Network Layer Overview

Ian Marsden, ZigBee NWG Chair

ZigBee Open House, Chicago, September 14\textsuperscript{th}, 2005
ZigBee is built upon the foundations provided by the IEEE 802.15.4 standard.
ZigBee NWK Layer Design Goals

- Enable low-cost, low-power embedded mesh networking.
- Support a wide variety of technical requirement and design tradeoffs.
  - Battery life vs. throughput.
  - Latency vs. spatial coverage.
  - Code size vs. “Ease of use” and “Feature richness”.
Architecture: NWK layer details

- ZigBee Device Types
- Stack Profile, Network Rules
- Network Management and Addressing
- Message Routing
- Route Discovery and Maintenance
- Security
Architecture:
Network Structure in ZigBee
Architecture: Stack Profile

Sets the rules that the network adheres to:

- nwkMaxDepth
- nwkMaxChildren
- nwkMaxRouters
- nwkSecurityLevel

And many more

- Table sizes
- Timeouts
- Route Cost Calculation Algorithm
ZigBee Coordinator (ZC)

- One and only one required for each ZigBee network.
  - First one to the party
- Initiates network formation.
  - Selects the time and place (Channel, PANId, Stack Profile)
- Acts as IEEE 802.15.4 2003 PAN coordinator (FFD).
- Also performs as router once network is formed.
- Not necessarily a dedicated device can perform an application too.
ZigBee Router (ZR)

- Optional network component.
- Discovers and associates with ZC or ZR.
  - Extends the network coverage
- Acts as IEEE 802.15.4 2003 coordinator (FFD).
- Manages local address allocation / de-allocation
- Participates in multi-hop / mesh routing of messages.
- Looks after its ZED’s when it comes to broadcasting and routing messages
ZigBee End Device (ZED)

- Optional network component.
- Discovers and associates with ZC or ZR.
- Acts as IEEE 802.15.4 2003 device (RFD).
- Can be optimised for very low power operation
- Relies on its parent to let it sleep
- Shall not allow association.
- Shall not participate in routing.
Network Initiation: ZC

1. ZigBee Coord. APL
   - NLME-NETWORK-FORMATION.request

2. ZigBee Coord. NWK
   - MLME-SCAN.request
     - Perform energy detection scan
     - MLME-SCAN.confirm
     - Select channel, PAN ID and logical address
     - MLME-SCAN.request
     - Perform active scan
     - MLME-SCAN.confirm

3. ZigBee Coord. MAC
   - MLME-SET.request
     - MLME-SET.confirm
     - MLME-START.request
     - MLME-START.confirm
**NLME-NETWORK-FORMATION.request**

| ScanChannels, ScanDuration, BeaconOrder, SuperframeOrder, PANId, BatteryLifeExtension |

Directs the MAC to start up a PAN with the specified parameters using the MLME-START.request{} primitive.
NLME-NETWORK-FORMATION.confirm

{  
    Status
}

Reports the results of a network formation attempt. Status values are INVALID_REQUEST, STARTUP_FAILURE or any status value returned from the MLME-START.confirm{ } primitive.
Network Association: ZR & ZED

1. **Select suitable PAN**
   - Child APL: NLME-NETWORK-DISCOVERY.request
   - Child NWK: MLME-SCAN.request
   - Child MAC: Perform active or passive scan

2. **Authentication procedure**
   - Child APL: NLME-JOIN.request
   - Child NWK: MLME-JOIN.confirm
   - Child MAC: MLME-JOIN.notify.indication

3. **Association procedure**
   - Child APL: NLME-JOIN.confirm
   - Child NWK: MLME-JOIN.notify.indication
   - Child MAC: MLME-JOIN.notify.indication

4. **Authentication procedure**
   - Child APL: NLME-START-ROUTER.request
   - Child NWK: MLME-START-ROUTER.confirm
   - Child MAC: MLME-START.request

5. **Association procedure**
   - Child APL: NLME-START-ROUTER.confirm
   - Child NWK: MLME-START.confirm
   - Child MAC: MLME-START.confirm
### NLME-NETWORK-DISCOVERY.request

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>ScanChannels,</td>
</tr>
<tr>
<td>ScanDuration</td>
</tr>
</tbody>
</table>

Directs the NWK layer to scan for existing networks on a specified set of channels. The parameters are passed through to MLME-SCAN.request{}.
Returns data on a list of found networks. Descriptor data includes PAN ID, channel, stack profile, ZigBee version, beacon order, superframe order, permit joining. Status reports the results of the MLME-SCAN.confirm.
NLME-JOIN.request

NLME-JOIN.request

Used on a ZigBee router or ZigBee end device to request association with a particular PAN. Some MLME-ASSOCIATE.request{} parameters are passed through, e.g. ScanChannels, some are synthesized, e.g. CapabilityInfo.

{  
PANId,  
JoinAsRouter,  
RejoinNetwork,  
ScanChannels,  
ScanDuration,  
PowerSource,  
RxOnWhenIdle,  
MACSecurity  
}

Created by Neevia Document Converter trial version http://www.neevia.com
### NLME-JOIN.confirm

| NLME-JOIN.confirm | { PANId, Status } |

Reports the results of an attempt to join a particular network. Status values are `INVALID_REQUEST`, `NOT_PERMITTED` or any status value returned from the MLME-ASSOCIATE and MLME-SCAN.confirm{} primitives.
On a ZigBee coordinator or ZigBee router reports the successful joining of a child device. The parameters are as received from the MLME-ASSOCIATION.indication{} primitive.
NLME-START-ROUTER.request

| {                          
| BeaconOrder,             
| SuperframeOrder,         
| BatteryLifeExtension    
| }                       

Used on a ZigBee router to start beaconing and other router activities after a network has been joined. Parameters are passed through to the MLME-START.request{} primitive.
NLME-START-ROUTER.confirm

{  
  Status
}  

Reports the results of an attempt to start up a ZigBee router. Status values are INVALID_REQUEST or any status value returned from the MLME-START.confirm primitive.
Addressing:
Tree-structured Address Assignment

- CSkip based address assignment
- Address determined from tree location
| NLDE-DATA.request | \{ \\
| DstAddr,  \\
| NsduLength,  \\
| Nsdu,  \\
| NsduHandle,  \\
| Radius,  \\
| DiscoverRoute,  \\
| SecurityEnable \\
| \} |

Used by higher layers for all data transmissions, broadcast and unicast.
| NLDE-DATA.confirm | \{  
|                  |   NsduHandle,  
|                  |   Status  
|                  | \}  

Reports the status of a transmission indexed by handle. Status values are INVALID_REQUEST or any status returned by the MCPS-DATA.confirm{} primitive.
### NLDE-DATA.indication

| {                          |
|                            |
| SrcAddress,               |
| NsduLength,               |
| Nsdu,                     |
| LinkQuality               |
| }                         |

Reports the receipt of a NWK data frame.
Tree Routing:

- The address tells you where the destination is.
- Simple equation gives ‘route up’ or ‘route down’.
- If $Local\Addr < Dest\Addr < Local\Addr + CSkip(d-1)$ Route Down.
- Else Route Up.

Obviously not necessarily the most efficient route.
A ZC or ZR maintains a table of devices in its neighbourhood.

If the target device is physically in range it can send the message directly.

But what happens if the destination is not in the local neighbourhood?
Mesh Routing:

- ZC or ZR maintains a routing table of next hop addresses
- If the target device has a routing table entry then the message can be sent using this route.

That's great, but where do the routing table entries come from?
Routing: Route Discovery

- A device wishing to discover a route issues a route request command frame which is broadcast throughout the network.

- When the intended destination receives the route request command frame it responds with a route reply command frame.

- Potential routes are evaluated with respect to a routing cost metric.

- Best route is added to the routing tables of all devices on the route.
Message Routing:
The Basic Algorithm

1. See if the destination is in the Neighbour Table
2. Check for a Routing Table entry
3. Finally resort to Tree Routing

NB. ZRs store messages for sleeping ZED’s
Broadcast: The Basic Algorithm

- Transmit broadcast message
- Rebroadcast by local ZRs if it is new.
- Time & radius limited.
- ZRs store messages for sleeping ZED's
- ZRs issue broadcasts on behalf of sleeping ZEDs
The Stack Profile defines the security level in use.

Uses Network Key unless Link Key has been applied.

Tool box offers both authentication and encryption facilities.

Auxiliary Header and Message Integrity Code add overhead to the packet.

<table>
<thead>
<tr>
<th>nibSecurityLevel</th>
<th>Security Suite</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>NONE</td>
</tr>
<tr>
<td>1</td>
<td>MIC-32</td>
</tr>
<tr>
<td>2</td>
<td>MIC-64</td>
</tr>
<tr>
<td>3</td>
<td>MIC-128</td>
</tr>
<tr>
<td>4</td>
<td>ENC</td>
</tr>
<tr>
<td>5</td>
<td>ENC-MIC-32</td>
</tr>
<tr>
<td>6</td>
<td>ENC-MIC-64</td>
</tr>
<tr>
<td>7</td>
<td>ENC-MIN-128</td>
</tr>
</tbody>
</table>
Network Layer Management Primitives

NLME-PERMIT-JOINING.request  NLME-RESET.request
NLME-PERMIT-JOINING.confirm  NLME-RESET.confirm

NLME-DIRECT-JOIN.request  NLME-GET.request
NLME-DIRECT-JOIN.confirm  NLME-GET.confirm

NLME-LEAVE.request  NLME-SET.request
NLME-LEAVE.confirm  NLME-SET.confirm
NLME-LEAVE.indication

NLME-SYNC.request
NLME-SYNC.confirm
NLME-SYNC.indication
To summarise the ZigBee network layer:

- Has 3 device types; ZC, ZR and ZED.
- Performs network discovery and formation
- Performs address allocation
- Performs message routing
- Configured by the stack profile
- Provides network wide security
- Allows low power devices to maximize their battery life

ZigBee turns 802.15.4 into a low power multi-hop mesh network.
Any Questions