

ECE 435 – Network Engineering

Lecture 28

Vince Weaver

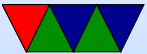
`https://web.eece.maine.edu/~vweaver`

`vincent.weaver@maine.edu`

7 April 2025

Announcements

- HW#10 (Ethernet) was posted



Direct Connection Ethernet

- Direct connect two machines with one cable
- Used to need special “crossover” cable to swap TX and RX lines
- Modern cards can detect direct connect and swap the wires for you



Ethernet Security

- Traditional hub, all machines saw all packets
- With tcpdump could monitor all packets on network, back in day all plain text. e-mail, web-browsing, chat, passwords, telnet
- tcpdump put card in “promiscuous” mode which let it intercept all packets instead of ignoring ones not to system
- Why so low security? Old day trust people at your work/office, also was probably expensive/difficult to get



an unauthorized UNIX workstation with Ethernet card
and root access on the local network

- Lights on hub



Power over Ethernet

- In 10/100 Base T, only 2 of the 4 pairs in Cat5 used.
So send voltage down spare pairs
- Gigabit, all pairs used, so what can you do?
send DC voltage down with the signals floating on top
- Standards
 - Original POE: 44 VDC, 15.4W
 - POE+ 25W



Power over Ethernet Switch

- Need special switch to send power, and device on other end has to support it.
- Takes time to negotiate power, so can take many seconds (or more) for full power available to device
- Raspberry Pi hat, used on new Pi4 cluster of mine



Wake on LAN

- Can have whole machine powered off, but just enough power to keep Ethernet card alive
- It watches for “Magic Frame”
- If it gets it, powers on the system



Classic 10Mbps is too slow!

- 10Mbps not fast enough! What can we do?
- FDDI and Fibrechannel (fast optic-ring), too expensive
- Can we just multiply all speeds by 10? Or else come up with some completely new better thing?
- IEEE group met, ended up with two groups
 - One wanted to keep everything same, just faster: 802.3u 1995
 - The other group went off and made 802.12 100BaseVG (which failed) (voice grade, 4 pairs of cat3)



“Fast” Ethernet 802.3u (100Mbps)

- 100BASE-TX most common
- BASE means baseband, not modulated on a carrier
- Bit time from 100nsec to 10nsec
- Uses twisted pair/switches, no coax
 - To use cat3 100BASE-T4 wiring needed 4 twisted pair and complex encoding, no Manchester, ternary
 - To use cat5 wiring 100BASE-TX. Two twisted pair, one to hub, one from.
- The pairs are differential pairs, like USB



100MBit MAC/PHY Split

- Often split between MAC (media access controller) and PHY (physical interface).
- Card configures the PHY via the MII (media independent interface)
- Goal was you could have same card but interchangeable PHY (twisted pair, fiber, etc)



MII Interface

- 4bit bus
- Interface requires 18 signals, only two can be shared if multiple PHY
- So RMII (reduced) was designed. Clock doubled, only 2-bit bus. Fewer signal wires.



100BASE-TX

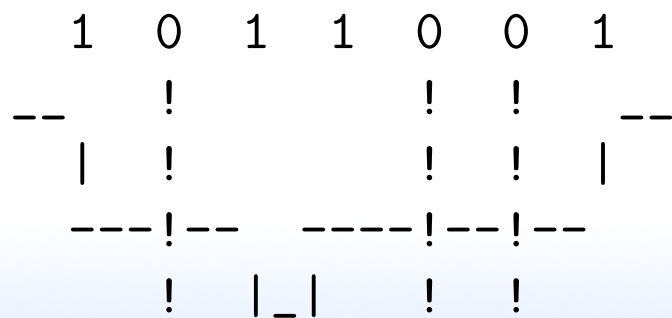
- 2 pairs inside cat5 cable
- One pair 100MB each direction, full duplex
- 100m distance
- Raw bits (4 bits wide at 25MHz at MII) go through 4B/5B encoding clocked at 125MHz
- 4B/5B means 4 bits encoded as 5, this allows at least two transitions per 5 bits, allows special patterns for end frame or error
- TX then goes through MLT-3 encoding



MLT-3 Encoding

- MLT-3 has three voltages, -1, 0, +1
- Cycles through them -1, 0, +1, 0, repeats
- For a 1, it transitions to next, for a 0, no transition
- 31.25MHz, like copper version of FDDI (CDDI/TP-PMD)
- Needs encoding that limits consecutive zeros

MLT-3



4B/5B Encoding

- At least two transitions per 5-bit code
- “Provides DC equalization and spectrum shaping”

What does that mean in simple terms? Hard to explain.
The PHY often filters out low-frequency values so 4B/5B makes life easier in that case



4B/5B Encoding (layout)

hex	4B	5B
0	0000	11110
1	0001	01001
2	0010	10100
3	0011	10101
4	0100	01010
5	0101	01011
6	0110	01110
7	0111	01111
8	1000	10010
9	1001	10011
A	1010	10110
B	1011	10111
C	1100	11010
D	1101	11011
E	1110	11100
F	1111	11101

Control Char	5B	meaning
H	00100	halt
I	11111	idle
J	11000	start #1
K	10001	Start #2
L	00110	Start #3
Q	00000	Quiet
R	00111	Reset
S	11001	Set
T	01101	End (terminate)



Interesting Gigabit Ethernet Article

- <https://resources.altium.com/p/gigabit-ethernet-101-basics-implementation>



Gigabit Ethernet

- Two task forces working in 1998/1999
 - 802.3z 1998 (fiber), 802.3ab 1999 (copper)
 - Could still use hub, problem was the CSMA/CD restriction.
 - About 200m for 100Mbps.
 - For Gb would have been 20m which is not very far.
 - Carrier extension: hardware transparently pads frames to 512 bytes
- Wasteful, 512 bytes to send 64 bytes of data



- Frame bursting: allow sender to send sequence of multiple frames grouped together
- Better solution is just use full duplex
- 1000Base-SX (fiber)/LX (fiber)/CX (shielded)/T (cat 5), more



Gigabit Ethernet – Fiber

- No Manchester, 8B/10B encoding.
- Chosen so no more than four identical bits in row,
- No more than six 0s or six 1s
- need transitions to keep in sync



Gigabit Copper – 1000BASE-T

- PAM-5, 5 voltage levels, 00, 01, 10, 11, or control. So 8 bits per clock cycle per pair,
- 4 pairs running at 125MHz, 2 bits per clock(?), so 1GBps
- simultaneous transmission in both directions with adaptive equalization (using DSPs)
 - Actually can send and receive at same time on one pair
 - Basically, when you send the values add up
 - However if you send 1 1 0 1 an you notice on the line



- 0 2 0 2 you can figure out the other side sent -1 1 0 1
- Slightly tricky because voltage loss on line
 - 5-level pulse-level modulation (PAM-5) [technically 100BASE-TX is PAM-3]. Diagram? looks sort of like a sine wave as cycle through the voltages.
 - four-dimensional trellis coded modulation (TCM) 6dB coding gain across the four pairs
Trellis coding also provides error correction
 - Autonegotiation of speed (see more details later)



Gigabit Ethernet – Other

- Try to balance 0s and 1s? keep DC component low so can pass through transformers?
- Fast enough that computers at time had trouble saturating such a connection



Gigabit Ethernet – Jumbo Frames

- 1500 byte MTU a bit limited these days, leads to lot of overhead and excessive number of packets
- Push for Jumbo frames. Most common size 9000 bytes (6×1500)

Possibly used in HPC setups?

- Lesser push for something slightly bigger than 1500 to allow encapsulating traditional frames
- Extra length not technically a problem when “type” used in frame header and OOB used to indicate frame end



- Problem is if all equipment does not support it, don't have way to allocate such big buffers
- Another issue, the Ethernet CRC less good at catching bit flips if 9000 bytes (and TCP/UDP checksum not good here either)
- One possibility (used by SCTP) is more advanced CRC, Castagnoli CRC polynomial



Even Faster Ethernet

http://www.theregister.co.uk/2017/02/06/decoding_25gb_ethernet_and_beyond/

- Misquote: Not sure what the network will be like in 30 years, but they will call it Ethernet.



10Gb Ethernet

- 10Gb: 802.3ae-2002. Full duplex, switches only
- Need Cat6a or Cat7 for links up to 100m
- Expensive. Lots of kinds.
- 10GBASE-T, 802.3an-2006 100m over cat6a, 55m Cat6
- additional encoding overhead, higher latency
- Tomlinson-Harashin precoding (THP), PAM15 in two-dimensional checkerboard DSQ128 at 800Msymbol/s



2.5Gb Ethernet

- 2.5Gb: 802.3bz (Sep 2016?)
- Like 10Gb but slower. Can't run 10Gb over Cat5e
- Power over Ethernet (for using on wireless access points)
- Power with signal overlaid on top.
- 2.5Gb on Cat 5e, 5Gb on Cat6



25Gb Ethernet

- 25Gb, 802.3by. 25GBASE-T, 50GBASE-T. Available, if copper only a few meters
- Introduced as 10GB not fast enough, but 40GB too expensive
- SFP28 transceivers, both optical and copper



40Gb Ethernet

- 40GBASE-T twisted pair 40Gbit/s 30m. QFSP+ connectors, like Infiniband
- Terabit? still under discussion
- 802.3ba – 2010 – 1m backplane, 100m multi-mode fiber, 10km single-mode fiber
- 802.3bg –
- 802.3bq – 2016 – 4-pair balanced twisted pair 30m
- QSFP+ transceivers, quad small form-factor pluggable, four 10GB lanes



- 802.3ba-2010, 802.3bg-2011, 802.3bj-2014, 802.3bm-2015



100Gb Ethernet

- 4 lanes of 25GB



Terabit Ethernet?

- Maybe someday?
- Still under discussion



Energy Efficient Ethernet

- Energy Efficient Ethernet (EEE) IEEE 802.3az (2010)
- Turn off when connection not up
- Use less power on shorter cables
- Low-power mode when idle



Autonegotiation

- How figure out line speed and duplex
- Series of pulses sent along periodically
- If not received for 150ms, link is detected as down
- Encoded in the 16-bit series of pulses is the speed / duplex available
- Only uses two pairs for this, can be trouble if pairs missing.



Auto-sensing vs Auto-negotiation

- When you connect, each side sends list of what it supports
- Problem, if they have the same list will they pick the same to use? Not always
- If somehow doesn't send list, has to guess, passively
- This is difficult, why fast ethernet uses auto-negotiating rather than auto-sensing
- need to detect speed, and duplex
- Speed – look at 1010101010 at start of frame



- Duplex – try to cause a collision



Auto-negotiation

- Similar to link integrity test pulses (LIT) unipolar positive-only pulses, 100ns
- That's how sense link
- For negotiation, fast-link pulse, 16-bit pattern describes link
- duplex, if it can't see a negotiation defaults to half-duplex as it's safe?



Other things to Investigate

- IEEE 1588 Precision Time Protocol (PTP)
- Synchronized Ethernet (SyncE)



What does your machine have

- skylake machine:

```
[ 18.240021] e1000e: eth0 NIC Link is Up 1000 Mbps Full Duplex, Flow Control: Rx/Tx
```

- Raspberry Pi:

```
[ 77.110505] smsc95xx 1-1.1:1.0 eth0: link up, 100Mbps, full-duplex, lpa 0xC5E1
```

- Haswell machine:

```
[ 3.907651] tg3 0000:03:00.0 eth0: Tigon3 [partno(BCM95761) rev 5761100]  
(PCI Express) MAC address f0:92:1c:f5:e8:f3  
[ 3.919115] tg3 0000:03:00.0 eth0: attached PHY is 5761  
(10/100/1000Base-T Ethernet) (WireSpeed[1], EEE[0])
```




```
[ 3.929838] tg3 0000:03:00.0 eth0: RXchecksums[1] LinkChgREG[0] MIirq[0] ASF[1] TSOcap[1]
[ 3.938174] tg3 0000:03:00.0 eth0: dma_rwctrl[76180000] dma_mask[64-bit]
[ 13.758613] IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
[ 15.404905] tg3 0000:03:00.0 eth0: Link is up at 100 Mbps, full duplex
[ 15.411479] tg3 0000:03:00.0 eth0: Flow control is on for TX and on for RX
```



Linux OS Support

- When frame comes in, interrupt comes in
- Allocates `sk_buff` copies in
- Old: `net_if_rx()` interrupt, `net_rx_action()`
interrupt/polling
- `net_if_receive_skb()`
- passes it to proper net level (`ip_rcv()`,
`ip_ipsv6_rcv()`, `arp_rcv()`)



- for send
- `net_tx_action()`
 `dev_queue_xmit()` and then deallocate `sk_buff`
- `qdisc_run()` selects next frame to transmit and calls `dequeue_skb()`



Other Wired Connections

- Note: all high-speed copper interconnects have similar issues to ethernet
- Low speeds (i2c, SPI, etc) just plain NRZ
- Faster like USB are?
Step up for each generation
- PCIe?

