Chapter 5: The Data Link Layer

Our goals:

- understand principles behind data link layer services:
  - error detection, correction
  - sharing a broadcast channel: multiple access
  - link layer addressing
  - reliable data transfer, flow control: done!

- instantiation and implementation of various link layer technologies
Link Layer

- 5.1 Introduction and services
- 5.2 Error detection and correction
- 5.3 Multiple access protocols
- 5.4 Link-Layer Addressing
- 5.5 Ethernet

- 5.6 Hubs and switches
- 5.7 PPP
- 5.8 Link Virtualization: ATM and MPLS
Link Layer: Introduction

Some terminology:

- hosts and routers are **nodes**
- communication channels that connect adjacent nodes along communication path are **links**
  - wired links
  - wireless links
  - LANs
- layer-2 packet is a **frame**, encapsulates datagram

**data-link layer** has responsibility of transferring datagram from one node to adjacent node over a link
Link layer: context

- Datagram transferred by different link protocols over different links:
  - e.g., Ethernet on first link, frame relay on intermediate links, 802.11 on last link

- Each link protocol provides different services
  - e.g., may or may not provide rdt over link

Transportation analogy

- trip from Princeton to Lausanne
  - limo: Princeton to JFK
  - plane: JFK to Geneva
  - train: Geneva to Lausanne

- tourist = datagram
- transport segment = communication link
- transportation access = link layer protocol
- travel agent = routing algorithm
Link Layer Services

- Framing, link access:
  - encapsulate datagram into frame, adding header, trailer
  - channel access if shared medium
  - “MAC” addresses used in frame headers to identify source, dest
    - different from IP address!

- Reliable delivery between adjacent nodes
  - we learned how to do this already (chapter 3)!
  - seldom used on low bit error link (fiber, some twisted pair)
  - wireless links: high error rates
    - Q: why both link-level and end-end reliability?
Link Layer Services (more)

- **Flow Control:**
  - pacing between adjacent sending and receiving nodes

- **Error Detection:**
  - errors caused by signal attenuation, noise.
  - receiver detects presence of errors:
    - signals sender for retransmission or drops frame

- **Error Correction:**
  - receiver identifies *and corrects* bit error(s) without resorting to retransmission

- **Half-duplex and full-duplex**
  - with half duplex, nodes at both ends of link can transmit, but not at same time
Adaptors Communicating

- Link layer implemented in “adapter” (aka NIC)
  - Ethernet card, PCMCI card, 802.11 card

- Sending side:
  - Encapsulates datagram in a frame
  - Adds error checking bits, rdt, flow control, etc.

- Receiving side
  - Looks for errors, rdt, flow control, etc
  - Extracts datagram, passes to rcving node

- Adapter is semi-autonomous

- Link & physical layers
Link Layer

- 5.1 Introduction and services
- 5.2 Error detection and correction
- 5.3 Multiple access protocols
- 5.4 Link-Layer Addressing
- 5.5 Ethernet
- 5.6 Hubs and switches
- 5.7 PPP
- 5.8 Link Virtualization: ATM
Error Detection

EDC = Error Detection and Correction bits (redundancy)
D   = Data protected by error checking, may include header fields

• Error detection not 100% reliable!
  • protocol may miss some errors, but rarely
  • larger EDC field yields better detection and correction
Parity Checking

**Single Bit Parity:**
Detect single bit errors

```
0111000110101011 0
```

**Two Dimensional Bit Parity:**
Detect and correct single bit errors

```
\[
\begin{array}{cccc}
  d_{1,1} & \cdots & d_{1,j} \\
  d_{2,1} & \cdots & d_{2,j} \\
  \vdots & \ddots & \vdots \\
  d_{i,1} & \cdots & d_{i,j} \\
  d_{i+1,1} & \cdots & d_{i+1,j} \\
\end{array}
\]
```

**Examples:**

- **No errors**
  - `101011` (column parity)
  - `111100` (row parity)
  - `011101` (column parity)
  - `001010` (row parity)

- **Correctable single bit error**
  - Change `1` to `0` in the first row to correct the error.
Internet checksum

**Goal:** detect “errors” (e.g., flipped bits) in transmitted segment (note: used at transport layer only)

**Sender:**
- treat segment contents as sequence of 16-bit integers
- checksum: addition (1’s complement sum) of segment contents
- sender puts checksum value into UDP checksum field

**Receiver:**
- compute checksum of received segment
- check if computed checksum equals checksum field value:
  - NO - error detected
  - YES - no error detected. *But maybe errors nonetheless? More later ....*
Checksumming: Cyclic Redundancy Check

- view data bits, \( D \), as a binary number
- choose \( r+1 \) bit pattern (generator), \( G \)
- goal: choose \( r \) CRC bits, \( R \), such that
  - \( <D,R> \) exactly divisible by \( G \) (modulo 2)
  - receiver knows \( G \), divides \( <D,R> \) by \( G \). If non-zero remainder: error detected!
  - can detect all burst errors less than \( r+1 \) bits
- widely used in practice

\[
D \cdot 2^r \op{XOR} R
\]

bit pattern

mathematical formula
**CRC Example**

Want:

\[ D \cdot 2^r \text{ XOR } R = nG \]

equivalently:

\[ D \cdot 2^r = nG \text{ XOR } R \]

equivalently:

if we divide \( D \cdot 2^r \) by \( G \), want remainder \( R \)

\[
R = \text{remainder}
\left[ \frac{D \cdot 2^r}{G} \right]
\]
Link Layer

- 5.1 Introduction and services
- 5.2 Error detection and correction
- 5.3 Multiple access protocols
- 5.4 Link-Layer Addressing
- 5.5 Ethernet

- 5.6 Hubs and switches
- 5.7 PPP
- 5.8 Link Virtualization: ATM
Multiple Access Links and Protocols

Two types of “links”:

- **point-to-point**
  - PPP for dial-up access
  - point-to-point link between Ethernet switch and host

- **broadcast** (shared wire or medium)
  - traditional Ethernet
  - Bluetooth
  - 802.11 wireless LAN
Multiple Access protocols

- single shared broadcast channel
- two or more simultaneous transmissions by nodes: interference
  - collision if node receives two or more signals at the same time

multiple access protocol

- distributed algorithm that determines how nodes share channel, i.e., determine when node can transmit
- communication about channel sharing must use channel itself!
  - no out-of-band channel for coordination
Ideal Multiple Access Protocol

Broadcast channel of rate $R$ bps

1. When one node wants to transmit, it can send at rate $R$.
2. When $M$ nodes want to transmit, each can send at average rate $R/M$
3. Fully decentralized:
   - no special node to coordinate transmissions
   - no synchronization of clocks, slots
4. Simple
MAC Protocols: a taxonomy

Three broad classes:

- **Channel Partitioning**
  - divide channel into smaller “pieces” (time slots, frequency, code)
  - allocate piece to node for exclusive use

- **Random Access**
  - channel not divided, allow collisions
  - “recover” from collisions

- **“Taking turns”**
  - Nodes take turns, but nodes with more to send can take longer turns
Channel Partitioning MAC protocols: TDMA

TDMA: time division multiple access
- access to channel in "rounds"
- each station gets fixed length slot (length = pkt trans time) in each round
- unused slots go idle
- example: 6-station LAN, 1,3,4 have pkt, slots 2,5,6 idle
Channel Partitioning MAC protocols: FDMA

FDMA: frequency division multiple access
- channel spectrum divided into frequency bands
- each station assigned fixed frequency band
- unused transmission time in frequency bands go idle
- example: 6-station LAN, 1,3,4 have pkt, frequency bands 2,5,6 idle

![Diagram of FDMA]

- TDM (Time Division Multiplexing): channel divided into N time slots, one per user; inefficient with low duty cycle users and at light load.
- FDM (Frequency Division Multiplexing): frequency subdivided.
Random Access Protocols

- When node has packet to send
  - transmit at full channel data rate \( R \).
  - no \textit{a priori} coordination among nodes
- two or more transmitting nodes \(\rightarrow\) “collision”,

random access MAC protocol specifies:
  - how to detect collisions
  - how to recover from collisions (e.g., via delayed retransmissions)

Examples of random access MAC protocols:
  - slotted ALOHA
  - ALOHA
  - \textit{CSMA, CSMA/CD, CSMA/CA}
Slotted ALOHA

**Assumptions**
- all frames same size
- time is divided into equal size slots, time to transmit 1 frame
- nodes start to transmit frames only at beginning of slots
- nodes are synchronized
- if 2 or more nodes transmit in slot, all nodes detect collision

**Operation**
- when node obtains fresh frame, it transmits in next slot
- no collision, node can send new frame in next slot
- if collision, node retransmits frame in each subsequent slot with prob. p until success
Slotted ALOHA

**Pros**
- single active node can continuously transmit at full rate of channel
- highly decentralized: only slots in nodes need to be in sync
- simple

**Cons**
- collisions, wasting slots
- idle slots
- nodes must be able to detect collision in less than time to transmit packet
- clock synchronization
**Slotted Aloha efficiency**

*Efficiency* is the long-run fraction of successful slots when there are many nodes, each with many frames to send.

- Suppose $N$ nodes with many frames to send, each transmits in slot with probability $p$.
- Prob that node 1 has success in a slot = $p(1-p)^{N-1}$.
- Prob that any node has a success = $Np(1-p)^{N-1}$.

- For max efficiency with $N$ nodes, find $p^*$ that maximizes $Np(1-p)^{N-1}$.
- For many nodes, take limit of $Np^*(1-p^*)^{N-1}$ as $N$ goes to infinity, gives $1/e = .37$.

*At best:* channel used for useful transmissions 37% of time!
Pure (unslotted) ALOHA

- unslotted Aloha: simpler, no synchronization
- when frame first arrives
  - transmit immediately
- collision probability increases:
  - frame sent at $t_0$ collides with other frames sent in $[t_0-1,t_0+1]$
**Pure Aloha efficiency**

\[
P(\text{success by given node}) = P(\text{node transmits}) \cdot \\
P(\text{no other node transmits in } [t_0-1,t_0]) \cdot \\
P(\text{no other node transmits in } [t_0,t_0+1]) \\
= p \cdot (1-p)^{N-1} \cdot (1-p)^{N-1} \\
= p \cdot (1-p)^{2(N-1)}
\]

... choosing optimum p and then letting n \(\to\) infty ...

\[= 1/(2e) = .18\]

**Even worse!**
CSMA (Carrier Sense Multiple Access)

**CSMA**: listen before transmit:
- If channel sensed idle: transmit entire frame
- If channel sensed busy, defer transmission

- Human analogy: don’t interrupt others!
CSMA collisions

collisions can still occur:
propagation delay means two nodes may not hear each other’s transmission

collision:
entire packet transmission time wasted

note:
role of distance & propagation delay in determining collision probability
CSMA/CD (Collision Detection)

**CSMA/CD**: carrier sensing, deferral as in CSMA
  - collisions detected within short time
  - colliding transmissions aborted, reducing channel wastage

- collision detection:
  - easy in wired LANs: measure signal strengths, compare transmitted, received signals
  - difficult in wireless LANs: receiver shut off while transmitting

- human analogy: the polite conversationalist
CSMA/CD collision detection
“Taking Turns” MAC protocols

channel partitioning MAC protocols:
- share channel efficiently and fairly at high load
- inefficient at low load: delay in channel access, 1/N bandwidth allocated even if only 1 active node!

Random access MAC protocols
- efficient at low load: single node can fully utilize channel
- high load: collision overhead

“taking turns” protocols
look for best of both worlds!
“Taking Turns” MAC protocols

Polling:
- master node “invites” slave nodes to transmit in turn
- concerns:
  - polling overhead
  - latency
  - single point of failure (master)

Token passing:
- control token passed from one node to next sequentially.
- token message
- concerns:
  - token overhead
  - latency
  - single point of failure (token)
Summary of MAC protocols

- What do you do with a shared media?
  - Channel Partitioning, by time, frequency or code
    - Time Division, Frequency Division
  - Random partitioning (dynamic),
    - ALOHA, S-ALOHA, CSMA, CSMA/CD
    - carrier sensing: easy in some technologies (wire), hard in others (wireless)
    - CSMA/CD used in Ethernet
    - CSMA/CA used in 802.11
  - Taking Turns
    - polling from a central site, token passing
LAN technologies

Data link layer so far:
  • services, error detection/correction, multiple access

Next: LAN technologies
  • addressing
  • Ethernet
  • hubs, switches
  • PPP
Link Layer

- 5.1 Introduction and services
- 5.2 Error detection and correction
- 5.3 Multiple access protocols
- 5.4 Link-Layer Addressing
- 5.5 Ethernet
- 5.6 Hubs and switches
- 5.7 PPP
- 5.8 Link Virtualization: ATM
MAC Addresses and ARP

- **32-bit IP address:**
  - *network-layer address*
  - used to get datagram to destination IP subnet

- **MAC (or LAN or physical or Ethernet) address:**
  - used to get frame from one interface to another physically-connected interface (same network)
  - 48 bit MAC address (for most LANs) burned in the adapter ROM
LAN Addresses and ARP

Each adapter on LAN has unique LAN address.

Broadcast address = FF-FF-FF-FF-FF-FF

LAN (wired or wireless)
LAN Address (more)

- MAC address allocation administered by IEEE
- manufacturer buys portion of MAC address space (to assure uniqueness)
- Analogy:
  (a) MAC address: like Social Security Number
  (b) IP address: like postal address
- MAC flat address → portability
  - can move LAN card from one LAN to another
- IP hierarchical address NOT portable
  - depends on IP subnet to which node is attached
ARP: Address Resolution Protocol

Question: how to determine MAC address of B knowing B’s IP address?

- Each IP node (Host, Router) on LAN has ARP table
- ARP Table: IP/MAC address mappings for some LAN nodes
  - IP address; MAC address; TTL
- TTL (Time To Live): time after which address mapping will be forgotten (typically 20 min)
**ARP protocol: Same LAN (network)**

- A wants to send datagram to B, and B’s MAC address not in A’s ARP table.
- A broadcasts ARP query packet, containing B’s IP address
  - Dest MAC address = FF-FF-FF-FF-FF-FF
  - all machines on LAN receive ARP query
- B receives ARP packet, replies to A with its (B’s) MAC address
  - frame sent to A’s MAC address (unicast)
- A caches (saves) IP-to-MAC address pair in its ARP table until information becomes old (times out)
  - soft state: information that times out (goes away) unless refreshed
- ARP is “plug-and-play”:
  - nodes create their ARP tables without intervention from net administrator
Routing to another LAN

walkthrough: send datagram from A to B via R
assume A know’s B IP address

- Two ARP tables in router R, one for each IP network (LAN)
- A creates datagram with source A, destination B
- A uses ARP to get R’s MAC address for 111.111.111.110
- A creates link-layer frame with R’s MAC address as dest, frame contains A-to-B IP datagram
- A’s adapter sends frame
- R’s adapter receives frame
- R removes IP datagram from Ethernet frame, sees its destined to B
- R uses ARP to get B’s MAC address
- R creates frame containing A-to-B IP datagram sends to B
Link Layer

- 5.1 Introduction and services
- 5.2 Error detection and correction
- 5.3 Multiple access protocols
- 5.4 Link-Layer Addressing
- 5.5 Ethernet

- 5.6 Hubs and switches
- 5.7 PPP
- 5.8 Link Virtualization: ATM
Ethernet

“dominant” wired LAN technology:
- cheap $20 for 100Mbs!
- first widely used LAN technology
- Simpler, cheaper than token LANs and ATM
- Kept up with speed race: 10 Mbps – 10 Gbps
Star topology

- Bus topology popular through mid 90s
- Now star topology prevails
- Connection choices: hub or switch (more later)
Ethernet Frame Structure

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

Preamble:
- 7 bytes with pattern 10101010 followed by one byte with pattern 10101011
- used to synchronize receiver, sender clock rates
Ethernet Frame Structure (more)

- **Addresses**: 6 bytes
  - if adapter receives frame with matching destination address, or with broadcast address (e.g., ARP packet), it passes data in frame to net-layer protocol
  - otherwise, adapter discards frame

- **Type**: indicates the higher layer protocol (mostly IP but others may be supported such as Novell IPX and AppleTalk)

- **CRC**: checked at receiver, if error is detected, the frame is simply dropped
Unreliable, connectionless service

- **Connectionless**: No handshaking between sending and receiving adapter.

- **Unreliable**: receiving adapter doesn’t send acks or nacks to sending adapter
  - stream of datagrams passed to network layer can have gaps
  - gaps will be filled if app is using TCP
  - otherwise, app will see the gaps
Ethernet uses CSMA/CD

- No slots
- Adapter doesn’t transmit if it senses that some other adapter is transmitting, that is, carrier sense
- Transmitting adapter aborts when it senses that another adapter is transmitting, that is, collision detection
- Before attempting a retransmission, adapter waits a random time, that is, random access
Ethernet CSMA/CD algorithm

1. Adaptor receives datagram from net layer & creates frame
2. If adapter senses channel idle, it starts to transmit frame. If it senses channel busy, waits until channel idle and then transmits
3. If adapter transmits entire frame without detecting another transmission, the adapter is done with frame!
4. If adapter detects another transmission while transmitting, aborts and sends jam signal
5. After aborting, adapter enters exponential backoff: after the mth collision, adapter chooses a K at random from \{0,1,2,...,2^m-1\}. Adapter waits K·512 bit times and returns to Step 2
Ethernet’s CSMA/CD (more)

**Jam Signal:** make sure all other transmitters are aware of collision; 48 bits

**Bit time:** .1 microsec for 10 Mbps Ethernet; for K=1023, wait time is about 50 msec

---

**Exponential Backoff:**

- **Goal:** adapt retransmission attempts to estimated current load
  - heavy load: random wait will be longer
- first collision: choose K from \{0,1\}; delay is K· 512 bit transmission times
- after second collision: choose K from \{0,1,2,3\}...
- after ten collisions, choose K from \{0,1,2,3,4,...,1023\}

---

See/interact with Java applet on AWL Web site: highly recommended!
CSMA/CD efficiency

- $T_{\text{prop}} = \text{max prop between 2 nodes in LAN}$
- $t_{\text{trans}} = \text{time to transmit max-size frame}$

\[
\text{efficiency} = \frac{1}{1 + 5\frac{T_{\text{prop}}}{t_{\text{trans}}}}
\]

- Efficiency goes to 1 as $T_{\text{prop}}$ goes to 0
- Goes to 1 as $t_{\text{trans}}$ goes to infinity
- Much better than ALOHA, but still decentralized, simple, and cheap
Question

Is it possible that:

A collision happens in Ethernet
But is not detected at the MAC layer

Remember: CSMA/CD does not use MAC layer ACKs
10BaseT and 100BaseT

- 10/100 Mbps rate; latter called “fast ethernet”
- T stands for Twisted Pair
- Nodes connect to a hub: “star topology”; 100 m max distance between nodes and hub

![Diagram of 10BaseT and 100BaseT network](image-url)
Hubs

Hubs are essentially physical-layer repeaters:
- bits coming from one link go out all other links
- at the same rate
- no frame buffering
- no CSMA/CD at hub: adapters detect collisions
- provides net management functionality
Gbit Ethernet

- uses standard Ethernet frame format
- allows for point-to-point links and shared broadcast channels
- in shared mode, CSMA/CD is used; short distances between nodes required for efficiency
- uses hubs, called here “Buffered Distributors”
- Full-Duplex at 1 Gbps for point-to-point links
- 10 Gbps now!
Link Layer

- 5.1 Introduction and services
- 5.2 Error detection and correction
- 5.3 Multiple access protocols
- 5.4 Link-Layer Addressing
- 5.5 Ethernet
- 5.6 Interconnections: Hubs and switches
- 5.7 PPP
- 5.8 Link Virtualization: ATM
Interconnecting with hubs

- Backbone hub interconnects LAN segments
- Extends max distance between nodes
- But individual segment collision domains become one large collision domain
- Can't interconnect 10BaseT & 100BaseT
Switch

- Link layer device
  - stores and forwards Ethernet frames
  - examines frame header and selectively forwards frame based on MAC dest address
  - when frame is to be forwarded on segment, uses CSMA/CD to access segment

- transparent
  - hosts are unaware of presence of switches

- plug-and-play, self-learning
  - switches do not need to be configured
Forwarding

- How do determine onto which LAN segment to forward frame?
- Looks like a routing problem...
Self learning

- A switch has a switch table
- entry in switch table:
  - (MAC Address, Interface, Time Stamp)
  - stale entries in table dropped (TTL can be 60 min)

- switch learns which hosts can be reached through which interfaces
  - when frame received, switch “learns” location of sender: incoming LAN segment
  - records sender/location pair in switch table
Filtering/Forwarding

When switch receives a frame:

index switch table using MAC dest address
if entry found for destination
    then{
        if dest on segment from which frame arrived
            then drop the frame
        else forward the frame on interface indicated
    }
else flood

forward on all but the interface on which the frame arrived
Switch example

Suppose C sends frame to D

Switch receives frame from C
- notes in bridge table that C is on interface 1
- because D is not in table, switch forwards frame into interfaces 2 and 3

Frame received by D
Switch example

Suppose D replies back with frame to C.

- Switch receives frame from D
  - notes in bridge table that D is on interface 2
  - because C is in table, switch forwards frame only to interface 1

- Frame received by C
Switch: traffic isolation

- switch installation breaks subnet into LAN segments
- switch **filters** packets:
  - same-LAN-segment frames not usually forwarded onto other LAN segments
  - segments become separate collision domains
Switches: dedicated access

- Switch with many interfaces
- Hosts have direct connection to switch
- No collisions; full duplex

Switching: A-to-A' and B-to-B' simultaneously, no collisions
More on Switches

- **cut-through switching**: frame forwarded from input to output port without first collecting entire frame
  - slight reduction in latency

- combinations of shared/dedicated, 10/100/1000 Mbps interfaces
So ...

What’s the difference between switches and routers?
Switches vs. Routers

- both store-and-forward devices
  - routers: network layer devices (examine network layer headers)
  - switches are link layer devices
- routers maintain routing tables, implement routing algorithms
- switches maintain switch tables, implement filtering, learning algorithms

![Diagram of network devices: Host, Bridge, Router, Host]
### Summary comparison

<table>
<thead>
<tr>
<th></th>
<th>hubs</th>
<th>routers</th>
<th>switches</th>
</tr>
</thead>
<tbody>
<tr>
<td>traffic isolation</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>plug &amp; play</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>optimal routing</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>cut through</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
</tbody>
</table>
Link Layer

- 5.1 Introduction and services
- 5.2 Error detection and correction
- 5.3 Multiple access protocols
- 5.4 Link-Layer Addressing
- 5.5 Ethernet
- 5.6 Hubs and switches
- 5.7 PPP
- 5.8 Link Virtualization: ATM
Point to Point Data Link Control

- one sender, one receiver, one link: easier than broadcast link:
  - no Media Access Control
  - no need for explicit MAC addressing
  - e.g., dialup link, ISDN line

- popular point-to-point DLC protocols:
  - PPP (point-to-point protocol)
  - HDLC: High level data link control (Data link used to be considered “high layer” in protocol stack!)
PPP Design Requirements [RFC 1557]

- **packet framing**: encapsulation of network-layer datagram in data link frame
  - carry network layer data of any network layer protocol (not just IP) *at same time*
  - ability to demultiplex upwards

- **bit transparency**: must carry any bit pattern in the data field

- **error detection** (no correction)

- **connection liveness**: detect, signal link failure to network layer

- **network layer address negotiation**: endpoint can learn/configure each other’s network address
PPP non-requirements

- no error correction/recovery
- no flow control
- out of order delivery OK
- no need to support multipoint links (e.g., polling)

Error recovery, flow control, data re-ordering all relegated to higher layers!
PPP Data Frame

- **Flag:** delimiter (framing)
- **Address:** does nothing (only one option)
- **Control:** does nothing; in the future possible multiple control fields
- **Protocol:** upper layer protocol to which frame delivered (eg, PPP-LCP, IP, IPCP, etc)
### PPP Data Frame

- **info:** upper layer data being carried
- **check:** cyclic redundancy check for error detection

<table>
<thead>
<tr>
<th>Flag</th>
<th>Address</th>
<th>Control</th>
<th>Protocol</th>
<th>Info</th>
<th>Check</th>
<th>Variable Length</th>
<th>Flag</th>
</tr>
</thead>
<tbody>
<tr>
<td>01111110</td>
<td>11111111</td>
<td>00000011</td>
<td>protocol</td>
<td>info</td>
<td>check</td>
<td>01111110</td>
<td>flag</td>
</tr>
</tbody>
</table>

1 1 1 1 1 or 2 variable length 2 or 4 1
Byte Stuffing

- "data transparency" requirement: data field must be allowed to include flag pattern <01111110>
  - Q: is received <01111110> data or flag?

- Sender: adds ("stuffs") extra < 01111110> byte after each < 01111110> data byte

- Receiver:
  - two 01111110 bytes in a row: discard first byte, continue data reception
  - single 01111110: flag byte
Byte Stuffing

Flag byte pattern in data to send

Flag byte pattern plus stuffed byte in transmitted data
PPP Data Control Protocol

Before exchanging network-layer data, data link peers must

- configure PPP link (max. frame length, authentication)
- learn/configure network layer information
  - for IP: carry IP Control Protocol (IPCP) msgs (protocol field: 8021) to configure/learn IP address
Link Layer

- 5.1 Introduction and services
- 5.2 Error detection and correction
- 5.3 Multiple access protocols
- 5.4 Link-Layer Addressing
- 5.5 Ethernet
- 5.6 Hubs and switches
- 5.7 PPP
- 5.8 Link Virtualization: ATM and MPLS
Virtualization of resources: a powerful abstraction in systems engineering:

- computing examples: virtual memory, virtual devices
  - Virtual machines: e.g., java
  - IBM VM os from 1960’s/70’s
- layering of abstractions: don’t sweat the details of the lower layer, only deal with lower layers abstractly
The Internet: virtualizing networks

1974: multiple unconnected nets
- ARPAnet
- data-over-cable networks
- packet satellite network (Aloha)
- packet radio network

... differing in:
- addressing conventions
- packet formats
- error recovery
- routing

"A Protocol for Packet Network Intercommunication",
V. Cerf, R. Kahn, IEEE Transactions on Communications,

5: DataLink Layer 5-83
The Internet: virtualizing networks

Internetwork layer (IP):
- addressing: internetwork appears as a single, uniform entity, despite underlying local network heterogeneity
- network of networks

Gateway:
- “embed internetwork packets in local packet format or extract them”
- route (at internetwork level) to next gateway
Cerf & Kahn’s Internetwork Architecture

What is virtualized?

- two layers of addressing: internetwork and local network
- new layer (IP) makes everything homogeneous at internetwork layer
- underlying local network technology
  - cable
  - satellite
  - 56K telephone modem
  - today: ATM, MPLS

... “invisible” at internetwork layer. Looks like a link layer technology to IP!
ATM and MPLS

- ATM, MPLS separate networks in their own right
  - different service models, addressing, routing from Internet
- viewed by Internet as logical link connecting IP routers
  - just like dialup link is really part of separate network (telephone network)
- ATM, MPLS: of technical interest in their own right
Asynchronous Transfer Mode: ATM

- 1990's/00 standard for high-speed (155Mbps to 622 Mbps and higher) Broadband Integrated Service Digital Network architecture
- **Goal**: integrated, end-end transport of carry voice, video, data
  - meeting timing/QoS requirements of voice, video (versus Internet best-effort model)
  - “next generation” telephony: technical roots in telephone world
  - packet-switching (fixed length packets, called “cells”) using virtual circuits
ATM architecture

- **adaptation layer**: only at edge of ATM network
  - data segmentation/reassembly
  - roughly analogous to Internet transport layer
- **ATM layer**: “network” layer
  - cell switching, routing
- **physical layer**
**ATM: network or link layer?**

**Vision:** end-to-end transport: “ATM from desktop to desktop”
- ATM is a network technology

**Reality:** used to connect IP backbone routers
- “IP over ATM”
- ATM as switched link layer, connecting IP routers
ATM Adaptation Layer (AAL)

- ATM Adaptation Layer (AAL): “adapts” upper layers (IP or native ATM applications) to ATM layer below
- AAL present only in end systems, not in switches
- AAL layer segment (header/trailer fields, data) fragmented across multiple ATM cells
  - analogy: TCP segment in many IP packets
ATM Adaptation Layer (AAL) [more]

Different versions of AAL layers, depending on ATM service class:

- **AAL1**: for CBR (Constant Bit Rate) services, e.g. circuit emulation
- **AAL2**: for VBR (Variable Bit Rate) services, e.g., MPEG video
- **AAL5**: for data (e.g., IP datagrams)
## ATM Layer

**Service:** transport cells across ATM network
- analogous to IP network layer
- very different services than IP network layer

<table>
<thead>
<tr>
<th>Network Architecture</th>
<th>Service Model</th>
<th>Guarantees?</th>
<th>Congestion feedback</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Bandwidth</td>
<td>Loss</td>
</tr>
<tr>
<td>Internet</td>
<td>best effort</td>
<td>none</td>
<td>no</td>
</tr>
<tr>
<td>ATM</td>
<td>CBR</td>
<td>constant rate</td>
<td>yes</td>
</tr>
<tr>
<td>ATM</td>
<td>VBR</td>
<td>guaranteed rate</td>
<td>yes</td>
</tr>
<tr>
<td>ATM</td>
<td>ABR</td>
<td>guaranteed minimum</td>
<td>no</td>
</tr>
<tr>
<td>ATM</td>
<td>UBR</td>
<td>none</td>
<td>no</td>
</tr>
</tbody>
</table>
ATM Layer: Virtual Circuits

- **VC transport**: cells carried on VC from source to dest
  - call setup, teardown for each call *before* data can flow
  - each packet carries VC identifier (not destination ID)
  - *every* switch on source-dest path maintain “state” for each passing connection
  - link, switch resources (bandwidth, buffers) may be *allocated* to VC: to get circuit-like perf.

- **Permanent VCs (PVCs)**
  - long lasting connections
  - typically: “permanent” route between two IP routers

- **Switched VCs (SVC)**:
  - dynamically set up on per-call basis
ATM VCs

- Advantages of ATM VC approach:
  - QoS performance guarantee for connection mapped to VC (bandwidth, delay, delay jitter)

- Drawbacks of ATM VC approach:
  - Inefficient support of datagram traffic
  - SVC introduces call setup latency, processing overhead for short lived connections
**ATM Layer: ATM cell**

- 5-byte ATM cell header
- 48-byte payload
  - **Why?** small payload -> short cell-creation delay for digitized voice
  - halfway between 32 and 64 (compromise!)

**Cell header**

```
+-----------------+-----------------+-----------------+
| VCI             | PT   | CLP | HEC |
|                 |      |     |     |
+-----------------+----------------+-----------------+
```

**Cell format**

```
+--------------------------+--------------------------+
| Cell Header              | ATM Cell Payload - 48 bytes |
| 3rd bit in PT field; 1   | SAR PDU                  |
| indicates last cell      | (AAL-Indicate bit)       |
+--------------------------+--------------------------+
```
**ATM cell header**

- **VCI**: virtual channel ID
  - will change from link to link thru net
- **PT**: Payload type (e.g. RM cell versus data cell)
- **CLP**: Cell Loss Priority bit
  - CLP = 1 implies low priority cell, can be discarded if congestion
- **HEC**: Header Error Checksum
  - cyclic redundancy check

<table>
<thead>
<tr>
<th>VCI</th>
<th>PT</th>
<th>CLP</th>
<th>HEC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

40 bits
**IP-Over-ATM**

**Classic IP only**
- 3 “networks” (e.g., LAN segments)
- MAC (802.3) and IP addresses

**IP over ATM**
- replace “network” (e.g., LAN segment) with ATM network
- ATM addresses, IP addresses
IP-Over-ATM
 Datagram Journey in IP-over-ATM Network

- **at Source Host:**
  - IP layer maps between IP, ATM dest address (using ARP)
  - passes datagram to AAL5
  - AAL5 encapsulates data, segments cells, passes to ATM layer

- **ATM network:** moves cell along VC to destination

- **at Destination Host:**
  - AAL5 reassembles cells into original datagram
  - if CRC OK, datagram is passed to IP
**IP-Over-ATM**

**Issues:**
- IP datagrams into ATM AAL5 PDUs
- from IP addresses to ATM addresses
  - just like IP addresses to 802.3 MAC addresses!
Multiprotocol label switching (MPLS)

- initial goal: speed up IP forwarding by using fixed length label (instead of IP address) to do forwarding
  - borrowing ideas from Virtual Circuit (VC) approach
  - but IP datagram still keeps IP address!

Diagram:
- PPP or Ethernet header
- MPLS header
- IP header
- remainder of link-layer frame

- label
- Exp
- S
- TTL
- 20
- 3
- 1
- 5
MPLS capable routers

- a.k.a. label-switched router
- forwards packets to outgoing interface based only on label value (don’t inspect IP address)
  - MPLS forwarding table distinct from IP forwarding tables
- signaling protocol needed to set up forwarding
  - RSVP-TE
    - forwarding possible along paths that IP alone would not allow (e.g., source-specific routing)!!
  - use MPLS for traffic engineering
- must co-exist with IP-only routers
MPLS forwarding tables

<table>
<thead>
<tr>
<th>in label</th>
<th>out label</th>
<th>dest</th>
<th>out interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td></td>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td>D</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>A</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>in label</th>
<th>out label</th>
<th>dest</th>
<th>out interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>6</td>
<td>A</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>9</td>
<td>D</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>in label</th>
<th>out label</th>
<th>dest</th>
<th>out interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>6</td>
<td>A</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>in label</th>
<th>out label</th>
<th>dest</th>
<th>out interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>-</td>
<td>A</td>
<td>0</td>
</tr>
</tbody>
</table>
Chapter 5: Summary

- principles behind data link layer services:
  - error detection, correction
  - sharing a broadcast channel: multiple access
  - link layer addressing

- instantiation and implementation of various link layer technologies
  - Ethernet
  - switched LANS
  - PPP
  - virtualized networks as a link layer: ATM, MPLS
Questions?
ATM Physical Layer (more)

Two pieces (sublayers) of physical layer:
- Transmission Convergence Sublayer (TCS): adapts ATM layer above to PMD sublayer below
- Physical Medium Dependent: depends on physical medium being used

TCS Functions:
- Header **checksum** generation: 8 bits CRC
- Cell **delineation**
- With "unstructured" PMD sublayer, transmission of **idle cells** when no data cells to send
ATM Physical Layer

Physical Medium Dependent (PMD) sublayer

- **SONET/SDH**: transmission frame structure (like a container carrying bits);
  - bit synchronization;
  - bandwidth partitions (TDM);
  - several speeds: OC3 = 155.52 Mbps; OC12 = 622.08 Mbps; OC48 = 2.45 Gbps, OC192 = 9.6 Gbps

- **TI/T3**: transmission frame structure (old telephone hierarchy): 1.5 Mbps/ 45 Mbps

- **unstructured**: just cells (busy/idle)
Manchester encoding

- Used in 10BaseT
- Each bit has a transition
- Allows clocks in sending and receiving nodes to synchronize to each other
  - no need for a centralized, global clock among nodes!
- Hey, this is physical-layer stuff!