get a timing signal.



- A line code where it is easy to extract the timing signal is called a transparent code. This is why many codes are designed the way they are.
- In practice the timing signal will be cleaned up by a narrowband bandpass filter before it is used to extract the bits in the input signal.

Timing Signal (cont.)

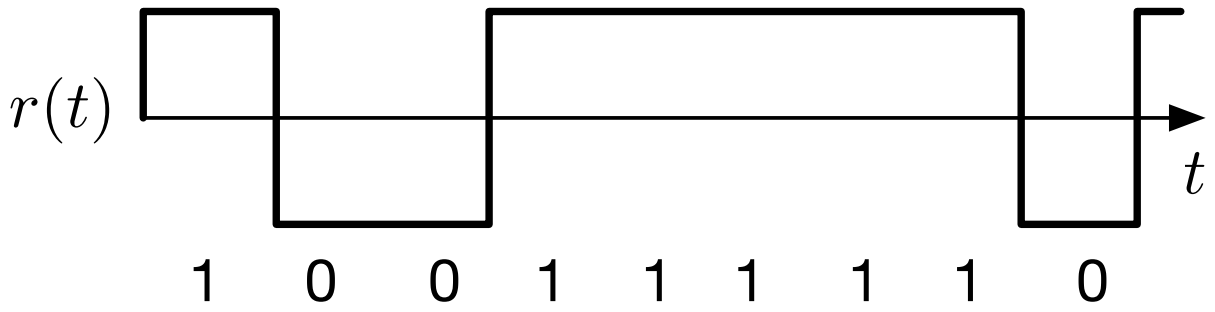
- Another approach is on-off keying (OOK), with either RZ or NRZ codes. RZ case is shown here.



- The RZ OOK signal is a RZ binary signal plus a RZ clock signal.

Timing Signal

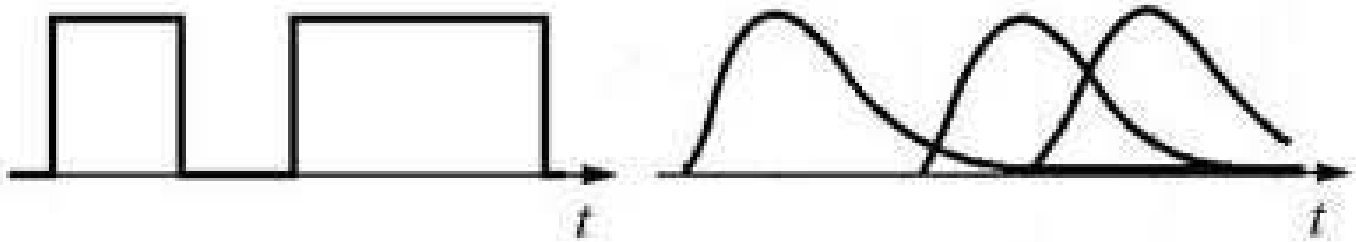
- NRZ codes can be more problematic. Long strings of 1's or 0's can cause loss of synchronization.



- Many codes limit the number of consecutive runs of 1's or 0's, and force bit changes after a given number of bits.
- The example we'll look at in the next lab forbids 6 1's in a row, and adds an extra zero bit after 5 1's.
- This is another example of bit stuffing that we saw last lecture.

Power Spectral Density of Line Codes

- The output distortion of a communication channel depends on power spectral density of input signal
- Input PSD depends on
  - pulse rate (spectrum widens with pulse rate)
  - pulse shape (smoother pulses have narrower PSD)
  - pulse distribution
- Distortion can result in smeared channel output; output pulses are (much) longer tthan input pulses
- Intersymbol interference (ISI): received pulse is affected by previous input symbols



Power Spectral Density (review)

For an energy signal $g(t)$, the energy spectral density is the Fourier transform of the autocorrelation:

$$\psi_g(t) = R_g(t) = \int_{-\infty}^{\infty} g(u)g(u+t)\,du \implies |G(f)|^2 = \mathcal{F}\{R_g(t)\}$$

The autocorrelation of a periodic signal is periodic.

$$R_g(t) = \frac{1}{T}\int_0^T g(u)g(u+t)\,du = \sum_{n=-\infty}^{\infty} |G_n|^2 e^{j2\pi nt}$$

For a power signal, autocorrelation and PSD are average over time. Define

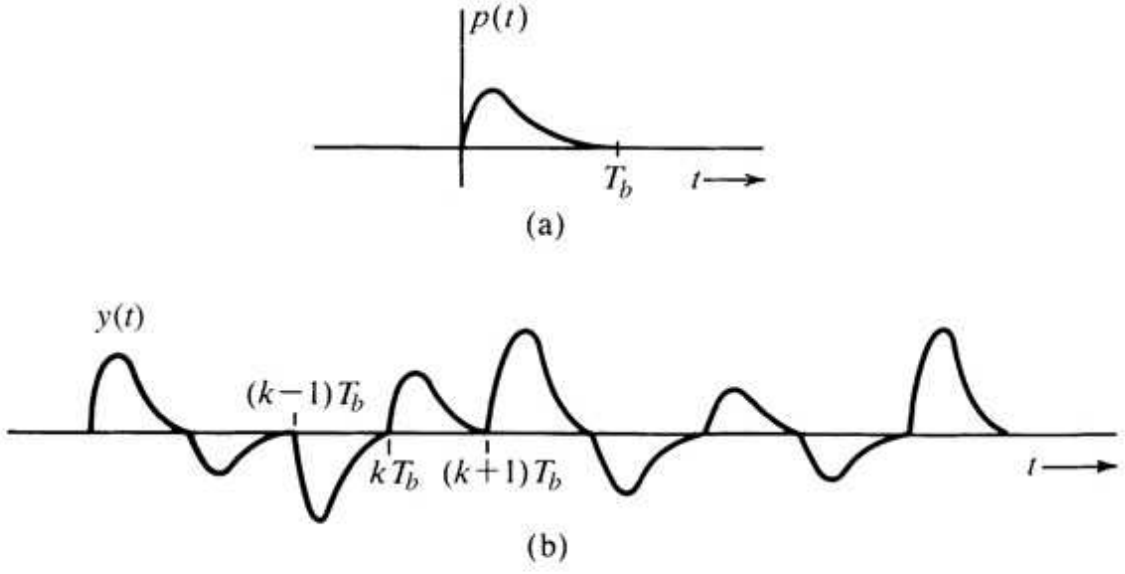
$$g_T(t) = \Pi(t/T)g(t) = \begin{cases} g(t) & |t| < T/2 \\ 0 & |t| > T/2 \end{cases}$$

Then

$$R_g(t) = \lim_{T\to\infty} \frac{R_{g_T}(t)}{T} \implies S_g(f) = \lim_{T\to\infty} \frac{|G_T(f)|^2}{T}$$

PSD of Line Codes

The PSD of a line code depends on the shapes of pulses that correspond to digital values. Consider PAM, pulse amplitude modulation.



The transmitted signal is the sum of weighted, shifted pulses.

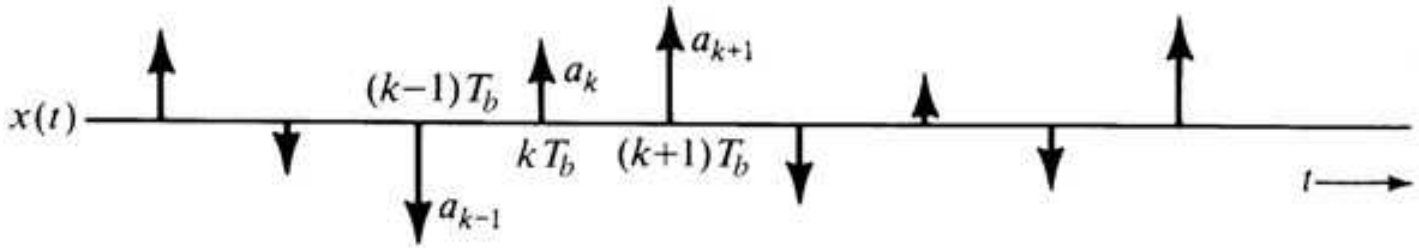
$$y(t) = \sum_{k=-\infty}^{\infty} a_k p(t - kT_b)$$

where T_b is pulse periox. (Pulses may be wider than T_b.)

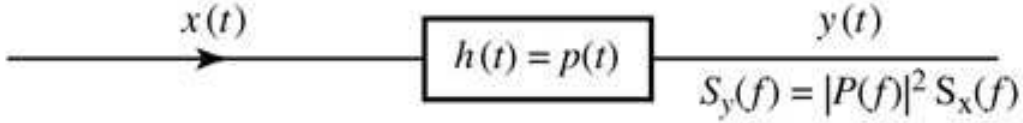
PSD of Line Codes (cont.)

PSD depends on pulse shape, rate, and digital values $\{a_k\}$.

We can simplify analysis by representing $\{a_k\}$ as impulse train.



(c)



(d)

PSD of $y(t)$ is $S_y(f) = |P(f)|^2 S_x(f)$.

- $P(f)$ depends only on the pulse, independent of digital values or rate.
- $S_x(f)$ increases linearly with rate $1/T_b$ and depends on distribution of values of $\{a_k\}$. E.g., $a_k = 1$ for all k has narrower PSD.

PSD of Impulse Train

Let's find the autocorrelation of an impulse train.

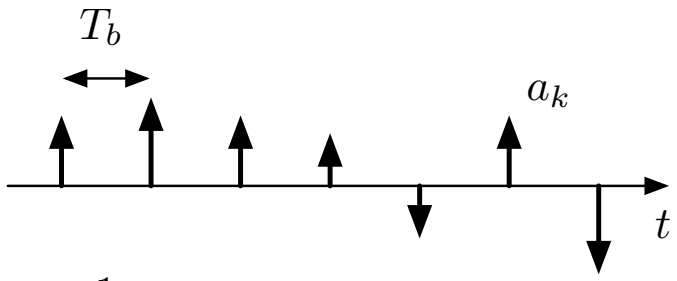
$$x(t) = \sum_{k=-\infty}^{\infty} a_k \delta(t - kT_b)$$

In discrete time the signal is

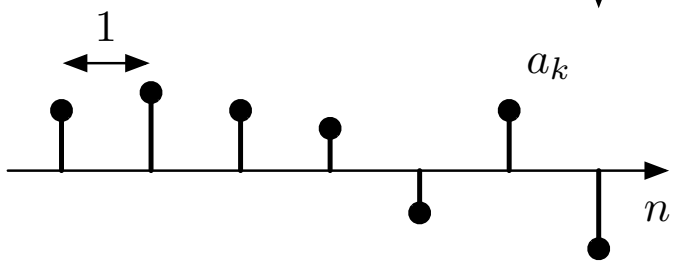
$$x[n] = \sum_{k=-\infty}^{\infty} a_k \delta[tn - k]$$

This is illustrated below.

$$x(t) = \sum_{k} a_k \delta(t - kT_b)$$



$$x[n] = \sum_{k} a_k \delta[n - k]$$



PSD of Impulse Train (cont.)

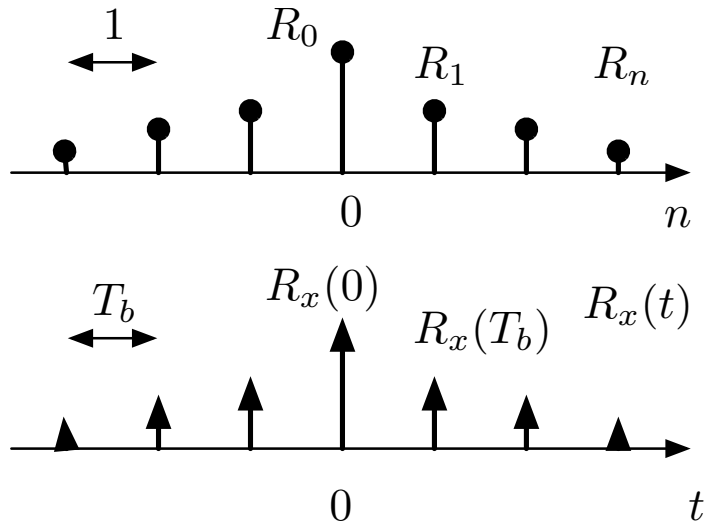
The autocorrelation in discrete time is

$$R_n = \lim_{N\to\infty} \frac{1}{2N} \sum_{k=-N}^{N} a_k a_{k-n}$$

The continuous time autocorrelation is therefore

$$R_x(t) = \frac{1}{T_b} \sum_{n=-\infty}^{\infty} R_n \delta(t - nT_b)$$

Here is a typical impulse train autocorrelation.



PSD of Impulse Train (cont.)

The power spectral density of impulse train is

$$\mathcal{F}\{R_x(t)\} = S_x(f) = \frac{1}{T_b}\sum_{n=-\infty}^{\infty} R_n e^{-jn2\pi fT_b}$$

Hence, if we know the discrete time autocorrelation of the transmitted bits, we know the continuous time power spectral density.

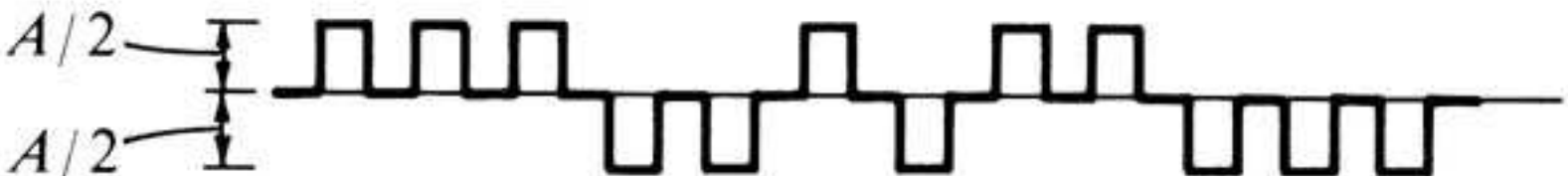
Given a PAM pulse sequence

$$y(t) = \sum_{k=-\infty}^{\infty} a_k p(t - kT_b)$$

the PSD of the encoded signal is

$$S_y(f) = |P(f)|^2 \left(\frac{1}{T_b}\sum_{n=-\infty}^{\infty} R_n e^{-jn2\pi fT_b}\right)$$

PSD of Polar Signaling



- $\mathbf{1} \rightarrow +p(t)\,,\ \mathbf{0} \rightarrow -p(t)$
- Since a_k and a_{k+n} $(n \neq 0)$ are independent and equally likely,

$$R_0 = \lim_{N\to\infty} \frac{1}{2N} \sum_{k=-N}^{N} a_k^2 = \lim_{N\to\infty} \frac{1}{2N} \sum_{k=-N}^{N} 1 = 1$$

$$R_n = \lim_{N\to\infty} \frac{1}{2N} \sum_{k=-N}^{N} a_k a_{k+n} = 0$$

$$S_y(f) = \frac{|P(f)|^2}{T_b} R_0 = \frac{|P(f)|^2}{T_b}$$

PSD of Polar Signaling (cont.)

Examples:

- NRZ (100% pulse)

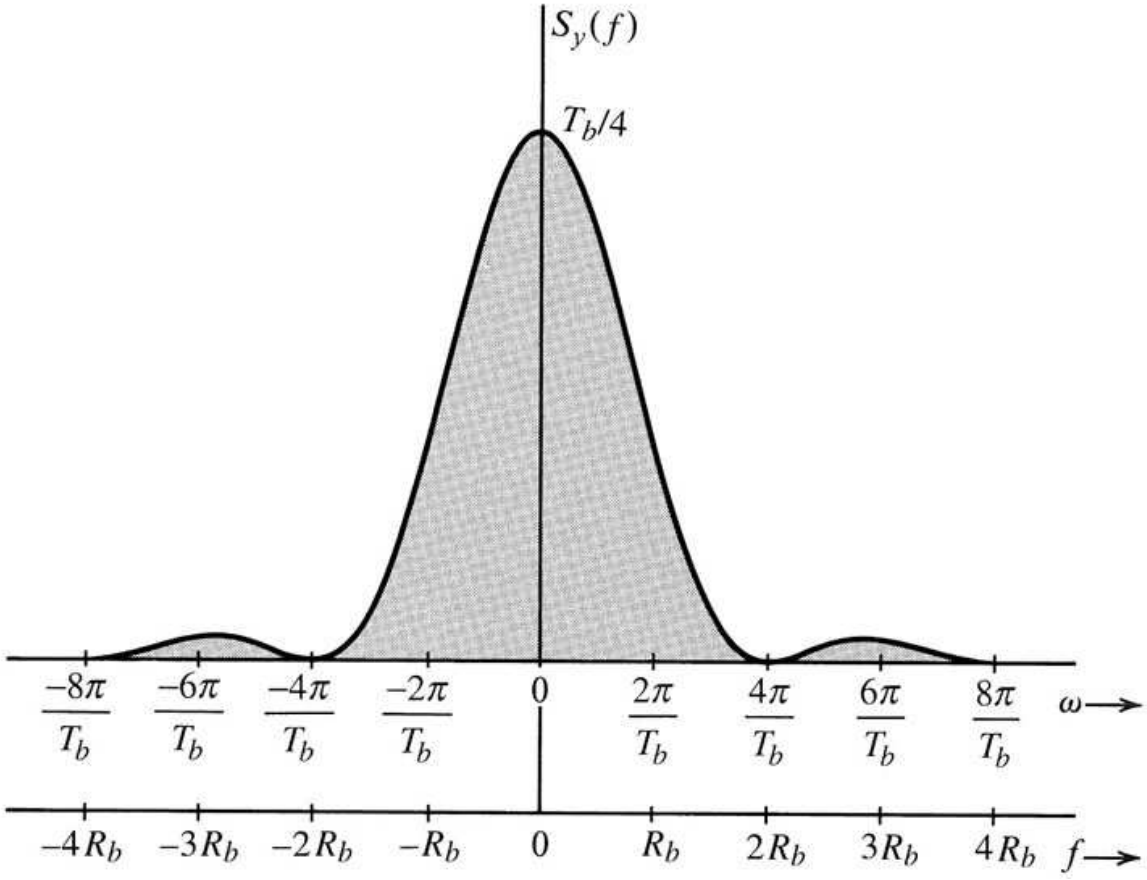
$$p(t) = \Pi(t/T_b)$$
$$P(f) = T_b \operatorname{sinc}(\pi T_b f)$$
$$|P(f)|^2 = T_b^2 \operatorname{sinc}^2(\pi T_b f)$$

- RZ half-width:

$$p(t) = \Pi(t/(T_b/2))$$
$$P(f) = \tfrac{1}{2} T_b \operatorname{sinc}(\tfrac{1}{2}\pi T_b f)$$
$$|P(f)| = \tfrac{1}{4} T_b^2 \operatorname{sinc}^2(\tfrac{1}{2}\pi T_b f)$$

PSD of Polar Signaling: Half-Width Pulse

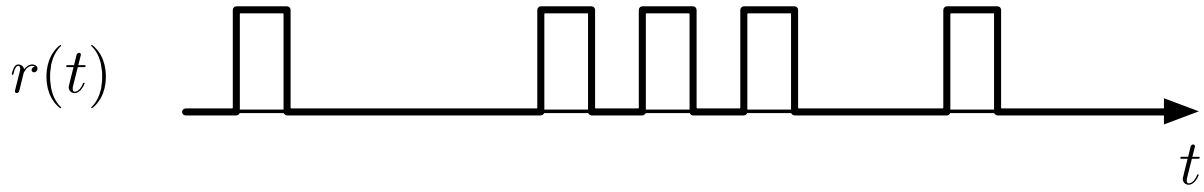
For RZ, $S_y(f) = \dfrac{|P(f)|^2}{T_b} = \dfrac{\frac{1}{4}T_b^2 \operatorname{sinc}^2(\frac{1}{2}\pi T_b f)}{T_b} = \dfrac{T_b}{4}\operatorname{sinc}^2\left(\dfrac{\pi T_b f}{2}\right)$



The bandwidth $2R_b$ is $4\times$ theoretical minimum of 2 bits/Hz/sec.

PSD of On-Off Keying

- OOK looks like



- As we saw earlier, OOK is level-shifted polar signaling:

$$y_{\text{on-off}}(t) = \tfrac{1}{2}\big(1 + y_{\text{polar}}(t)\big)$$

- R_0 is $\frac{1}{2}$ because half the time the signals are 1 and half the time they are 0.

$$R_0 = \left(\tfrac{1}{2}\right) 1 + \left(\tfrac{1}{2}\right) 0 = \tfrac{1}{2}$$

- The issue is with all the higher order terms. If we look at R_n, $1/4$ of the time two bits separated by n are both 1, $1/2$ the time one is one and one is zero, and $1/4$ the time they are both zero. The autocorrelation is then

$$R_n = \left(\tfrac{1}{4}\right) 1 + \left(\tfrac{1}{2}\right) 0 + \left(\tfrac{1}{4}\right) 0 = \tfrac{1}{4}$$

  This contributes a DC term of $1/4$.

PSD of On-Off Keying (cont.)

- The DC term results in impulses in the PSD:

$$S_y(f) = \frac{|P(f)|^2}{4T_b}\left(1 + \frac{1}{T_b}\sum_n \delta(f - n/T_b)\right)$$

  These are undesirable. This corresponds to extra frequency components that don't carry information and DC current that just heats up the wires!

- We can eliminate impulses by using a pulse $p(t)$ with

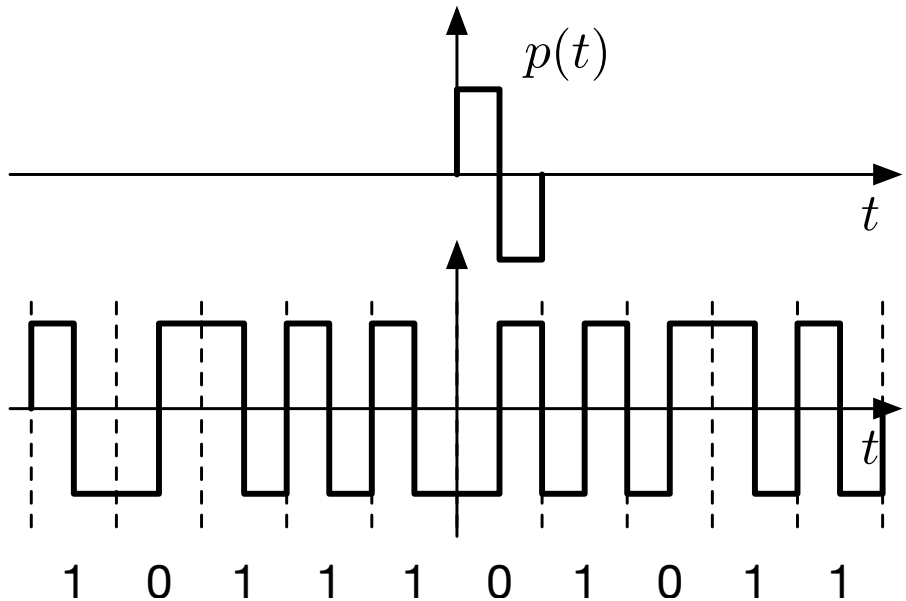
$$P\left(\frac{n}{T_b}\right) = 0\,, \quad n = 0, \pm 1, \pm 2, \ldots$$

- Overall, on-off keying is inferior to polar. For a given average power, noise immunity is less than for bipolar signaling.

- However, OOK is very simple (you just have to gate an oscillator on and off), so it shows up widely in lower power systems (like key fobs) or very high frequency systems (where modulation can be difficult).

Split Phase (Manchester) Encoding

- As we saw with OOK, line codes with nonzero DC value lower performance, because a DC component with no information heats up the wires.
- There are two alternatives
  - Use pulses $p(t)$ that have zero average value (split phase, or Manchester encoding, now)
  - Use sequences of pulses that have average values that go to zero (bipolar signaling, next)
- Split phase encoding looks like this:



Split Phase Encoding (cont.)

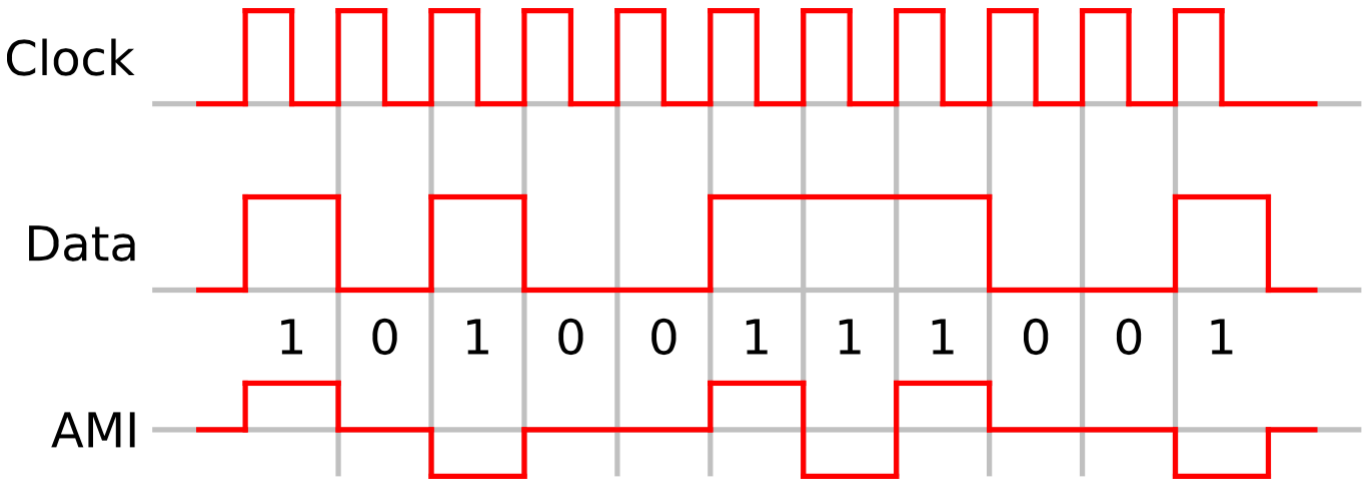
- The PSD is

$$S_y(f) = |P(f)|^2 \left(\frac{1}{T_b} \sum_{n=-\infty}^{\infty} R_n e^{-jn2\pi f T_b} \right) = |P(f)|^2 \frac{1}{T_b}$$

- With this PSD, $R_n = 0$ for $n \neq 0$.
  - $R_0 = 1$
  - However, $P(0) = 0$
- We'll see what this looks like in a few slides
- It is also very easy to get the timing signal from the coding waveform.
- This was first introduced with the development of magnetic disk drives in the late 1940's and early 1950's. Read heads were only sensitive to transitions in magnetization, so this approach guaranteed at least one transition per bit.
- This is widely used in wired ethernet. Also common in RF, particularly in low power near field RF (NFRF) devices.

Alternate Mark Inversion (Bipolar) Signaling

AMI encodes 0 as 0 V and 1 as $+V$ or $-V$, with alternating signs.



AMI was used in early PCM systems.

- Eliminates DC build up on cable.
- Reduces bandwidth compared to polar.
- Provides error detecting; every bit error results in bipolar violation.
- Guarantees transitions for timing recovery with long runs of ones.

AMI is also called *bipolar* and *pseuodternary*.

PSD of AMI Signaling

- If the data sequence $\{a_k\}$ consists of equally likely and independent 0's and 1s, then the autocorrelation function of the sequence is for R_0 is

$$R_0 = \left(\frac{1}{2}\right) 1 + \left(\frac{1}{2}\right) 0 = \frac{1}{2}$$

- For $R_{\pm 1}$ there are four possibilities, 11, 01, 10, and 00.

  Since the signs change for successive 1's, and all the others have autocorrelations of zero,

$$R_{\pm 1} = \left(\frac{1}{4}\right) (-1) + \left(\frac{3}{4}\right) 0 = -\frac{1}{4}$$

- For $n = 2$, the various permutations of 1's and 0's are either zero, or cancel out and give $R_2 = 0$. This continues for $n > 2$.

PSD of AMI Signaling (cont.)

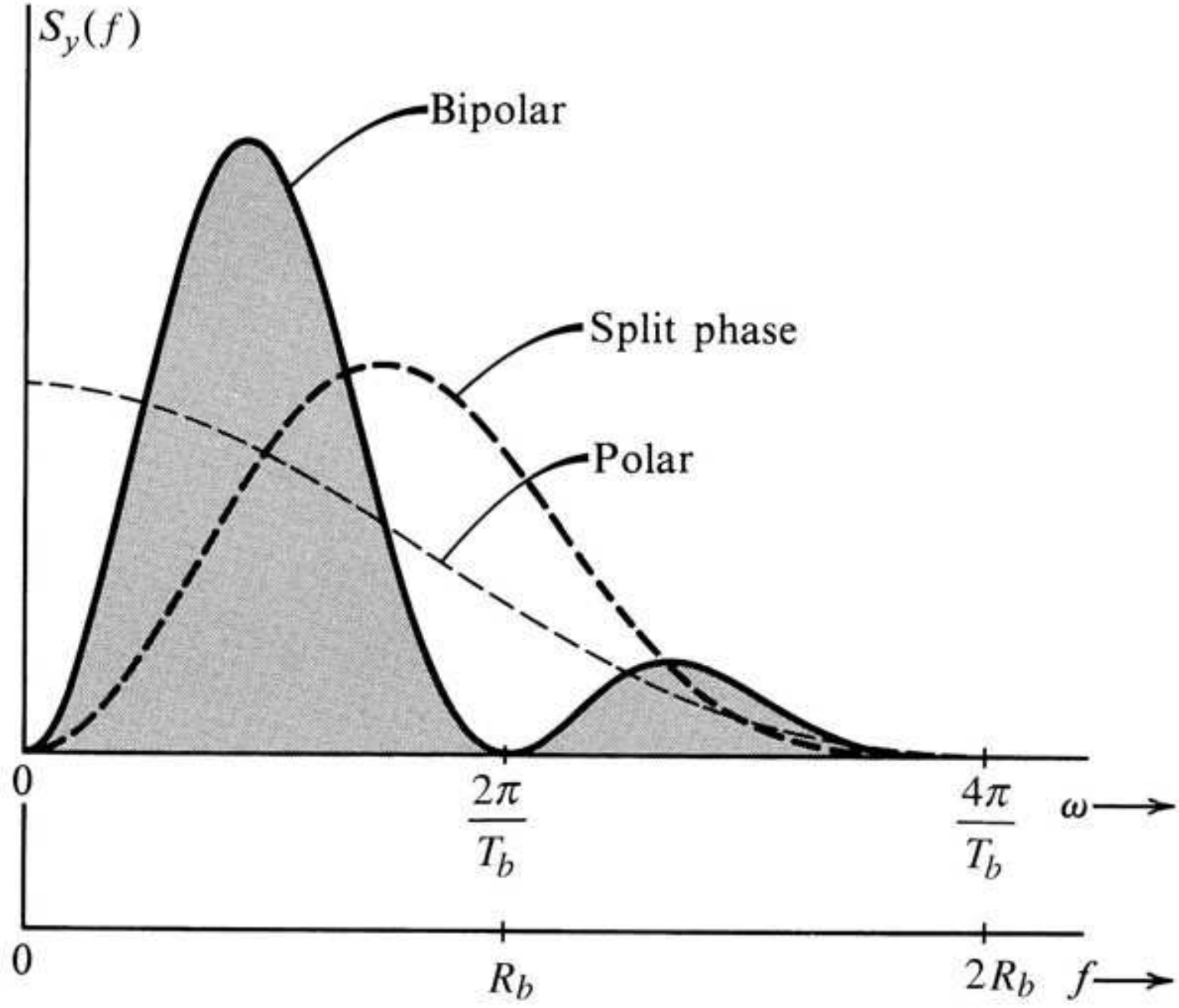
- Therefore

$$\begin{aligned} S_y(f) &= \frac{|P(f)|^2}{T_b} \sum_n R_n e^{j2\pi n f T_b} \\ &= \frac{|P(f)|^2}{T_b} \left(R_0 + R_{-1} e^{j2\pi f T_b} + R_1 e^{-j2\pi f T_b} \right) \\ &= \frac{|P(f)|^2}{T_b} \left(\frac{1}{2} - \left(\frac{1}{4} \right) 2\cos(2\pi f T_b) \right) \\ &= \frac{|P(f)|^2}{2T_b} \left(1 - \cos(2\pi f T_b) \right) \\ &= \frac{|P(f)|^2}{T_b} \sin^2(\pi T_b f) \end{aligned}$$

  This PSD falls off faster than $\mathrm{sinc}(\pi T_b f)$.

- The PSD has a null at DC, which aids in transformer coupling.

PSD of AMI Signaling (cont.)



Data Transfer in Digital System

- In a synchronous digital system, a common clock signal is used by all devices.

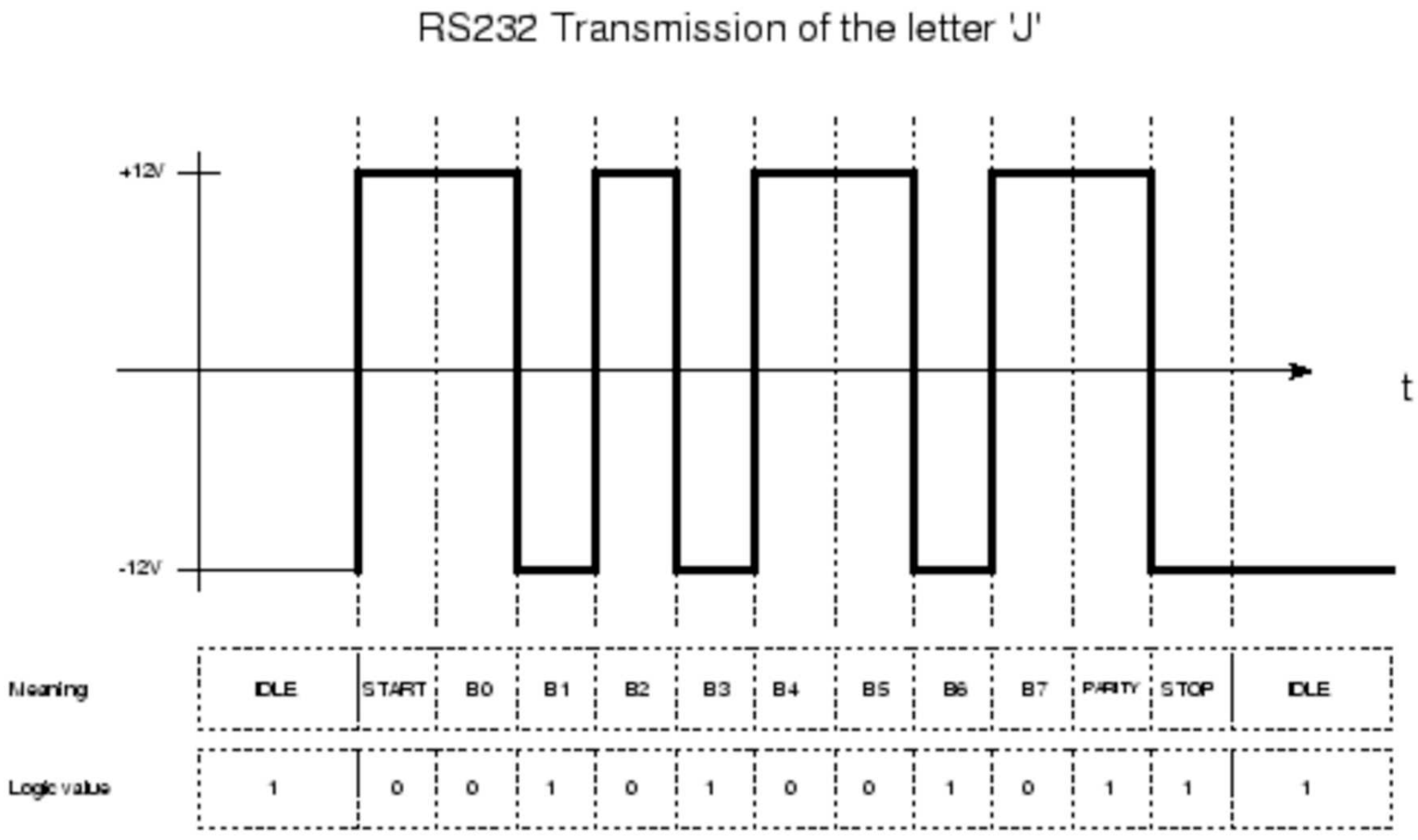
  $$\text{data} + \text{clock}$$

  Multiple data signals can be transmitted in parallel using a single clock signal.

- Serial peripheral communication schemes (RS-232, USB, FireWire) use various clock extraction methods
  - RS-232 is asynchronous with (up to) 8 data bits preceded by a start bit (0) and followed by optional parity bit and stop bit (1); clock recovery by "digital phase-locked loop"
  - USB needs a real phase-locked loop and uses *bit stuffing* to ensure enough transitions
  - FireWire has differential data and clock pairs; clock transitions only when data does not

Serial Communication: RS-232 Signaling

RS-232 is a standard for *asynchronous* serial communication.



Each transition resynchronizes the receiver's bit clock.

Serial Communication: USB Signaling

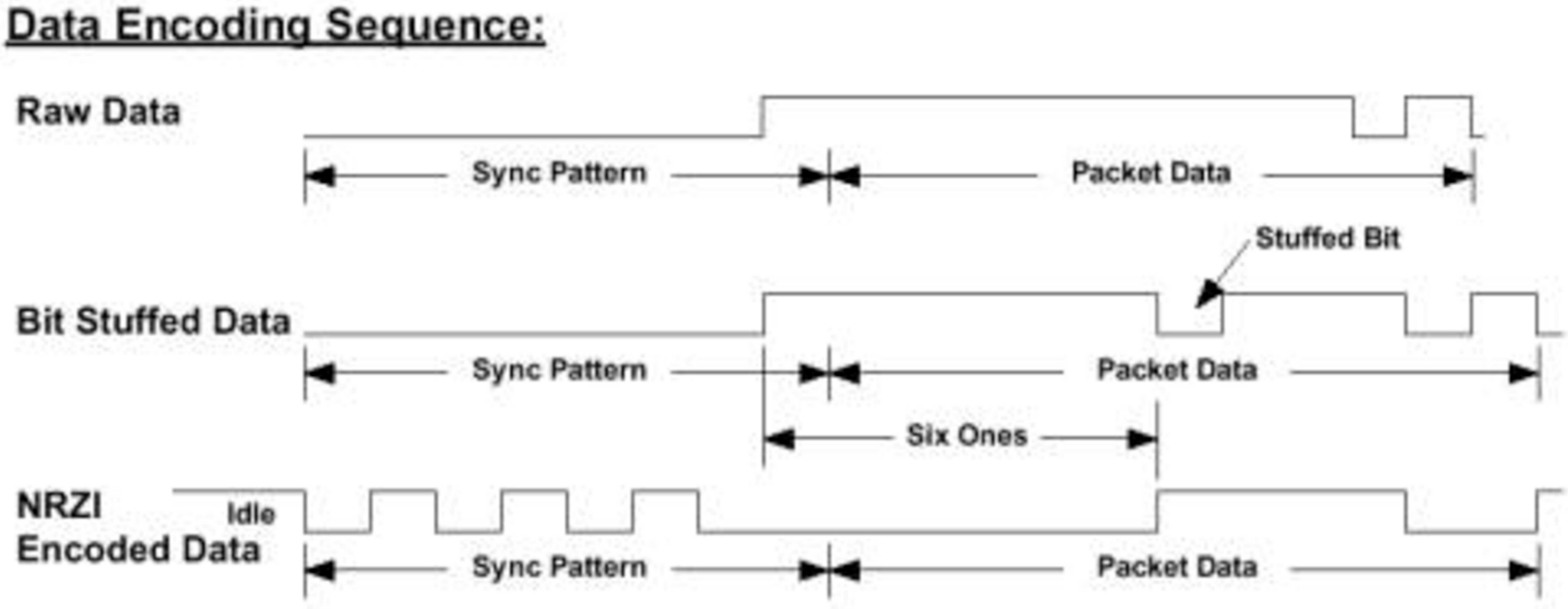


Figure 7-13. Bit Stuffing