Data Signal Analysis

Prof. Alan Kaminsky
Department of Computer Science
Rochester Institute of Technology
Rochester, NY, USA
ark@cs.rit.edu

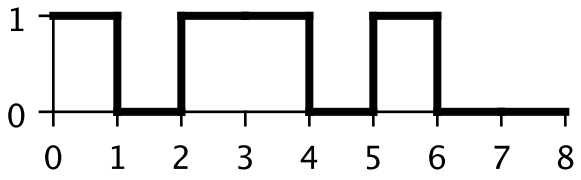
March 2, 2014

1 Signals

Signal: A function of time, $h(t)$

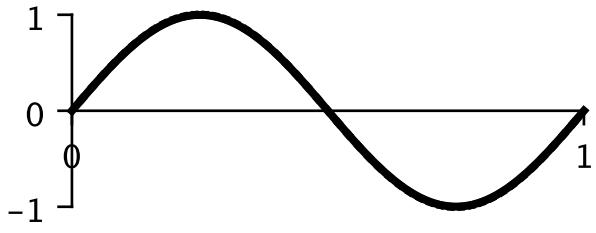
Digital signal: A signal with discrete values

Example: The bits 10110100 as a digital signal

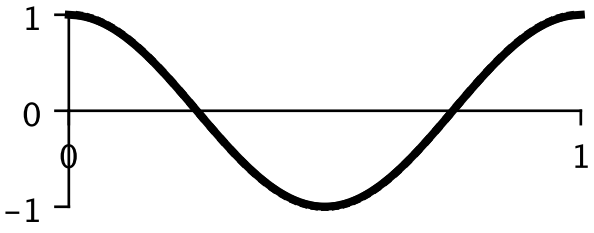


Analog signal: A signal with continuous values

Example: $h(t) = \sin 2\pi t$



Example: $h(t) = \cos 2\pi t$



Sinusoidal signal: $h(t) = a(t)\sin(\phi(t))$

$a(t) =$ **amplitude** (function of time)
$\phi(t) =$ **phase** (function of time)

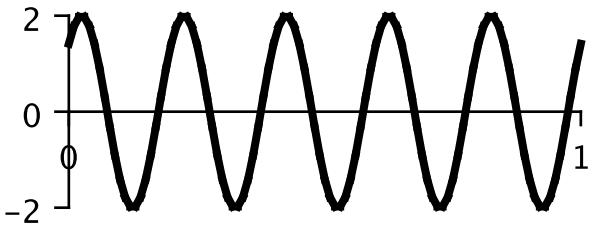
$$f(t) = \frac{1}{2\pi}\frac{\mathrm{d}\phi(t)}{\mathrm{d}t}$$

$f(t) =$ **frequency** (function of time)

$$\phi(t) = 2\pi\int_0^t f(\tau)\mathrm{d}\tau + \phi(0)$$

Time in units of seconds (sec)
Frequency in units of Hertz (Hz) (1/sec)
Phase in units of radians (rad)

Example: $h(t) = 2\sin(10\pi t + \pi/4)$
$a(t) = 2$ (constant)
$f(t) = 5$ (constant)
$\phi(0) = \pi/4$
$\phi(t) = 10\pi t + \pi/4$



2 Modulation

A **modulator** converts a digital signal to an analog signal, typically a sinusoidal signal
A **demodulator** converts a modulated (analog) signal back to the original unmodulated (digital) signal
A **modem** converts an outgoing digital signal to an outgoing modulated signal, *and* converts an incoming modulated signal to an incoming digital signal

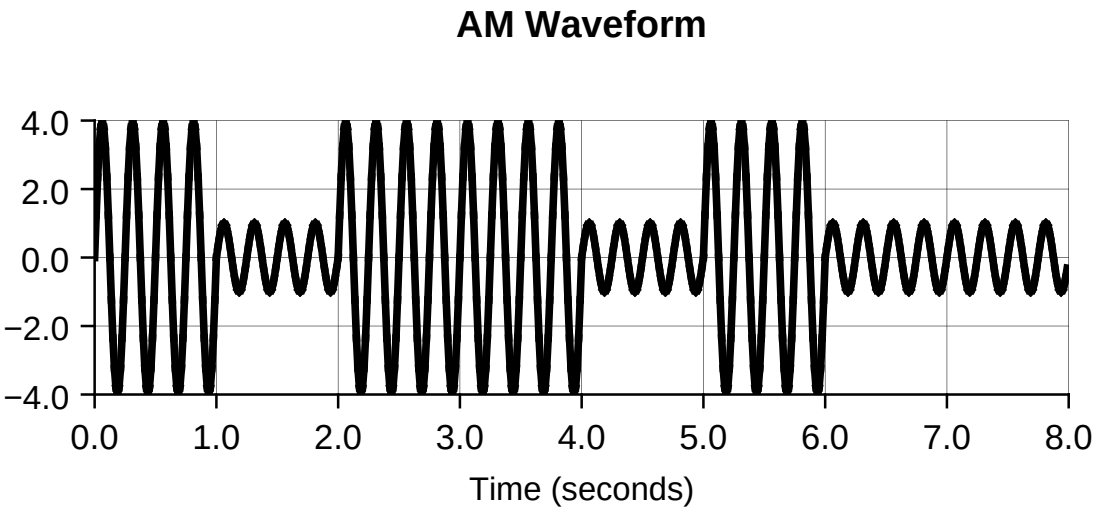
Why? To change the signal's bandwidth so it can be transmitted on a limited-bandwidth communication channel (like a telephone line or a cable TV channel) without too much distortion — more on this later

How? By using the digital bit stream to vary the sinusoidal signal's amplitude $a(t)$ and/or phase $\phi(t)$

2.1 Amplitude Modulation (AM)

$f(t) = f$ — **carrier frequency,** constant
$\phi(t) = 2\pi ft$
$a(t) = a_0$ for a 0 bit, a_1 for a 1 bit

Example: Bit stream = 10110100, bit rate = 1 bps, $f = 4$ Hz, $a_0 = 1$, $a_1 = 4$

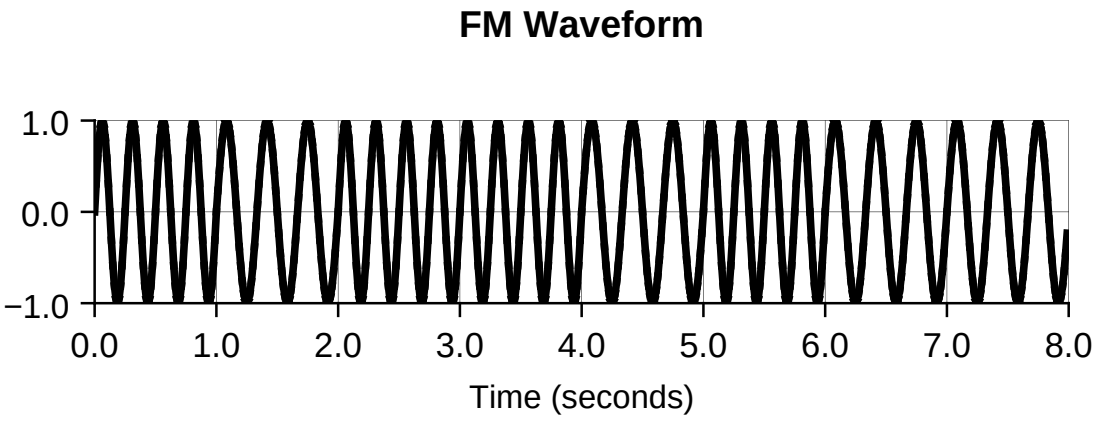


AM demodulation
In each symbol interval, measure the amplitude of the peaks in the waveform
Compare the amplitude to a **threshold**
If the amplitude is on one side of the threshold, it's a 0 bit, otherwise it's a 1 bit

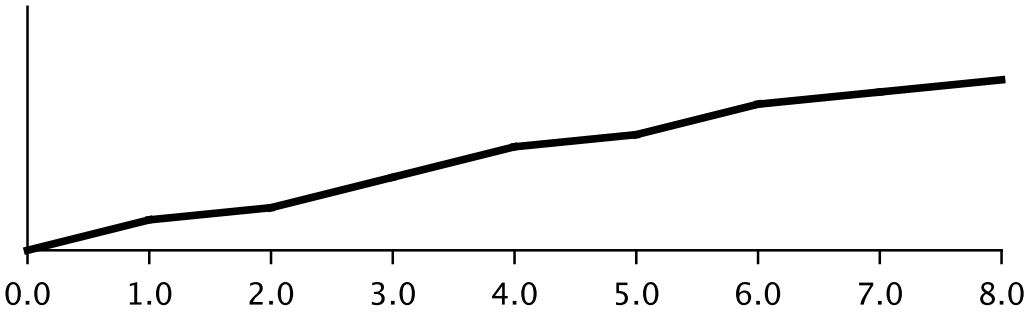
2.2 Frequency Modulation (FM)

Also known as **frequency shift keying (FSK)**
$a(t) = a$ — constant amplitude
$f(t) = f_0$ for a 0 bit, f_1 for a 1 bit

Example: Bit stream = 10110100, bit rate = 1 bps, $a = 1$, $f_0 = 3$ Hz, $f_1 = 4$ Hz



The phase ramps up continuously, but with different slopes during different bit intervals



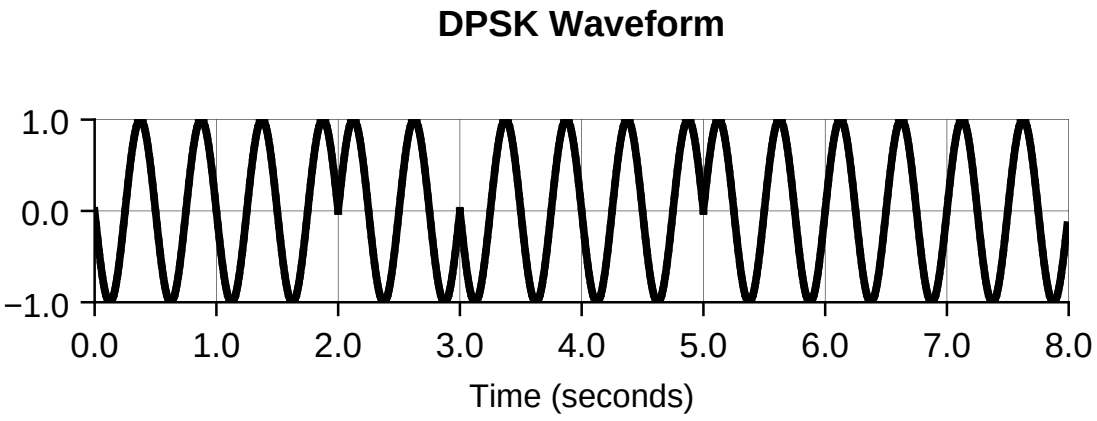
The Bell 103 modem standard: An actual telephone modem that used FSK
Bit rate = 300 bps
Outgoing ("originate") signal: $f_0 = 1270$ Hz, $f_1 = 1070$ Hz
Incoming ("answer") signal: $f_0 = 2225$ Hz, $f_1 = 2025$ Hz
"Mark" = 0 bit, "mark frequency" = f_0
"Space" = 1 bit, "space frequency" = f_1

FM demodulation
Feed the FM signal through a pair of **filters,** one tuned to frequency f_0, the other tuned to frequency f_1
The signal will pass through a filter only if the signal matches the filter's frequency
If we get a signal out of the f_0 filter, it's a 0 bit
If we get a signal out of the f_1 filter, it's a 1 bit
Easy to do with analog circuitry

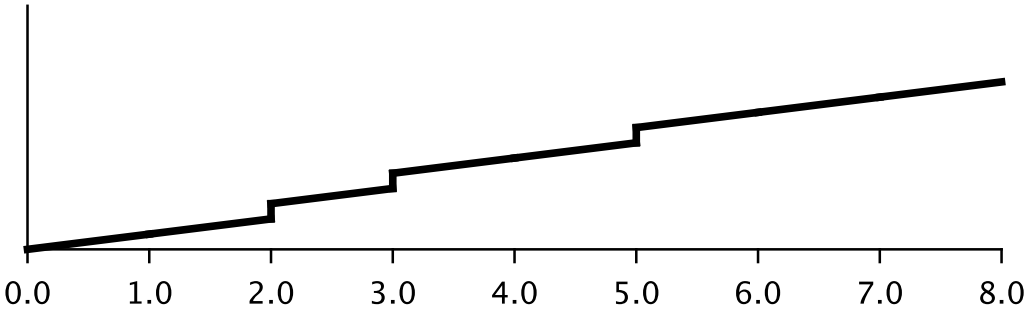
2.3 Differential Phase Shift Keying (DPSK)

Constant amplitude a
Constant frequency f
The phase ramps up continuously at a constant slope (frequency)
At the start of each bit interval, the phase *shifts* (increases discontinuously) by an amount $\Delta\phi$ determined by the bit value: $\Delta\phi_0 = 0$, $\Delta\phi_1 = \pi$

Example: Bit stream = 10110100, bit rate = 1 bps, $a = 1$, $f = 2$ Hz



Here is a plot of the phase showing the phase shifts:



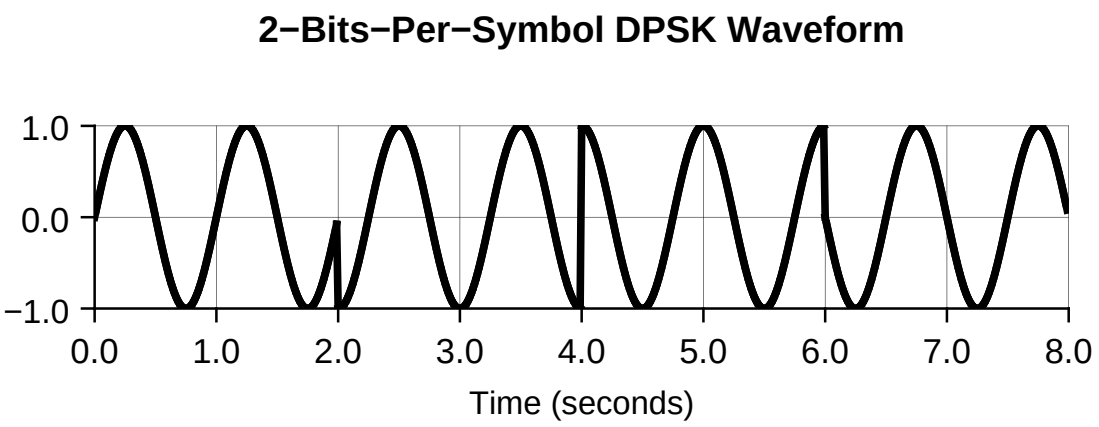
DPSK demodulation
Typically done with an analog circuit called a **phase locked loop** (PLL)
The PLL keeps the phase of an internally-generated sine function in sync to the phase of the incoming signal
The PLL generates an output signal which is 0 if the internal sine function's phase is the same as the incoming signal's phase
When the incoming signal's phase shifts, the PLL's output signal becomes nonzero until the PLL pulls the internal sine function's phase back in sync with the incoming signal's phase
At each bit boundary, look at the PLL's output signal to detect what the phase shift was, then convert that to a 0 or 1 bit

2.4 Bits Versus Symbols

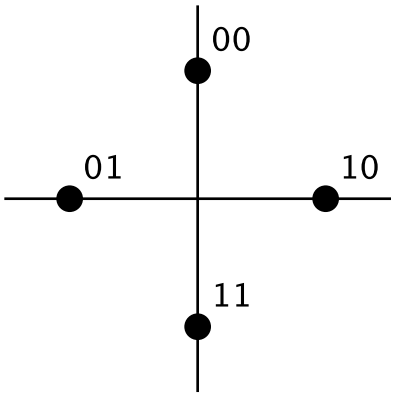
Bit: The original digital value (0 or 1)
Symbol: The modulated analog waveform
Hitherto, each symbol has encoded one bit
However, each symbol can encode more than one bit
If there are n bits per symbol, the **symbol rate** (also known as the **baud rate**) in symbols per second is $1/n$ times the **bit rate** in bits per second
If there are n bits per symbol, there are 2^n different symbols

The Bell 212A modem standard: DPSK with multibit symbols
Bit rate = 1200 bps
$n = 2$ bits per symbol
Symbol rate = 600 symbols per second
4 different symbols
Phase shifts: $\Delta\phi_{00} = \pi/2$, $\Delta\phi_{01} = \pi$, $\Delta\phi_{10} = 0$, $\Delta\phi_{11} = 3\pi/2$
Outgoing ("originate") signal: carrier frequency $f = 1200$ Hz
Incoming ("answer") signal: carrier frequency $f = 2400$ Hz

Example: Bit stream = 10110100, bit rate = 1 bps, symbol rate = 0.5 symbols per second, $a = 1$, $f = 1$ Hz, $\Delta\phi_{00} = \pi/2$, $\Delta\phi_{01} = \pi$, $\Delta\phi_{10} = 0$, $\Delta\phi_{11} = 3\pi/2$



Constellation diagram for a DPSK waveform:

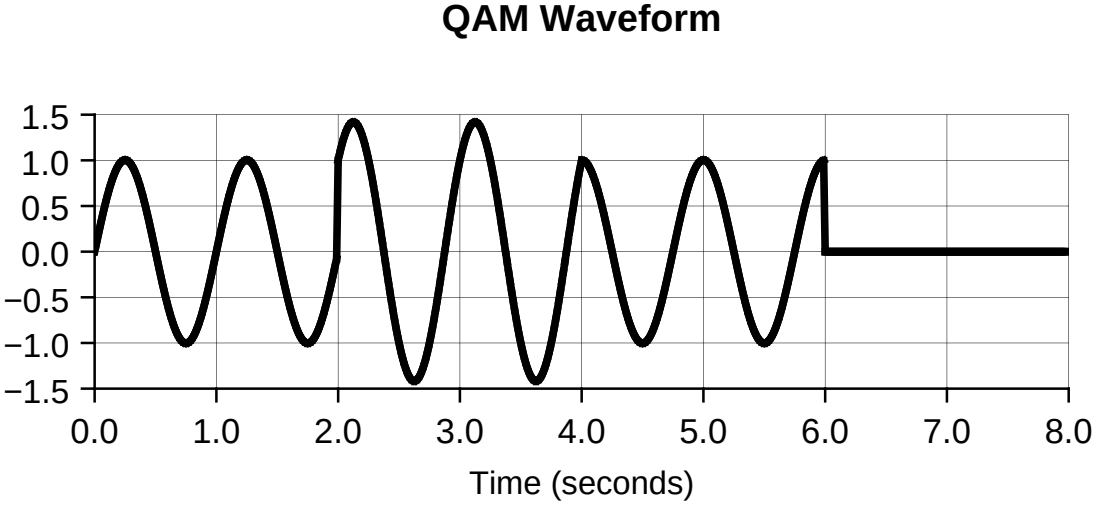


Each point represents one symbol
A point's distance from the origin gives the symbol's amplitude
A point's angle with respect to the positive X axis gives the symbol's phase shift

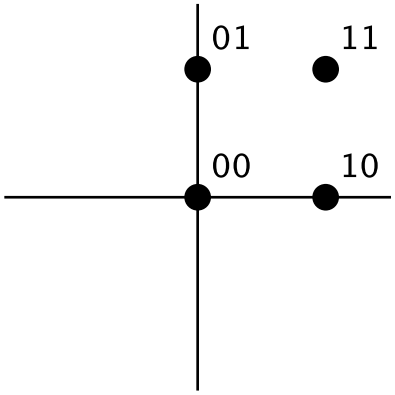
2.5 Quadrature Amplitude Modulation (QAM)

A multibit per symbol waveform that is the sum of an amplitude modulated sine function and an amplitude modulated cosine function at the same, constant carrier frequency f
Even bits determine the sine function amplitude $s(t)$: $s_0 = 0$, $s_1 = a$
Odd bits determine the cosine function amplitude $c(t)$: $c_0 = 0$, $c_1 = a$
$h(t) = s(t) \sin 2\pi ft + c(t) \cos 2\pi ft$

Example: Bit stream = 10110100, bit rate = 1 bps, symbol rate = 0.5 symbols per second, $a = 1$, $f = 1$ Hz



Constellation diagram for a QAM waveform, showing the symbols' amplitudes and absolute phases (instead of phase shifts):



You can have QAM with any number of bits per symbol
Typically, half of the symbol bits encode the sine amplitude and half of the symbol bits encode the cosine amplitude

Example: 2-bits-per-symbol QAM
1 bit encodes sine amplitude, 2 different sine amplitudes
1 bit encodes cosine amplitude, 2 different cosine amplitudes

Example: 4-bits-per-symbol QAM
2 bits encode sine amplitude, 4 different sine amplitudes
2 bits encode cosine amplitude, 4 different cosine amplitudes

Example: 6-bits-per-symbol QAM
3 bits encode sine amplitude, 8 different sine amplitudes
3 bits encode cosine amplitude, 8 different cosine amplitudes

Today's high-speed telephone modems use QAM combined with an error correcting code (trellis code) to achieve a maximum data rate of 33,600 bps (V.34: both channels, V.90: outgoing channel only)

Today's cable modems also use QAM, at much higher bit rates than telephone modems

QAM demodulation
In each symbol interval:
Correlate the QAM waveform with a sine function to extract the sine amplitude s
Correlate the QAM waveform with a cosine function to extract the cosine amplitude c
Convert s and c back to the original bits

The correlation of two signals $h_1(t)$ and $h_2(t)$:

$$\text{Corr}(h_1(t), h_2(t)) = \int h_1(t) h_2(t)\, \mathrm{d}t$$

Using correlation to extract the sine amplitude from a QAM waveform:
Multiply the QAM waveform by a sine function at the same frequency f and integrate over the symbol interval
Example: bit rate = 1 bps, symbol rate = 0.5 symbols per second, symbol interval = 2 seconds, $f = 1$ Hz
QAM waveform in one symbol interval $h(t) = s \sin 2\pi t + c \cos 2\pi t$

$$\begin{aligned}\int_0^2 h(t) \sin 2\pi t \, \mathrm{d}t &= \int_0^2 (s \sin 2\pi t + c \cos 2\pi t) \sin 2\pi t \, \mathrm{d}t \\ &= \int_0^2 s \sin^2 2\pi t \, \mathrm{d}t \; + \; \int_0^2 c \cos 2\pi t \sin 2\pi t \, \mathrm{d}t\end{aligned}$$

Using the trigonometric identities $\sin^2 x = (1 - \cos 2x)/2$ and $\cos x \sin x = (\sin 2x)/2$:

$$\begin{aligned}\int_0^2 h(t) \sin 2\pi t \, \mathrm{d}t &= \frac{s}{2} \int_0^2 (1 - \cos 4\pi t) \, \mathrm{d}t \; + \; \frac{c}{2} \int_0^2 \sin 4\pi t \, \mathrm{d}t \\ &= \frac{s}{2} \int_0^2 \mathrm{d}t \; - \; \frac{s}{2} \int_0^2 \cos 4\pi t \, \mathrm{d}t \; + \; \frac{c}{2} \int_0^2 \sin 4\pi t \, \mathrm{d}t \\ &= \frac{s}{2} \cdot 2 \; - \; \frac{s}{2} \cdot 0 \; + \; \frac{c}{2} \cdot 0 \; = \; s, \text{ the sine amplitude}\end{aligned}$$

Using correlation to extract the cosine amplitude from a QAM waveform:
Multiply the QAM waveform by a cosine function at the same frequency f and integrate over the symbol interval
Example: bit rate = 1 bps, symbol rate = 0.5 symbols per second, symbol interval = 2 seconds, $f = 1$ Hz
QAM waveform in one symbol interval $h(t) = s \sin 2\pi t + c \cos 2\pi t$

$$\begin{aligned}\int_0^2 h(t) \cos 2\pi t \, \mathrm{d}t &= \int_0^2 (s \sin 2\pi t + c \cos 2\pi t) \cos 2\pi t \, \mathrm{d}t \\ &= \int_0^2 s \sin 2\pi t \cos 2\pi t \, \mathrm{d}t \; + \; \int_0^2 c \cos^2 2\pi t \, \mathrm{d}t\end{aligned}$$

Using the trigonometric identities $\sin x \cos x = (\sin 2x)/2$ and $\cos^2 x = (1 + \cos 2x)/2$:

$$\begin{aligned}\int_0^2 h(t) \sin 2\pi t \, \mathrm{d}t &= \frac{s}{2} \int_0^2 \sin 4\pi t \, \mathrm{d}t \; + \; \frac{c}{2} \int_0^2 (1 + \cos 4\pi t) \, \mathrm{d}t \\ &= \frac{s}{2} \int_0^2 \sin 4\pi t \, \mathrm{d}t \; + \; \frac{c}{2} \int_0^2 \mathrm{d}t \; + \; \frac{c}{2} \int_0^2 \cos 4\pi t \, \mathrm{d}t \\ &= \frac{s}{2} \cdot 0 \; + \; \frac{c}{2} \cdot 2 \; + \; \frac{c}{2} \cdot 0 \; = \; c, \text{ the cosine amplitude}\end{aligned}$$

It is very easy to perform correlation using **digital signal processing** (DSP) techniques
DSP circuits are easy to build out of digital logic gates
Digital logic gates are easy to put on integrated circuit chips — no nasty analog circuit components needed
IC chips are cheap
Therefore, QAM modem chips are cheap, which explains the popularity of QAM

3 Frequency Analysis

We've been studying *how* modems work, now we'll study *why* we use modems

3.1 Fourier Synthesis

Demonstration: The Sound Synthesizer program (see the Computer Science Course Library)

Any periodic signal, like a sound, can be created as the summation of sinusoidal functions:

$$h(t) = a_0 + \sum_{j=1}^{\infty} a_j \sin(2\pi j f_1 t + \phi_j)$$

f_1 = fundamental frequency
j = harmonic number
jf_1 = frequency of the j-th harmonic, a.k.a. the j-th frequency component
a_j = amplitude of the j-th frequency component
ϕ_j = phase of the j-th frequency component
a_0 = 0-frequency component or DC component (DC stands for "direct current")

A signal can thus be represented in two equivalent ways:
Time domain representation — plot $h(t)$ as a function of time
Frequency domain representation — plot the frequency components' amplitudes a_j (and phases ϕ_j) as a function of frequency
The latter plot is also called a **frequency spectrum**

The frequency spectrum shows the frequencies where the signal has much power or little power

Given the frequency components, it's easy to compute the signal in the time domain — **Fourier synthesis**
Given the signal in the time domain, it's trickier to compute the frequency components — **Fourier analysis**

3.2 Fourier Analysis with the Fourier Transform

A function of time, $h(t)$, in the time domain can be transformed into a function of frequency, $H(f)$, in the frequency domain using the **Fourier transform:**

$$H(f) = \int_{-\infty}^{+\infty} h(t) \mathrm{e}^{2\pi \mathrm{i} f t} \mathrm{d}t$$

A function of frequency, $H(f)$, in the frequency domain can be transformed back into a function of time, $h(t)$, in the time domain using the **inverse Fourier transform:**

$$h(t) = \int_{-\infty}^{+\infty} H(f)\mathrm{e}^{-2\pi \mathrm{i} f t}\mathrm{d}f$$

In practice, you don't compute those integrals analytically
Instead, you **sample** your waveform $h(t)$ to get a **time series** h_j:

$$h_j = h(j\Delta t), j = 0, 1, 2, ...N-1$$

$\Delta t =$ **sampling interval** (seconds)
$1/\Delta t =$ **sampling rate** (samples per second)
$N =$ number of samples

Then you transform your time series h_j to a **frequency series** H_k (also known as a **frequency spectrum**) using the **discrete Fourier transform (DFT):**

$$H_k = \sum_{j=0}^{N-1} h_j \mathrm{e}^{-2\pi \mathrm{i} jk/N}, k = 0, 1, 2, ...N-1$$

You can transform your frequency series H_k back to a time series h_j using the **inverse discrete Fourier transform:**

$$h_j = \frac{1}{N}\sum_{k=0}^{N-1} H_k \mathrm{e}^{2\pi \mathrm{i} jk/N}, j = 0, 1, 2, ...N-1$$

In general, computing the DFT takes $O(N^2)$ time
If N is a power of 2, then the **Fast Fourier Transform (FFT)** algorithm can compute the DFT in $O(N \log N)$ time
The Data Communications and Networks II Course Library includes Java classes for time series, frequency series, and the FFT algorithm

3.3 Interpretation of the Frequency Series

The frequency series H_k consists of **frequency components** of the waveform:
$H_0 =$ component at $f = 0$
$H_1 =$ component at $f = \Delta f$
$H_2 =$ component at $f = 2\Delta f$
. . .
$H_k =$ component at $f = k\Delta f$
. . .
$H_{N/2} =$ component at $f = (N/2)\Delta f$

$$\Delta f = \frac{1}{N\Delta t}$$

Example: Sampling rate = 8000 samples per second, sampling interval $\Delta t = 125$ μsec, $N = 512$ samples, $\Delta f = 1/(N\Delta t) = 8000/512 = 15.63$ Hz, frequency components go from 0 to 4000 Hz

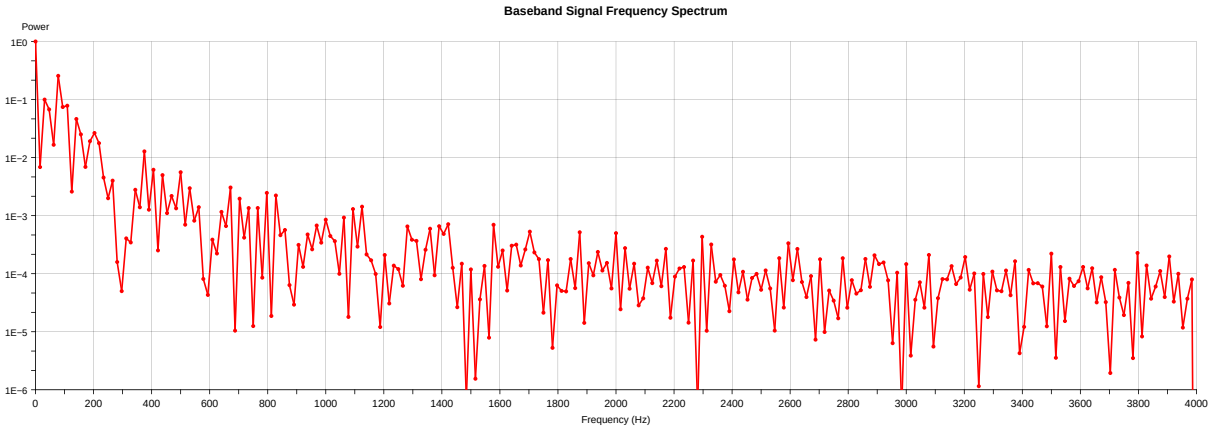
(The frequency components at indexes greater than $N/2$ correspond to *negative* frequencies, but for a real-valued signal the negative frequency components contain the same information as the positive frequency components, so we generally ignore the negative frequency components)

The amplitude of the frequency component at frequency $k\Delta f$ equals the magnitude of the complex number H_k
The power of the frequency component at frequency $k\Delta f$ equals the square of the magnitude of the complex number H_k

So a signal's frequency spectrum — a plot of $|H_k|^2$ versus frequency $k\Delta f$ — shows the frequencies where the signal has much power or little power

3.4 Bandwidth

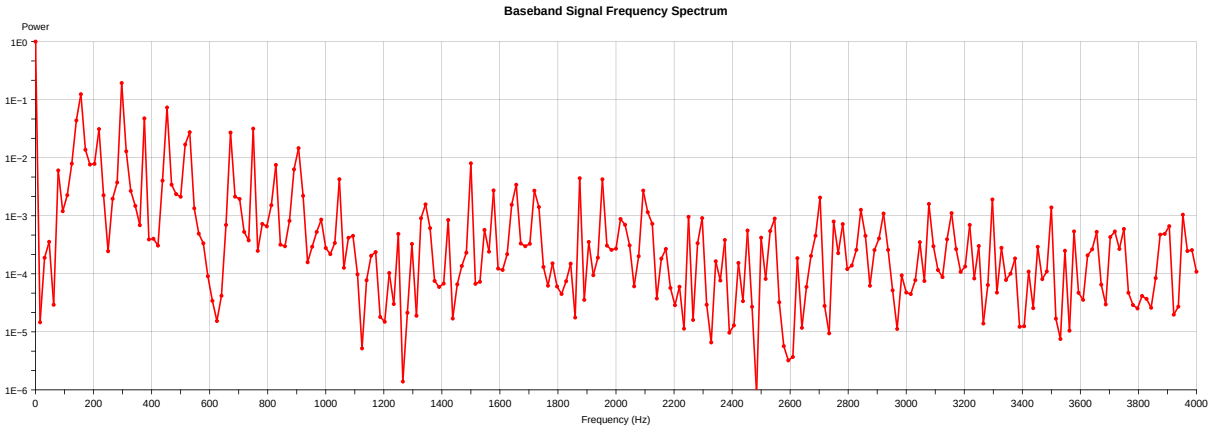
Example of an unmodulated digital signal's frequency spectrum at a bit rate of 300 bps:



A signal's **bandwidth** is the range of frequencies where "most" of the energy is located
A convenient criterion for determining the bandwidth is to set a threshold, say 0.01 times the largest frequency component, and include all frequencies from the smallest frequency component above the threshold to the largest frequency component above the threshold

The bandwidth of the 300-bps unmodulated digital signal in the above example (threshold = 0.01) is about 0 to 400 Hz, with most of the power down near 0 Hz

Example of an unmodulated digital signal's frequency spectrum at a bit rate of 1200 bps:



The bandwidth of the 1200-bps unmodulated digital signal in the above example (threshold = 0.01) is about 0 to 1600 Hz, with most of the power down near 0 Hz
The higher the bit rate, the wider the bandwidth of the signal

A communication channel, like a telephone line or a cable TV channel, also has a bandwidth
The channel will only transmit energy at frequencies inside its bandwidth
Energy at frequencies outside the channel's bandwidth will be **attenuated** (reduced or eliminated)
If you send a signal into a communication channel, the signal's energy outside the channel's bandwith will be lost, and the signal coming out the far end will be distorted
If too much signal energy is lost, the received signal will be indecipherable, and communication cannot take place (or will experience too many bit errors)

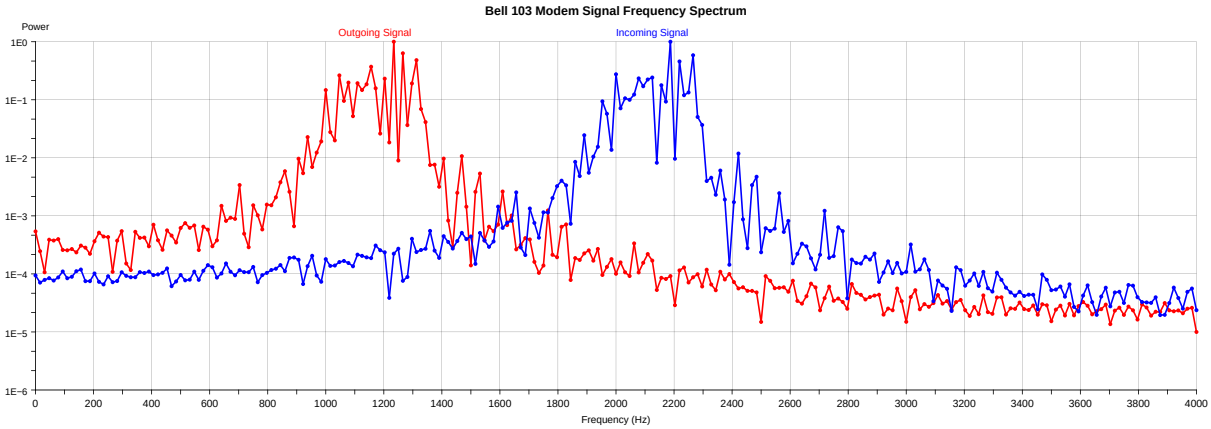
A voice telephone line has (by design) a bandwidth of about 300 Hz to 3400 Hz
An unmodulated digital signal typically cannot survive transmission through a voice telephone line — the signal has too much energy outside the channel's bandwidth, particularly around 0 Hz
The solution: **Modulating the digital signal shifts the signal's energy to frequencies inside the channel's bandwidth**
As a result, the modulated signal can be transmitted over a telephone line with little or no distortion, then demodulated back again at the far end

The same is true of digital signals transmitted over a cable TV channel, whose bandwidth is about 6 MHz wide

THIS, FINALLY, IS WHY WE USE MODEMS!

3.5 FM Frequency Spectrum

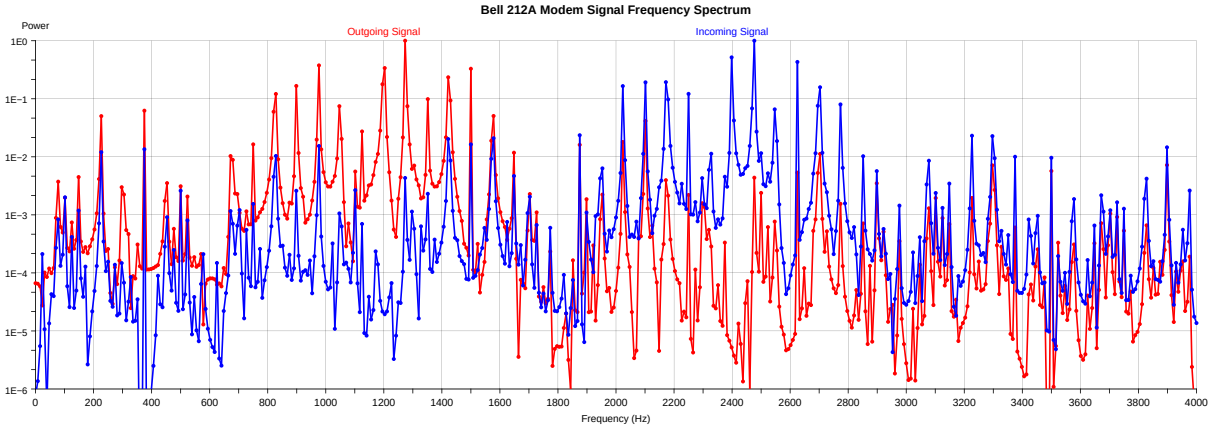
Example of a 300-bps FM Bell 103 modem signal's frequency spectrum:



Note that the incoming and outgoing signals' bandwidths do not overlap
At the frequencies where the outgoing signal power is strong, the incoming signal power is much smaller, and vice versa, so the modem has no trouble separating the two signals

3.6 DPSK Frequency Spectrum

Example of a 1200-bps multibit DPSK Bell 212A modem signal's frequency spectrum:



Note the fuller utilization of the channel bandwidth compared to FM
Again, at the frequencies where the outgoing signal power is strong, the incoming signal power is much smaller, and vice versa, so the modem has no trouble separating the two signals

4 Baseband Signals

4.1 Baseband Versus Broadband

A **broadband** communication channel has a bandwidth that goes from some nonzero lower frequency to some nonzero upper frequency
Example: Telephone channel — 300 Hz to 3400 Hz
Example: Cable TV channel — 6 MHz wide, in the 5–42 MHz range for outgoing (upstream) signals, in the 450–860 MHz range for incoming (downstream) signals

Before transmitting a digital signal on a broadband channel, the digital signal must be *modulated* to shift the signal's frequencies into the channel's bandwidth

A **baseband** communication channel has a bandwidth that goes from zero frequency (DC) to some upper frequency
Example: Ethernet cable

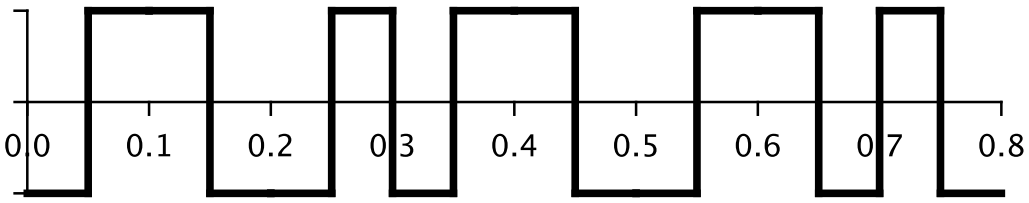
A digital signal can be transmitted directly on a baseband channel; no modulation is needed to shift its frequencies
However, the digital signal may be **encoded** for other reasons

4.2 Manchester Encoding — 10-Mbps Wired Ethernet

Used in 10-Mbps Ethernet LANs (IEEE 802.3)

Two voltage levels, L (low, –0.85 V) and H (high, +0.85 V)
Each bit is encoded as a pair of voltage levels: 0 = HL, 1 = LH

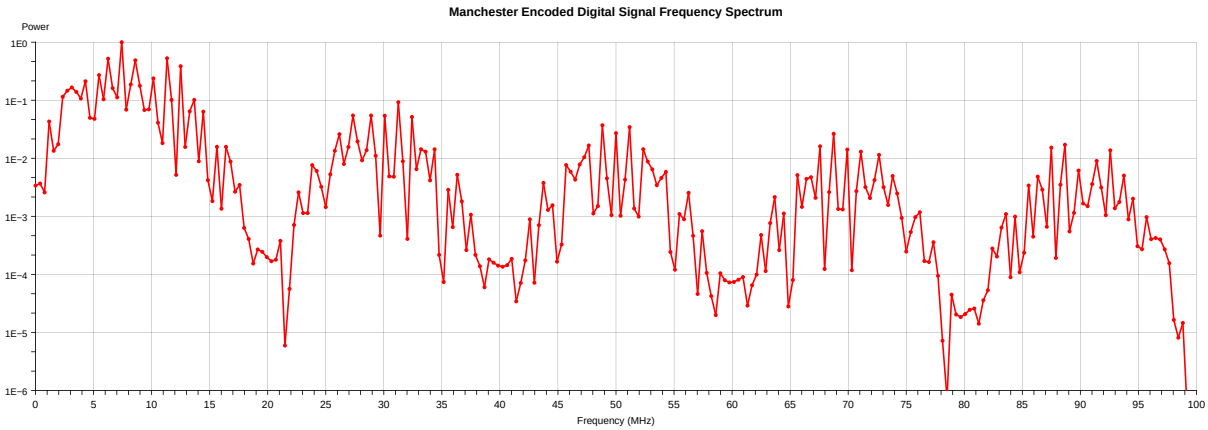
Example: Bit stream = 10110100, bit interval = 0.1 μsec



The receiver detects bit boundaries in an Ethernet signal as follows
Each Ethernet frame begins with a *preamble*: 7 bytes of 10101010 plus 1 byte of 10101011
The 10101010 bytes, when Manchester encoded, yield a *square wave* with a transition in the center of each bit interval
The receiver synchronizes its clock to these transitions
When the receiver sees two consecutive 1 bits, it signals the end of the preamble and the start of the actual frame data

Since there is a transition at the center of every bit interval, the receiver keep its clock synchronized by detecting these transitions
Thus, Manchester encoding yields a **self-clocking** waveform — there is no need for a separate clock signal to synchronize the transmitter's and receiver's bit boundaries

Example of a Manchester encoded waveform's frequency spectrum:

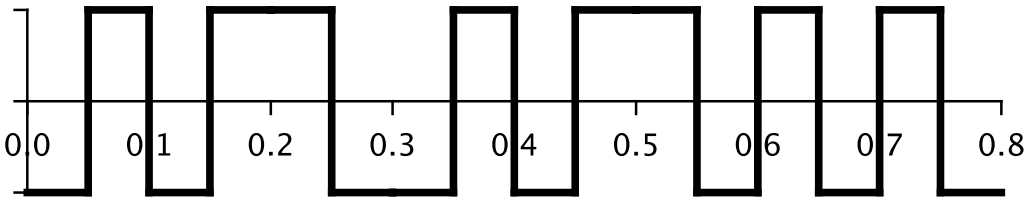


4.3 Differential Manchester Encoding

Used in token ring LANs (IEEE 802.5)

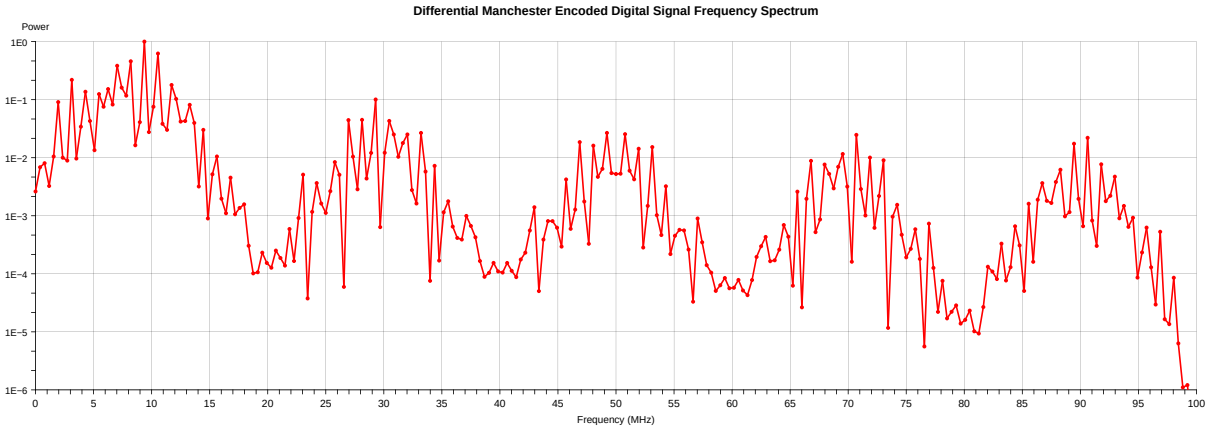
Two voltage levels
There is a transition at the center of each bit interval
For a 0 bit, there is a transition at the beginning of the bit interval (analogous to a phase shift of 0)
For a 1 bit, there is no transition at the beginning of the bit interval (analogous to a phase shift of π)

Example: Bit stream = 10110100, bit interval = 0.1 μsec



Like Manchester encoding, differential Manchester encoding yields a **self-clocking** waveform

Example of a differential Manchester encoded waveform's frequency spectrum:



4.4 MLT-3 Encoding – 100-Mbps Wired Ethernet

Used in 100-Mbps Ethernet LANs (IEEE 802.3) carried over Category 5 unshielded twisted pair cables (100Base-TX), the most common kind of 100-Mbps Ethernet cabling nowadays

There are three steps in encoding a stream of data bits:

Step 1. 4B/5B-NRZI Encoding
Each group of 4 bits is converted to a group of 5 bits
The 5-bit code groups are defined with enough transitions to provide self-clocking, like Manchester encoding
But while the Manchester encoded bit rate is 2 times the original bit rate (i.e., 200 Mbps), the 4B/5B-NRZI encoded bit rate is only 1.25 times the original bit rate (i.e., 125 Mbps)
This lower encoded bit rate results in a lower bandwidth signal, which can be transmitted over a Category 5 cable without excessive distortion

The 4B/5B-NRZI encodings are:

Original	Encoded
0000	10100
0001	01110
0010	11000
0011	11001
0100	01100
0101	01101
0110	01011
0111	01010

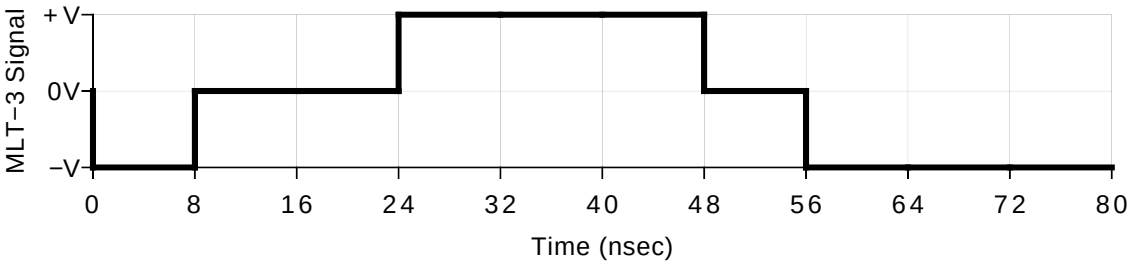
Original	Encoded
1000	11100
1001	11101
1010	11011
1011	11010
1100	10011
1101	10010
1110	10111
1111	10110

Step 2. Scrambling
The 4B/5B-NRZI encoded bit stream is exclusive-ored with a pseudorandom bit stream
This spreads the signal energy more evenly across the frequency spectrum, making the signal more resistant to noise

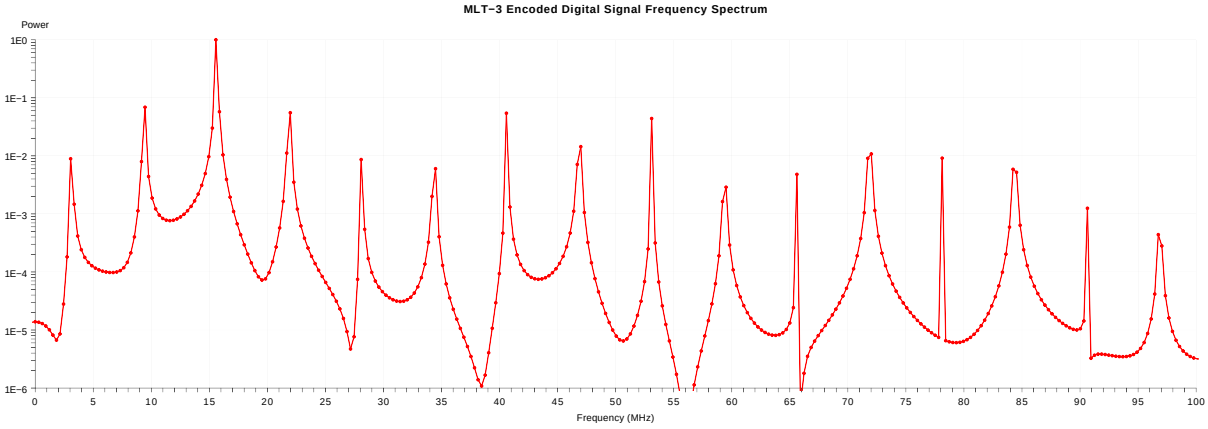
Step 3. MLT-3 Encoding
The 4B/5B-NRZI encoded, scrambled bit stream is further encoded using MLT-3 encoding
This shifts most of the signal energy below about 50 MHz, which reduces the amount of energy lost due to radiation out of the cable and reduces interference with adjacent cables

There are three voltage levels: –V, 0V, and +V
For a 0 bit, there is no transition at the start of the bit interval
For a 1 bit, there is a transition at the start of the bit interval; four cases:
If the previous transition was 0V → –V, transition –V → 0V
If the previous transition was –V → 0V, transition 0V → +V
If the previous transition was 0V → +V, transition +V → 0V
If the previous transition was +V → 0V, transition 0V → –V

Example: Original bit stream = 10110100
4B/5B-NRZI encoded bit stream = 1101001100
(Scrambling is omitted in this example)
MLT-3 encoded waveform:



Frequency spectrum:



4.5 1-Gbps Wired Ethernet

Carried over Category 5 unshielded twisted pair cables (1000Base-T), the same cables as 100-Mbps Ethernet

Each cable has four pairs of signal wires

100-Mbps Ethernet transmits a 125 million bits per second data stream on one pair and receives a 125 million bits per second data stream on another pair

1-Gbps Ethernet simultaneously transmits and receives a 125 million *symbols* per second, 2 bits per symbol data stream on all four pairs
$(125 \times 10^6$ symbols/second/pair) $\times$ (2 bits/symbol) $\times$ (4 pairs) $= 1 \times 10^9$ bits/second

100-Mbps Ethernet uses 3-level pulse amplitude modulation (PAM-3)—the MLT-3 waveform

1-Gbps Ethernet uses 5-level pulse amplitude modulation (PAM-5)

100-Mbps Ethernet does not do forward error correction

1-Gbps Ethernet does **forward error correction** by encoding the symbol stream using a **trellis code**
This is needed because PAM-5 is more susceptible to interference than PAM-3

With 5 amplitude levels on each pair and 4 pairs, $5^4 = 625$ different symbols can be transmitted
To send 8 bits with no redundancy, 256 symbols are needed
The trellis code adds redundant information, encoding 8 bits with 512 symbols (2X redundancy)
The redundant information lets the decoder correct transmission errors
(The remaining 113 symbols are used for control information)

5 Wireless Signals

5.1 Interference

With wireless networks, interference with and from signals that are not part of the network becomes a major concern

For example, some wireless Ethernet networks use frequencies in the unregulated Industrial/Scientific/Medical (ISM) band of frequencies around 2.4 GHz
"Unregulated" means anyone can transmit signals in this band without needing a license

So multiple wireless Ethernet networks; other wireless industrial, scientific, or medical devices; Bluetooth devices; and microwave ovens can all be putting out signals in this same band of frequencies at the same time
If we don't design the signals carefully, to avoid interfering with other signals, and to tolerate interference from other signals, we will get chaos and nothing will work

This is not so much of a problem with wired networks because the *cabling* is designed to minimize interference, so the *signal* doesn't have to
This becomes more of a problem with high data rate wired networks
For example, interference considerations drove the design of the 100-Mbps Ethernet signal (MLT-3 encoding)

5.2 Wireless Ethernet Standards

Standard	Data Rate (Mbps)	Frequency Band (GHz)	Encoding Method	Modulation Method
802.11 (1997)	1	2.4	FHSS	1-bit-per-symbol FM
	2	2.4	FHSS	2-bit-per-symbol FM
	1	2.4	DSSS	1-bit-per-symbol DPSK
	2	2.4	DSSS	2-bit-per-symbol DPSK
802.11b (1999)	5.5	2.4	DSSS	CCK
	11	2.4	DSSS	CCK
802.11a (1999)	6	5	OFDM	1-bit-per-symbol DPSK
	9	5	OFDM	1-bit-per-symbol DPSK
	12	5	OFDM	2-bit-per-symbol DPSK
	18	5	OFDM	2-bit-per-symbol DPSK
	24	5	OFDM	4-bit-per-symbol QAM
	36	5	OFDM	4-bit-per-symbol QAM
	48	5	OFDM	6-bit-per-symbol QAM
	54	5	OFDM	6-bit-per-symbol QAM
802.11g (2003)	6–54	2.4	Various	Various
802.11n (2009)	7.2–72.2	2.4, 5	OFDM	Various
	15–150	2.4, 5	OFDM	Various
802.11ad (2012)	up to 6912	60	OFDM	Various
802.11ac (2014)	up to 866.7	5	OFDM	Various

FHSS = Frequency Hopping Spread Spectrum
DSSS = Direct Sequence Spread Spectrum
OFDM = Orthogonal Frequency Division Multiplexing
FM = Frequency Modulation
DPSK = Differential Phase Shift Keying
CCK = Complementary Code Keying
QAM = Quadrature Amplitude Modulation

5.3 The 802.11 1-Mbps DSSS Signal

The signal uses one of the following carrier frequencies (in the U.S.):

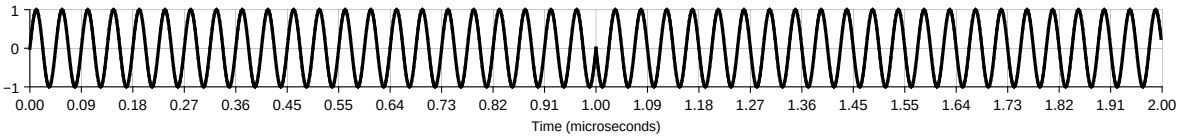
Channel	Frequency
1	2.412 GHz
2	2.417 GHz
3	2.422 GHz
4	2.427 GHz
5	2.432 GHz
6	2.437 GHz
7	2.442 GHz
8	2.447 GHz
9	2.452 GHz
10	2.457 GHz
11	2.462 GHz

The 1-Mbps bit stream modulates the carrier using DPSK with these phase shifts:
$\Delta\phi_0 = 0, \Delta\phi_1 = \pi$
Thus, there are 1 million symbols per second

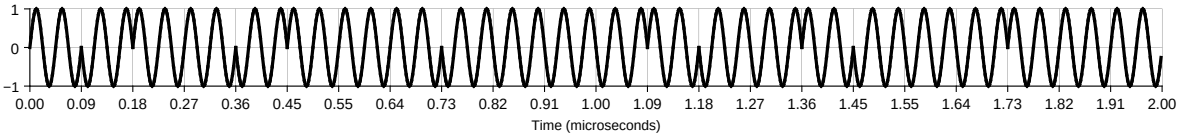
The DPSK-modulated signal is multiplied by a **chipping sequence**, a.k.a. a **spreading sequence**, a.k.a. a **pseudonoise (PN) sequence**

Each chip is $+1$ or -1
The chipping sequence consists of the following 11 chips continually repeated at a chip rate of 11 million chips per second:
$+1, -1, +1, +1, -1, +1, +1, +1, -1, -1, -1$
Thus, each modulated symbol is multiplied by the chipping sequence

Example of an 802.11 1-Mbps DSSS signal for the data bits 01
(The carrier frequency is reduced so the phase shifts will be visible)
The DPSK-modulated signal before being multiplied by the chipping sequence:



The DPSK-modulated signal after being multiplied by the chipping sequence:



To demodulate a received signal, first multiply by the chipping sequence to get back the original DPSK waveform, then demodulate the DPSK waveform to get back the data bits

Why is this called a *spread spectrum* signal?
In effect, multiplying by the chipping sequence changes the 1-Mbps bit stream into some other 11-Mbps bit stream
The transmitted signal's bit rate is effectively 11 times the original bit rate
Therefore, the bandwidth of the signal after chipping is 11 times the bandwidth of the signal before chipping
Chipping causes the signal's frequency spectrum to spread out — hence, "spread spectrum"

Why send the spread spectrum signal? Why not just send the original 1-MBps signal?

Without spread spectrum, the signal's energy is concentrated, giving a large signal power in a small range of frequencies — a *narrowband* signal
If there is another device transmitting a narrowband signal at the same or nearly the same frequency, the two signals will get mixed up, and neither device will work

With spread spectrum, the signal's energy is spread out, giving a small signal power in a wide range of frequencies
If there is another device transmitting a narrowband signal at the same or nearly the same frequency, the narrowband signal will affect only a small portion of the energy in the spread spectrum signal, and the spread spectrum signal can still be received successfully
Conversely, the spread spectrum signal just looks like low-power background noise to the narrowband signal, and the narrowband signal can still be received successfully

Spread spectrum *reduces interference*, which is very important for wireless signals

5.4 The 802.11 2-Mbps DSSS Signal

The 2-Mbps signal uses the same carrier frequencies as the 1-Mbps signal

The 2-Mbps bit stream modulates the carrier using two-bits-per-symbol DPSK with these phase shifts:
$\Delta\phi_{00} = 0, \Delta\phi_{01} = \pi/2, \Delta\phi_{10} = 3\pi/2, \Delta\phi_{11} = \pi$
Thus, there are 1 million symbols per second

The 1 million symbols per second DPSK-modulated signal is multiplied by the same 11 million chips per second chipping sequence:
$+1, -1, +1, +1, -1, +1, +1, +1, -1, -1, -1$
Thus, each modulated symbol is multiplied by the chipping sequence