Network Adaptors

AKA Network Interface Cards (NIC)
Network Adaptors

- Components
- Options for Use
  - Data Motion
  - Event Notification
- Potential performance bottlenecks
- Programming device drivers
Network Adaptors

- Processor
- Cache
- Memory
- Network Adaptor

Communication?

memory bus (MBUS)

input/output bus (I/O BUS)
Network Adaptors

- **Adaptor Implements:**
  - Encoding
  - Framing
  - Error detection
  - Medium access control

- **Data Motion**
  - Direct Memory Access (DMA)
  - Programmed Input/Output (PIO)
Network Adaptor: DMA

- Scatter
  - Adaptor memory
  - Host memory

- Gather
  - Host memory
  - Adaptor memory
Network Adaptor: DMA
Network Adaptor: PIO

Host memory | Adaptor memory | Adaptor memory | Host memory

Processor

© CS 438 Staff, University of Illinois
Network Adaptor: PIO
Network Adaptor Use

- Data Motion
  - Direct Memory Access (DMA)
    - Processor free to do other things
    - Can be faster than memory copy through CPU
    - Start up cost
  - Programmed Input/Output (PIO)
    - Processor manages each access (loads/stores)
    - Faster than DMA for small amounts of data
Network Adaptor Use

- **Event Notification**
  - Hardware interrupts
    - Processor free to do other things
    - Events delivered immediately
    - State (register) save/restore expensive
    - Context switches more expensive
  - Event polling
    - Processor must periodically check
    - Events wait until next check
    - No extra state changes
Network Adaptor Performance

- Potential bottlenecks
  - Link capacity
  - I/O bus bandwidth
  - Memory bus bandwidth
  - Processor computing power
Programming Device Drivers

- Sample device driver in P&D
- Better examples in Linux
- Key Features
  - Memory-mapped control registers
  - Interrupt driven
  - Handler code must execute quickly
  - Logically concurrent with other processors
Direct Link Examples

- Goal
  - Explain real systems in terms of direct link topics
- TCP transport layer
- IP network layer
- Two examples of data link/physical layers
  - Ethernet
  - FDDI
- merely case studies—no need to memorize details
Example

- **TCP transport layer (reliable transmission)**
  - sliding window algorithm
  - adaptive window sizes
    - heuristics to address contention
    - aim at global optimum
    - see P&D 6.3 for details or wait until April

- **IP network layer (error detection)**
  - IP checksum
  - backs up stronger data link barriers (usually CRC)
Example

- 10 Mbps Ethernet (Xerox)
  - Encoding
    - Manchester
    - 10 Mbps, so transitions at 20 MHz
  - Error detection
    - Cyclic redundancy check (probably CRC-32)
  - Framing
    - Sentinel marks end-of-frame
    - Bit-oriented (similar to HDLC)
    - Variable length
    - Data-dependent length
  - Medium access control
    - CSMA/CD
10Mb Ethernet Frame Format
**Ethernet Frame Components**

- **Preamble + Start of Frame**
  - 7 bytes of 10101010, 1 byte of 10101011
  - Encoded as 10MHz square wave
  - Synchronize receiver’s clock

- **Source and Destination Address**
  - Unique unicast Ethernet addresses
    - 20 bit manufacturer prefix + 28 bit ID
  - Broadcast address: FF:FF:FF:FF:FF:FF
  - Multicast address: MSB set (80:00:...)
Ethernet Frame Components

- **Type**
  - 2 – bytes
  - Used to demultiplex higher layers

- **Body + Padding**
  - Minimum data size = 46 (minimum frame size = 64)
  - Data padded to minimum value
  - Maximum data size = 1500
Ethernet Frame Components

- CRC
  - 4 byte
- End of frame marker
  - 1 byte
- Total of 27 bytes header and trailer
- Xerox vs. 802.3
  - 802.3 replaces type with length
  - 802.3 drops EOF
IEEE 802.11 Frame Format

- **Types**
  - control frames, management frames, data frames

- **Sequence numbers**
  - important against duplicated frames due to lost ACKs

- **Addresses**
  - receiver, transmitter (physical), BSS identifier, sender (logical)

- **Miscellaneous**
  - sending time, checksum, frame control, data
IEEE 802.11 Data Frame Format

<table>
<thead>
<tr>
<th>bytes</th>
<th>Frame Control</th>
<th>Duration/ID</th>
<th>Address 1</th>
<th>Address 2</th>
<th>Address 3</th>
<th>Sequence Control</th>
<th>Address 4</th>
<th>Data</th>
<th>CRC</th>
</tr>
</thead>
<tbody>
<tr>
<td>bits</td>
<td>2</td>
<td>2</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>2</td>
<td>6</td>
<td>0-2312</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Protocol version</th>
<th>Type</th>
<th>Subtype</th>
<th>To DS</th>
<th>From DS</th>
<th>More Frag</th>
<th>Retry</th>
<th>Power Mgmt</th>
<th>More Data</th>
<th>WEP</th>
<th>Order</th>
</tr>
</thead>
</table>
# IEEE 802.11 Control Frame Format

- **Acknowledgement**
  - ACK
  - **bytes**: 2 2 6 4
  - **Frame Control** | **Duration** | **Receiver Address** | **CRC**

- **Request To Send**
  - RTS
  - **bytes**: 2 2 6 6 4
  - **Frame Control** | **Duration** | **Receiver Address** | **Transmitter Address** | **CRC**

- **Clear To Send**
  - CTS
  - **bytes**: 2 2 6 4
  - **Frame Control** | **Duration** | **Receiver Address** | **CRC**