

# **ECE 435 – Network Engineering**

## **Lecture 26**

Vince Weaver

`https://web.eece.maine.edu/~vweaver`

`vincent.weaver@maine.edu`

2 April 2025

# Announcements

- HW#9 Due Friday
- There will probably be a week gap before HW#10 assigned



# Error Control / Reliability

- You detect an error, what can you do?
  - Drop it on the floor? (“Best Effort”) Maybe hope another layer helps
  - Get an acknowledgement saying was correct?
  - What if something happens and the entire frame lost? Receiver never gets it one way or another. Sender waits forever?
  - Use a timer. If no response send again
  - What happens if you send multiple times and then



eventually both get there? Often have a sequence number to track if there are multiple.

- Very quickly end up re-implementing the net layer at this layer.



# Error Detection/Correction

- Are errors a problem? If sending 1000 bit frames, and error rate is .001 per bit, then if even distributed on average each frame have an error. Are errors evenly distributed? What if 1000 in a row then none? (bursty)
- Error-Detection Codes – let you tell if an error happened what to do if error happens? resend. doable if errors infrequent (reliable connection)
- Error-Correcting Codes – let you fix an error



# Hamming Distance

- Number of bits that differ
- Can calc by exclusive oring then counting the ones.
- $0101\ 1101 = 1000 = 1$
- If hamming distance of  $N$  then takes  $N$  single-bit errors to convert between the two
- To detect  $N$  errors you need hamming distance of  $N+1$  to ensure than  $N$  errors can create another valid code



- To correct  $N$  errors you need  $2N+1$  distance, that way even with  $N$  errors it is still closer to changed value than any other
- parity bit. Chosen so code word is always even (or odd) can detect single bit error (essentially 1-bit CRC)
- Hamming code for detecting errors



# Error Detecting Codes

- One way: arrange bits in rectangle, take parity bits across both rows and columns
- Polynomial codes: CRC (cyclic redundancy check)





# Checksum

- Like in TCP/UDP
- Easy to calculate in software



# Cyclic Redundancy Check (CRC)

- Redundancy because adding extra bits to message
- Easier to calculate with hardware
- Polynomial, 110001 means  $x^5 + x^4 + x^0$
- Agree on generator in advance. High and low bits must be 1. CRC is calculated. Value to check must be longer than generator
- Append CRC on end, and when run through the result is zero. Any remainder means an error
- IEEE 802 uses  $x^{32} + x^{26} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} +$



$$x^8 + x^7 + x^5 + x^4 + x^2 + x^1 + 1$$

which can detect any burst error less than 32 and all odd number bits

- might seem hard, but easy to make in hardware with a shift register and some xor gates.
- CRC can find single bit errors, double bit errors, bursts of errors less than length of polynomial.
- Explaining how it works is “mathematically complex” says open source approach book

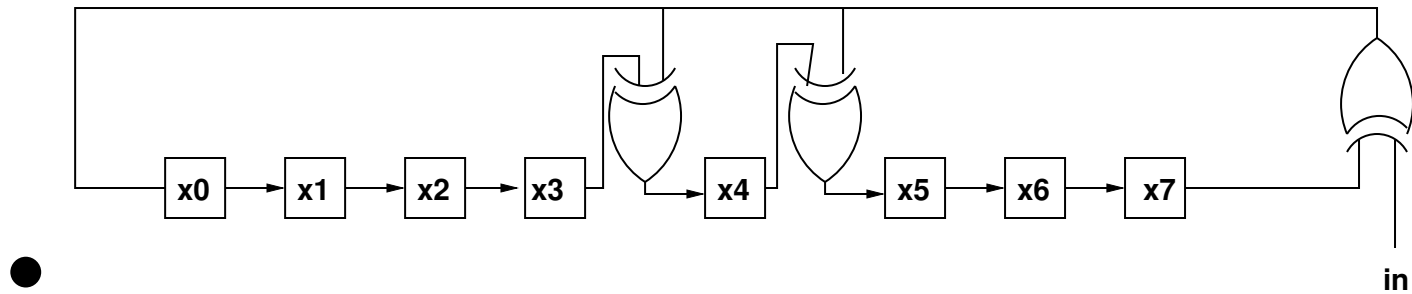


# 1-wire CRC check example

- Usually used in hardware, harder to implement in software
- Can detect all double-bit errors, any double bit errors, any cluster within an 8-bit window
- if CRCs with itself gets 0 at the end, how hardware detects correct address.
- $X^8 + X^5 + X^4 + X^1$



- Fill with zero, shift values in.



# Link Layer Protocols (Obsolete/Fading)

- Token Ring
- HIPPI, FDDI – fiber distributed data interface
- Fibre Channel
- ATM
- ISDN
- X.25
- AX.25
- PPP / PPPoE (? fading)



# Link Layer Protocols (Current)

- Ethernet (802.3)
- WLAN (802.11)
- Bluetooth (802.15)
- LTE/WiMAX (802.16)
- DSL



# PPP / HDLC

- Older textbooks like to talk about PPP
- Point to Point Protocol, used when serial/modems were the rage
- Remnant still exists, ASDL often tunneled via PPPoE (PPP over Ethernet)
- PPP based on earlier protocol called HDLC
- We will skip over it as not being super relevant anymore





# Earliest Ethernet History

- Proposed by Bob Metcalfe in 1973  
(went on to found 3Com)  
2022 Turing Award
- Metcalfe, Boggs, Thacker, and Lampson listed on patent
- Inspired by ALOHAnet, a wireless network in Hawaii, allow users on various islands to connect to server on Oahu
- Various competing local networks, Ethernet won in the end



# Interesting article on the history of Ethernet

<https://lostintransit.se/2024/08/21/ethernet-history-deepdive-why-do-we-have-different-frame-types/>



# Token Ring (Ethernet Competitor)

- Guaranteed Deterministic Delivery  
(vs Ethernet: best effort)
- Dates to 1970s
- Standardized by IBM, 1984, IEEE 802.5 (note, not RFC)
- 4Mbps, eventually shielded twisted pair, eventually 16Mbps, 100Mbps and 1Gbps
- 3-byte frame passed around gives permission to transmit
- More complex, no crossover cable (direct connect two machines),



- Supports multiple identical MAC addresses (?)
- Deterministic time to get to transmit
- Frames can have different access priorities
- Empty token passed around. If data to transmit, put in. Then passes around until it gets to receiver, removed, and back to passing empty token. When gets back to originator it knows it has been received.
- Token Bus, GM, IEEE802.4 (withdrawn) like ring, but virtual ring. Needed to know neighbors to pass token. Guaranteed worst case transmit time.



# Why did Ethernet win?

- Other competitors: FDDI, ATM, DQDB (?)
- Why did it win? Simpler and thus cheaper.
- Why simpler?  
No priority mechanism, no QoS, no central control
- Both started out using expensive cabling, but Ethernet moved to cheaper unshielded twisted pair (token ring eventually did too)
- Token ring cards generally a lot more expensive than Ethernet



- Token ring network equipment (MAU) more expensive than a hub/switch



# The Ethernet Progression

- Low speed (3Mbps) → High speed (400Gbps)
- Shared media → dedicated media
- LAN → WAN



# More Early Ethernet History

- 1972 – experimental 3Mbps (Xerox Alto)
- 1981 – DIX (DEC/Intel/Xerox) ver 1 (10Mbps)
- Standardized in 1981
- 1982 – DIX ver 2





# Ethernet Naming

- Naming: Speed/BROAD, BASE, PASS/PHY
- Almost all is baseband (narrow frequency, vs broadband).
- PHY originally was distance could travel (in 100m) but now medium type.



# Thick Ethernet

- 1983 – IEEE 802.3/10BASE5
- “Thick Ethernet”, up to 500m often yellow or orange (standard suggests yellow)
- Looks like garden hose.
- Vampire Tap, AUI connector, drill into cable, at 2.5m intervals (to avoid reflections)
- Terminated on each end, one bad connection could ruin for all



# Naming note on IEEE 802.3

- Why do IEEE network standards start with 802?
- Probably just next available?
- Rumor because first meeting was Feb. 1980  
That's probably just a co-incidence
- For example, IEEE floating point is IEEE 754 but first meeting was not April 1975



# Thin Ethernet

- 1985 – 10BASE2
- “thin net”, thinner connections, BNC connectors, T connectors (185m, rounded up to 200m)
- 50 ohm terminator
- Issues with grounding loops (one and only one must be tied to ground, otherwise can get current flow in shielding)
- How to detect network problem? Send pulse, look for echo



# 10BASE-T

- 1990
- 10BASE-T – twisted pair (Cat3) , needed hub
- 1993 – 10BASE-F – fiber



# Faster Ethernet

- We will talk about this in more detail later
- 1995 – 100BASE-T, 100BASE-TX 4B5B MLT-3 cat5  
two twisted pairs
- 1997 – Full-duplex
- 1998/1999 – 1000BASE-TX PAM-5, four twisted pairs,  
can transfer in both directions on one pair using  
DSP/echo cancellation
- 2006 – 10GBASE-T
- 2010 – 40G and 100G



- 2017 – 400GB



# “Classic” Ethernet Overview

- Not really used anymore, but a classic example of what a relatively easy-to-understand link-level interface is like





# Ethernet MAC

- CSMA/CD “Carrier sense multiple access with collision detection”
- First senses cable (how? – see later)
- If busy, waits
- Sends. If collision, jams the cable aborts transmission, waits random back off time before retrying.
- Exponential backoff. Randomly choose time from 0 to



$2^k - 1$  where  $k$  is number of tries (capping at 10). Time slot is 512 bits for 10/100, 4096 for 1Gbs

- on newer full-duplex links no need for carrier sense and collision detection not needed



# Ethernet Collisions

- In order to work properly, twice round-trip time needs to be less than time needed to transmit minimal (64-byte) frame, otherwise not possible to notice collision in time and frame loss
- This limits network size to collision domain
- Bits wasted is not bad, collision often caught in the preamble

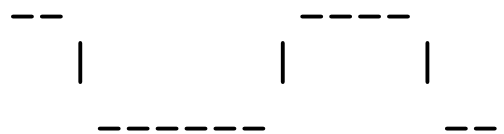


# Physical Layer Encoding Background: NRZ/NRZI

For initial comparison, these are the simplest encodings.


NRZ (non-return to zero) is just 0 low, 1 high

1 0 0 1 1 0



NRZI (NRZ Inverted) (1 flip, 0 same)

1 0 1 1 0 0 1



!

!

!



# Physical Layer – Voltages

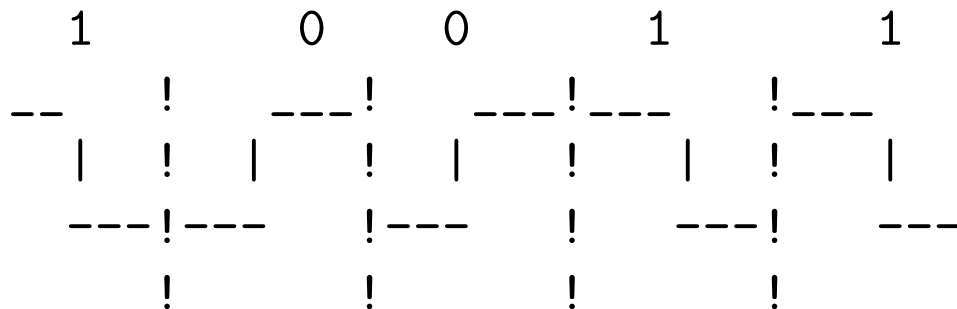
- Could just use 0V for 0 and 5V for 1
- What if you want to be able to detect idle/unused?
- Have Idle be 0V, and  $+1V/-1V$  for 1/0
- This also avoids having a DC bias to your signal



# Manchester Encoding (Ethernet)

- 1 is high to low transition.
- 0 is low to high transition.
- Always a transition in the middle of an interval.
- Disadvantage, need twice as much bandwidth

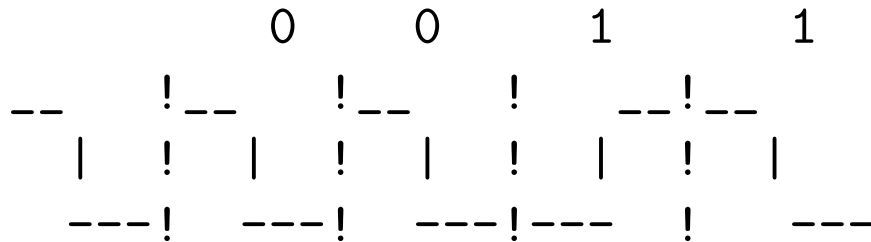
Manchester (10BASET Ethernet)



# Differential Manchester (Token Ring)

- transition at start of interval means 0
- lack of transition means 1
- Still transition in the middle
- More complex but better noise handling

Differential Manchester (Token Ring)





# Ethernet on the Wire

- Manchester Encoding
- High 0.85V and low -0.85V

