FRAME FORMATS

AN APPROACH BASED ON MINIMAL CHANGES

The Emphasis

- The emphasis was on changing the functionality as little as possible.
  - Defer that to another paper (94/171)
  - Lots of history behind fields
    - Don't want to break something by accident
    - Look at each functionality change, one by one
  - Focus on field order
    - How to PARSE the frame
  - Better definition for some Elements
The First Byte

- Version Field
  » Reduce it to two bits
  » Provide two reserved bits

- Type Field
  » There are only 9 types currently defined
  » do not need the distinction for data, management etc.
  » 4 bit field

Second Byte

- The Control Bits
  » To AP, From AP (no change)
    - History
  » Elements Present, More (no change)
  » Power Management Bits (no change)
  » Contention Free
    - Not gone yet, two bits
  - Retry
    » Redundant, but....
  » Vote to put back if CF goes.
**Network ID, Sequence #**

- Put NID here for consistency
  - Needed in nearly every frame type

- Sequence #
  - MSDU-ID is a poor choice of name
  - Only need to have a "negligible" rate of packet duplication
    - A count is easier and good enough
    - What was the HASH algorithm

---

**Elements**

- Could go in any frame TYPE
  - Needed for PHY specific needs
    - Microcells for DS PHY
  - Backward compatibility

- Every frame is handled the same way
  - Elements are handled consistently
  - Always in the same place
  - Refer to flow chart for parse
  - Placed before any TYPE dependant fields

- Many undefined elements
Version Field and Elements

• How does the version field work?
  » VERSION 1 device will:
    • Reject higher version frames
    • Backward compatibility
      – Table driven?

• Elements
  » VERSION 1 device will:
    • Ignore Elements it doesn't understand
      – It can PARSE them but ignore contents
    • Added only when needed

TYPE Dependant Fields

• Duration there etc.
• Minor change
  » Beacon timestamp, now a fixed field
CRC

- Is the CRC 8 really OK?
  - NO
    - PHY work suggests CRC 16 OK for very short fields
  - MAC should standardize on one CRC
    » IEEE 32 CRC

Frame Lengths

No significant change in frame lengths

<table>
<thead>
<tr>
<th>Frame Type</th>
<th>Old</th>
<th>New</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTS</td>
<td>17</td>
<td>16</td>
</tr>
<tr>
<td>CTS</td>
<td>8</td>
<td>12</td>
</tr>
<tr>
<td>POLL</td>
<td>20</td>
<td>18</td>
</tr>
<tr>
<td>DATA (fragmented)</td>
<td>27(min)</td>
<td>25(min)</td>
</tr>
<tr>
<td>ACK (fragmented)</td>
<td>9</td>
<td>13</td>
</tr>
</tbody>
</table>
**Parsing the Frame**

- Consistent for every frame type
  - Simple state machine or code
- Remember that the frame might be bad
  - May need to back everything out if CRC bad
  - Need ‘temporary’ storage anyway

**94/171 - Fixing the Fields**

- PHY Specific Elements
- Time Stamp in Beacon
- Removing Asynchronous Contention Free
- Fragmentation Field only when needed
- New LOAD Element
- Restore ‘RETRY’ bit