Background Material 1: Getting stuff from here to there
Or
How I learned to love OSI layers 1-3

Outline

• Switching and Multiplexing

• Link-Layer

• Routing-Layer

• Physical-Layer Encoding

Packet vs. Circuit Switching

• Packet-switching: Benefits
  • Ability to exploit statistical multiplexing
  • More efficient bandwidth usage

• Packet switching: Concerns
  • Needs to buffer and deal with congestion:
  • More complex switches
  • Harder to provide good network services (e.g., delay and bandwidth guarantees)

Amplitude and Frequency Modulation

0 0 1 1 0 0 1 1 0 0 0 1 1 1 0 0 0 1 1 0 0 0 1 1 1 0

0 1 1 0 1 1 0 0 0 1 1 1 0 0 0 1 1 0 0 0 1 1 0

0 1 1 0 1 1 0 0 0 1 1 0 0 0 1 1 0
**Capacity of a Noisy Channel**

- Can’t add infinite symbols - you have to be able to tell them apart. This is where noise comes in.
- Shannon’s theorem:
  - $C = B \times \log(1 + S/N)$
  - $C$: maximum capacity (bps)
  - $B$: channel bandwidth (Hz)
  - $S/N$: signal to noise ratio of the channel
  - Often expressed in decibels (dB). $10 \log(S/N)$.
- Example:
  - Local loop bandwidth: 3200 Hz
  - Typical $S/N$: 1000 (30dB)
  - What is the upper limit on capacity?
  - Modems: Teleco internally converts to 56kbit/s digital signal, which sets a limit on $B$ and the $S/N$.

**Time Division Multiplexing**

- Different users use the wire at different points in time.
- Aggregate bandwidth also requires more spectrum.

**Frequency Division Multiplexing: Multiple Channels**

- Determines Bandwidth of Link
- Determines Bandwidth of Channel
- Different Carrier Frequencies

**Frequency versus Time-division Multiplexing**

- With frequency-division multiplexing different users use different parts of the frequency spectrum.
  - I.e. each user can send at the same time at reduced rate
  - Example: roommates
- With time-division multiplexing different users send at different times.
  - I.e. each user can send at full speed some of the time
  - Example: a time-share condo
- The two solutions can be combined.
  - Example: a time-share roommate
  - Example: GSM
Outline

- Switching and Multiplexing
- Link-Layer
  - Ethernet and CSMA/CD
  - Bridges/Switches
- Routing-Layer
- Physical-Layer

Ethernet MAC (CSMA/CD)

- Carrier Sense Multiple Access/Collision Detection

- Ethernet Backoff Calculation

  - Exponentially increasing random delay
  - Infer senders from # of collisions
  - More senders $\rightarrow$ increase wait time
  - First collision: choose K from $\{0,1\}$; delay is $K \times 512$ bit transmission times
  - After second collision: choose K from $\{0,1,2,3\}$…
  - After ten or more collisions, choose K from $\{0,1,2,3,4,\ldots,1023\}$

Collisions
Minimum Packet Size

- What if two people sent really small packets
  - How do you find collision?

- Consider:
  - Worst case RTT
  - How fast bits can be sent

Ethernet Collision Detect

- Min packet length > 2x max prop delay
  - If A, B are at opposite sides of link, and B starts one link prop delay after A
  - Jam network for 32-48 bits after collision, then stop sending
  - Ensures that everyone notices collision

Ethernet Frame Structure

- Sending adapter encapsulates IP datagram (or other network layer protocol packet) in Ethernet frame

<table>
<thead>
<tr>
<th>Preamble</th>
<th>Dest Address</th>
<th>Source Address</th>
<th>Data</th>
<th>CRC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Ethernet Frame Structure (cont.)

- Addresses: 6 bytes
  - Each adapter is given a globally unique address at manufacturing time
    - Address space is allocated to manufacturers
      - 24 bits identify manufacturer
      - E.g., 0:0:15:* \(\rightarrow\) 3com adapter
    - Frame is received by all adapters on a LAN and dropped if address does not match
  - Special addresses
    - Broadcast – FF:FF:FF:FF:FF:FF is “everybody”
    - Range of addresses allocated to multicast
      - Adapter maintains list of multicast groups node is interested in
Summary

- CSMA/CD → carrier sense multiple access with collision detection
  - Why do we need exponential backoff?
  - Why does collision happen?
  - Why do we need a minimum packet size?
    - How does this scale with speed? (Related to HW)
- Ethernet
  - What is the purpose of different header fields?
  - What do Ethernet addresses look like?
  - What are some alternatives to Ethernet design?

Scale

- What breaks when we keep adding people to the same wire?
- Only solution: split up the people onto multiple wires
  - But how can they talk to each other?

Problem 1 – Reconnecting LANs

- When should these boxes forward packets between wires?
- How do you specify a destination?
- How does your packet find its way?
### Transparent Bridges / Switches

- **Design goals:**
  - Self-configuring without hardware or software changes
  - Bridge do not impact the operation of the individual LANs

- **Three parts to making bridges transparent:**
  1. Forwarding frames
  2. Learning addresses/host locations
  3. Spanning tree algorithm

### Frame Forwarding

- A machine with MAC Address lies in the direction of number port of the bridge
- For every packet, the bridge “looks up” the entry for the packet’s destination MAC address and forwards the packet on that port.
  - Other packets are broadcast – why?
  - Timer is used to flush old entries

### Spanning Tree Bridges

- More complex topologies can provide redundancy.
  - But can also create loops.
  - What is the problem with loops?
  - Solution: spanning tree

### Outline

- Switching and Multiplexing
- Link-Layer
- Routing-Layer
  - IP
  - IP Routing
  - MPLS
- Physical-Layer
IP Addresses

- Fixed length: 32 bits
- Initial classful structure (1981) (not relevant now!!)
- Total IP address size: 4 billion
  - Class A: 128 networks, 16M hosts
  - Class B: 16K networks, 64K hosts
  - Class C: 2M networks, 256 hosts

<table>
<thead>
<tr>
<th>High Order Bits</th>
<th>Format</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>7 bits of net, 24 bits of host</td>
<td>A</td>
</tr>
<tr>
<td>10</td>
<td>14 bits of net, 16 bits of host</td>
<td>B</td>
</tr>
<tr>
<td>110</td>
<td>21 bits of net, 8 bits of host</td>
<td>C</td>
</tr>
</tbody>
</table>

Original IP Route Lookup

- Address would specify prefix for forwarding table
  - Simple lookup
- www.cmu.edu address 128.2.11.43
  - Class B address – class + network is 128.2
  - Lookup 128.2 in forwarding table
  - Prefix – part of address that really matters for routing
- Forwarding table contains
  - List of class+network entries
  - A few fixed prefix lengths (8/16/24)
- Large tables
  - 2 Million class C networks

IP Address Classes (Some are Obsolete)

<table>
<thead>
<tr>
<th></th>
<th>Network ID</th>
<th>Host ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class A</td>
<td>8</td>
<td>16</td>
</tr>
<tr>
<td>Class B</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Class C</td>
<td>110</td>
<td></td>
</tr>
<tr>
<td>Class D</td>
<td>1110</td>
<td></td>
</tr>
<tr>
<td>Class E</td>
<td>1111</td>
<td></td>
</tr>
</tbody>
</table>

Subnet Addressing RFC917 (1984)

- Class A & B networks too big
  - Very few LANs have close to 64K hosts
  - For electrical/LAN limitations, performance or administrative reasons
- Need simple way to get multiple “networks”
  - Use bridging, multiple IP networks or split up single network address ranges (subnet)
- CMU case study in RFC
  - Chose not to adopt – concern that it would not be widely supported ☹️
Aside: Interaction with Link Layer

- How does one find the Ethernet address of a IP host?
  - ARP (Address Resolution Protocol)
    - Broadcast search for IP address
      - E.g., "who-has 128.2.184.45 tell 128.2.206.138" sent to Ethernet broadcast (all FF address)
      - Destination responds (only to requester using unicast) with appropriate 48-bit Ethernet address
        - E.g, "reply 128.2.184.45 is-at 0:d0:bc:f2:18:58" sent to 0:c0:4f:d:ed:c6

Classless Inter-Domain Routing (CIDR) – RFC1338

- Allows arbitrary split between network & host part of address
  - Do not use classes to determine network ID
  - Use common part of address as network number
  - E.g., addresses 192.4.16 - 192.4.31 have the first 20 bits in common. Thus, we use these 20 bits as the network number \( \rightarrow 192.4.16/20 \)
  - Enables more efficient usage of address space (and router tables) \( \rightarrow \) How?
    - Use single entry for range in forwarding tables
    - Combined forwarding entries when possible

Host Routing Table Example

<table>
<thead>
<tr>
<th>Destination</th>
<th>Gateway</th>
<th>Genmask</th>
<th>Iface</th>
</tr>
</thead>
<tbody>
<tr>
<td>128.2.209.100</td>
<td>0.0.0.0</td>
<td>255.255.255.255</td>
<td>eth0</td>
</tr>
<tr>
<td>128.2.0.0</td>
<td>0.0.0.0</td>
<td>255.255.0.0</td>
<td>eth0</td>
</tr>
<tr>
<td>127.0.0.0</td>
<td>0.0.0.0</td>
<td>255.0.0.0</td>
<td>lo</td>
</tr>
<tr>
<td>0.0.0.0</td>
<td>128.2.254.36</td>
<td>0.0.0.0</td>
<td>eth0</td>
</tr>
</tbody>
</table>

- From "netstat –rn"
- Host 128.2.209.100 when plugged into CS ethernet
- Dest 128.2.209.100 \( \rightarrow \) routing to same machine
- Dest 128.2.0.0 \( \rightarrow \) other hosts on same ethernet
- Dest 127.0.0.0 \( \rightarrow \) special loopback address
- Dest 0.0.0.0 \( \rightarrow \) default route to rest of Internet
- Main CS router: gigrouter.net.cs.cmu.edu (128.2.254.36)

Routing to the Network

- Packet to 10.1.1.3 arrives
- Path is R2 – R1 – H1 – H2
Routing Within the Subnet

- Packet to 10.1.1.3
- Matches 10.1.0.0/23

Routing table at R2

<table>
<thead>
<tr>
<th>Destination</th>
<th>Next Hop</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>lo0</td>
</tr>
<tr>
<td>Default or 0/0</td>
<td>10.1.1.1</td>
<td>10.1.1.2</td>
</tr>
<tr>
<td>10.1.0.24</td>
<td>10.1.8.1</td>
<td>10.1.8.1</td>
</tr>
<tr>
<td>10.1.2.023</td>
<td>10.1.2.1</td>
<td>10.1.2.1</td>
</tr>
<tr>
<td>10.1.0.024</td>
<td>10.1.2.2</td>
<td>10.1.2.2</td>
</tr>
</tbody>
</table>

Routing table at H1

<table>
<thead>
<tr>
<th>Destination</th>
<th>Next Hop</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>lo0</td>
</tr>
<tr>
<td>Default or 0/0</td>
<td>10.1.1.1</td>
<td>10.1.1.2</td>
</tr>
<tr>
<td>10.1.1.024</td>
<td>10.1.1.2</td>
<td>10.1.1.2</td>
</tr>
<tr>
<td>10.1.3.31</td>
<td>10.1.2.1</td>
<td>10.1.2.1</td>
</tr>
</tbody>
</table>

Routing Within the Subnet

- Packet to 10.1.1.3
- Matches 10.1.1.1/31
- Longest prefix match

Routing table at R1

<table>
<thead>
<tr>
<th>Destination</th>
<th>Next Hop</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>lo0</td>
</tr>
<tr>
<td>Default or 0/0</td>
<td>10.1.1.1</td>
<td>10.1.1.2</td>
</tr>
<tr>
<td>10.1.0.24</td>
<td>10.1.8.1</td>
<td>10.1.8.1</td>
</tr>
<tr>
<td>10.1.1.024</td>
<td>10.1.1.1</td>
<td>10.1.1.1</td>
</tr>
<tr>
<td>10.1.2.023</td>
<td>10.1.2.2</td>
<td>10.1.2.2</td>
</tr>
<tr>
<td>10.1.1.231</td>
<td>10.1.1.2</td>
<td>10.1.1.2</td>
</tr>
</tbody>
</table>

Routing Within the Subnet

- Packet to 10.1.1.3
- Direct route
  - Longest prefix match

Routing table at H1

<table>
<thead>
<tr>
<th>Destination</th>
<th>Next Hop</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>lo0</td>
</tr>
<tr>
<td>Default or 0/0</td>
<td>10.1.1.1</td>
<td>10.1.1.2</td>
</tr>
<tr>
<td>10.1.1.024</td>
<td>10.1.1.2</td>
<td>10.1.1.2</td>
</tr>
<tr>
<td>10.1.3.31</td>
<td>10.1.1.2</td>
<td>10.1.1.2</td>
</tr>
</tbody>
</table>

IP Addresses: How to Get One?

Network (network portion):
- Get allocated portion of ISP’s address space:

<table>
<thead>
<tr>
<th>ISP’s block</th>
<th>Organization 0</th>
<th>Organization 1</th>
<th>Organization 2</th>
<th>Organization 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>11001000</td>
<td>11001000</td>
<td>11001000</td>
<td>11001000</td>
<td>11001000</td>
</tr>
<tr>
<td>00010111</td>
<td>00010000</td>
<td>00010111</td>
<td>00010111</td>
<td>00010111</td>
</tr>
<tr>
<td>00001000</td>
<td>00000000</td>
<td>00010010</td>
<td>00010100</td>
<td>00011110</td>
</tr>
<tr>
<td>00000000</td>
<td>200.23.16.0/20</td>
<td>200.23.18.0/23</td>
<td>200.23.20.0/23</td>
<td>200.23.30.0/23</td>
</tr>
</tbody>
</table>
IP Addresses: How to Get One?

- How does an ISP get block of addresses?
  - From Regional Internet Registries (RIRs)
    - ARIN (North America, Southern Africa), APNIC (Asia-Pacific), RIPE (Europe, Northern Africa), LACNIC (South America)
- How about a single host?
  - Hard-coded by system admin in a file
  - DHCP: Dynamic Host Configuration Protocol: dynamically get address: "plug-and-play"
    - Host broadcasts "DHCP discover" msg
    - DHCP server responds with "DHCP offer" msg
    - Host requests IP address: "DHCP request" msg
    - DHCP server sends address: "DHCP ack" msg

IP Service Model

- Low-level communication model provided by Internet
- Datagram
  - Each packet self-contained
    - All information needed to get to destination
    - No advance setup or connection maintenance
  - Analogous to letter or telegram

<table>
<thead>
<tr>
<th>IPv4 Packet Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>version</td>
</tr>
<tr>
<td>Identifier</td>
</tr>
<tr>
<td>TTL</td>
</tr>
<tr>
<td>Source Address</td>
</tr>
<tr>
<td>Options (if any)</td>
</tr>
</tbody>
</table>

IP Fragmentation Example

- Base-level protocol (IP) provides minimal service level
  - Allows highly decentralized implementation
  - Each step involves determining next hop
  - Most of the work at the endpoints
  - ICMP provides low-level error reporting
- IP forwarding → global addressing, alternatives, lookup tables
- IP addressing → hierarchical, CIDR
- IP service → best effort, simplicity of routers
- IP packets → header fields, fragmentation, ICMP
Distance-Vector Routing

**Idea**
- At any time, have cost/next hop of best known path to destination
- Use cost = when no path known

**Initially**
- Only have entries for directly connected nodes

Initial Table for A

<table>
<thead>
<tr>
<th>Dest</th>
<th>Cost</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>A</td>
</tr>
<tr>
<td>B</td>
<td>4</td>
<td>B</td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>2</td>
<td>E</td>
</tr>
<tr>
<td>F</td>
<td>6</td>
<td>F</td>
</tr>
</tbody>
</table>

Distance Vector: Link Cost Changes

- Good news travels fast
- Bad news travels slow - "count to infinity" problem!

Distance Vector: Split Horizon

If Z routes through Y to get to X:
- Z does not advertise its route to X back to Y

Distance Vector Update

- Update(x,y,z)
  \[ d' = c(x,z) + d(z,y) \]
  # Cost of path from x to y with first hop z
  if \( d < d(x,y) \)
  # Found better path
  return \( d',z \) # Updated cost / next hop
  else
  return \( d(x,y) \), nexthop(x,y) # Existing cost / next hop
Distance Vector: Poison Reverse

If Z routes through Y to get to X:
- Z tells Y its (Z's) distance to X is infinite (so Y won't route to X via Z)
- Eliminates some possible timeouts with split horizon
- Will this completely solve count to infinity problem?

![Diagram of Distance Vector: Poison Reverse](image)

Poison Reverse Failures

- X Wants to Send Information
  - Sends on all outgoing links
- When Node B Receives Information from A
  - Send on all links other than A

![Diagram of Poison Reverse Failures](image)

Link State Protocol Concept

- Every node gets complete copy of graph
- Every node “floods” network with data about its outgoing links
- Every node computes routes to every other node
  - Using single-source, shortest-path algorithm
- Process performed whenever needed
- When connections die / reappear

![Diagram of Link State Protocol Concept](image)

Sending Link States by Flooding

- X Wants to Send Information
  - Sends on all outgoing links
- When Node B Receives Information from A
  - Send on all links other than A

![Diagram of Sending Link States by Flooding](image)
Comparison of LS and DV Algorithms

**Message complexity**
- **LS:** with n nodes, E links, \( O(nE) \) messages
- **DV:** exchange between neighbors only \( O(E) \)

**Space requirements:**
- LS maintains entire topology
- DV maintains only neighbor state

**Speed of Convergence**
- **LS:** Complex computation
  - But...can forward before computation
  - may have oscillations
- **DV:** convergence time varies
  - may be routing loops
  - count-to-infinity problem
  - (faster with triggered updates)

Routing Hierarchies

- Flat routing doesn’t scale
  - Storage: Each node cannot be expected to store routes to every destination (or destination network)
  - Convergence times increase
  - Communication: Total message count increases

**Key observation**
- Need less information with increasing distance to destination
- Need lower diameters networks

**Solution:** area hierarchy

Routing Hierarchy

- Area-Border Router
- Backbone Areas
- Lower-level Areas

- Partition Network into “Areas”
  - Within area
    - Each node has routes to every other node
  - Outside area
    - Each node has routes for other top-level areas only
    - Inter-area packets are routed to nearest appropriate border router
- Constraint: no path between two sub-areas of an area can exit that area

Area Hierarchy Addressing

- 1
- 2
- 3
- 1.1
- 1.2
- 2.1
- 2.2
- 3.1
- 3.2
**IP Multicast Control Plane**

- Service model
  - Host-to-router protocol (IGMP)

**Multicast routing protocols (various)**

**IP Multicast Service Model (rfc1112)**

- Each group identified by a single IP address
- Groups may be of any size
- Members of groups may be located anywhere in the Internet
- Members of groups can join and leave at will
- Senders need not be members
- Group membership not known explicitly

**Analogy:**
- Each multicast address is like a radio frequency, on which anyone can transmit, and to which anyone can tune-in.

**How IGMP Works**

- On each link, one router is elected the “querier”
- Querier periodically sends a Membership Query message to the all-systems group (224.0.0.1), with TTL = 1
- On receipt, hosts start random timers (between 0 and 10 seconds) for each multicast group to which they belong

**Multicast Routing Protocols (Part 2 of Control Plane)**

- Basic objective – build distribution tree for multicast packets
- Flood and prune
  - Begin by flooding traffic to entire network
  - Prune branches with no receivers
  - Examples: DVMRP, PIM-DM
  - *Unwanted state where there are no receivers*
- Link-state multicast protocols
  - Routers advertise groups for which they have receivers to entire network
  - Compute trees on demand
  - Example: MOSPF
  - *Unwanted state where there are no senders*
**BGP - Border Gateway Protocol**

- Covered next week

---

**NAT: Opening Client Connection**

- Client 10.2.2.2 wants to connect to server 198.2.4.5:80
  - OS assigns ephemeral port (1000)
  - Connection request intercepted by firewall
    - Maps client to port of firewall (5000)
    - Creates NAT table entry

---

**NAT: Client Request**

- Firewall acts as proxy for client
  - Intercepts message from client and marks itself as sender

---

**NAT: Server Response**

- Firewall acts as proxy for client
  - Acts as destination for server messages
  - Relabels destination to local addresses
Extending Private Network

- Supporting Road Warrior
  - Employee working remotely with assigned IP address 198.3.3.3
  - Wants to appear to rest of corporation as if working internally
    - From address 10.6.6.6
    - Gives access to internal services (e.g., ability to send mail)
- Virtual Private Network (VPN)
  - Overlays private network on top of regular Internet

Supporting VPN by Tunneling

- Concept
  - Appears as if two hosts connected directly
- Usage in VPN
  - Create tunnel between road warrior & firewall
  - Remote host appears to have direct connection to internal network

Implementing Tunneling

- Host creates packet for internal node 10.6.1.1.1
- Entering Tunnel
  - Add extra IP header directed to firewall (243.4.4.4)
  - Original header becomes part of payload
  - Possible to encrypt it
- Exiting Tunnel
  - Firewall receives packet
  - Strips off header
  - Sends through internal network to destination

Virtual Circuit IDs/Switching: Label (“tag”) Swapping

- Global VC ID allocation -- ICK! Solution: Per-link uniqueness. Change VCI each hop.

<table>
<thead>
<tr>
<th>Input Port</th>
<th>Input VCI</th>
<th>Output Port</th>
<th>Output VCI</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1:</td>
<td>1</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>R2:</td>
<td>2</td>
<td>9</td>
<td>4</td>
</tr>
<tr>
<td>R4:</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>
Simplified Virtual Circuits Example

Source Routing Example

Global Address Example

Comparison

<table>
<thead>
<tr>
<th></th>
<th>Source Routing</th>
<th>Global Addresses</th>
<th>Virtual Circuits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header Size</td>
<td>Worst</td>
<td>OK – Large address</td>
<td>Best</td>
</tr>
<tr>
<td>Router Table Size</td>
<td>None</td>
<td>Number of hosts (prefixes)</td>
<td>Number of circuits</td>
</tr>
<tr>
<td>Forward Overhead</td>
<td>Best</td>
<td>Prefix matching (Worst)</td>
<td>Pretty Good</td>
</tr>
<tr>
<td>Setup Overhead</td>
<td>None</td>
<td>None</td>
<td>Connection Setup</td>
</tr>
<tr>
<td>Error Recovery</td>
<td>Tell all hosts</td>
<td>Tell all routers</td>
<td>Tell routers and Tear down circuit and re-route</td>
</tr>
</tbody>
</table>
**MPLS core, IP interface**

![MPLS network diagram]  

**Take Home Points**

- Costs/benefits/goals of virtual circuits
- Cell switching (ATM)
  - Fixed-size pkts: Fast hardware
  - Packet size picked for low voice jitter. Understand trade-offs.
  - Beware packet shredder effect (drop entire pkt)
- Tag/label swapping
  - Basis for most VCs.
  - Makes label assignment link-local. Understand mechanism.
- MPLS - IP meets virtual circuits
  - MPLS tunnels used for VPNs, traffic engineering, reduced core routing table sizes

**Outline**

- Switching and Multiplexing
- Link-Layer
- Routing-Layer
- Physical-Layer
  - Encodings

**From Signals to Packets**

- Analog Signal
- "Digital" Signal
- Bit Stream
- Packets
- Packet Transmission
Encoding

- We use two discrete signals, high and low, to encode 0 and 1
- The transmission is synchronous, i.e., there is a clock used to sample the signal
  - In general, the duration of one bit is equal to one or two clock ticks

Non-Return to Zero (NRZ)

- 1 \rightarrow high signal; 0 \rightarrow low signal
- Long sequences of 1’s or 0’s can cause problems:
  - Sensitive to clock skew, i.e. hard to recover clock
  - Difficult to interpret 0’s and 1’s

Non-Return to Zero Inverted (NRZI)

- 1 \rightarrow make transition; 0 \rightarrow signal stays the same
- Solves the problem for long sequences of 1’s, but not for 0’s.

Ethernet Manchester Encoding

- Positive transition for 0, negative for 1
- Transition every cycle communicates clock (but need 2 transition times per bit)
- DC balance has good electrical properties
4B/5B Encoding

- Data coded as symbols of 5 line bits → 4 data bits, so 100 Mbps uses 125 MHz.
  - Uses less frequency space than Manchester encoding
  - Uses NRI to encode the 5 code bits
  - Each valid symbol has at least two 1s: get dense transitions.
- 16 data symbols, 8 control symbols
  - Data symbols: 4 data bits
  - Control symbols: idle, begin frame, etc.
- Example: FDDI.

Framing

- A link layer function, defining which bits have which function.
- Minimal functionality: mark the beginning and end of packets (or frames).
- Some techniques:
  - out of band delimiters (e.g. FDDI 4B/5B control symbols)
  - frame delimiter characters with character stuffing
  - frame delimiter codes with bit stuffing
  - synchronous transmission (e.g. SONET)

Dealing with Errors

Stop and Wait Case

- Packets can get lost, corrupted, or duplicated.
  - Error detection or correction turns corrupted packet in lost or correct packet
  - Duplicate packet: use sequence numbers.
  - Lost packet: time outs and acknowledgements.
    - Positive versus negative acknowledgements
    - Sender side versus receiver side timeouts
  - Window based flow control: more aggressive use of sequence numbers (see transport lectures).