

# Physical Layer

- **Signals propagate over a physical medium.**
  - *E.g.:* Copper wires, fiber, space
- **Encode bits in the signal**
  - Sender varies *something* (*e.g.*, turns light on and off)
  - Receiver samples and reverses the transform.
- **Fundamental Issues**
  - Speed of light:  $3 \times 10^8$  m/s, in copper  $\sim 2 \times 10^8$
  - Information transmission limits (Shannon, Nyquist)
  - Multiplexing the medium (if shared)

## Why not high=1v, low=0v? (square waves)

- They're not square when bandwidth-limited.
- Bandwidth – the frequencies that a channel propagates well.
  - signals consist of many frequency components.
  - attenuation (strength ↓) is a function of frequency.
  - delay is a function of frequency.
- Result: square signals don't propagate well over long distances.

## Using Carriers

- **Only send frequencies that are transmitted well.**
- ***Modulate* the signal to encode bits.**
  - Amplitude: vary the strength of the signal
  - Frequency: switch between a small number of frequencies.
  - Phase: change phase of signal to encode bits

# Nyquist's Theorem

Nyquist's Theorem places an upper bound on how quickly we can modulate, based on the bandwidth of the channel.

If a signal has been passed through a filter of bandwidth less than  $L$ , then  $2L$  samples/sec are sufficient to fully represent the signal.

There's simply no point in changing the signal more often than  $2L$ /sec. The channel can not represent enough detail to recover the original signal.

# Nyquist Example

Telephones remove all signals below 400hz, and above 3400hz. That leaves a bandwidth of 3khz (and explains why voices sound different over the phone).

When transmitting data over phone lines:

- **Sampling at greater than 6000hz is pointless.**
  - There's no more information available.
- **Sampling at less than 6000hz is wasteful.**
  - More information could have been sent.

How do modems transmit more than 6kbps?

## A Nyquist “loophole”?

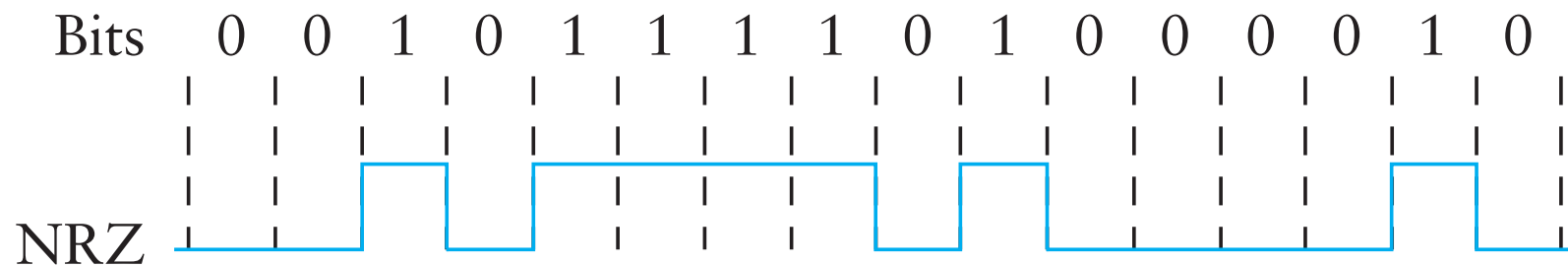
- We can only change  $2L/\text{sec}$ , so let's encode more bits per sample. (baud vs bits/sec)
- Suppose the channel passes frequencies from 1kHz to 2kHz.
  - 1 bit per sample: alternate between 1kHz and 2kHz signal.
  - 2 bits: Send one of 1kHz, 1.33khs, 1.66kHz, 2.khz.
  - n bits: Choose among  $2^n$  frequencies between 1kHz & 2kHz.
- Party time! No limit to information capacity!

# How much can we really encode?

- **Depends on frequency, and signal/noise ratio**
- **Shannon:**  $C = B \log_2(1 + S/N)$ 
  - $C$  is channel capacity in bps
  - $B$  is bandwidth of line
  - $S$  and  $N$  are average signal & noise power
- **Example: Telephone line**
  - 3 KHz b/w, 30 db S/N =  $10^{30/10} = 1000$
  - $C \approx 30$  Kbps (so 56 Kbps modems need better S/N ratio)

# NRZ

- Now assume we can somehow modulate signal
- How to encode binary data onto signals?
- One approach: Non-return to zero (NRZ)
  - Transmit 0 as low, 1 as high



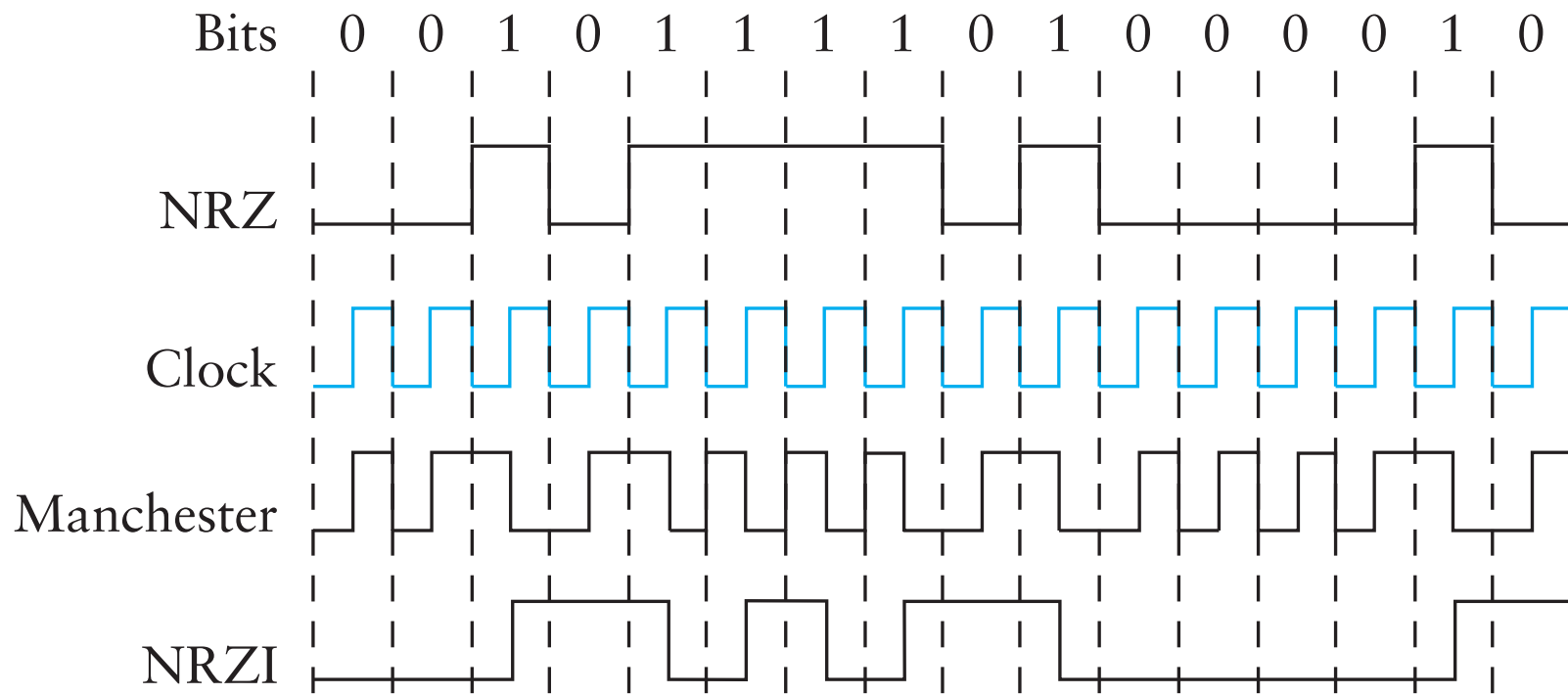
## NRZ drawbacks

- **Consecutive 1s or 0s are problematic**
- **Non signal could be interpreted as 0s (or vice versa)**
- **“Baseline wander” problem**
  - Where is threshold between low and high?
  - Could compare signal to average value, but avg. will drift
- **Clock recovery problem**
  - For long stretch of consecutive bits, might miscount

# Alternate Encodings

- **Non-return to Zero Inverted (NRZI)**
  - Encode 1 with transition from current signal
  - Encode 0 by staying at same level
  - At least solves problem of consecutive 1s
- **Manchester**
  - XOR of NRZ symbol with square-wave clock
  - But now lose 50% maximum potential data rate.

# Illustration



## **4B/5B**

- **Every 4 bits of data encoded in a 5-bit code (see Table 2.4 in text)**
- **Remaining 13 codes used for other purposes**
  - 11111 – line idle, 00000 – line dead, 00100 halt
- **5-bit codes selected to have no more than one leading 0 and no more than two trailing 0s**
  - thus, never get more than three consecutive 0s
- **resulting 5bit codes are transmitted using NRZI**
- **achieves 80% efficiency**

# Wireless Concerns

- Power
- Noise
- Security

Wireless devices tend to be mobile, running on battery power. The medium is noisy, and anyone can listen in.

# RF Technology

- **Spread Spectrum technology**

- spread signal over a wide bandwidth
- insensitive to noise
- frequency hopping
  - signal moves across several frequencies sequentially
  - pseudo-random sequence known only to sender and receivers
  - used in 802.11b (though sequence is well-known)
- direct sequence
  - digital input xor'd with higher frequency bit string
  - modulated on carrier using differential phase-shift keying
  - result is widely spread signal, but same average power

# Software Tools: netcat

netcat (nc) is a “swiss-army knife” for TCP and UDP interactions.

```
edge> nc -l -p 3333
```

```
edge> nc localhost 3333
```

```
<type something here> (it shows up above)
```

```
edge> cat backup.iso | nc -l -p 3333
```

```
shrub> nc hostname 3333 > backup.iso
```

UDP, too. Use -u

## Test the server

```
> snowcast_server 3003 file1.mp3 file2.mp3 }
```

```
> nc localhost 3003
```

```
HELLO
```

```
^B,server did not receive a valid HELLO command
```

```
> perl -e 'print "\0\4\4"' | nc localhost 3003 > response.txt
```

```
> ls -l response.txt
```

```
-rw-rw-r-- 1 jj nogroup 3 Jan 29 09:55 response.txt
```

## Test the client

```
> nc -l -p 3004
```

```
> snowcast_control localhost 3004 2000
```

You should see the 3 byte hello command. Redirect it to a file for inspection.

## Coming up

- **Fri 30: Snowcast Checkpoint**
- **Tue 3: Link layer**
- **Wed 4: 2pm-4pm Claudiu's hours**
- **Thu 5: Switching, 6pm-8pm Rassi's hours**
- **Fri 6: Snowcast due.**