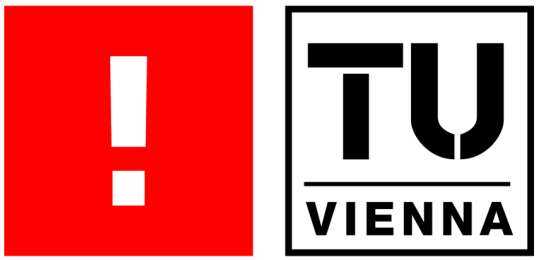

Bachelor Seminar Paper

ZigBee

Markus Gerstner
0305174, E535
markus.gerstner@gmx.at
WS 2009/10

betreut durch
Ao. Univ. Prof. Dr. Wolfgang Kastner

Inhaltsverzeichnis

1 ZigBee – Übersicht

Kabellose Übertragungstechniken existieren in einer großen Vielfalt an Ausprägungen, Eigenschaften und Fähigkeiten., um nur einige zu nennen: WiMAX™, WiFi™, Bluetooth™, Active RFID, Wibree, auch einige sehr spezialisierte Standards wie Wireless USB.

Was unterscheidet ZigBee von all diesen Standards und wo ist sein Anwendungsbereich?

1.1 Einordnung

Viele Wireless Standards wurden dahingehend entwickelt hohe Datenraten zu gewährleisten, was auch stets einen relativ hohen Hardwarebedarf bedeutet. ZigBee hingegen ist ein klassisches Beispiel für eine Lowcost-Implementierung bei der nicht hohe Datenraten sondern hohe Einsetzbarkeit und minimale Anforderungen an die Hardware die Ziele der Entwicklung waren. (Entwurfskriterium ZigBee 2006: muss in einer 8-Bit Mikrocontroller Architektur ausführbar sein)

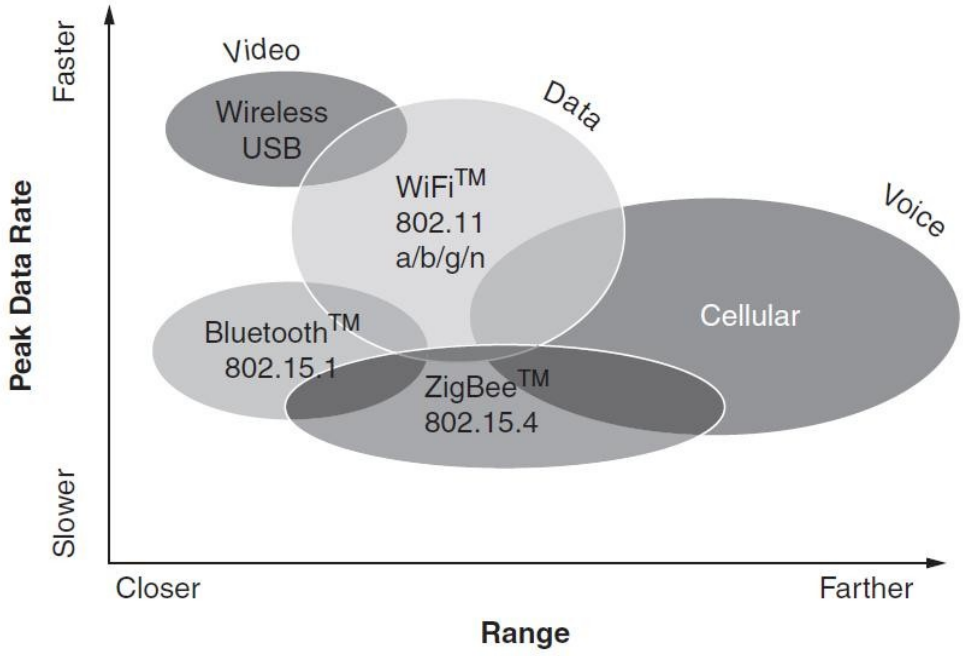

Abbildung 1: Einordnung von ZigBee

Damit ist ZigBee eine „Wireless Feldbustechnik“ und dazu gedacht, unterschiedlichste Sensoren, Aktuatoren und Steuergeräte mit hoher Verlässlichkeit miteinander zu verbinden.

ZigBee ist ein Standard der eine Menge an Kommunikationsprotokollen und RF-Techniken definiert, welche hoch zuverlässig, eine niedere Datenrate bei geringer Reichweite für kabellose Netzwerke von Sensoren, Aktuatoren und Steuergeräte sicherstellen soll.

Großes Augenmerk wurde bei der Definition ZigBees auf die Zuverlässigkeit gelegt. (engl. Reliability ist definiert als: *„The ability of an item to perform a required function under given conditions for a given time interval“ EC 60050, 191-02-06*)

Des Weiteren sollten ZigBee Systeme auch ohne Einsatz teurer Hardware in rauen Umgebungen

eine akzeptable Durchsatz-Rate bei uneingeschränkter Funktionalität bieten. Dazu wurde auf bekannte Techniken und Standards zurückgegriffen.

ZigBee wurde von der ZigBee Alliance entwickelt, eine Dachorganisation von Unternehmen aus verschiedenen Industriesparten wie Software, Chip-Manufacturing, Elektrotechnik und RF-Technik.

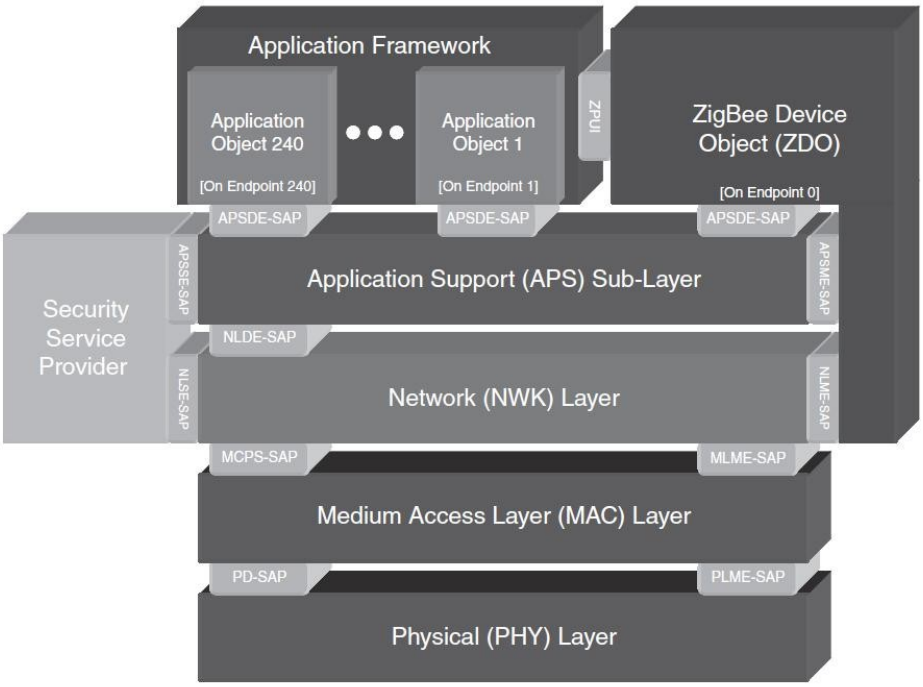

Abbildung 2: Schematische Darstellung des ZigBee Standards

Wie Abbildung 2 zeigt, werden die untersten 2 Netzwerkschichten durch den IEEE 802.15.4 - 2003 Standard definiert und sind damit unabhängig vom ZigBee Standard. Daher mussten nur Network Layer und Application Layer sowie eine Security Schicht definiert werden, welche man als *„protocol stack software“*(Farahani 2008) zusammenfasst.

Zur Gewährleistung des hohen Grades an Zuverlässigkeit werden unter anderen folgende Techniken implementiert.

- IEEE 802.15.4 - 2003
- CSMA-CA
- 16-bit CRCs
- „Acknowledgments“ nach jedem „Hop“
- „Mesh-networking“
- „End-to-end acknowledgments“ zur Bestätigung des Datentransfers

Der ZigBee Protocol Stack ist heute in drei unterschiedlichen Versionen verfügbar, wobei die ersten beiden sich nur so marginal unterscheiden, dass sie die gleiche Stack Profile ID bekamen. Abbildung 3 zeigt eine Auflistung der heute gängigen ZigBee Protocol Stacks und ihren groben Funktionsumfang. In dieser Arbeit soll jedoch nur ein grober Überblick über den ZigBee Standard

selbst gegeben werden und nicht auf die Unterschiede der Stack Versionen eingegangen werden.

Feature	ZigBee 2006	ZigBee 2007	ZigBee Pro
Size in ROM/RAM	Smallest	Small	Bigger
Stack Profile	0x01	0x01	0x02
Maximum hops	10	10	30
Maximum nodes in network	31,101	31,101	65,540
Mesh networking	✓	✓	✓
Broadcasting	✓	✓	✓
Tree routing	✓	✓	–
Frequency Agility	–	✓	✓
Bandwidth Used By Protocol	Least	More	Most
Fragmentation	–	✓	✓
Multicasting	–	–	✓
Source routing	–	–	✓
Symmetric Links	–	–	✓
Standard Security (AES 128 bit)	✓	✓	✓
High Security (SKKE)	–	–	✓

Abbildung 3: Funktionsumfang der ZigBee Stack Profiles

1.2 IEEE 802.15.4

Der Ende der 1990er Jahre entwickelte Standard IEEE 802.15.4 definiert ein Übertragungsprotokoll für WPAN (Wireless Personal Arae Networks) auf den untersten zwei Schichten des OSI-Modells, also auf der Bitübertragungs- und der MAC-Schicht.

Höhere Funktionalität, wie Routing oder Mesh-Networking unterliegen nicht diesem Standard und müssen von darüber liegenden Schichten übernommen werden.

Wesentliche Entwicklungsziele waren geringe Leistungsaufnahme, kostengünstige Hardware, sichere Übertragung und Nutzung kostenfreier ISM Bänder.

1.2.1 Devices und Topologien

IEEE 802.15.4 sieht zwei Typen von Netzknoten mit unterschiedlicher Funktionalität vor. Dies sind FFDs (Full Functional Device) und RFDs (Reduced Functional Device), wobei RFDs nur eine Teilmenge der Funktionalität des Standards überdecken und daher kostengünstiger produziert werden können. Die bedeutendste Einschränkung von RFDs ist, dass es ihnen nur möglich ist, mit FFDs zu kommunizieren. FFDs hingegen implementieren den vollen Funktionsumfang und können sowohl mit RFDs als auch mit anderen FFDs kommunizieren. Ein FFD übernimmt in einem Netz die spezielle Funktion des PAN Coordinators, welcher den PAN Identifier festlegt, der das Netzwerk von anderen IEEE 802.15.4 Netzen in Funkreichweite abgrenzt. Des weiteren übernimmt ein FFD im Slotted Mode die Synchronisation aller Netzknoten im Netz, in welchem maximal 254 Geräte verbunden sein können (inklusive PAN Coordinator, die Adresse 255 stellt einen Broadcast dar).

Daraus resultieren 3 verschiedene Netztopologien:

1. Stern-Topologie: bei dieser kommunizieren alle Devices mit dem Netz-Koordinator.

2. Peer-to-Peer: Wie in jedem Netz gibt es auch hier nur einen Netz-Koordinator, jedoch können die Knoten auch untereinander kommunizieren.
3. Baumstruktur (Cluster Tree): Hier stellt ein Teil der Geräte, typischerweise RFDs, die Blätter des Baums dar, welche mit FFDs verbunden sind, die für diesen Teil des Netzes die Funktion des Netzkoordinators übernehmen. Diese Stamm-FFDs sind wiederum direkt oder indirekt über andere FFDs mit Koordinatorfunktion mit dem PAN Koordinator des Netzes verbunden. Diese Netztopologie stellt lediglich eine Mischung der anderen beiden Topologien dar, da mit den Mitteln des Standards keine Vermaschung oder Routing von Nachrichten durchgeführt werden kann.

Der Standard sieht keine Vermittlungsschicht (Network Layer) vor, daher müssen Routing und andere Funktionen durch übergeordnete Schichten definiert werden, erst dann sind echte vermaschte Netze realisierbar.

Ein IEEE 802.15.4 Network, unabhängig von seiner Topologie, wird immer von einem PAN Coordinator gebildet und hat folgende Funktionalität zu gewährleisten:

- Zuweisen einer eindeutigen Adresse für ein Device (16 Bit „short address“ oder 64 Bit „extended address“)
- Initiieren, Terminieren und Routen von Messages durch das Netzwerk
- Auswahl eines 16 Bit eindeutigen PAN-Identifiers für das Netzwerk. Die PAN ID erlaubt es innerhalb des Netzwerks nur 16 Bit breite Geräteadressen zu verwenden und trotzdem über PAN Grenzen hinweg mit anderen Devices zu kommunizieren.

Es existiert nur ein einziger PAN Coordinator pro Netzwerk, welcher über längere Zeiträume hinweg aktiv sein muss, daher ist dieser (meist) nicht „battery powered“ sondern an das Stromversorgungsnetz oder an eine unabhängige Stromversorgung (USV) angeschlossen. Das kleinste mögliche Netzwerk besteht aus zwei Knoten, dem PAN Coordinator und einem Enddevice.

1.2.2 CSMA-CA

Um einen durch den PAN Coordinator festgelegten Frequenzkanal optimal auszunutzen, verwendet IEEE 802.15.4 – 2003 CSMA-CA (Carrier Sense Multiple Access with Collision Avoidance), um Kollisionen während eines gleichzeitigen Übertragens zu vermeiden.

1.2.3 Netzwerke und Beacons

Für ein Netzwerk mit einem einzigen gemeinsamen Medium gibt es prinzipiell zwei unterschiedliche Zugriffsvarianten:

- *contention-based (unslotted mode, non-beacon)*
- *contention-free (slotted mode, beacon enabled mode)*

Bei *contention based networks* verwenden alle Devices den CSMA-CA Algorithmus.

Bei *contention free networks* wird jedem im Netzwerk beteiligten Device ein *Guaranteed Time Slot* (GTS) zugewiesen. (Hinweis: Implementierung von Realtime Protokollen auf Basis 802.15.4) Damit es jedoch nicht zu Kollisionen während der Datenübertragung kommt, müssen alle Devices zueinander synchronisiert werden.

Der *Beacon* ist eine Nachricht von besonderem Format, welche zur Synchronisation aller im „Channel“ und Netzwerk befindlichen Knoten dient(beacon-enabled PAN). Sind die Knoten einmal synchronisiert, müssen sie nur mehr zyklisch aufwachen, um sich erneut zu synchronisieren und abzufragen, ob Daten für sie beim mit ihnen assoziierten FFD vorhanden sind.

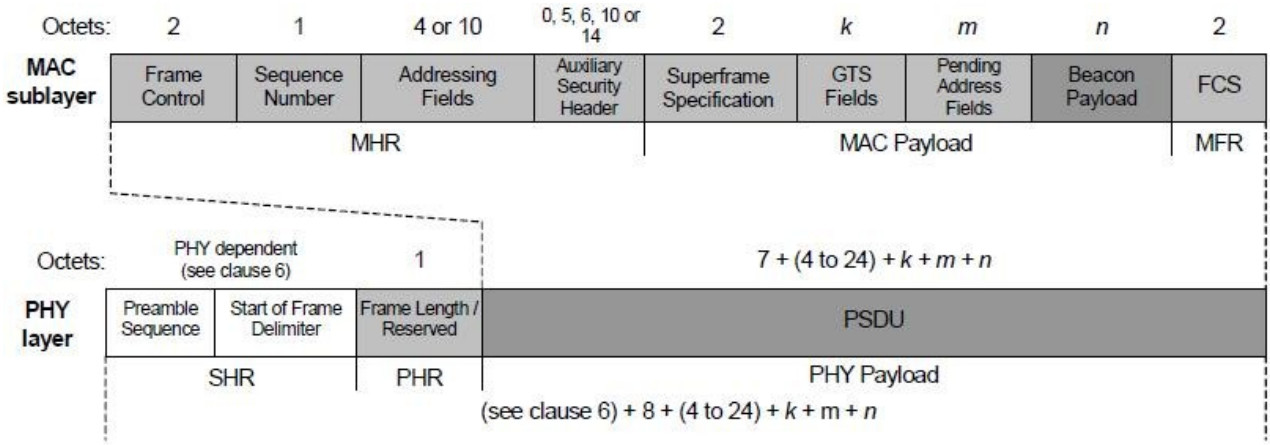

Abbildung 4: Schematische Darstellung eines Beacon Frame und des PHY Pakets

Ohne Synchronisation müssten Endgeräte ständig online sein und den Kontakt mit ihren *Full-Functional-Devices* (FFD) halten, womit eine beträchtliche Einschränkung der Lebensdauer bei batteriebetriebenen *Enddevices* (ED) gegeben wäre.

Weitere Vorteile des beacon-enabled PAN sind die garantierte Übertragungsrate und das Wegfallen des CSMA-CA Algorithmus.

1.2.4 Datentransferarten in IEEE 802.15.4

Drei mögliche Fälle werden unterschieden:

- Datentransfer vom Device zum PAN Coordinator
- Datentransfer vom PAN Coordinator zum Device
- Peer-to-Peer Datentransfer (Datentransfer zwischen zwei Peerdevices)

Datentransfer vom Device zum PAN Coordinator

Im beacon-enabled PAN wartet das Device ab, bis es einen Beacon erhält, synchronisiert sich und sendet anschließend Daten zum Coordinator. Ein ACK ist optional und wird als Bit in der Nachricht mit kodiert.

Durch das Setzen eines Urgent-Bits wird das Device dazu veranlasst, die Nachricht sofort zu senden. Dann verwendet es den CSMA-CA Algorithmus und sendet die Nachricht bei Freiwerden des Kanals (gleiche Funktionsweise wie in non-beacon PAN's).

Datentransfer vom PAN Coordinator zum Device

Im beacon-enabled PAN:

Wenn der PAN Coordinator eine Nachricht an ein Device übertragen möchte, so sendet er einen Beacon aus, in dem er signalisiert, dass Daten für das Device vorhanden sind. Das Device wacht auf, synchronisiert sich, und zeigt mit der Transmission einer Requestmessage, dass es „ready-to-receive“ ist. Dies quittiert der PAN Coordinator mit einem ACK, und sendet dann die eigentlich zu übertragende Nachricht. Das ACK des Devices ist optional.

Im nonbeacon PAN:

Im nonbeacon PAN gibt es laut Spezifikation keine Möglichkeit einem Device zu signalisieren, dass es Daten zu erhalten gibt. Der PAN Coordinator wird gezwungen zu warten, bis sich das Device bei ihm mit einem Request-Data meldet. Wenn Daten vorhanden sind, sendet der PAN Coordinator ein ACK mit Angabe der Länge der zu übertragenden Daten. Sind die Daten aufgrund von mangelnder Aktualität verworfen worden, sendet der PAN Coordinator ein *„Acknowledge“* der anzeigt, dass keine weiteren Daten vorhanden sind. Alternativ (der Standard ist hier offen) kann auch eine Datennachricht mit Länge 0 gesendet werden.

Das ACK des Devices ist wiederum optional.

1.2.5 IEEE 802.15.4 Adressierung

IEEE 802.15.4 sieht zwei Adressformate für Devices innerhalb eines Netzwerks vor.

Im s*hort adressing mode* werden den Devices im Netzwerk 16 Bit lange, im e*xtended addressing mode* 64 Bit breite Adressen zugeordnet.

Die Vorteile des short addressing modes liegen in der Verkürzung der Nachrichten und dem daraus resultierenden kleineren Speicherverbrauch des Protokollstacks.

Die Kombination zwischen PAN ID und Device Adresse lässt auch eine Kommunikation über die Netzgrenzen hinweg zu. IEEE 802.15.4 standardisiert hier jedoch gar nichts, sodass diese Aufgabe der übergeordneten Schicht, der Vermittlungsschicht, zu geordnet wird.

1.2.6 Assoziation und Disassoziation

Association und *Disassociation* sind Services, welche noch durch die Sicherungsschicht von IEEE 802.15.4 der NWK Layer zur Verfügung gestellt werden. Ihre Aufgabe ist es, einen Kommunikationspartner in einem PAN anzumelden oder abzumelden. Will ein Kommunikationspartner sich in ein PAN eingliedern, sendet dieser einen *association request* an den PAN Coordinator, welcher diesen ablehnen oder annehmen kann. Will ein physisches Gerät das PAN verlassen, so sendet es einen *disassociation request*, daraufhin löscht der PAN Coordinator den Eintrag in den Bindingtabellen und signalisiert dem ZigBee Protocol Stack, dass sich ein Gerät abgemeldet hat. Der ZigBee Protocol Stack gibt darauf hin die endpoint address des Devices oder die gesamte ZigBee address wieder frei.

Die von der MAC Layer der NWK Layer angebotenen *service primitives* sind:

- MLME-ASSOCIATE.request
- MLME-ASSOCIATE .indication
- MLME-ASSOCIATE .response
- MLME-ASSOCIATE .confirm
- MLME-COMM-STATUS.indication
- MLME-DISASSOCIATE.request
- MLME-DISASSOCIATE..indication
- MLME-DISASSOCIATE.response

- MLME-DISASSOCIATE.confirm

Abbildungen 5 und 6 zeigen den Ablauf von Association und Disassociation.

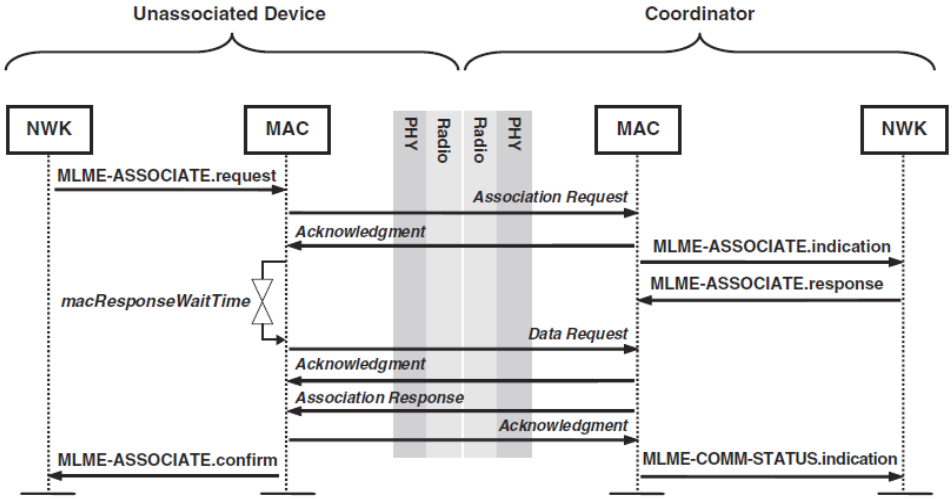

Abbildung 5: Ablauf der Anmeldung eines Device am PAN

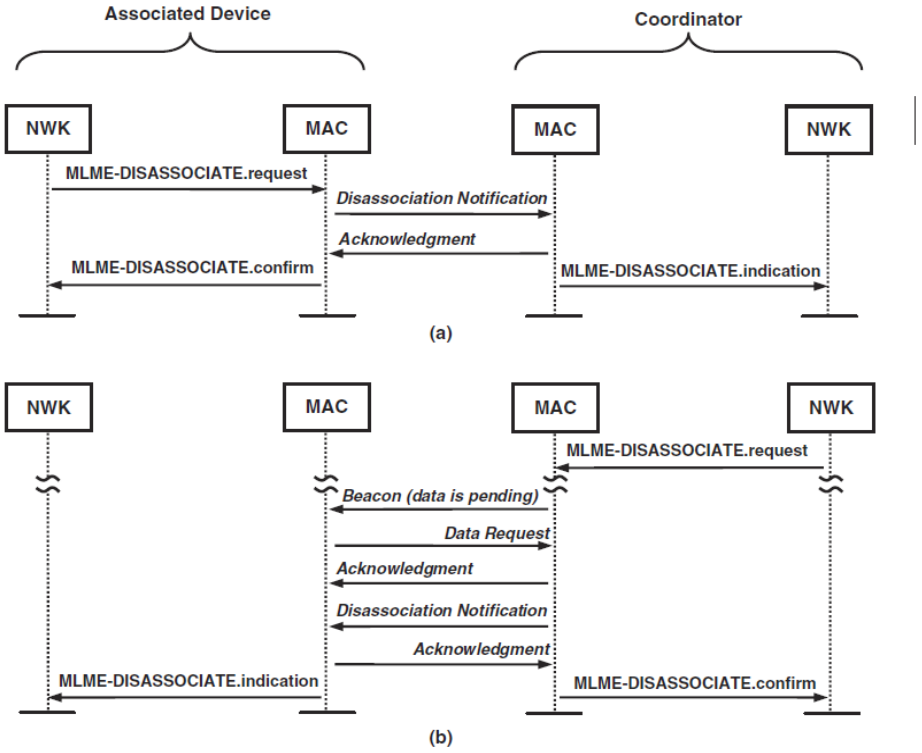

Abbildung 6: Abmeldung eines Device vom Netzwerk ausgelöst vom

a) Device b) Coordinator;

1.2.7 Self-Forming and Self-Healing Networks

IEEE 802.15.4 Netzwerke sind selbst formend, dass heißt, sie bedürfen keiner intialen Konfiguration, um sich zu erstellen. Wie bereits erwähnt konfiguriert sich das erste im „Äther“ befindliche FFD als PAN Coordinator und wartet auf andere FFDs oder RFDs, um die Netzstruktur aufzubauen.

2 Die Netzschicht (NWK Layer)

Die Netzschicht ist das Bindeglied zwischen MAC (Media Access Control Layer) und APL (Application Protocoll Layer). Ihre Aufgaben im Rahmen des ZigBee Standards sind Management der Netzwerk Informationen und das Routing von Nachrichten innerhalb des Netzes.

Wie bereits erwähnt, definiert der Standard IEEE 802.15.4 keine Funktionen der Vermittlungsschicht, sondern deckt alle Funktionen der Sicherungsschicht ab. Genau an diesem Punkt beginnt der eigentliche *ZigBee Protocol Stack*. Die Netzschicht des ZigBee Protokolls weist jeder Device Adresse eine eigene 16 Bit breite Netzwerkadresse zu, da die Transaktionen der Netzschicht eigene Netzwerkadressen benötigen.

Jeder Transmitter im Netzwerk erhält also eine eigene IEEE Adresse und eine ZigBee Netzadresse.

Nur der ZigBee Coordinator und die Routing Knoten können neue Routen finden und sind auch für die Wartung dieser zuständig (Das bedeutet das Auffinden unterbrochener Routen das Erstellen neuer Routen die Kommunikation zum Enddevice wieder her zu stellen). Die Netzschicht des ZigBee Coordinators ist zusätzlich noch für die Auswahl der Netztopologie verantwortlich (Tree, Star oder Mesh).

Die Netzschicht bietet mehrere Services die in zwei Gruppen unterteilt werden können:

- Data (über NWK Layer Data Entity, NLDE)
- Management (über NWK Layer Management Entity, NLME)

Die Attribute und Konfigurationsvariablen werden in der NIB (Network Information Base) gespeichert, welche von der APL durch die Serviceprimitive NLME-Get und NLME-Set verändert und ausgelesen werden kann.

Die Netzschicht eines ZigBee Coordinators (ZigBee Coordinator ist gleichzeitig PAN Coordinator) weist jedem Device in einem Netzwerk eine 16-Bit Netzwerkadresse zu, welche im Fall, dass auf MAC Ebene das short address format (16-Bit MAC Adresse) benutzt wird, identisch ist.

Die Netzschicht limitiert weiters die Distanz die ein Frame über das Netzwerk hinweg übertragen werden kann. Dieser Parameter, Radius genannt, wird zu jedem NWK-Frame hinzugefügt und bei jedem *relay* dekrementiert. Erreicht er 0, wird der Frame nicht mehr weiter geleitet.

Die Kommunikation wird in drei Gruppen untergliedert: Broadcast, Multicast und Unicast.

2.1 Kommunikationsarten

2.1.1 Broadcast

Prinzipiell existieren zwei Arten von Broadcasts. Bei der ersten wird eine der Broadcast-Adressen in das Destination Address Field geschrieben, alle Devices innerhalb eines PANs erkennen diese Adresse als ihre eigene Adresse an. Die Broadcast-Adressen sind:

- 0xfff - gilt für alle Geräte innerhalb eines ZigBee Networks. FFDs müssen für ihre schlafenden Kindknoten den Broadcast cachen.
- 0xffd - alle nicht schlafenden Geräte eines Netzwerks werden adressiert
- 0xffc - nur der ZigBee Coordinator und die ZigBee Router

Bei der anderen Art der Adressierung wird 0xffff in den PAN Identifier geschrieben. Dieser *Broadcast PAN Identifier* wird über mehre Netzwerke entlang eines *channel* weitergereicht.

Obwohl IEEE 802.15.4 den PAN Broadcast, also den Broadcast über mehrere Netzwerke unterstützt, verbietet ZigBee den Broadcast über Netzwerkgrenzen hinweg; erlaubt aber jedem Device in einem Netzwerk einen Broadcast zu initiieren.

In großen Netzwerken ist es nicht von Vorteil, dass jedes Device einen ACK an den *Originator* zurücksendet. Anstatt dessen stellen der PAN Coordinator (ZigBee Coordinator) und die ZigBee Router sicher, dass ihre Nachbardevices die Broadcastmessage erfolgreich weitergereicht haben. Nach dem Versenden des Broadcasts geht der Router oder PAN Coordinator in den Receive-Mode und wartet bis der selbe Frame wieder als Broadcast empfangen wird, was die erfolgreiche Weitergabe signalisiert. Dies wird als *passive acknowledgement mechanism* bezeichnet.

ZigBee Coordinator und Router warten eine Tabelle aller Broadcasts, BTT Broadcast Transaction Table genannt. Der einzelne Eintrag ist der BTR Broadcast Transaction Record und enthält neben der Sequence Number noch die Herkunfts-MAC Adresse. Der BTT ist einzig und allein für die Retransmission von verlorenen Broadcasts zuständig.

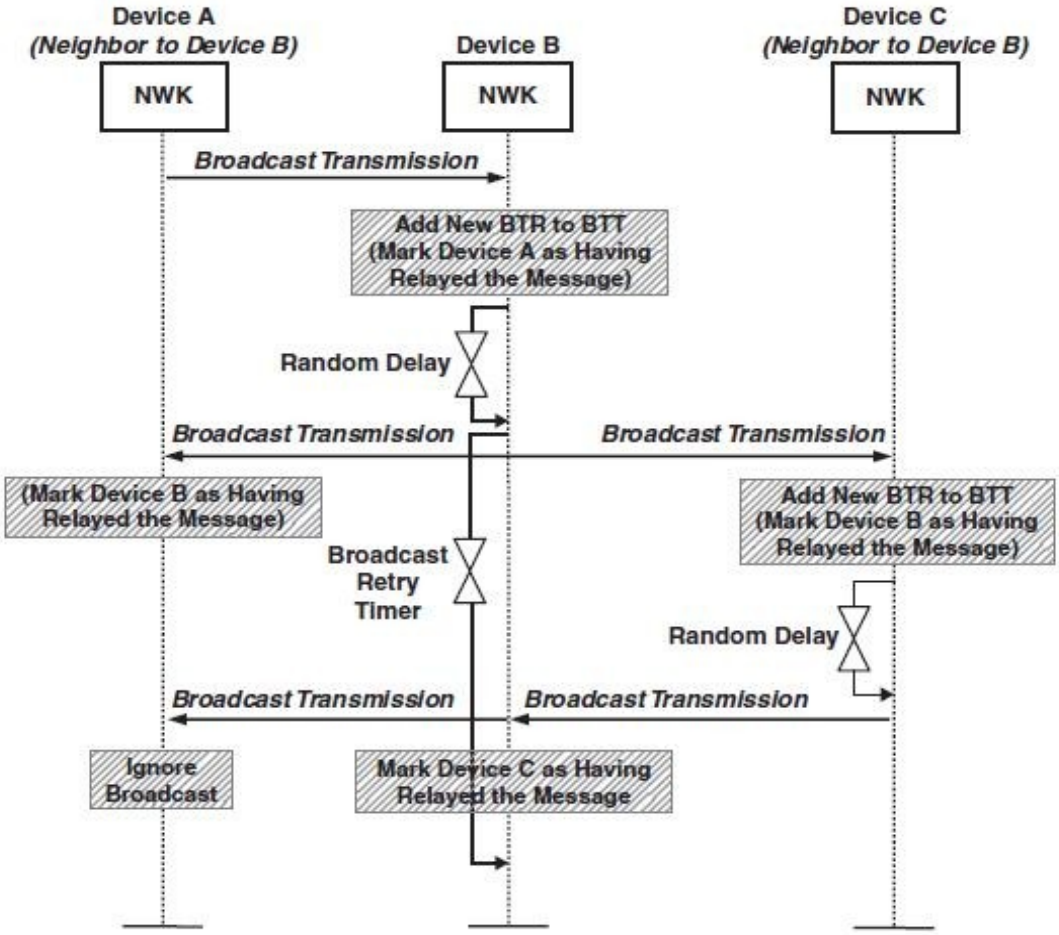

Abbildung 7: Broadcast mit "Passive Acknowledgement"

Während des Broadcasts wird der Frame von mehreren Devices übertragen, welches zu Collisions aufgrund des *Hidden Node Problem* führen kann. Damit die Chance auf Collisions minimiert wird, wartet der Router / ZigBee Coordinator eine zufällige Zeit lang, bevor er den Broadcast Frame erneut überträgt. Der dafür zuständige Parameter wird von der Netzschicht verwaltet und heißt *Broadcast Jitter.*

Abbildung 7 zeigt: A initiiert Broadcast, B empfängt und wartet ob andere Devices ebenfalls den

Broadcast senden möchten und überträgt den Broadcast Frame erneut. Empfängt Device C den Broadcast nicht innerhalb von *nwkPassiveAckTimeout ,* sendet es ihn erneut.

2.1.2 Multicasting

Multicasts richten sich nicht an alle Teilnehmer des Netzes sondern nur an eine Teilmenge dessen. Die Anwendungsmöglichkeit liegt auf der Hand, werden Devices zu einer logischen Gruppe zusammengeschlossen (Lichtkontroller, Lichtschalter, Dimmer und Luminanzsensor) wäre es unnötig alle Devices mit einem Unicast über ein Event zu benachrichtigen, anstatt dessen wird ein Multicast initiiert, um den Netzwerktraffic niedrig zu halten.

Multicast Gruppen werden über den *Multicast Group Identifier* identifiziert. Devices können mehreren Multicastgruppen zugehörig sein.

Die Unterstützung von Multicast Paketen erfordert von ZigBee zwei verschiedene Typen von Services, member mode und non-member mode.

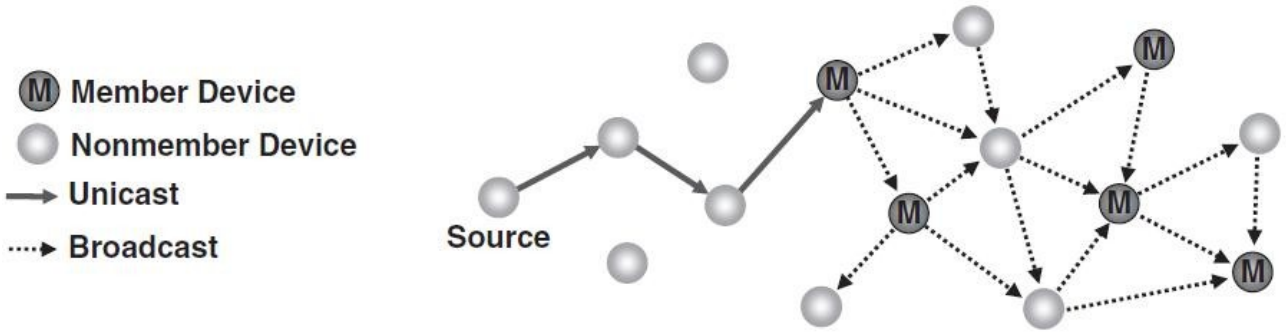

Abbildung 8: Multicastübertragung mit Hilfe von Non-members

Der NWK Data Frame enthält das Feld *multicast mode*, welches anzeigt, ob der Frame über ein Mitglied (member mode) oder nicht übertragen wurde. Obige Grafik zeigt einen Multicast, welcher von einem nonmember node aus an eine bestimmte Multicastgruppe gerichtet ist. Das Source-Device weiß, dass es nicht Mitglied der Multicastgruppe ist, durch einen Vergleich des Frames den es von der APL erhalten hat, mit seiner eigenen *multicast table*, in dem die Group-ID der Multicast-Gruppen eingetragen werden, in denen sich das Device befindet.

Angenommen ein Route Discovery zu einem Multicastgruppenmitglied wurde zuvor bereits durchgeführt, so wird der Frame von einem Node zum Nächsten per Unicastverbindung weitergereicht. Empfängt ein Node, welcher Mitglied der Multicastgruppe ist, den Frame, so wird dieser nicht mehr durch Unicast weitergegeben, sondern die Zieladresse wird auf 0xffff verändert und der Frame per Broadcast weitergegeben. Dieser „Multicast“ wird von den Mitgliedern der Multicastgruppe als normaler Broadcast behandelt. Der Frame wird in den *broadcast transaction table* (BTT) eingetragen, jedoch gibt es bei Multicast im Gegensatz zu Broadcast kein passive acknowledgement.

Dieser Broadcast wird von allen umliegenden Geräten empfangen, welche daraufhin eine beliebige Zeit kleiner als *nwkcMaxBroadcastJitter* warten und den Frame erneut übertragen. Dabei ist es unwichtig, ob das empfangende Device ein Member oder Non-member Node der Multicastgruppe ist. Für Non-member Nodes kann einzig und allein ein Non-member Radius definiert werden, welcher bei Übertragung über je einen nonmember node dekrementiert wird (und wenn 0 erreicht ist, wird der Frame folglich nicht mehr weiter gesendet). Wiederum einzige Ausnahme: wird der nonmember radius auf 7 gesetzt, so gibt es keine Beschränkung mehr und der Frame kann beliebig

oft per Broadcast ausgesandt werden. In allen Fällen gilt jedoch, wird ein Frame von einem member node aufgefangen, so wird der nonmember radius wieder auf den Maximalwert gesetzt, dieser Maximalwert wird ebenfalls im Header mitgesandt.

Multicasts werden im ZigBee Standard nur für Datentransmissionen benutzt. NWK Layer Kommandonachrichten dürfen nicht per Multicast versandt werden.

2.1.3 Many-to-One Communication

Abbildung 9 zeigt eine spezielle Art der Kommunikation in ZigBee. Diese ist als Many-to-One Communication bekannt, dabei erstellt ein spezieller Knoten der als *sink* oder *concentrator* bezeichnet wird, Routen von allen ZigBee Routern und dem ZigBee Coordinator zu sich selbst, so lange diese Knoten sich innerhalb eines bestimmten Radius befinden.

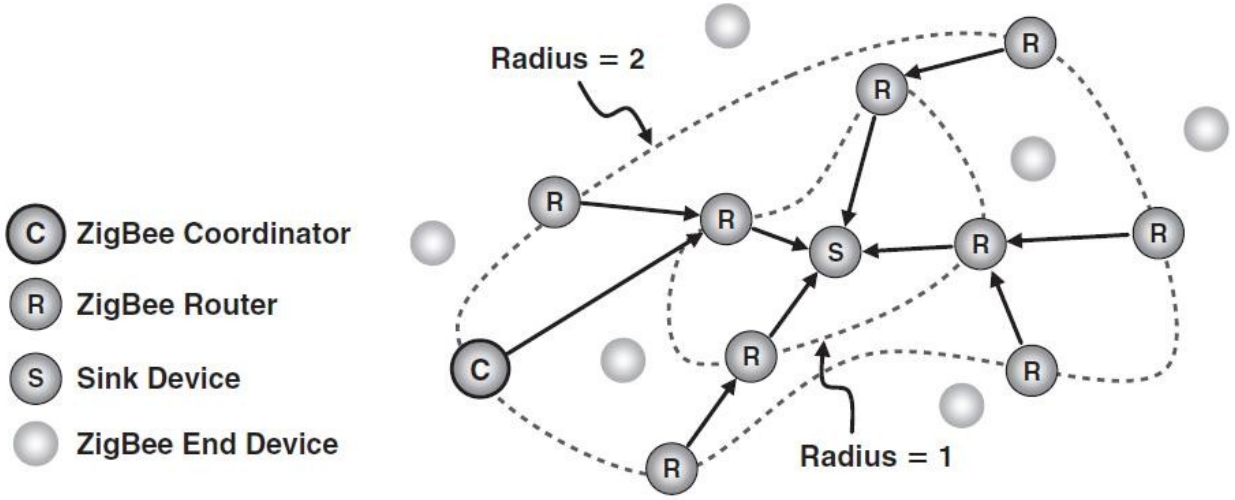

Abbildung 9: Many-to-one Communication

2.2 Adressierungsarten und Topologien

2.2.1 Hierarchical-Tree Topology

Bei der Hierarchical-Tree Topology baut der ZigBee Coordinator eine baumartige Adressierungsstruktur auf, wobei er selbst die Wurzel des Baumes repräsentiert. ZigBee Router können als Baum/Elternknoten innerhalb des Baumes fungieren. ZigBee Enddevices können nur Blatt-/Endknoten sein, da sie keine Routingfunktion unterstützen.

Dabei stellt der ZigBee Standard einen Adressierungsmechanismus zur Verfügung, der als *default distributed address allocation* bekannt ist, welcher jedoch durch eine vom Application Developer programmierte Routine ersetzt werden kann.

Ist im ZigBee Coordinator die Variable nwkUseTreeAddrAlloc gesetzt, weist der ZigBee Coordinator jedem potentiellen Elternknoten im Baum einen eigenen Subblock des Adressraums zu. Devices, die sich an einen solchen Elternknoten anmelden, bekommen aus diesem Adressblock eine Adresse zugewiesen. Dabei wird die maximale Anzahl an Kindknoten beim Erstellen des Netzwerks durch den ZigBee Coordinator festgelegt.

Bei der default distributed address allocation kommen vier Parameter für die Adressallokation und

das Routing zum Einsatz:

- nwkMaxDepth: Die maximale Anzahl an Ebenen im Netzbaum (mD).
- nwkMaxChildren: Die maximale Anzahl an Kindern die durch einen Baumknoten adressiert werden können (mC).
- nwkMaxRouters: Die obere Grenze an Kindknoten die selbst Router im Baum sein können (mR).
- depth: die Tiefe eines Knotens im Netzwerkbaum (d).

Das Adressierungsverfahren startet mit der Zuweisung der Adresse 0 für den ZigBee Coordinator. Alle anderen Adressen können durch eine einfache Funktion Cskip(d) berechnet werden:

$$Cskip(d) := \begin{cases} 1 + mC \times (mD - d - 1) & , if\ mR = 1 \\ \dfrac{1 + mC - mD - mR - (mC \times mR(d))}{(1 - mR)} & , otherwise \end{cases}$$

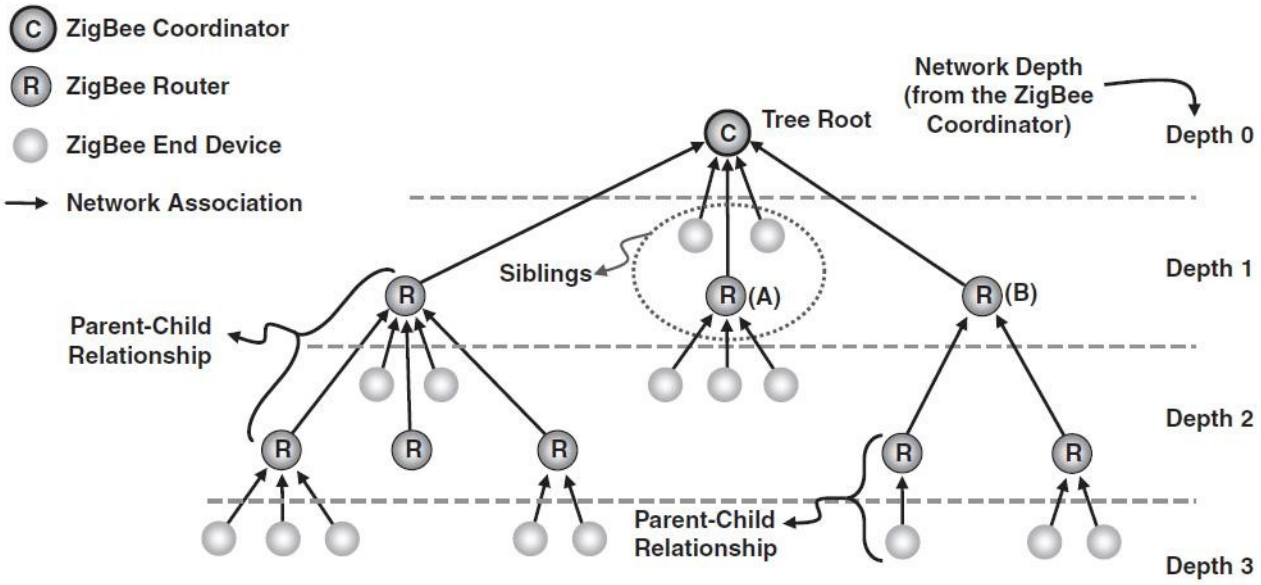

Abbildung 10: Schema eines Hierarchical-Tree-Routing

Wird der berechnete Wert von Cskip(d) 0, so bedeutet dies, dass dieses Device keine Kinderknoten akzeptieren und adressieren darf.

Die Adresse für ein ZigBee Enddevice n auf der Ebene d wird anders berechnet.

$$Address^{n}_{Child} = Address_{Parent} + (Cskip(d) \times mR)$$

Cskip(d) wird daher auch für das Routing benutzt. Möchte ein Device wissen ob die empfangene weiterzuleitende Message mit Destination Address D innerhalb seines Adressblocks liegt, führt es einen einfachen Vergleich durch.

$$Address_{relaying\ Device} < Destination < Address_{relaying\ Device} + Cskip(d-1)$$

Um die Adresse für den nächsten Hop zu berechnen, benötigt man daher:

$$A_{next\ Hop} = A_{relaying\ Device} + 1 + rounddown\left(\frac{A_{Destination} - (A_{relaying\ Device} + 1)}{Cskip}(d)\right) \times Cskip(d)$$

In einer hierarchischen Topologie wird die Message nur unter einer von zwei zutreffenden Bedingungen an die Kindknoten weitergeleitet.

- Die Message wurde von einem Kindknoten empfangen und die Destination Address ist wiederum ein Nachfolger des Routers.
- Die Message wurde von einem Elternknoten empfangen und die Destination Address ist ein Nachfolger des Routers.

Ansonst wird die Nachricht an den Elternknoten die Baumstruktur hochgereicht. Die Sterntopologie ist ein Sonderfall des eben besprochenen Addressierungsmechanismus.

Parameter: nwkMaxDepth = 1; nwkMaxRouters = 0; nwkMaxChildren = MaxCoordinatorChildren;

Dieser Addressierungs und Routingalgorithmus wird in ZigBee 2006 und ZigBee 2007 verwendet. ZigBee Pro unterstützt weiters noch das Stochastic Adressing.

2.2.2 Mesh-Topology

Bei der Mesh-Topology existieren keine hierarchischen Strukturen, stattdessen ist es jedem Device erlaubt, mit jedem anderen Device im ZigBee Netzwerk Verbindungen aufzubauen, entweder direkt oder über ein Routing Device basierend auf dem Message Originator. In vermaschten Netzen wird das Routing nur auf Basis der Source und Destination Address durchgeführt. Kann eine bereits etablierte Route nicht mehr benutzt werden, so können die ZigBee Router miteinander kooperieren, um ein neues Routing zu finden (siehe Abschnitt Route Discovery and Maintenance).

2.2.3 LQI – Link Quality Information

Der LQI wird von der IEEE 802.15.4 MAC Schicht für jedes empfangenes Datagramm bestimmt. In ihn gehen Signalstärke des empfangenen Signals, CRC Fehlerrate, sowie Anzahl der Retries beim Transmitting ein. Der LQI ist damit eine Repräsentation der SNR (*Signal-to-Noise Ratio*).

2.3 Routing

2.3.1 Routing

In ZigBee, wie in anderen Protokollen, basiert das Routing auf bestimmten Annahmen und ständig generierten Route-spezifischen Parametern. Diese Parameter sind *link quality*, *number of hops* und *energy conservation considerations*.

Diese Parameter fließen in einen Link Cost Parameter ein, der die Wahrscheinlichkeit einer erfolgreichen Übertragung repräsentiert. Je geringer die Wahrscheinlichkeit einer erfolgreichen Übertragung desto höher ist der Wert von Link Cost.

$$C(l) = MIN\left(7\ ,\ RoundDown\left(\frac{1}{P^4}\right)\right)$$

Der Parameter P wird üblicherweise über den Durchschnittswert des LQI für diesen Link bestimmt.

Verglichen mit verschiedenen Pfaden, welche aus Ketten von Links bestehen, kann für jeden Pfad ein Parameter Path Cost definiert werden, welcher die Summe all seiner Linkkosten ist.

Der Pfad mit den geringsten Kosten besitzt die höchste Wahrscheinlichkeit zur erfolgreichen

Übertragung.

Wird eine valider minimaler Pfad durch das Netzwerk gefunden, wird dieser in den Routing Table der ZigBee Router und Coordinators eingetragen. Dabei dient die Destinationadresse als Schlüssel für das Finden des nächsten Hops.

Ein weiterer Datensatz, der von FFDs gewartet werden muss, ist die Route Discovery Table. Sie enthält unter anderem die Kosten vom Source Device der Route zum Relaying Device und vom Relaying Device zum Destination Device. Der Inhalt der Route Discovery Table ist nicht persistent und wird nach einer gewissen voreingestellten Zeit gelöscht.

Jedes Device unterhält auch noch eine Neighbour Table, der bei jedem empfangenen Datagramm upgedated wird und alle Devices in Reichweite beinhalten soll. Die Neighbour Table wird bei einem Rejoin eines Devices und beim Suchen nach einem Parent-Device benutzt.

Die APL kann die Service-Primitive NLME-ROUTE-DISCOVERY-Request verwenden, um die Netzschicht anzuweisen, neue Routen für eine Unicast, Multicast oder Many-to-One Verbindung zu suchen.

2.3.2 Route Discovery

Ohne genauer auf die MAC Schicht IEEE 802.15.4 eingegangen zu sein, kann nur ein sehr rudimentärer Überblick über das Route Discovery in ZigBee gewährt werden, da in diesem besonderen Fall die MAC Schicht und die ZigBee NWK Schicht besonders eng miteinander kooperieren und die MAC Schicht bereits etliche Problemstellungen oder etwaige auftretende Komplikationen ausblenden kann (Bsp.: Hidden Node Problem, Broadcast Sequencing).

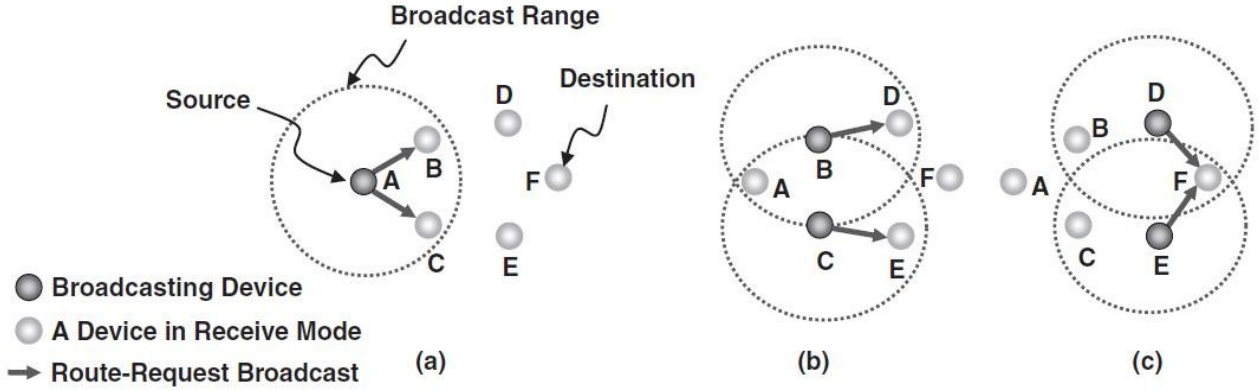

Abbildung 11: Beispiel eines Route Discovery

Node A setzt einen Route Discovery Broadcast an alle Devices in Hörweite ab und wartet darauf, dass in diesem Falle Node B und Node C die Message weiterreichen. Hierbei wird kein Acknowledge von Knoten B und C an Knoten A gerichtet. Da Knoten A in Reichweite von Knoten B und C sein muss, reicht es, dass Knoten A den weitergeleiteten Route Discovery Broadcast mit derselben Destination Address von Knoten B und C empfängt. Diesen Vorgang nennt man *passive acknowledgement*.

Die Nodes B und C prüfen, ob sie noch freie Routing Kapazität haben. Falls dies der Fall ist legen sie einen neuen Eintrag in der Routing Table an und tragen gleichzeitig die Path Cost basierend auf dem LQI, bestimmt durch die MAC, in das path-cost field des routing request ein und broadcasten das Ergebnis.

Node D könnte nun mehrere Routing Request Broadcasts empfangen, die dieselbe Destination Address aufweisen. In diesem Fall bestimmt Device D die Path-Cost aller eingehenden möglichen Requests und updated seinen Routing Table mit dem Pfad der die geringsten Path-Costs aufweist.

Dies ist für den Zeitpunkt der Rückmeldung des Routing Ergebnisses an den Source Node erforderlich.

Empfängt nun der Destination Node einen Routing Request Broadcast, wartet er noch eine Zeit lang, bis er alle möglichen Routing Request Broadcasts empfangen hat. Aus allen möglichen Routen wählt er dann jenen Pfad mit den geringsten Path-Costs aus und sendet auf diesem das Route-Reply Command zurück an den Source Node.

Die Route von Source zu Destination Address nennt man Forward Route, in umgekehrter Richtung daher Backward Route. Routing kann auf zwei Arten durchgeführt werden. Im Falle symmetrischen Routinge entspricht die Backward Route genau der Forward Route. Im anderen Fall wird nun erneut ein Route-Request Command als Broadcast abgesetzt, um eine Route von Destination zu Source zu finden.

Multicast Route Discovery funktioniert aus der NWK Sicht sehr ähnlich. Der Source Knoten setzt einen Route Request Broadcast ab, jedes ihn empfangende Device mit freien Routing Ressourcen gleicht nun seinen eigenen Multicast Group Table mit der Destination Address des Requests ab, um zu bestimmen, ob es selbst Mitglied der Gruppe ist. Ist es Mitglied der Multicastgruppe und existiert noch keine Gruppe, so setzt das empfangende Device einen Route Reply ab. Ist im Routing Table bereits ein Eintrag vorhanden mit der selben Destination Address und dem gleichen Route Request Identifier, so gleicht es die Routen anhand ihrer Pfadkosten ab und behält die optimale. War die optimale Route bereits bekannt, wird ein Route Reply Kommando an den Source Node gesendet. Ist die optimale Route die neue, wird wiederum ein Route Reply Kommando gesendet.

2.3.3 Source Routing

ZigBee NWK sieht Source Routing vor. Dabei erzeugt der Originator der Nachricht eine Liste aller Knoten innerhalb des Netzes ,die zu passieren sind, welche in den NWK Header geschrieben wird. Weiters wird noch ein Indexpointer mitgesandt, der immer auf den nächsten Knoten zeigt, somit muss der nachfolgende Relaying Node nicht seine eigene Lookup Table auswerten, sondern kann direkt den nächsten Node aus der Liste im NWK Header bestimmen. Dies lässt theoretisch Routing Devices zu, die selbst keine Routing Table warten können.

2.3.4 Route Maintenance and Repair

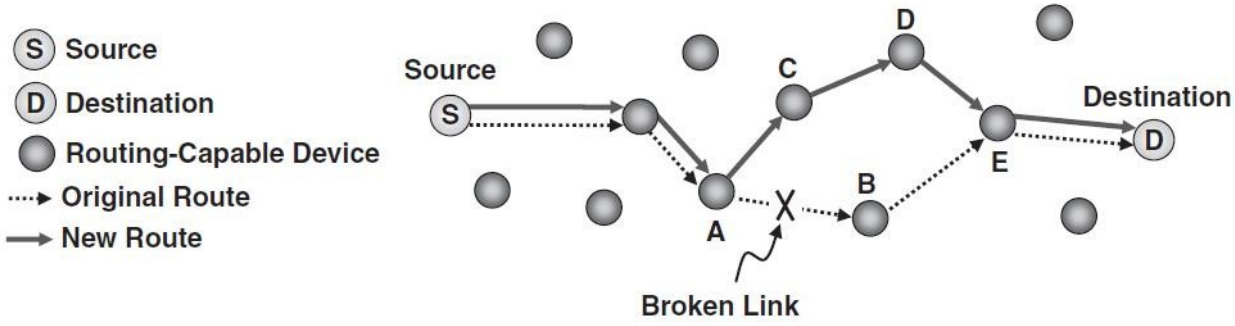

Abbildung 12: Beispiel eines Route Repair

Im Falle eines Ausfalls eines Link auf der bereits etablierten Route wird versucht die ursprünglich hergestellte Route zu reparieren. Wie die Grafik zeigt ist der Link zwischen A und B ausgefallen. Device A merkt aufgrund des Ausbleibens des passive Acknowledgement, dass Node B die Message nicht weitergereicht hat. In diesem Fall versucht nun Node A eine neue Route zum Destination Device D zu finden und setzt ein Route Request Command per Broadcast ab.

Wird eine neue Route von A nach D gefunden, wird das Routing von Nachrichten von S nach D nun über das neue Teilstück abgewickelt. Kann keine neue Route gefunden werden, wird an das Source Device S eine Route Error Nachricht gesandt, welche den Failure des Routings und des Route Repairs indiziert.

Um Route Requests zum initialen Erstellen einer Route und zum Reparieren einer Route unterscheiden zu können, wird im NWK Header ein Flag gesetzt.

2.3.5 NWK Layer Data Service

Die Application Layer übergibt dem Network Layer Daten zum Senden über die NLDE-DATA.Request Primitive, welche einen Transfer initiiert. Um einen Datentransfer zu starten muss die NWK Schicht noch mit den Parametern Radius, Destination und Originator ID versorgt werden., um die Reihenfolge der übertragenen Daten zu gewährleisten, wird jedes übertragene Datensegment mit einer Random Number signiert, welche beim Übertragen des nächsten Segments inkrementiert wird.

Erhält der NWK Layer von der MAC Schicht Daten, um sie an den APS Sublayer weiterzuleiten, geschieht dies über die NLDE-Data.Indication Primitive, welche zusätzlich noch Parameter wie LQI und Data Sequence Number übergibt.

3 APL Layer

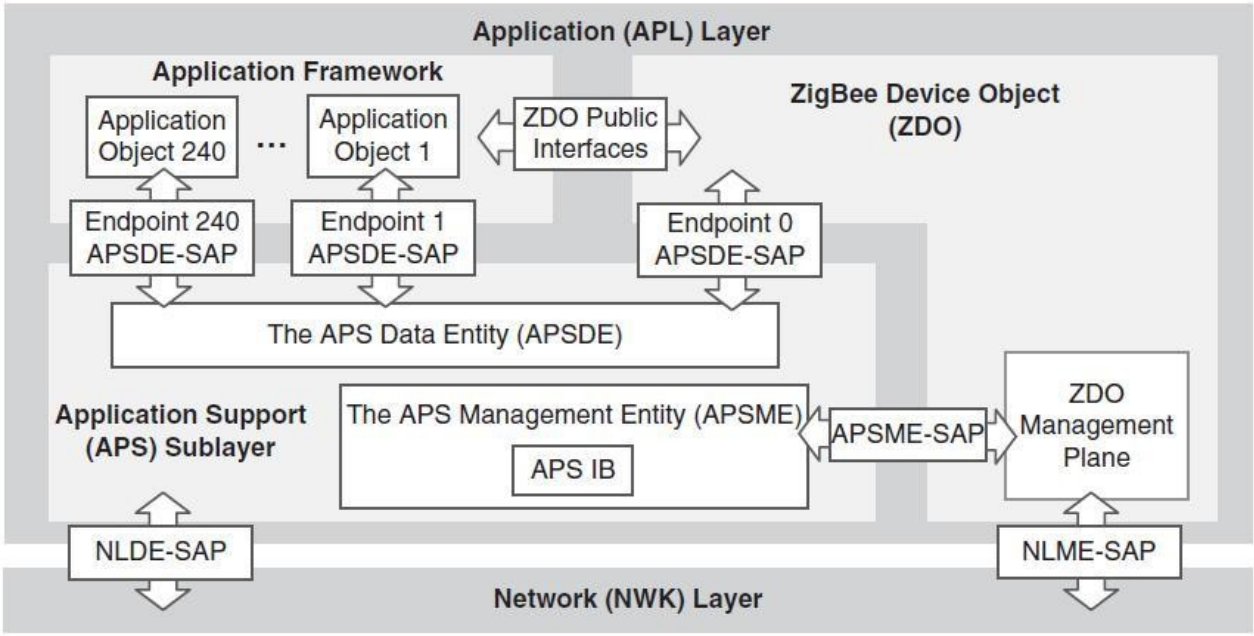

Abbildung 13: Schematischer Aufbau der Application Protocol Layer

Der ZigBee Application Layer (APL) ist die oberste Schicht des ZigBee Standards mit der höchsten Ebene an Abstraktion. Er selbst besteht aus drei Sublayern:

- Application Framework
- Application Support Sublayer (APS)
- ZigBee device object (ZDO)

3.1 Application Framework

Mit dem ZigBee Standard wurden *application profiles* eingeführt. Wenn ein ZigBee Device mehrere *application objects* hostet, dann sollten diese möglichst effizient und fehlerfrei miteinander kommunizieren und arbeiten können. Um dies zu gewährleisten, mussten Meta-Standards geschaffen werden, die a*pplication profiles* oder *ZigBee profiles*. Jeder data request, sei er ausgehend oder eingehend wird über ein application profile abgewickelt.

Jedes application profile wird über einen 16-Bit breiten *profile identifier* referenziert, welcher durch die ZigBee Alliance vergeben wird. *public profiles* haben einen identifier im Bereich 0x0001 bis 0x7fff, *manufacturer profiles* im Bereich 0xbf00 bis 0xffff.

Es bestehen keine Einschränkungen bezüglich der Anzahl oder Art an application profiles innerhalb eines Netzes oder eines Knotens. Eine besondere Rolle nehmen die public profiles innerhalb des Standards ein. Sie sind so definiert, dass Produkte verschiedener Hersteller out-of-the-box miteinander kommunizieren können.

Ein application profile besteht wiederum aus mehreren Datenstrukturen zweier Typen:

- cluster descriptions
- device descriptions

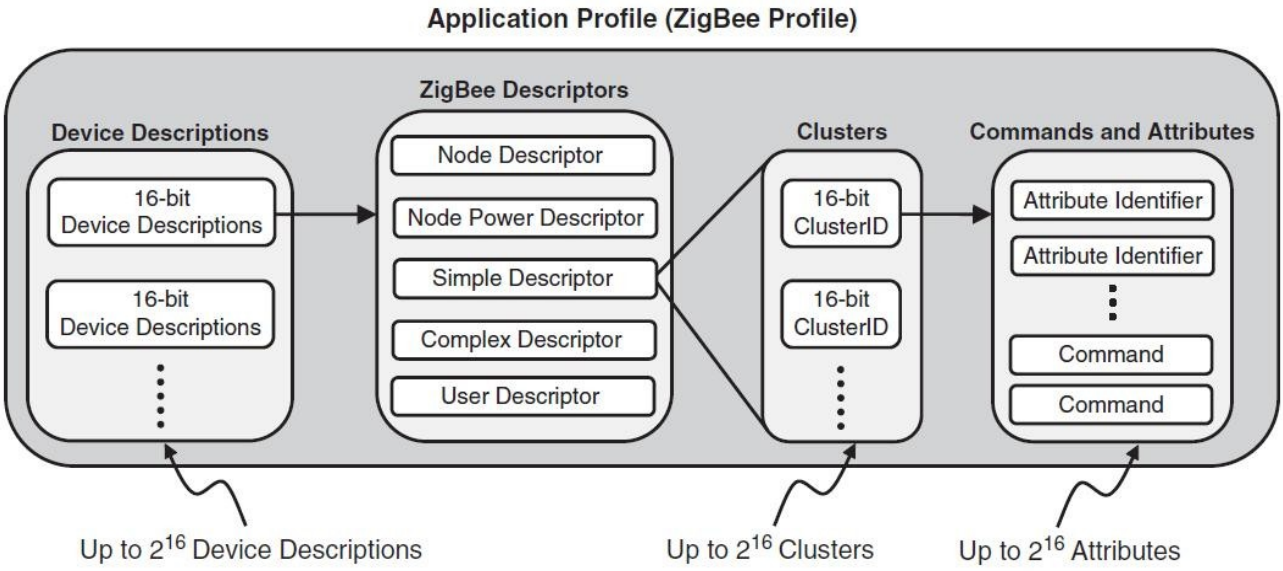

Abbildung 14: Aufbau von Application Profiles

3.1.1 Cluster

Während network address und endpoints Adressierungskonzepte sind, haben cluster nur applikationsspezifische Bedeutung. Die Idee des cluster ist in ZigBee jedoch kein Set von application objects sondern nur ein Set von Variablen die zu einem cluster zusammengefasst werden. Cluster besitzen eine eindeutige 16 Bit cluster ID, und bestehen wiederum aus *attributes,* welche ebenfalls einen 16 Bit *attribute identifier* aufweisen. Clusters und attributes werden dazu benutzt, um applikationsspezifische Informationen zu sammeln und zu präsentieren.

Wie aus Abbildung 14 leicht ersichtlich, sind cluster stark durch ihren vom application profile definierten Kontext abhängig. Application profiles sind keine großen Datenstrukturen, die selbst Cluster-Instanzen besitzen, sondern listen nur cluster identifiers auf, die benutzt werden. Beim Kommissionieren ist es daher unablässig, eine detaillierte Wissensbasis über die eingesetzten application profiles und die darin verwendeten cluster aufzubauen.

Cluster beinhalten jedoch nicht nur data (attributes) sondern auch code (commands).

3.1.2 Commands

public profiles benutzen die ZigBee Cluster Library (ZCL), welche zwei neue Konzepte in den eigentlich ZigBee Standard einführt. *Commands* und *Attributes*. Die ZCL macht es sehr einfach Attribute eines clusters über commands zu manipulieren, wobei diese commands *cluster-specific* oder *cross-cluster* Einfluss nehmen können.

Commands sind standardisierte Frames, die von der APS oder vom ZDO versendet werden, auf welche im Zielobjekt reagiert wird. Welche Funktion bei der Reaktion aufgerufen wird, hängt vom Kontext des adressierten Clusters ab. Dies können cluster der ZDO, der APS oder eines application

profiles sein. Dementsprechend wird die Reaktion über das ZigBee Device Profile oder ein application profile festgelegt.

3.1.3 Device Descriptions

Jeder endpoint beinhaltet einen profile identifier und auch einen device identifier. Somit kann der ZigBee node seiner Umgebung, anzeigen welche Geräte an ihn angeschlossen sind. Prinzipiell sind Device ID's zwei Zwecken dienlich:

- die Anzeige einer Erklärung oder Icons auf einem Human-readable Display
- device identifier erlauben es ZigBee-Kommissionierungswerkzeugen intelligenter zu agieren.

Diese device identfier referenzieren abhängig vom application profile device descriptions, welche die eigentliche Beschreibung der Geräte beinhalten (unterstütze Frequenzkanäle, Gerätetyp (Coordinator, Enddevice), Batteriekapazität). Tabelle 1 zeigt einen kurzen Auszug der vom Home Automation Profile unterstützten Geräte.

Name	Identifier	Name	Identfier
Range Extender	0x0008	Light Sensor	0x0106
Mains Power Outlet	0x0009	Shade	0x0200
On/Off Light	0x0100	Shade Controller	0x0201
Dimmable Light	0x0101	Heating/Cooling Unit	0x0300
On/Off Light Switch	0x0103	Thermostat	0x0301
Dimmer Switch	0x0104	Termperatur Senor	0x0302

Tabelle 1: Device IDs verfügbar im Home Automation Profile

Jede device description wird in einer fünfteiligen descriptor data structure abgespeichert:

1. Node Descriptor: Logischer Typ des Device und Manufactorer Code
2. Node Power Descriptor: „main supply“ oder „battery powered“ und Angabe der verbleibenden Restkapazität der Batterie.
3. Simple Descriptor: Application Profile Identifier und Cluster
4. Complex Descriptor: optional, und beinhaltet Informationen wie Seriennummer und Modelname
5. User Descriptor: optional, 16 ASCII Zeichen

3.1.4 Node Descriptor

Abbildung 15 zeigt die einzelnen Felder des node descriptors, deren Bezeichnungen selbsterklärend sind. Das APS Flag kennzeichnet die Version und unterstützte Funktionalität der verwendeten APS.

Field Name	Length (Bits)
Logical type	3
Complex descriptor available	1
User descriptor available	1
Reserved	3
APS flag	3
Frequency band	5
MAC capacity flags	8
Manufacturer code	16
Maximum buffer size	8
Maximum transfer size	16
Server mask	16

Abbildung 15: Darstellung eines Node Descriptors

In einem ZigBee Netzwerk können Geräte auch spezielle Funktionalität für das ZigBee Netzwerk bereitstellen. Diese Services werden durch Einträge in der server mask des node descriptors beschrieben.

- **primary/backup trust center:** Jener node, welcher von allen Knoten im ZigBee Netzwerk als vertrauenswürdig angesehen wird. Er übernimmt Verteilung, Austausch und Authentifikation der Security-Keys, welche benutzt werden, um eine sichere Kommunikation im Netzwerk aufzubauen.
- **primary/backup binding table cache:** Jener Knoten, der es allen anderen nodes erlaubt, ihre binding tables auf ihn selbst auszulagern.
- **primary/backup discovery cache:** ist ein ZigBee Coordinator oder Router, der die node descriptoren anderer Geräte cached. Enthält ein Netzwerk Geräte, welche in sleeping mode wechseln können, ist ein solcher primary discovery cache vorgeschrieben. Möchte ein device Informationen über einen sleeping node abfragen, kann der primary discovery cache antworten beziehungsweise es veranlassen, dass der gewünschte node seine normale Funktion wieder aufnimmt.

3.2 Application Support Sublayer (APS)

Der APS stellt notwendige Daten-Schnittstellen zwischen NWK Layer und Application Framework zur Verfügung. So wie in den bereits darunter liegenden Schichten existieren auch hier zwei Arten von Schnittstellen:

- APSDE: Application Sublayer Data Entity
- APSME: Application Sublayer Management Enity

In den Aufgabenbereich des APS fallen folgende Aktivitäten:

- Herausfiltern von Frames an nicht registrierte Endpoints

- Herausfiltern von Paketen an endpoints, deren Profile ein anderes ist, als im Paket angegeben.
- Generieren von *end-to-end acknowledgement with retries* und als Folge davon das Herausfiltern von duplizierten Paketen. Abbildung 16 zeigt ein Beispiel einer gestörten Transmission.
- Wartung der lokalen *binding table*
- Wartung der lokalen *group table (APS group table)*
- Wartung der lokalen *address map*

Der application support sublayer enthält alle wichtigen *application-level tables*. Diese können aber nicht direkt von application objects aus dem application framework manipuliert werden, sondern können nur über das spezielle ZigBee Device Object (ZDO) und seine Anbindung an die APS-Management Entity Schnittstelle verändert werden. In der sogenannten APS IB (APS Information Base) werden die Informationen der *address map*, *APS group table* und des *binding table* abgelegt.

Binding ist das Verbinden eines endpoints mit einem oder mehreren endpoints auf anderen ZigBee nodes. Mit groups auf APS Ebene meint man eine beliebige Menge an application objects, die auf (nicht notwendigerweise) verschiedene nodes verteilt sind. Die address map assoziiert die 64bit IEEE oder MAC-Adressen mit den 16bit NWK-Adressen.

Zusammen formen APS und AF das ZigBee Interface, welches den application objects zur Verfügung gestellt wird. Die darunter liegenden Schichten können nicht direkt angesprochen werden, sondern werden über das ZDO und den APS beeinflusst.

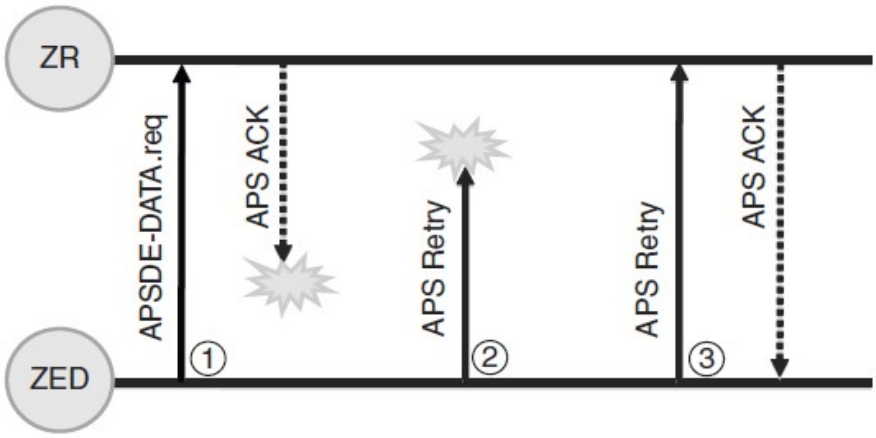

Abbildung 16: Selbstständiges end-to-end Acknowledgement der APS Layer

Der APS Sublayer bietet sowohl den application objects im application framework, also auch dem ZigBee Device Object seine Funktionalität an. Die APS data entity übernimmt eine PDU und fügt dann den APSDE Header hinzu und übergibt das daraus entstandene Paket an den NWK layer.

Die APS bietet über die APSME (APS Management entity) auch Managementfunktionen dem ZigBee Device Object an.

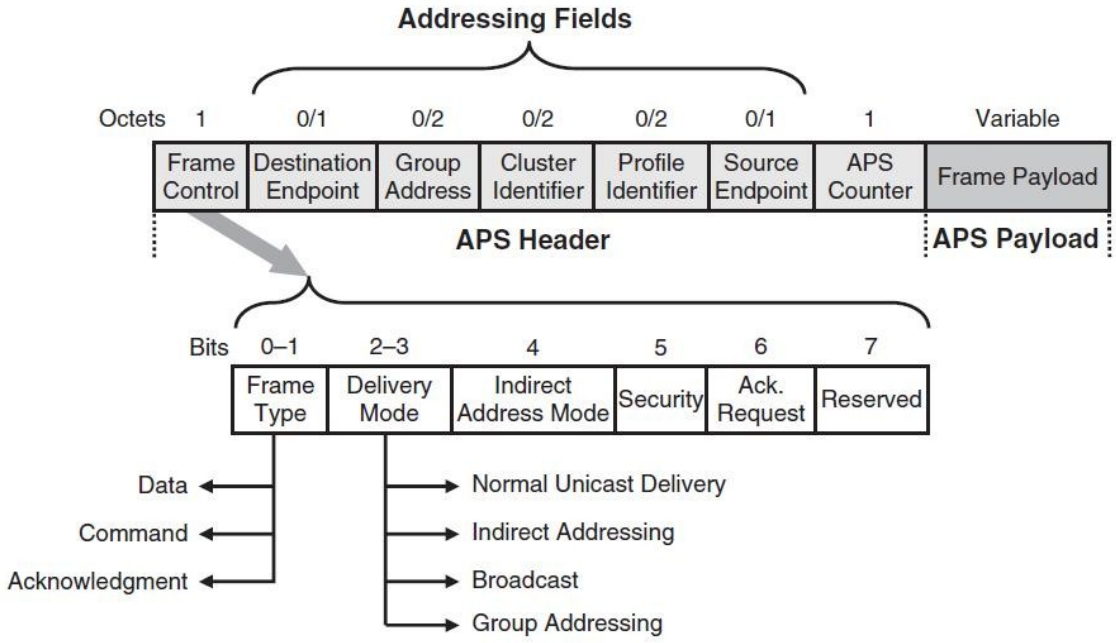

Abbildung 17: Das APS Header Format

Die APSME bietet Funktionen für bind und group management sowie für die AIB (APS Information Base) an.

Neben den bereits bekannten Unicast, Broadcast und Multicast Mechanismen, bietet die APS Sublayer noch eine zusätzlich Art der Nachrichten Übertragung. Beim *indirect addressing* muss ein Knoten mit limitierten Ressourcen nicht über die Adresse des Zielnodes Bescheid wissen. *Indirect transmissions* werden statt dessen zum ZigBee Coordinator geschickt, welcher eine *binding table* hostet. Aufgrund der Quelladresse der Nachricht, der *endpoint adress* und der *cluster ID,* kann der ZigBee Coordinator die daran gebundene(n) Adresse(n) herausfinden und die Nachricht weiterleiten.

Abbildung 17 zeigt die allgemeine Struktur eines solchen APS frame, im ZigBee 2006 Standard. Der ZigBee Pro Standard sieht noch ein weiteres optionales Feld namens *extended header* vor.

Man unterscheidet drei Typen von APS frames: *data, command, acknowledgment* , wobei der *APS frame type* vom *frametype* Feld im *frame control byte* bestimmt wird. Die Bits von *delivery mode* kontrollieren die von der APS angeforderte Übertragungsart. Das *security* Subfeld wird vom Security Service Provider (SSP) geschrieben.

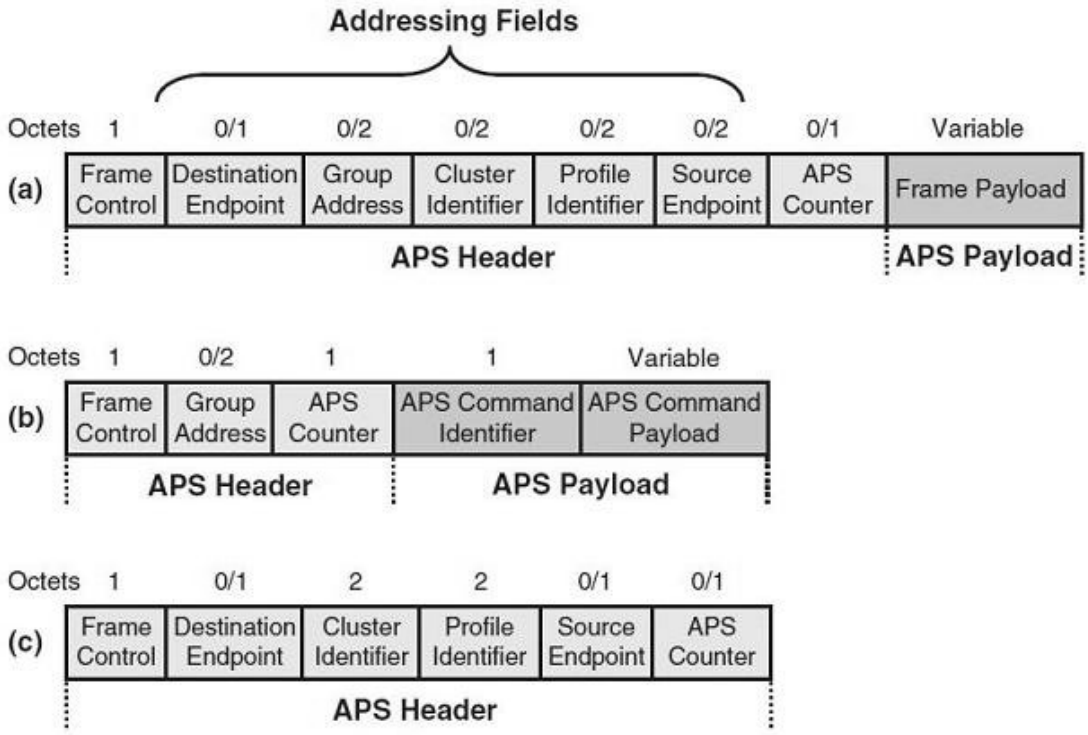

Abbildung 18: APS Layer: a) Data Frame b) Command Frame c) Acknowledgment Frame Format

Wird ein frame im *indirect adressing mode* übertragen, wird das *indirect address mode field* auf 1 gesetzt, wenn der Frame vom Quellknoten zum ZigBee Coordinator übertragen wird. Außerdem wird das *destination address field* nicht befüllt, da die Zieladresse erst bestimmt wird. Im ZigBee Coordinator wird die Zieladresse aufgelöst und das *indirect address mode field* auf 0 gesetzt. Dies kennzeichnet nun, dass der Frame vom ZigBee Coordinator zum Zieldevice übertragen wird. Gleichzeitig wird die Quelladresse unterdrückt, da diese beim Acknowledge erst wieder vom Coordinator bestimmt werden muss (Die Anwort muss nicht immer an den gleichen endpoint geschickt werden, wenn überhaupt eine geschickt wird).

Ist das *group adress field* gesetzt, so werden alle endpoints der Multicastgruppe angesprochen, logischerweise können daher niemals *group adress field* und *destination endpoint field* gleichzeitig gesetzt sein. Beim Verteilen einer Nachricht an eine APS group wird zusätzlich zur group ID auch noch die profile ID des endpoints geprüft. Damit ist es möglich, dass selbst in einer APS group mehrere application profiles existieren.

Das *cluster identifier field* ist nur in einer *binding operation* gesetzt. Der ein Byte große APS counter wird bei jedem neuen APS frame, um eins inkrementiert und hilft dabei Duplikate zu erkennen und zu ignorieren.

3.2.1 endpoints

Abbildung 13 zeigt die *endpoints* zwischen APS Sublayer und Application Framework. Diese endpoints sind einfache Datenstrukturen vom Typ *simple descriptor,* mit Hilfe derer man den Kontext verschiedener application objects beschreibt. Die untere Tabelle zeigt den Aufbau des simple descriptor, der auch beim binding eine wesentliche Rolle einnimmt.

```
typedef struct zbSimpleDescriptor_t {
    zbEndPoint_t   EndPoint;
    zbProfileId_t  aAppProfId;
    zbDeviceId_t   aAppDeviceId;
    uint8_t        appDevVerAndFlag;
    zbCounter_t    appNumInClusters;
    uint8_t        *pAppInClusterList;
    zbCount_t      appNumOutClusters;
    uint8_t        *pAppOutClusterList;
} zbSimpleDescriptor_t;
```

Es ist möglich bis zu 240 unterschiedliche application objects in einem einzigen ZigBee node zu hosten, diese werden dann auf die endpoints 1 bis 240 abgebildet. Es existieren noch andere endpoints, diese sind jedoch für spezielle Zwecke reserviert. Zum Beispiel referenziert die endpoint address 0 das ZigBee Device Object (ZDO) des nodes und endpoint address 255 ist ein Broadcast an alle application objects innerhalb des Devices.

Wie bereits erwähnt, lässt sich mit Hilfe der endpoints und der Datenstruktur simple descriptor der Kontext eines application objects kontrollieren. Im Detail bedeutet dies, dass es endpoints erlauben, dass innerhalb eines ZigBee Knotens mehrere application profiles existieren. Da mit den endpoints unterschiedliche Kontrollpunkte existieren, können auch unterschiedliche Geräte an den ZigBee node angeschlossen werden.

Die application objects bedienen sich eines APSDE-SAP (Service Access Point), um Daten mit anderen Schichten oder Objekten im Device oder auf einem anderen Device auszutauschen.

3.2.2 ZigBee Device Object (ZDO)

Das ZigBee device object (ZDO) ist ein application object, welches am für ihn reservierten endpoint 0 registriert ist. Es stellt ein Interface zwischen Application Framework, APS und NWK Layer dar, überwacht den Zustand des ZigBee nodes und des Netzwerks und übernimmt die Initialisierung der APS und NWK Schicht und des SSP (Security Service Providers) . Zum Beispiel übernimmt es die „applikationsseitige“ Initialisierung des Knotens, dies bedeutet auch einen besonderen Einfluss beim Formen des ZigBee Netzwerks (Welche Rolle soll der ZigBee node innerhalb des Netzwerks spielen: ZigBee Coordinator, ZigBee Router, ZigBee Enddevice, Cache, Data Concentrator?).

Das ZDO verwendet selbst die von der APS Management Entity zur Verfügung gestellten Primitive des APSME-SAP und bietet dem Application Framework das ZDO public Interface an.

Für das ZDO gibt es ein spezielles application profile, das ZigBee Device Profile (ZDP). Es hat die vom Standard festgelegte profile id 0x0000 und bietet die Funktionen für das Suchen, die Konfiguration und die Wartung von ZigBee Geräten und Services über das ZigBee Netzwerk an.

Das ZDP besteht wiederum aus cluster und device descriptions. Der eigentliche Zweck des ZigBee Device Profile liegt aber im Support von Device Discovery, Service Discovery und Binding Management.

Möchte ein Device von einem anderen Device applikationsspezifische oder devicespezifische Informationen abfragen, so veranlasst es das ZDO zu einem Service Discovery (Node_Desc_Req), welcher vom Zieldevice detaillierte Informationen wie profile identifier, ZigBee Descriptoren oder

die Listen der Input und Output Cluster erfragt. Diese Clusterlisten sind dann für ein darauf folgendes Binding notwendig.

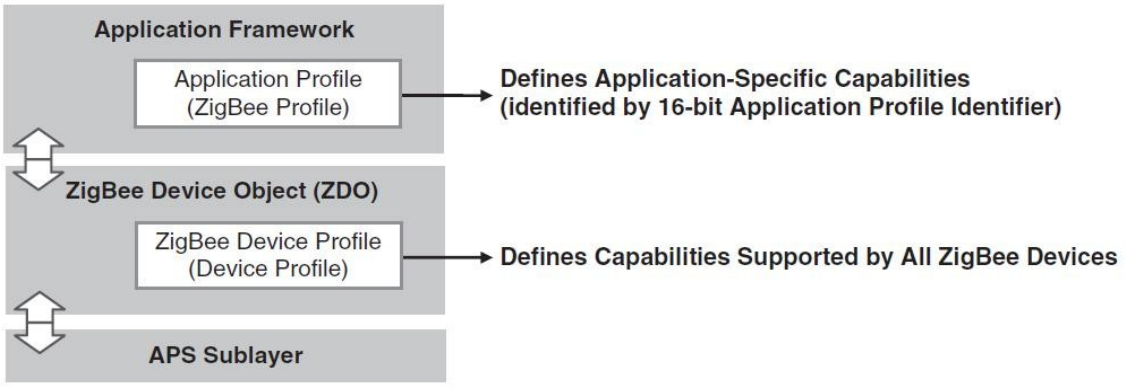

Abbildung 19: ZDO fungiert als Interface zwischen APS und Application Framework

ZigBee Device Profile Kommandos werden über das APS Data Service in einem speziellen Format übergeben. An den Kopf der Daten wird immer eine *transaction sequence number* gestellt, da NWK und MAC Schicht nicht die Reihenfolge der Kommandonachrichten garantieren kann. Der erste Teil ist daher eine 8 Bit lange *transaction sequence number* gefolgt vom *transaction data field*, das die eigentlichen Kommandos und Parameter beinhaltet.

Jedes application object hat einen eigenen counter welcher in das *transaction number field* kopiert wird. Somit kann die Reihenfolge der Kommandos überwacht und eingehalten werden.

Weiters unterscheidet man die ZDP commandos in zwei Arten: client und server services, welche wiederum in vier Gruppen unterteilt werden können.

- device and service discovery
- bind Management
- network Management
- security Management

3.2.3 Device Discovery Services des ZDP

Das ZDP definiert commands für den Austausch von wesentlichen Informationen zum Aufbau und Betrieb eines ZigBee Netzwerks unter den beteiligten nodes. Tabelle 2 zeigt alle im ZDP definierten commands für das device discovery. Zu beachten ist, dass im Standard als optional definierte services, abhängig von der Rolle des nodes nicht unterstützt werden müssen.

Device Discovery Services	Unicast / Broadcast	Client Transmission (Request)	Server Processing (Response)
NWK_addr_req	U;B	O	M
IEEE_addr_req	U	O	M
Node_Desc_req	U	O	M
Power_Desc_req	U	O	M
Complex_Desc_req	U	O	O
User_Desc_req	U	O	O
User_Desc_set	U	O	O
Device_annce	B	O	M

Tabelle 2: Device Discovery Services des ZDP
(O)...optional, (M)...mandatory

Der wohl wichtigste Service ist das *device announce command*. Device announce kann nicht durch ein application object hervorgerufen werden sondern wird nur durch den ZigBee Stack generiert. Ein abgesetzter device announce veranlasst alle am ZigBee Netzwerk beteiligten Knoten, die in irgendeiner Form vom ausstellenden node abhängig sind, ihre internen Tabellen zu aktualisieren. Davon sind zumindest folgende 3 Tabellen betroffen:

- address map
- neighbour table
- binding table

Daraus ist ersichtlich, dass ein device announce Auswirkungen auf alle Ebenen des ZigBee Stack hat.

Ein device announce wird vom node abgesetzt, wenn dieser den Kontakt zu seinem Elternknoten verloren hat, oder wenn er zum ersten mal Kontakt zum Netzwerk aufzubauen will. Aber auch wenn ein Kindknoten seinen Elternknoten anweisen will, dass er alle requests puffern soll, so lang sich der Kindknoten im sleep mode befindet.

3.2.4 Service Discovery des ZDP

Dreh- und Angelpunkt beim Kommissionieren ist es zu wissen, welche Services auf welchen Knoten zur Verfügung stehen. Aufgrund dieser zentralen Rolle wurden auch diese commands in das ZDP aufgenommen, jedoch sind nur einige davon für alle ZigBee nodes verpflichtend.

Service Discovery Services	Unicast / Broadcast	Client Transmission (Request)	Server Processing (Response)
Simple_Desc_req	U	O	M
Extended_Simple_Desc_req	U	O	O
Active_EP_req	U	O	M
Extended_Active_EP_req	U	O	O
Match_Desc_req	B	O	M
System_Server_Discover_req	B	O	O
Find_node_cache_req	B	O	O
Discovery_Cache_req	U	O	O
Discovery_store_req	U	O	O
Node_Desc_store_req	U	O	O
Power_Desc_store_req	U	O	O
Active_EP_store_req	U	O	O
Simple_Desc_store_req	U	O	O
Remove_node_cache_req	U	O	O

Tabelle 3: Service Discovery Services des ZDP

Das ZDO kann über einen Active Endpoint Request alle in einem ZigBee node aktiven endpoints abfragen. Im Anschluss wird mit dem Simple Descriptor Request der simple descriptor aller endpoints von Interesse abgeholt. Damit stehen dem application object alle notwendigen Informationen zur Verfügung, um ein Binding durchzuführen.
Der Match Descriptor Request (Match_Desc_req) ist ein Service, um innerhalb des Netzwerks alle passenden endpoints zu finden. Als Input wird dem Match_Desc_req ein simple descriptor eines endpoints mitgegeben. Ein den Request emfpangender Knoten vergleicht daraufhin alle auf ihm befindlichen enpoints, und schickt dem Sender eine Liste der passenden endpoints zurück.

3.3 Binding

Unter binding versteht man dass Erstellen von unidirektionalen logischen Verbindungen zwischen zwei endpoints des Netzwerks. z.B.: Ein application object eines ZigBee nodes, welches einen Schalter überwacht, muss an das application object, welches eine Lampe steuert, gebunden werden (umgekehrt jedoch nicht). Die Zuordnungen dieser logical links werden in der lokal gespeicherten *binding table* abgelegt, welche durch die APS gewartet wird. Jeder Eintrag im binding table speichert den source endpoint, die destination address, den destination endpoint oder die destination group sowie eine zugehörige cluster ID.

Src EP	Destination Addr	Addr/Grp	Dst EP	Cluster ID
5	0x1234	A	12	0x0006
6	0x79F6	A	240	0x0006
5	0x9999	G	-	0x0006
5	0x5678	A	44	0x0006

Tabelle 4: Vereinfachtes Beispiel einer binding table

Lokales binding, also binding innerhalb eines nodes wird von der APS Layer selbst unterstützt. Sogenanntes *over-the-air binding* wird über commands des ZDP ausgeführt (ZDO).

ZDP Binding Services	Client Transmission (Request)	Server Processing (Response)
End_Device_Bind_req	O	O
Bind_req	O	O
Unbind_req	O	O

Tabelle 5: Binding Services des ZDP

Für das Binding von Endpoints wird, wie bereits beschrieben ein, service discovery durchgeführt, der alle geeigneten endpoints auflistet. Die endpoints, die in Frage kommen, müssen erstens über die selbe cluster ID verfügen, zweitens muss der cluster im endpoint, als input beziehungsweise auf der gebundenen Seite als output cluster markiert sein. Nur dann kann ein sinnvolles binding erstellt werden. Danach setzt der node, der das binding durchführen will, einen Bind_req an den gewünschten Ziel endpoint ab. Wird dieser bestätigt, kann in den lokalen binding table das neue binding eingefügt werden.

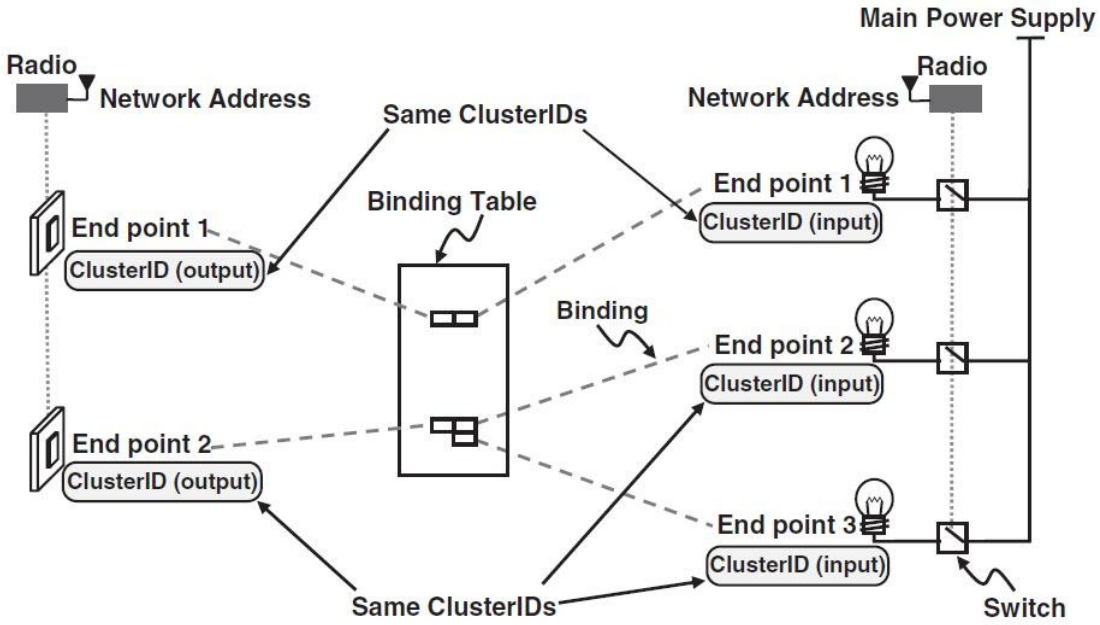

Abbildung 20: Ablauf des Bindings

Abgesehen von der ersten Methode des bindings, bei der die binding tables in den beteiligten ZigBee nodes abgespeichert sind, daher die binding informationen verteilt über das gesamte Netzwerk liegen, gibt es noch eine andere, zentrale Art des bindings, bei der in die binding table des ZigBee Coordinators binding informationen anderer nodes eingetragen werden. Dabei setzt ein ZigBee node der einen seiner endpoints mit einem anderen binden will, den End_Device_Bind_Req ab, welcher vom ZigBee Coordinator akzeptiert wird.

Der ZigBee Coordinator wartet dann eine gewisse Zeit auf den end device bind request des nodes mit dem zugehörigen input/output clusters. Kommt dieser rechtzeitig an, wird das binding zwischen input (Schalter) und output (Lampe) hergestellt. Dieser Mechanismus wird *simple binding mechanism* genannt, und ist auch vom application profile abhängig. Wie die oben angeführte Tabelle über die vom ZDP festgelegten services zeigt, sind alle services optional, es können daher auch eigene binding mechanisms festgelegt werden.

Die Verwaltung einer zentralen binding table hat Vorteile. Zum einen kann effizientes routing bereits im ZigBee Coordinator anhand der endpoints durchgeführt werden. Zum anderen ist der ZigBee Coordinator meist auch Data Concentrator und an weitere Geräte angeschlossen, um zum Beispiel debugging zu ermöglichen.

4 SSP Security Service Provider

Von großer Bedeutung ist Kommunikationssicherheit in Wireless Networks. Ohne gesicherte Kommunikation zwischen unterschiedlichen Partnern könnte jeder Zeit eine Fülle von Attacken durchgeführt werden.

Auf NWK Schichtebene könnte durch permanentes Stören oder Simulieren von *hidden nodes* ein route discovery nicht durchgeführt werden, als Beispiel einer Denial of Service Attacke.

ZigBee wird vor allem in billigen, leistungsschwachen Geräten eingesetzt. Aufgrund der Kürze der

Nachrichten und der eingeschränkten Ressourcen werden nur symmetrische Verschlüsselungsverfahren eingesetzt. ZigBee greift auf den weithin bekannten AES Verschlüsselungsalgorithmus zurück und unterstützt Schlüssellängen von 128 Bits, dabei beträgt die Blocklänge der AES Daten ebenfalls maximal 128 Bit.

ZigBee unterscheidet zwei Arten von Keys: link key und network key. Der link key sichert die Kommunikation zweier Nodes während der network key für die allgemeine Kommunikation und Broadcasts benutzt wird. Innerhalb eines ZigBee Netzwerk gibt es ein ausgewiesenes Gerät, welches den network key und die device-device-key (link key) tripplets verwaltet. Der ZigBee Coordinator definiert die Adresse des *trust centers* des Netzwerks und speichert dessen Adresse in der AIB (Attribut: *apsTrustCenterAddress*).

Im Allgemeinen existieren drei Möglichkeiten eine gesicherte verschlüsselte Verbindung zwischen zwei Nodes aufzubauen:

1. ein Network-Key, und optional link-keys werden vorinstalliert.
2. *key transport*
3. *key establishment*

Einfachste Variante ist die Installation eines vom Hersteller definierten Keys. Somit kann der node ein verschlüsseltes Netzwerk joinen ohne ein Trust-Center kontaktieren zu müssen.

Bei der *key transport* Methode erfragt der joinende node das trust center nach dem security key des Netzwerks und bedient sich hierbei der APS Sublayer. Das trust center antwortet mit dem security key in klartext oder es verschlüsselt diesen mit einem key-transport key, mit welchem das trust center jeden anderen key überträgt.

key establishment ist ein Algorithmus zum Aushandeln eines zufälligen key zwischen zwei Teilnehmern, ohne den key selbst über eine ungesicherte Verbindung zu übertragen und basiert auf dem *Symmetric-Key Key Establishment* Protokoll (SKKE). Dabei greifen die Devices auf einen *master key* zurück, welcher konfiguriert oder fix installiert wird, oder aber vom trust center in Klartext bezogen werden kann.

Beim key establishment überträgt der *initiator* Daten an den *responder,* welche mit dem master key verschlüsselt werden. Der responder leitet dann von den empfangenen Daten den eigentlichen link key ab.

Das trust center selbst kennt zwei Arbeitsmodi: *commercial* und *residential*. Im commercial mode speichert das trust center alle im Netzwerk verwendeten keys (network key, master key und link keys), während es im residential mode einzig und allein den network key speichert.

In ZigBee ist jede involvierte Protokollschicht für das Verschlüsseln eines Frames der Schicht mit dem key des nodes zuständig. Das bedeutet: im initiierenden Knoten werden APS, NWK und MAC Schicht die zu übertragende Nachricht mit dem key des nodes verschlüsseln. Gibt der Node nun die Nachricht weiter, wird sie an den nächsten node weitergereicht. Dabei bleibt die APS Schicht unangetastet, während sich nun bei der Weitergabe, NWK und MAC oder nur MAC Schicht den Schlüssel wechseln.

5 Commissioning

„Commissioning is the process of connecting ZigBee applications to each other. This process, while simple in concept, can be fairly complicated to do well and can involve quite a few steps.“ (Drew Gislason; ZigBee wireless networking; p. 337)

Es ist schwer von „dem Kommissionieren“ als Methode oder Vorgang zu sprechen, da *commissioning primitives* in allen Komponenten des ZigBee Stacks vorhanden sind und der Prozess des Kommissionierens alle Layers des Standards umfasst und viele verschiedene Herangehensweisen bietet.

Ein Überblick über die Aufgaben und Teilschritte des Kommissionierens erklärt warum:

- Aufbau oder Auffinden eines geeigneten Netzwerks oder des Zielnetzwerks
- Joinen des Netzwerks (Austausch von Keys, Authentifizierung, Anmelden an einem parent node, Laden allgemeiner Netzwerkparameter vom Coordinator)
- Mit welchen Knoten oder Applikationen soll kommuniziert werden? (Binding, Routing)
- Art der Kommunikation bestimmen (Broadcasts, Group Messages)
- Was soll bei fehlerhafter oder aussetzender Kommunikation geschehen (neues Netzwerk suchen, neuen parent node suchen)?

commissioning primitives befinden sich, wie bereits erwähnt, in allen ZigBee Komponenten, der NWK Layer, der APS, des ZDO, des ZDP und in der noch nicht angesprochenen ZigBee Cluster Library (ZCL):

- Das ZDO beinhaltet Methoden zum Auffinden, Auswählen und joinen von Netzwerken. Und steuert den NWK Layer.
- Im ZDP sind device und service discovery services und remote-table management Funktionen definiert.
- Die ZigBee Cluster Library, eine vordefinierte Sammlung an Clustern bietet eine Auswahl an over-the-air *group* und *scene management functions* an, und den *commissioning cluster*, der commands und attributes für das Setup von security keys, PAN IDs, channel mask und anderen wichtigen Parametern des Netzwerks bietet.

Man kann jedoch grob drei Arten von commissioning unterscheiden.

1. Simple commissioning
2. Butterfly commissioning
3. Custom commissioning

5.1 Simple Commissioning

Simple commissioning ist die primitivste Art der Kommissionierung und kann ohne Verwendung des commissioning clusters der ZCL durchgeführt werden. Die einzelnen nodes joinen nach ihrer Aktivierung das erste Netzwerk, welches in ihrer Reichweite liegt. Zu diesem Zeitpunkt wissen alle Geräte nur über ihre eigenen Parameter und vom ZigBee Coordinator Bescheid. Danach warten alle Geräte auf eine Eingabe durch den Benutzer, welche den Kommissionierungsvorgang auslöst. Meist wird dann ein Match Descriptor Request mit dem zu bindenden endpoint als Argument abgesetzt,

und im Anschluss mit End Device Bind Request ein simple binding durchgeführt. Der Coordinator übernimmt beim simple commissioning also alle Aufgaben der Kommissionierung.

5.2 Butterfly Commissioning

Die eigentliche Standard ZigBee commissioning Methode folgt dem in der Literatur als „butterfly concept“ beschriebenen Modell. Nachdem ein ZigBee Device aktiv wird, sucht es das erste Netzwerk, welches sich anbietet, joined es und holt daraus weitere Kommissionierungsinformationen ab. Nach einigen Iterationen dieses Schritts ist der ZigBee node fertig konfiguriert. In diesem Fall muss für die initiale Konfiguration des ZigBee Device ein *commissioning network* erstellt werden, in welchem sich der node alle weiteren Informationen zum joinen des eigentlichen Arbeitsnetzwerks holen kann.

Bsp.: Ein ZigBee Knoten holt sich aus dem ersten commissioning network alle Informationen zum Aufbau einer Verbindung zum Security Center ab. Danach wird eine gesicherte Verbindung mit einem neuen Netzwerk erstellt, über welche die eigentlichen Kommissionierungsinformationen ausgetauscht werden. Nach dieser abschließenden Konfiguration wird das eigentliche Arbeitsnetzwerk „gejoined“.

Nicht notwendigerweise müssen, obwohl meist die Regel, einzelne gesicherte Netzwerke für diesen Vorgang erstellt werden, man könnte auch auf Basis von Gruppenzugehörigkeit einen solchen Vorgang durchführen.

5.3 Rolle des ZDO

Das ZDO ist jener Teil ZigBee's der die Entscheidung fällt, welches Netzwerk zu joinen ist. Diese Entscheidung basiert auf mehreren Feldern in den Informationen Bases unterschiedlicher Schichten. Wird ein *commissioning tool*, etwa ein PC, ein Handheld oder ein speziell konfiguriertes ZigBee Device benutzt, kann durch den commissioning cluster der ZCL das ZDO angewiesen werden, das ZigBee Device mit bestimmten Parametern neu zu starten. In jedem Fall ist aber immer das ZDO die einzige Instanz die das joinen eines Netzwerks durchführt.

Tabelle 7 zeigt alle für die Entscheidung vom ZDO benutzten Felder. Nochmals sei darauf hingewiesen dass diese in verschiedenen Layer liegen. (NWK, APS) Für ZigBee public profiles, sind alle diese auf 0x00 oder 0xff gesetzt und enthalten nur die eigene MAC Adresse des Geräts. In einem private profile wären sie gesetzt. Dir Primitive des ZDP, ZCL und des commissioning clusters werden erst nach dem joinen des Netzwerks verwendet.

Field	Description
MAC Address	The MAC address is a unique 64-bit number assigned at manufacturing time. It is never changed.
apsChannelMask	The channel mask describes which channels should be scanned when forming or joining. The field is a 32-bit bit-mask of 802.15.4 channels. Only channels 11 through 26 are valid for ZigBee, which operates in the 2.4 GHz RF spectrum. Examples: 0x00000800 channel 11 (bit 11) 0x04001000 channels 12 and 26 (bits 12 and 26) 0x07fff800 all channels (11–26)
apsUseExtendedPANID	This is a 64-bit address (similar to the MAC address). Set this to all 0x00s for ZigBee Application Profiles. It is sometime set to a specific number for private profiles.
nwkPANId	Always set to 0xffff (no PAN ID). This will choose a random PAN ID.
nwkNetworkAddress	Set to 0xffff to indicate no short address.
nwkStackProfile	Set this to the preferred stack profile (0x01 or 0x02). Most Application Profiles allow the node to join either stack profile.
TrustCenterAddress	Normally set to 0x0000. Can be set to a different short address if the TC is not on the ZigBee Coordinator.
NetworkKey	Set to all 0x00s if no network key. Set to a specific network key for preconfigured keys. Usually given to the node by the network in Home Automation.
NwkKeySeqNum	Only relevant to over-the-air commissioning tools.
TrustCenterMasterKey	Only relevant High Security networks. This is defined either by the Application Profile or a commissioning tool.

Tabelle 6: ZDO Startup Fields; Drew Gislason; ZigBee wireless networking; p. 335

Die wichtigsten vom ZDP benutzten commissioning commands wurden bereits in en Abschnitten 3.3 und 3.2.4 vorgestellt: Bind-Request, Unbind-Request, End-Device-Bind, Simple-Descriptor-Request, Active-Endpoint-Request, Match-Descriptor-Request, IEEE Address Request, Mgmt-Bind, Permit-Joining-Request.

5.4 Der Commissioning Cluster

Die Spezifikation des ZDP inkludierte nicht die Fähigkeit verschiedene Startup Parameter over-the-air abzufragen, um den Kern des ZigBee Stack klein zu halten, um auch ressourcenarme Geräte mit simple commissioning einbinden zu können. Da das ZDP in jedem ZigBee Device benutzt wird, wurde der commissioning cluster in die ZCL aufgenommen.

Der commissioning cluster besteht selbst aus vier wesentlichen attribute sets:

- Startup Attribute Set (SAS)
- Join Parameters Attribute Set (JPAS)
- End-Device Parameters Attribute Set
- Concentrator Parameters Attribute Set

Das Startup Attirbute Set enthält alle Informationen, um das gewünschte Netzwerk zu finden und um den SSP mit den richtigen Parametern zu versorgen. Die Attribute des SAS werden über APSME-Set und NLME-Set Instruktionen gesetzt und beeinflussen damit die information bases der APS und der NWK Layer.

Die SAS Parameter werden erst mit einem Neustart des ZigBee nodes über den Restart Device Request, ein command des commissioning cluster geladen.

Innerhalb des SAS ist das *StartupControl* jenes Feld welchem man am meisten Aufmerksamkeit schenken wird. Es beeinflusst die Art und Weise des Resettings des nodes.

- ***silent join:*** Der node kennt bereits alle notwendigen Informationen des Netzwerks welches er joinen soll, so als ob er bereits dem Netzwerk beigetreten wäre, das beinhaltet auch Informationen wie security keys. Der silent join kann nur dann ausgeführt werden, wenn der node bereits im Netzwerk registriert ist.
- ***form a network:*** Klarerweise ist dieser Parameter nur für ZigBee Coordinator capable devices valid.
- ***rejoin:*** weist den node an, das Netzwerk wieder zu joinen, als ob der node den Kontakt zum parent node verloren hätte. Diese Option ist vorallem dann interessant, wenn permit-join im ZigBee Coordinator deaktiviert ist. Nach dem die Kommissionierungsinformationen an alle Devices ausgegeben wurde, kann der ZigBee Coordinator das Netzwerk „absperren“.
- ***associate join a network:*** der node muss sich nochmals erneut anmelden, dabei muss permit-join erlaubt sein.

SAS Attribute	Description
ShortAddress	This is the 16-bit network short address. Set to 0x0000–0xfff7 for valid addresses, or 0xffff to indicate it is not yet established.
ExtendedPANId	Which extended PAN ID will the node form or join when reset? ExtendedPANId 0x00f0c27710000000 is the commissioning extended PAN. Set to all 0x00s to indicate any extended PAN.
PANId	Normally starting out as 0xffff, which means choose a random PAN ID. Most applications care only about the extended PAN ID. Both PANId and ExtendedPANId are found in beacons.
ChannelMask	A 32-bit mask for deciding which channels to search when forming or joining. Only bits 11–26 may be set. It takes time (about 1/3 to 1 second) for each channel scanned.
ProtocolVersion	Always set to 0x02.
StackProfile	Set to 0x01 or 0x02, the preferred stack profile.
StartupControl	0x00—silent join. 0x01—form a network. 0x02—rejoin a network. 0x03—associate join a network.
TrustCenterAddress	A short address to find the trust center. This is required in high security. Normally 0x0000 (the ZC).
TrustCenterMasterKey	The master key is used to establish a link key with the trust center through SKKE.
NetworkKey	The network key
UseInsecureJoin	Set to TRUE for standard security, FALSE for high security.
PreconfiguredLinkKey	Assumes SKKE has already been performed.
NetworkKeySeqNum	Key sequence number for the network key. A node may have more than 1 network key (old and new).
NetworkKeyType	Set to 0x01 for standard security, 0x05 for high security.
NetworkManagerAddress	Normally set to 0x0000 (the ZC). This node is in charge of frequency agility, if enabled.

Tabelle 7: Startup Attribute Set des Commissioning Clusters; Drew Gislason; ZigBee wireless networking; p. 338

Wird der node ein neues Netzwerk joinen oder einen Rejoin am alten durchführen, so können auch noch die Werte von JPAS gesetzt sein. *RejoinInterval* bestimmt, mit welcher Frequenz der node joined oder einen rejoin versuchen wird. Bei den meisten ZigBee Profilen wird der node zu Beginn regelmäßig einen Rejoin versuchen und später die Frequenz des Rejoins drosseln, um Energie zu sparen in der Annahme, dass das Netzwerk später wieder hochfährt.

Join Parameters Attribute	Description
ScanAttempts	From 1 to 0xff. 0xff means forever.
TimeBetweenScans	From 1 to 0xffff, in milliseconds.
RejoinInterval	Lower bounds for rejoining, in seconds. Defaults to 60.
MaxRejoinInterval	Upper bounds for rejoining, in seconds. Defaults to 1 hour.

Tabelle 8: Join Parameter Attribute Set des Commissioning Clusters; Drew Gislason; ZigBee wireless networking; p. 339

Mit dem End Device Attribute Set kann der Commissioning Cluster das polling interval (*IndirectPollingInterval*) von ZED bestimmen, was die Lebenszeit bei batteriebetriebenen Geräten wesentlich beeinflusst. *ParentRetryThreshold* beeinflusst das Suchverhalten nach neuen parents, wenn der Kontakt zum Elternknoten einmal abgerissen ist und wirkt damit auf Endknoten, welche in sleeping mode wechseln können (*RxOnIdle FALSE*) wie auch auf ZEDs die ständig online sind.

End Device Parameters Attribute Set	Description
IndirectPollRate	The rate, in milliseconds, to poll the parent
ParentRetryThreshold	The number of failed attempts to contact a parent that will cause a “find new parent” procedure to be initiated

Tabelle 9: Beschreibung des End Device Parameter Set des Commissioning Clusters; Drew Gislason; ZigBee wireless networking; p. 339

Beim commissioning müssen auch sogenannte *concentrators*, spezielle Knoten die den Nachrichtenverkehr innerhalb ihres *concentrator radius* bündeln und auch das Routing für die an sie gebundene Knoten übernehmen. Many-To-One Routing (siehe Abschnitt 2.1.3) erlaubt es dem Concentrator als Relay innerhalb eines ZigBee Netzes zu dienen, somit können andere nodes beim Mesh-Networking Ressourcen sparen und der Nachrichtenaustausch beim Routing wird vermindert.

ZEDs innerhalb des concentrator radius müssen keine neighbour tables oder routing tables mehr warten. Sie übergeben einfach ihre Nachrichten an den concentrator, welcher alle Routen für die Weiterleitung an das Zieldevice kennt. Umgekehrt müssen auch ZigBee Knoten außerhalb des concentrator radius (Router und der Coordinator) nicht mehr eigene Routingeinträge für alle Knoten, die mit dem concentrator assoziiert sind warten, sondern können Nachrichten direkt an den concentrator übergeben, welcher diese korrekt zustellt.

Concentrator Parameters Attribute Set	Description
ConcentratorFlag	After restarting, will this node be a concentrator or not? Assumes that the commissioning tool already knows.
ConcentratorRadius	To what radius will this discover the many-toone route? The default (and maximum) is 5.
ConcentratorDiscoveryTime	Many-to-one route discovery can occur automatically (Interval: 0x0001 - 0xffff seconds) or manually (0x0000).

Tabelle 10: Informationen zum Concentrator Parameter Attribute Set des Commissioning Clusters; Drew Gislason; ZigBee wireless networking; p. 339

Die Idee von Commands im Commissioning Cluster ist, dass ein spezielles Device ein *commissioning network* formt, in dem sich die aktivierten ZigBee nodes alle notwendigen Informationen für das Arbeitsnetzwerk holen. Manchen Zigbee nodes ist es sogar möglich mehrere Commissioning Cluster zu speichern und zu laden. Zum Beispiel einen vom User erstellten und „factory default“, oder mehrere für verschiedene Bereiche eines Gebäudes (Home Automation Profile).

ID	Command	Mandatory/Optional
0x00	Restart Device Request	Mandatory
0x01	Save Startup Parameters Request	Optional
0x02	Restore Startup Parameters Request	Optional
0x03	Reset Startup Parameters Request	Mandatory

Tabelle 11: Commissioning Cluster Commands; Drew Gislason; ZigBee wireless networking; p. 338

5.5 Custom Commissioning

Custom Commissioning ist prinzipiell vom Hersteller des Geräts selbst definiert und kann, muss aber nicht, sich der bereits erwähnten Komponenten aus ZDO, ZDP und ZCL bedienen.

6 Schlussbetrachtungen

Die Möglichkeiten die ZigBee bietet, gehen weit über die Verschränkung von Smart Sensoring hinaus, viel mehr ist es als Protokoll mit großem Potential für autonom agierende und verteilte Systeme zu sehen, wie in dem von Junli Wan, Yanqiu Wang, Qin Qin und Yanqin Li geschriebenen Paper über „Multi-Robots' Communication System Based On ZigBee Network".

Für zukünftige Anwendungen von ZigBee möchte ich auch noch auf ein sehr interessantes Paper über „Research on Data Fusion Algorithm in ZigBee Protocol" aus Quelle 5 hinweisen. Im Paper wird die Möglichkeit aufgezeigt, wie man in einem ZigBee Netzwerk Statusabfragen oder Informationsaustausch ähnlich wie bei Datenbanken organisieren könnte.

Mittlerweile gibt es eine große Anzahl an Developer Kits für ZigBee, jedoch scheint es, dass diese Wireless Technologie von der Consumer-Electronic-Industrie, obwohl selbst ein von der Industrie geschaffener Standard, noch kaum angenommen wird. ZigBee verspricht aber aufgrund seiner Vielfältigkeit im Einsatz, vom kleinsten batteriebetriebenen Gerät, bis hin zu leistungsstarken Data Concentrators noch großes Potential zu bergen.

7 Abbildungsverzeichnis

8 Literaturverzeichnis

[1] **Drew Gislason;** „ZigBee wireless networking“; s.a. [2007]; s.l. Butterworth Heinemann; Auflage: Pap/Onl (13. Oktober 2007). ISBN 978-0750685979

[2] **Shahin Farahani;** „ZigBee wireless networks and transceivers“; 2008; Burlington et al. Elsevier ISBN: 978-0-7506-8393-7

[3] **KF Tsang, WC Lee, KL Lam, HY Tung and Kai Xuan;** „An Integrated ZigBee Automation System: An Energy Saving Solution“; IEEE 2007

[4] **Junli Wan, Yanqiu Wang, Qin Qin, Yanqin Li;** „Multi-Robots' Communication System Based On ZigBee Network“; 2009; Electricity Technology and Electrical Information College, China Three Gorges University Hubei; China; The Ninth International Conference on Electronic Measurement & Instruments ICEMI'2009

[5] **Ping Wang, Ming-Chuan Cheng, and Wen-Zao Shi;** „Research on Data Fusion Algorithm in ZigBee Protocol“; 2008; Department of Electronic and Information Fujian Normal University; Fuzhou, China; 2008 International Symposium on Computer Science and Computational Technology

Onlinequellen:

[6] http://standards.ieee.org/getieee802/download/802.15.4-2006.pdf

[7] http://www.zigbee.org/zigbee/en/spec_download/spec_download.asp

(ZigBee Standards, Papers)