

Hauptseminar

„Aktuelle Trends in Kommunikationsnetzen“

**Prof. H. G. Hegering
Prof. Dr. Linnhoff-Popien**

LMU / TU München

Ad Hoc Protokolle

IEEE 802.11 – Bluetooth – Infrarot

Nicolas Weber, Daniel Brügge

Stand: Oktober 2002

Inhaltsverzeichnis

1	EINFÜHRUNG.....	3
2	AD HOC NETZWERKE.....	4
2.1	EINFÜHRUNG.....	4
2.2	GERÄTEUNTERSCHIEDE IN EINEM AD HOC NETZWERK.....	4
2.3	ENERGIEVERWALTUNG IN AD HOC NETZWERKEN.....	4
2.4	MEDIA ACCESS CONTROL (MAC).....	5
2.4.1	<i>Synchrone und Asynchrone MAC Protokolle.....</i>	<i>5</i>
3	IEEE 802.11	6
3.1	EINFÜHRUNG.....	6
3.2	PROTOKOLLE.....	6
3.2.1	<i>Protokollstack.....</i>	<i>6</i>
3.2.2	<i>Physical Layer (PHY).....</i>	<i>6</i>
3.2.3	<i>Logical Link Control (LLC).....</i>	<i>7</i>
3.2.4	<i>Media Access Control (MAC).....</i>	<i>7</i>
3.2.5	<i>MAC Protokolle.....</i>	<i>7</i>
3.2.5.1	IEEE 802.11 MAC Probleme.....	8
3.2.5.1.1	Hidden Terminal Problem.....	8
3.2.5.1.2	RTS-CTS Problematik.....	9
3.2.5.1.3	Exposed Node Problem.....	10
3.2.5.2	Senderinitiierte MAC Protokolle.....	12
3.2.5.2.1	MACA.....	12
3.2.5.2.2	MACAW.....	14
3.2.5.2.3	PAMAS.....	14
3.2.5.2.4	DBTMA.....	16
3.2.5.3	Empfängerinitiierte MAC Protokolle.....	17
3.2.5.3.1	MACA-BI.....	17
3.2.5.3.2	MARCH.....	19
4	BLUETOOTH	21
4.1	EINFÜHRUNG.....	21
4.2	ARCHITEKTUR.....	21
4.2.1	<i>Piconet.....</i>	<i>21</i>
4.2.2	<i>Scatternet.....</i>	<i>22</i>
4.3	PROTOKOLLE.....	24
4.3.1	<i>Protokollstack.....</i>	<i>24</i>
4.3.2	<i>Bluetooth Kernprotokolle.....</i>	<i>25</i>
4.3.2.1	Baseband.....	25
4.3.2.2	Link Manager Protokoll (LMP).....	26
4.3.2.3	Logical Link Control und Adaptation Protokoll (L2CAP).....	26
4.3.3	<i>Host Controller Interface (HCI).....</i>	<i>27</i>
4.3.4	<i>Bluetooth Service Discovery Protokoll (SDP).....</i>	<i>27</i>
4.3.5	<i>RFCOMM.....</i>	<i>28</i>
4.3.6	<i>TCS Binary.....</i>	<i>28</i>
4.3.7	<i>Aufgesetzte Protokolle.....</i>	<i>29</i>
4.3.7.1	PPP.....	29
4.3.7.2	TCP/UDP/IP.....	29
4.3.7.3	OBEX.....	30
4.3.7.4	Inhaltsformate vCard und vCalendar.....	30

4.3.7.5	WAP.....	30
4.4	BLUETOOTH AUDIO.....	30
4.5	BLUETOOTH MAC	31
4.6	PAKETSTRUKTUR.....	31
4.7	ADRESSIERUNG	32
5	INFRAROT.....	33
5.1	EINFÜHRUNG.....	33
5.2	IRDA CONTROL	33
5.3	IRDA DATA	34
5.3.1	<i>IrDA Data Protokollstack</i>	34
5.3.2	<i>Zwingend erforderliche Protokolle</i>	34
5.3.2.1	PHY (Physical Signalling Layer).....	34
5.3.2.2	IrLAP (Link Access Protocol).....	34
5.3.2.3	IrLMP (Link Management Protocol)	36
5.3.3	<i>Optionale Protokolle</i>	36
5.3.3.1	Tiny TP.....	36
5.3.3.2	IrCOMM.....	37
5.3.3.3	IrOBEX	37
5.3.3.4	IrLAN.....	38
6	ZUSAMMENFASSUNG	39
	QUELLENVERZEICHNIS	40
	ABBILDUNGSVERZEICHNIS	42
	INDEX.....	43

1 Einführung

Um zu gewährleisten, dass bestehende Anwendungen drahtlose Netzwerktechniken nutzen können, muss es Kernprotokolle geben, auf die herkömmliche Protokolle (TCP/IP, PPP; ...) aufsetzen können. Kollisionsvermeidung, und Energieeffizienz sind wichtige Punkte, die bei Ad Hoc Protokollen berücksichtigt werden müssen. Stellvertretend für alle drahtlosen Netzwerktechnologien werden wir im folgenden IEEE 802.11, Bluetooth, und den etwas älteren Standard, Infrarot betrachten. Da durch die verschiedenen Strukturen dieser Standards auch verschiedene Probleme auftreten, werden wir bei IEEE 802.11 besonders die Media Access Control Protokolle und bei Bluetooth die unteren Kernprotokolle und die verschiedenen Übertragungstechniken betrachten.

2 Ad Hoc Netzwerke

2.1 Einführung

Da in Ad Hoc Netzwerken andere Faktoren eine Rolle spielen als in festen Netzwerken, benötigen diese auch eigene Protokolle. In Ad Hoc Netzwerken ist es normal, dass laufend Geräte dazu kommen oder andere Geräte das Netz verlassen. Im Gegensatz zu festen Netzen kann in Funknetzen nicht jedes Gerät alle anderen Geräte im gleichen Netzwerk direkt erreichen, da manche Geräte sich zwar im gleichen Netzwerk befinden, aber außerhalb der Reichweite des Senders.

Des Weiteren werden alle mobilen Geräte mit Strom aus Akkus betrieben, die eine sehr begrenzte Lebensdauer haben. Daher werden Protokolle gebraucht die sowohl Kollisionen vermeiden, als auch Strom sparend sind.

Protokolle werden auch gebraucht um zu bestimmen, wer Daten senden darf und wer diese empfängt. Dies ist vor allem der Fall in Netzen in denen es nur gleichberechtigte Geräte gibt (IEEE 802.11), da es hier keinen Master gibt der für alle anderen bestimmt wer wann Daten senden darf. Zusätzlich müssen die Protokolle aber auch Fehler in Datenübertragungen erkennen und beheben können und unempfindlich gegen Interferenzen sein.

Höhere Protokollschichten herkömmlicher Protokolle müssen sich auf die Ad Hoc Protokolle aufsetzen lassen, damit die drahtlosen Verbindungen auch mit Software, die nicht speziell für WLAN geschrieben wurde, genutzt werden können.

2.2 Geräteunterschiede in einem Ad Hoc Netzwerk

Für Ad Hoc Netzwerke kann ein Großteil der bekannten Protokolle nicht mehr verwendet werden und aus diesem Grund müssen neue Protokolle entwickelt werden, die wir in den späteren Kapiteln noch kennen lernen werden.

Beim Entwerfen solcher Protokolle muss darauf geachtet werden, dass es in einem Ad Hoc Netzwerk erhebliche Geräteunterschiede geben kann und dass dadurch nicht jedes Gerät die gleichen Funktionen in diesem Netz übernehmen kann.

So kann es zum Beispiel nicht angehen, dass ein leistungsschwaches Mobiltelefon anspruchsvollere Aufgaben wie z.B. Server- oder Service-Provider-Aufgaben übernimmt, als ein leistungsstärkerer Laptop, der sich in dem selben Netzwerk befindet.

Außer den Energie- und Leistungsunterschieden muss auch auf die Ausstattung der einzelnen Geräte geachtet werden. Ein Palm-Organizer ist beispielsweise nicht in der Lage hochauflösende Grafiken darzustellen, während es für einen Laptop kein Problem ist.

2.3 Energieverwaltung in Ad Hoc Netzwerken

Die meisten herkömmlichen Protokolle berücksichtigen keine Energieverwaltung. Dies war bis jetzt auch nicht nötig da alle Geräte in festen Netzen an eine feste Stromversorgung angeschlossen waren. Mobile Geräte in Ad Hoc Netzwerken sind abhängig von Batterien. Allerdings hält eine geladene Li-ion Batterie nicht länger als 2-3 Stunden. Diese begrenzte Operationszeit erfordert eine sparsame Energieverwaltung von den Protokollen.

Dazu kommt noch, dass mobile Geräte in Ad Hoc Netzen nicht nur als Endgeräte fungieren sondern auch zwischen anderen Geräten Pakete weiterleiten (packet forwarding), was zusätzlich Energie verbraucht.

2.4 Media Access Control (MAC)

Media Access Control wird gebraucht um den Funkkanal den Geräten so zuzuteilen, dass keine Kollisionen entstehen und kein Gerät benachteiligt ist. Dies ist besonders bei konkurrierenden Netzwerken (IEEE 802.11, Hiperlan, ...) der Fall, da es keinen Master gibt der den Geräten den freien Kanal zuteilt.

Deswegen können die herkömmlichen Verfahren wie Zeitmultiplex (TDMA – Time Division Multiple Access) oder Frequenzmultiplex (FDMA – Frequency Division Multiple Access) nicht verwendet werden. Herkömmliche MAC Protokolle (Media Access Control) unterstützen keine mobilen Geräte.

Daher müssen Protokolle geschaffen werden, die auch ohne Kontrolle durch einen Master die freien Kanäle so verteilen, dass es nicht zu Kollisionen kommt. Dadurch, dass zwar die Nachbarn der Geräte aber nicht unbedingt deren Nachbarn in der Reichweite eines Gerätes liegen müssen entstehen viele Probleme.

2.4.1 Synchrone und Asynchrone MAC Protokolle

Bei den Media Access Protokollen eines Ad Hoc Netzwerkes unterscheidet man zwischen synchronen und asynchronen MAC Protokollen.

Bei den synchronen Protokollen werden alle Knoten eines Netzwerkes auf die gleiche Zeit gesetzt. Dieses geschieht durch einen Master Knoten, der ständig ein Funksignal an die anderen Knoten aussendet, welche nach diesem Signal ihre Uhren synchronisieren. Deswegen ist bei synchronen MAC Protokollen eine zentrale Kontrollstelle erforderlich.

Im Gegensatz zu den synchronen Protokollen, ist es bei den asynchronen nicht erforderlich, dass die Uhren bei allen Knoten synchronisiert werden. Da alle Knoten miteinander konkurrieren, müssen neue Kontrollmechanismen herangezogen werden, wie wir in den folgenden Kapiteln noch sehen werden. Im nächsten Kapitel zeigen wir Anhand des IEEE 802.11 Standards welche Probleme bei asynchronen MAC Protokolle auftreten und wie diese gelöst werden.

3 IEEE 802.11

3.1 Einführung

Der IEEE 802.11 Standard wurde 1997 vom IEEE¹ entwickelt und ist der erste Standard für drahtlose Netze (WLAN).

Der Standard definiert die Media Access Control (MAC) Schicht und die physikalische Schicht (PHY) für Netzwerke mit drahtloser Funktionalität.

IEEE 802.11 ist für Übertragungsbandbreiten von einem und zwei MBit/s und sendet im lizenzierungsfreien ISM²-Band auf 2,400 – 2,485 GHz. Dieses macht IEEE 802.11 natürlich für viele Zwecke interessant. Leider sind auf dieser Frequenz auch viele Störquellen wie z.B. Mikrowellen zu finden, weswegen eine Kommunikation vielfach erschwert wird.

Wir werden uns in diesem Kapitel über IEEE 802.11 den Protokollstack mit seinen Schichten näher ansehen und einiges dazu sagen. Der Schwerpunkt wird aber eindeutig bei den Kollisionsproblemen und bei den Media Access Control Protokollen liegen.

3.2 Protokolle

3.2.1 Protokollstack

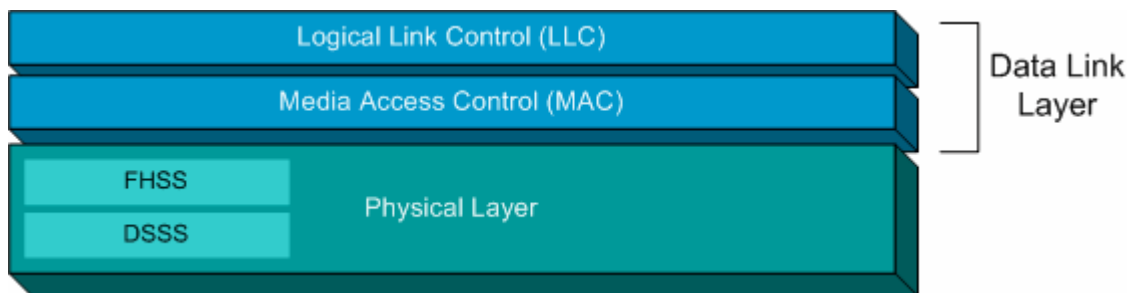


Abbildung 3-1: Vereinfachter IEEE 802.11 Protokollstack

Abbildung 3-1 zeigt den IEEE 802.11 Protokollstack in vereinfachter Form. Der Stack unterteilt sich in zwei Hauptschichten (Layer), den *Data Link Layer* und den *Physical Layer*. Beim Data Link Layer definiert der Standard nur die unterste Schicht, also den Media Access Control (MAC) Layer.

3.2.2 Physical Layer (PHY)

Der Physical Layer ist intern in zwei Schichten aufgeteilt. Eine Schicht ist *Physical Layer Convergence Protocol (PLCP)*, welche eine von der Übertragungstechnologie unabhängige Schnittstelle zur Data Link Schicht definiert. Diese setzt auf der *Physical Media Dependent (PMD)* Schicht auf, welche unterschiedliche Übertragungsverfahren festlegt. Zwei Verfahren sind das *Frequency Hopping Spread Spectrum (FHSS)* und das *Direct Sequence Spread Spectrum (DSSS)*. Beide sollen Störeinflüsse soweit wie möglich verhindern, wobei DSSS dieses durch breitere Verteilung der Signale versucht und FHSS eine

¹ IEEE = Institute of Electrical and Electronical Engineers (Normungsverband US-amerikanischer Ingenieure)

² „Industrial, Scientific und Medical“ (engl.: industriell, wissenschaftlich und medizinisch)

Frequenzsprungverfahren einsetzt, bei dem über das ganze ISM-Band die Signale in unterschiedlichen Frequenzen gesendet werden.

3.2.3 Logical Link Control (LLC)

Der LLC-Layer ist für die logische Steuerung der Verbindungen zuständig und ist für alle IEEE 802 Standards identisch, also nicht extra für IEEE 802.11 definiert worden. Auf diese Weise können Protokolle der höheren Schichten unabhängig vom Zugriffsmechanismus und der physikalischen Realisierung auf die Kommunikationsdienste zugreifen.

Höhere Protokolle wie beispielsweise TCP/IP haben so also keine Probleme, auf drahtlose Netze zuzugreifen und können sich genauso verhalten wie bei drahtgebundenen Ethernet Protokollen.

3.2.4 Media Access Control (MAC)

Der MAC (*Media Access Control*) Layer stellt die notwendigen Funktionalitäten zur Verfügung, um Daten zuverlässig zu den höheren Protokollschichten zu transportieren.

Standardmäßig verfügt die MAC Schicht über das Kollisionsvermeidungsverfahren CSMA/CA (*Carrier-Sense Multiple Access with Collision Avoidance*), welches bei einem „shared“ Medium wie IEEE 802.11 unabdingbar ist, da hier viel leichter Kollisionen von Datenpaketen auftreten können, als bei herkömmlichen Netzwerken.

CSMA/CA läuft nach folgendem Schema ab:

1. Wenn eine Station Daten senden möchte, dann wird erst einmal „gelauscht“, ob gerade eine Nachbarstation Daten sendet.
2. Falls gerade eine Datenübertragung stattfindet wird gewartet.
3. Wenn die Datenübertragung vom Nachbarknoten beendet ist, wird eine vorgeschriebene plus eine zufällige Zeitspanne (Backoff Time) gewartet.
4. Falls in der Wartezeit wieder eine Datenübertragung belauscht wird, wird der Zeitspannen-Timer angehalten. So wird eine gewisse Priorität gewährleistet, da Stationen mit angehaltenem Timer eher drankommen, als Stationen die sich erst am Anfang der Wartezeit befinden.
5. Falls nach der Zeitspanne keine Datenübertragung festgestellt wird, wird gesendet.

CSMA/CA ist keine perfekte Lösung und deshalb wurden auch bessere Kollisionsvermeidungsprotokolle entwickelt, die wir im nächsten Kapitel noch kennen lernen werden.

3.2.5 MAC Protokolle

Ad Hoc Netzwerke unterscheiden sich von statischen Netzwerken vor allen Dingen dadurch, dass es anstelle von unbeweglichen Knotenpunkten jetzt auf einmal mobile Knoten gibt, die sich jederzeit im Netz bewegen können.

Aus diesem Grund greifen die herkömmlichen Netzwerkprotokolle zum größten Teil nicht mehr und es müssen neue Lösungen gefunden werden.

Dieses gilt insbesondere für die *Media Access Protokolle* (MAC-Protokolle), die die Kommunikation einzelner Knoten untereinander kontrollieren und organisieren.

3.2.5.1 IEEE 802.11 MAC Probleme

3.2.5.1.1 Hidden Terminal Problem



Abbildung 3-2: A sendet Daten an B



Abbildung 3-3: C sendet Daten an B... Kollision!

Beim Hidden Terminal Problem senden zwei voneinander getrennte Geräte (das eine Gerät ist außerhalb der Reichweite des anderen Gerätes) gleichzeitig Informationen zu einem Empfängergerät. Da keines der sendenden Geräte merkt, dass schon ein anderes sendet kommt es zu einer Kollision (Abbildung 3-3).

Dies kann verhindert werden indem alle benachbarten Geräte des Empfängers über die Absicht zu Senden eines Gerätes informiert werden. Hierzu wird der zum Senden verwendete Kanal durch Kontrollnachrichten reserviert. Wenn ein Gerät einen freien Kanal benötigt schickt es ein RTS-Signal (Request To Send) zum Empfänger (Abbildung 3-4). Dieser sendet ein CTS-Signal (Clear To Send), das von allen Geräten in dessen Reichweite empfangen wird (Abbildung 3-5). Nun sind alle Geräte die ein CTS-Signal empfangen haben, aber kein RTS-Signal geschickt haben blockiert (Abbildung 3-6). Auf diese Weise wird sichergestellt, dass der Kanal des sendewilligen Gerätes frei ist, und Kollisionen vermieden werden. Nach der Übertragung sendet das empfangende Gerät ein ACK-Signal (ACKnowledgement) (Abbildung 3-7). Hiermit wird dem Sender der Empfang bestätigt und bei den übrigen Geräten wird die Sendesperre aufgehoben.

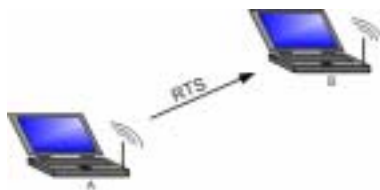


Abbildung 3-4: Hidden Terminal Handshake 1. Schritt (RTS)

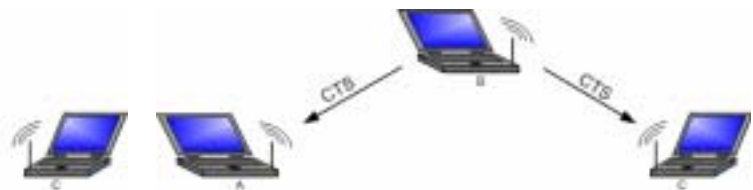


Abbildung 3-5: Hidden Terminal Handshake 2. Schritt (CTS)

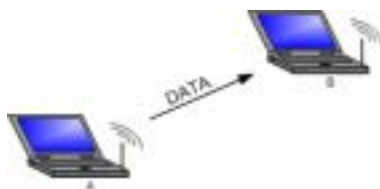


Abbildung 3-6: Hidden Terminal Handshake 3. Schritt (Daten)

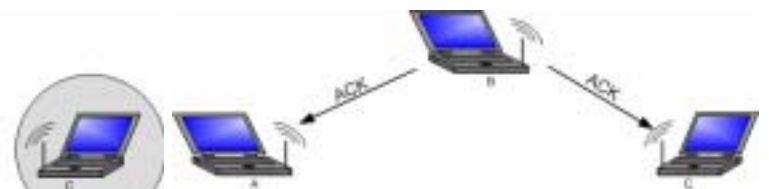


Abbildung 3-7: Hidden Terminal Handshake 4. Schritt (ACK)

3.2.5.1.2 RTS-CTS Problematik

Die RTS-CTS Methode ist keine perfekte Lösung für das Hidden Terminal Problem. In einigen Fällen kann es trotzdem zu Kollisionen kommen. Wenn Knoten A einen Kanal anfordert, um zu Knoten B Daten zu senden (RTS) und D (außerhalb der Reichweite von B) auch einen freien Kanal von C anfordert, kollidiert das RTS-Signal von D mit dem CTS Signal von B (Abbildung 3-8). Da C wegen dieser Kollision kein CTS Signal sendet, schickt D nach einiger Zeit ein neues RTS Signal. Das darauf folgende CTS Signal von C kollidiert mit den Daten die von A kommen.

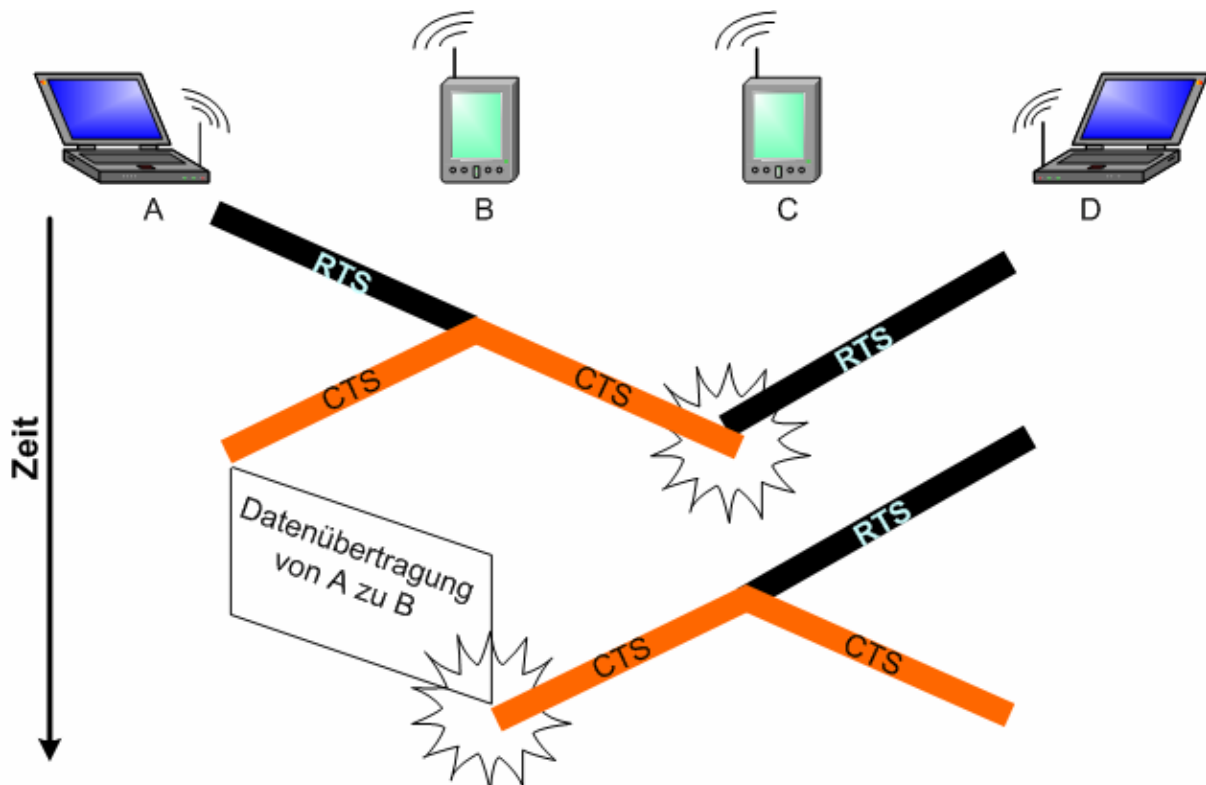


Abbildung 3-8: Probleme der RTS-CTS Methode

Ein weiteres Problem entsteht wenn mehrere benachbarte Geräte gleichzeitig die Freigabe zum Senden von Daten geben. Dies kann geschehen wenn von mehreren benachbarten Geräten, von denen eines außerhalb der Reichweite des anderen steht, CTS-Signale gesendet werden. Wenn von Gerät A ein RTS-Signal gesendet wird erwidert B dies mit einer Sendefreigabe (CTS). Annähernd gleichzeitig sendet C eine Sendeanforderung (RTS). Da D außerhalb der Reichweite von B liegt wird dieses Gerät nicht durch das CTS Signal (von B) blockiert. Deswegen fängt C an Daten zu senden die mit den Daten, die A (nach dem Empfang des CTS-Signal) sendet, kollidieren (Abbildung 3-9). Also kann es trotz der Kommunikation über RTS-CTS Signale zu Kollisionen kommen.

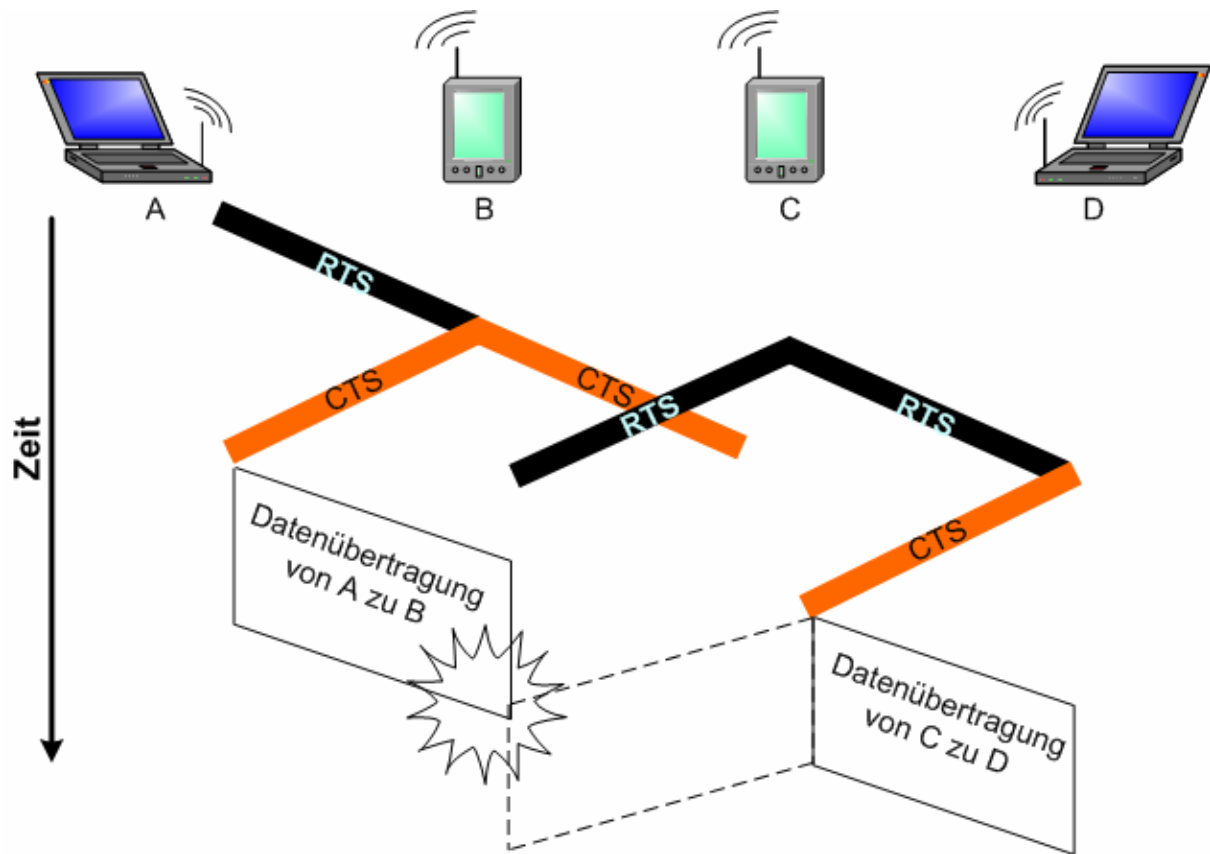


Abbildung 3-9: RTS-CTS Problem (Datenkollision)

3.2.5.1.3 Exposed Node Problem

Unter einem sogenannten „Exposed Node“ (deutsch: Ausgesetzter Knoten) versteht man einen Knoten der vollständig blockiert ist, da ein Nachbarknoten mit einem dritten Knoten kommunizieren will und somit alle Knoten, die in Reichweite des übertragenden Knoten sind, aber nicht an der Kommunikation teilnehmen dürfen bzw. wollen, für eigene Übertragungen gesperrt sind.

Da der blockierte Knoten somit zu einer eigenen Datenübertragung nicht mehr in der Lage ist, findet eine Bandbreitenverschwendung statt.

Die Abbildung 3-10 zeigt zum Beispiel vier Knoten A, B, C und D, wobei Knoten C eine Datenübertragung mit dem Knoten D vornehmen möchte.

Knoten B ist derweilen blockiert, da er in der Reichweite von C ist, möchte aber eine eigene Übertragung zum Knoten A vornehmen, welches aber durch die Blockierung nicht möglich ist.

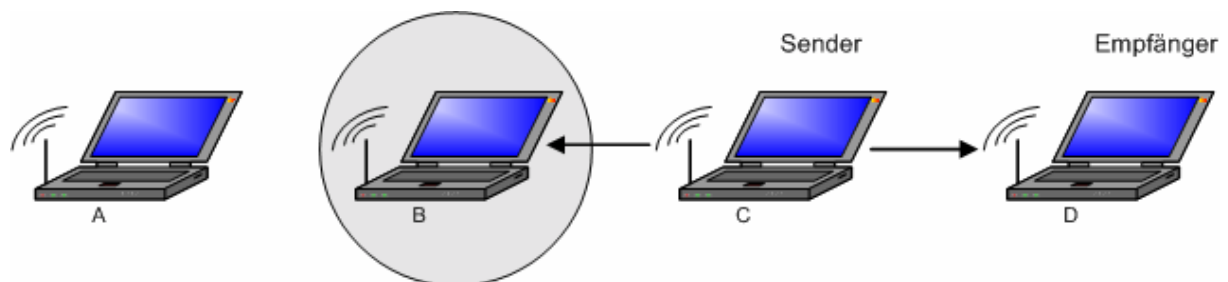


Abbildung 3-10: Exposed Node Problem

Dieses Problem tritt aber nur auf, falls man ungerichtete Antennen benutzt, also Antennen, die in alle Richtungen ihr Signal aussenden (Abbildung 3-11). Eine Lösung wäre es deswegen gerichtete Antennen zu verwenden.

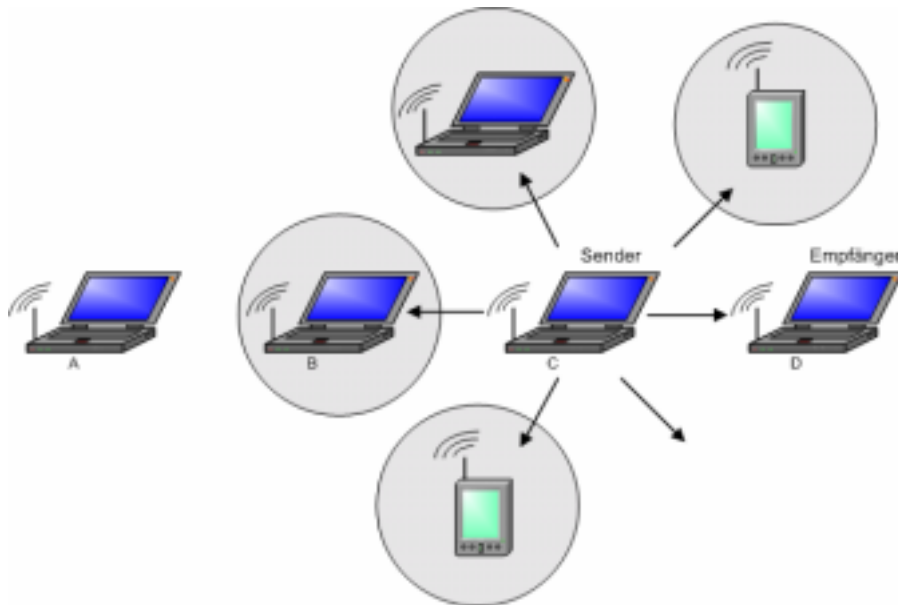


Abbildung 3-11: Exposed Node (Blockierungen bei ungerichteten Antennen)

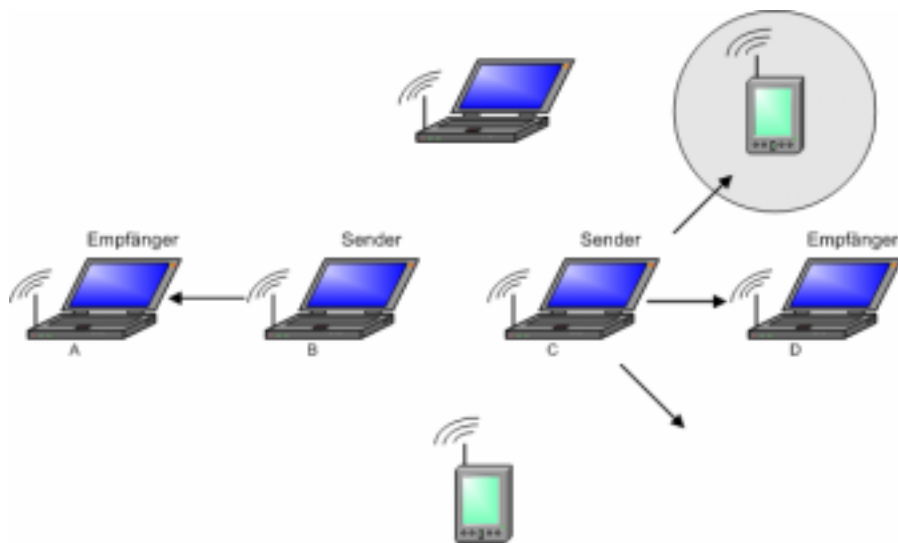


Abbildung 3-12: Exposed Node Lösung (Gerichtete Antennen)

3.2.5.2 Senderinitiierte MAC Protokolle

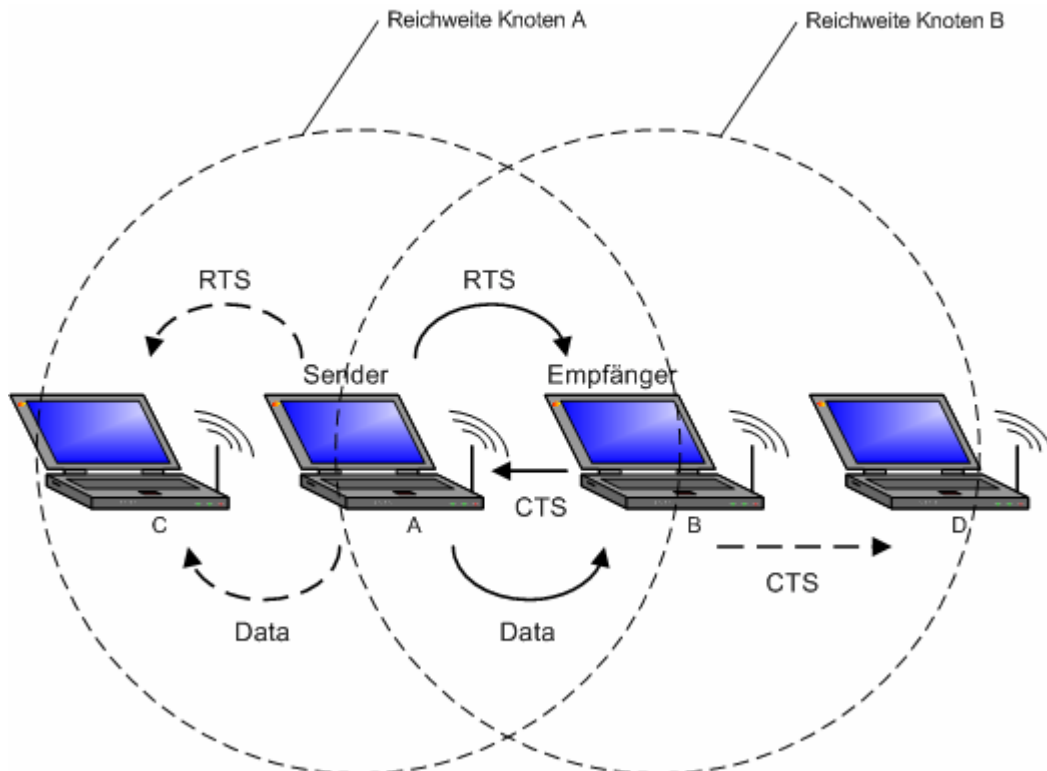


Abbildung 3-13: Senderinitiiertes MAC Protokoll

Die meisten MAC Protokolle, die wir später im Laufe der Ausarbeitung kennen lernen werden, sind senderinitiierte MAC Protokolle. Der Sender verfügt über Daten, die er versenden möchte und teilt dieses seinen Nachbarn über ein sogenanntes RTS (*Request-To-Send*) Signal mit.

Nachbarknoten, die „Interesse“ an einer Datenübertragung haben, müssen dem Sender dieses durch ein CTS (*Clear-To-Send*) Signal bestätigen. Abbildung 3-13 veranschaulicht diese Kommunikation zwischen den Knoten.

3.2.5.2.1 MACA

MACA steht für *Multiple Access with Collision Avoidance* und ist eines der etwas einfacheren MAC Protokolle für Ad Hoc Netzwerke. Es wurde vor allen Dingen zur Lösung des Hidden Terminal Problems und des Exposed Node Problems entwickelt und benutzt den Standard-3-Wege-Handshake (RTS – CTS – Daten).

Abbildung 3-14, Abbildung 3-15 und Abbildung 3-16 verdeutlichen wie der Handshake bei MACA abläuft:

1. Der Sender sendet ein RTS-Signal an seine Nachbarknoten aus, um signalisieren, dass er Daten zum Versand zur Verfügung.
2. Nachbarknoten, die nicht an der Kommunikation teilhaben wollen, blockieren³ jetzt, damit sie später nicht die Datenübertragung stören können

³ Knoten können weder Daten senden noch empfangen

3. Der Empfänger sendet nun als Bestätigung ein CTS-Signal an den Sender, um seine Bereitschaft zum Empfang der Daten zu zeigen.
4. Nachbarknoten, außer dem Sender, die dieses CTS-Signal mithören blockieren nun ebenfalls, um Störungen zu vermeiden
5. Der Sender kann nun seine Daten an den Empfänger senden.

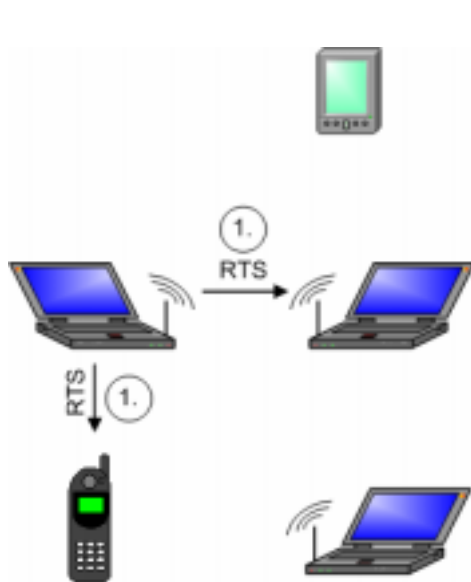


Abbildung 3-14: MACA Handshake (RTS)

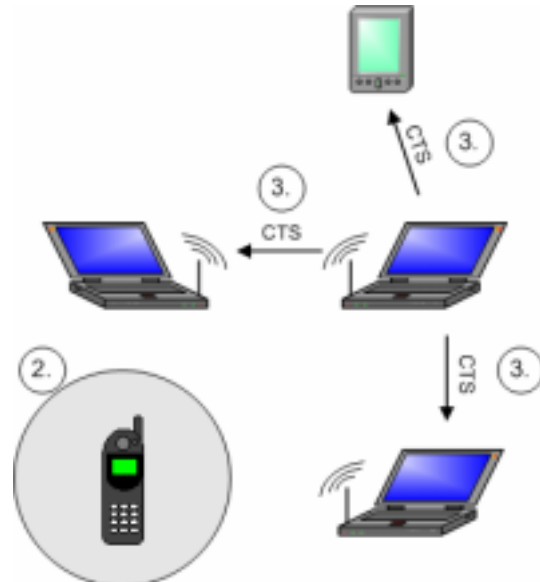


Abbildung 3-15: MACA Handshake (CTS)

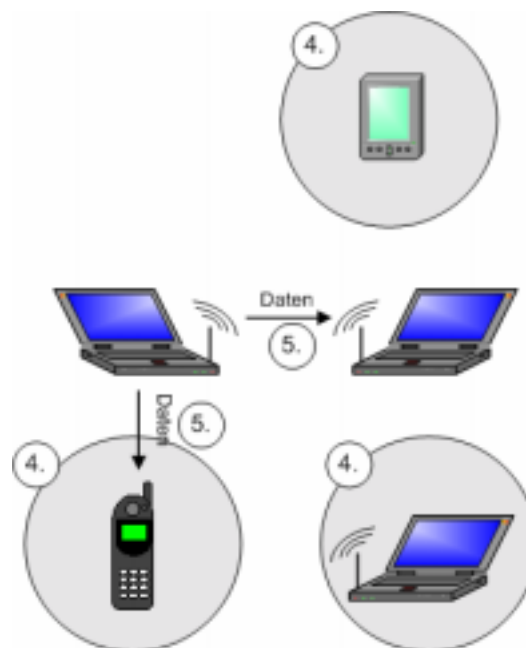


Abbildung 3-16: MACA Handshake (Daten)

Ein Vorteil von MACA und die Lösung für das Exposed Node Problem ist es, dass die Leistung von Knoten kontrolliert geregelt werden und so zusätzliche Leistung herausgeholt werden kann.

Wenn zum Beispiel ein Knoten A Daten an einen Knoten B gesendet hat, weiß A nach dieser Kommunikation wie viel Leistung er benötigt, um B zu erreichen.

Nun erhält Knoten B von einem Knoten C, der außerhalb der Reichweite von A liegt, ein RTS-Signal. Knoten B antwortet mit einem CTS-Signal, welches natürlich auch Knoten A erreicht. Jetzt kann Knoten A aber seine Leistung soweit drosseln, dass er B nicht mehr stört, aber trotzdem selber noch empfangen bzw. senden kann.

So kann man neue Verbindungen nutzen, die mit einer komplett Abschaltung von Knoten (Exposed Node) nicht möglich wäre.

3.2.5.2.2 MACAW

Da das MACA Protokoll den normalen *RTS-CTS-Data* Handshake benutzt, ist MACA auch von der RTS-CTS Problematik betroffen, die wir in einem der oberen Kapitel schon kennen gelernt haben. MACAW (*MACA with Acknowledgment*) beseitigt dieses Problem durch die Erweiterung des Handshakes um ein neues Signal: Acknowledgment (ACK).

Das ACK-Signal dient der Fehlerkorrektur und wird nach einer erfolgreichen Datenübertragung von dem Empfänger an den Sender übertragen und sorgt dafür, dass dieser erfährt, dass die Übertragung fehlerfrei verlaufen ist. Das ACK wird beispielsweise nicht gesendet, falls die Daten nicht korrekt angekommen sind. Der Sender weiß dadurch, dass er die Datenübertragung wiederholen muss.

Sollte hingegen das ACK-Signal nach einer erfolgreichen Datenübertragung nicht korrekt übertragen werden, so wird bei der nächsten RTS-Anfrage für eine erneute Datenübertragung, das ACK-Signal noch mal gesendet.

Der neue Handshake mit dem ACK-Signal sieht nun folgendermaßen aus:

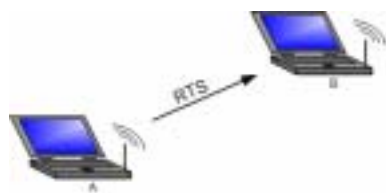


Abbildung 3-17: MACAW Handshake (RTS)

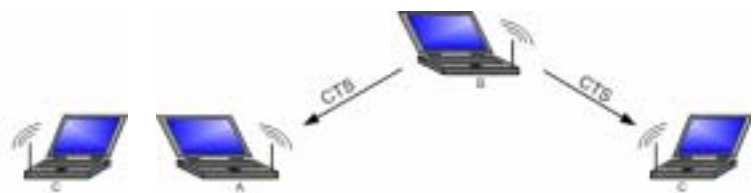


Abbildung 3-18: MACAW Handshake (CTS)

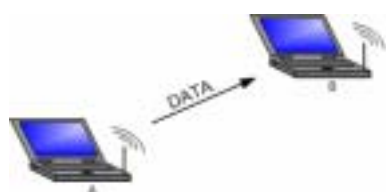


Abbildung 3-19: MACAW Handshake (Daten)



Abbildung 3-20: MACAW Handshake (ACK)

3.2.5.2.3 PAMAS

Ein Gerät in einem Ad Hoc Netzwerk verbraucht Energie, wenn es ein Paket empfängt bzw. ein Paket sendet. Dieser Energieverbrauch tritt aber auch dann auf, falls das Gerät Datenpakete „belauscht“, die eigentlich gar nicht für das Gerät bestimmt waren, sondern für einen Nachbarn. Hier wird also Energie verschwendet! [16]

Das kann man sich schon an einem Netz mit drei Teilnehmern vorstellen. Knoten A überträgt Daten an einen benachbarten Knoten B. Ein anderer Nachbar C belauscht diese Übertragung

und verbraucht unnötige Energie, da die Pakete gar nicht für ihn bestimmt waren (Abbildung 3-21). Eine Lösung wäre es, den Knoten C abzuschalten und genau hier setzt das PAMAS Protokoll an.

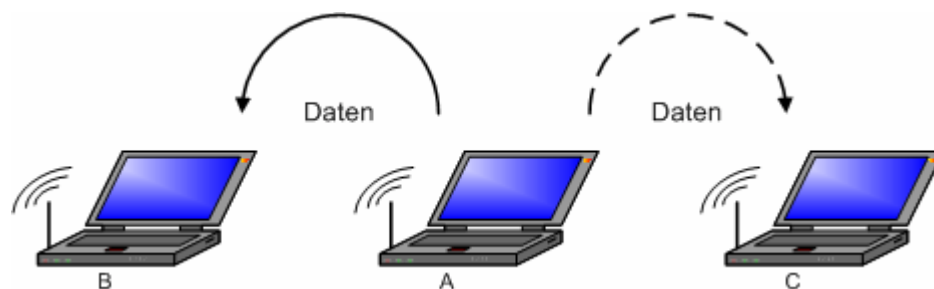


Abbildung 3-21: Energieverschwendung ohne PAMAS

Das PAMAS Protokoll (*Power-Aware Multi-Access with Signaling*) basiert auf dem MACA Protokoll. Der Unterschied besteht darin, dass es noch einen zusätzlichen Kanal für Signale hat. Über diesen Kanal läuft der Austausch der RTS-CTS Signale. Anhand des separaten Signal-Kanal können die Geräte bestimmen über welche Zeitspanne sie sich abschalten müssen.

Das PAMAS Protokoll ist ein stromsparendes Protokoll. Dies wird erreicht, indem Geräte die weder senden noch empfangen für einen bestimmten Zeitraum ausgeschaltet werden. Wenn ein Gerät Daten zu einem anderen senden will, schickt es ein RTS Signal und geht in den *'wait-for-CTS'* Zustand. Wenn es das CTS Signal empfängt, fängt das Gerät an, Daten zu senden. Empfängt es allerdings kein CTS Signal, wechselt es nach einer gewissen Zeit in einen energiesparenden Schlafzustand und sendet das RTS Signal nach einer gewissen Zeitspanne noch einmal.

Beim Empfänger läuft es ähnlich ab. Nachdem er das CTS Signal gesendet hat, geht er in den *'await-data'* Zustand. Sobald Daten empfangen werden, sendet er das Besetzt-Signal (busy tone) über den Signal-Kanal und geht über in den *'receive-data'* Zustand.

Beim PAMAS Protokoll schalten sich die einzelnen Geräte selbständig aus, wenn sie Übertragungen bemerken, die nicht an sie gerichtet sind. Die Bedingungen unter denen die Geräte selbstständig abschalten sind folgende:

- Ein Gerät hat keine Datenpakete zum Versenden und einer seiner Nachbarknoten sendet gerade Daten an ein anderes Gerät.
- Ein Gerät verfügt über Daten, die es versenden möchte, aber mindestens einer seiner Nachbarn sendet Daten an ein anderes empfangendes Gerät. In diesem Fall sollte der Transceiver abgeschaltet werden.

Bemerkung: Bei Abschalten des Transceivers kann weder gesendet noch empfangen werden.

Der Zeitpunkt in dem ein Knoten wieder aktiviert werden sollte, muss doch Ausprobieren herausgefunden werden.

Um ein besseren Datendurchsatz zu erreichen, könnte man statt beide Kanäle abzuschalten, nur den Datenkanal in einen Ruhezustand versetzen und den Signalkanal eingeschaltet lassen [19].

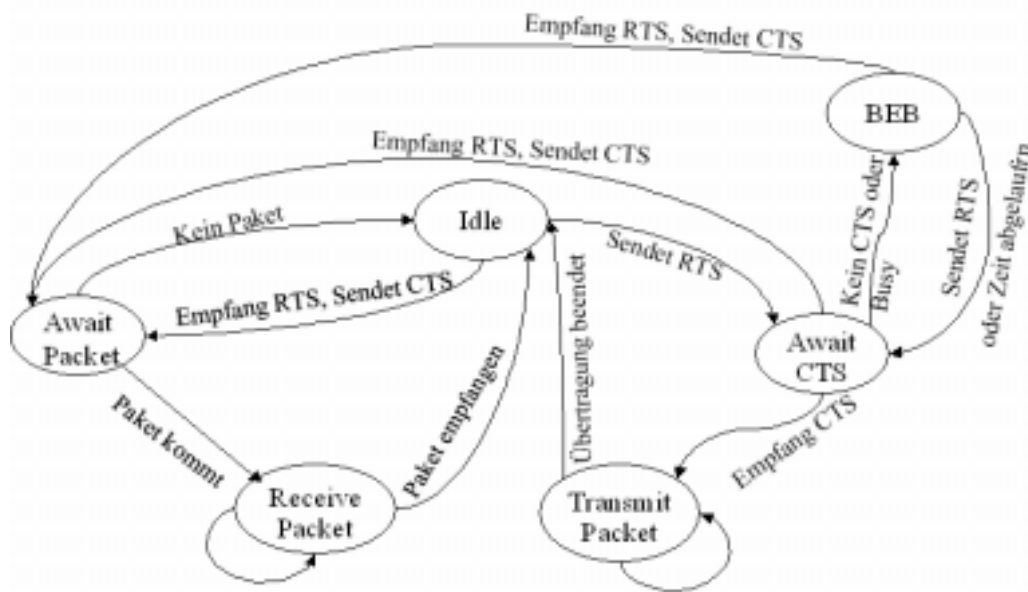


Abbildung 3-22: PAMAS Zustandsübergangsdiagramm [16]

3.2.5.2.4 DBTMA

Mit dem DBTMA Protokoll (*Dual Busy Tone Multiple Access*) soll das Hidden Terminal Problem gelöst werden. Beim DBTMA Protokoll gibt es zwei verschiedene Besetzt-Töne mit denen benachbarte Geräte über Datenübertragungen informiert werden, einen Ton für *transmit-busy* und einen für *receive-busy*. Diese Besetzt-Töne können, dadurch dass sie in verschiedenen Frequenzbereichen gesendet werden, unterschieden werden. Zusätzlich wird der eine Kanal geteilt in Daten- und Kontroll-Kanal. Daten werden über den Daten-Kanal und RTS-CTS Signale werden über den Kontroll-Kanal gesendet.

Beim DBTMA Protokoll sendet das Gerät, das Daten verschicken will, ein RTS Signal. Der Empfänger sendet den *receive-busy* Besetzt-Ton und schickt dann das CTS Signal. Alle benachbarten Knoten empfangen den *receive-busy* Ton und sind gesperrt. Vor der Datenübertragung schickt der Sender den *transmit-busy* Ton. Benachbarte Geräte empfangen den Besetzt-Ton und ignorieren alle Datenübertragungen, die sie empfangen. Abbildung 3-23 stellt den Sendeablauf der Signaltöne dar.

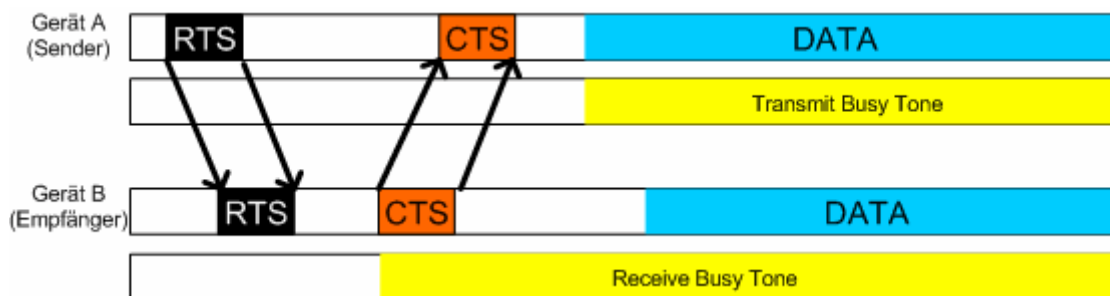


Abbildung 3-23: Dual Busy Ton beim DBTMA Protokoll

3.2.5.3 Empfängerinitiierte MAC Protokolle

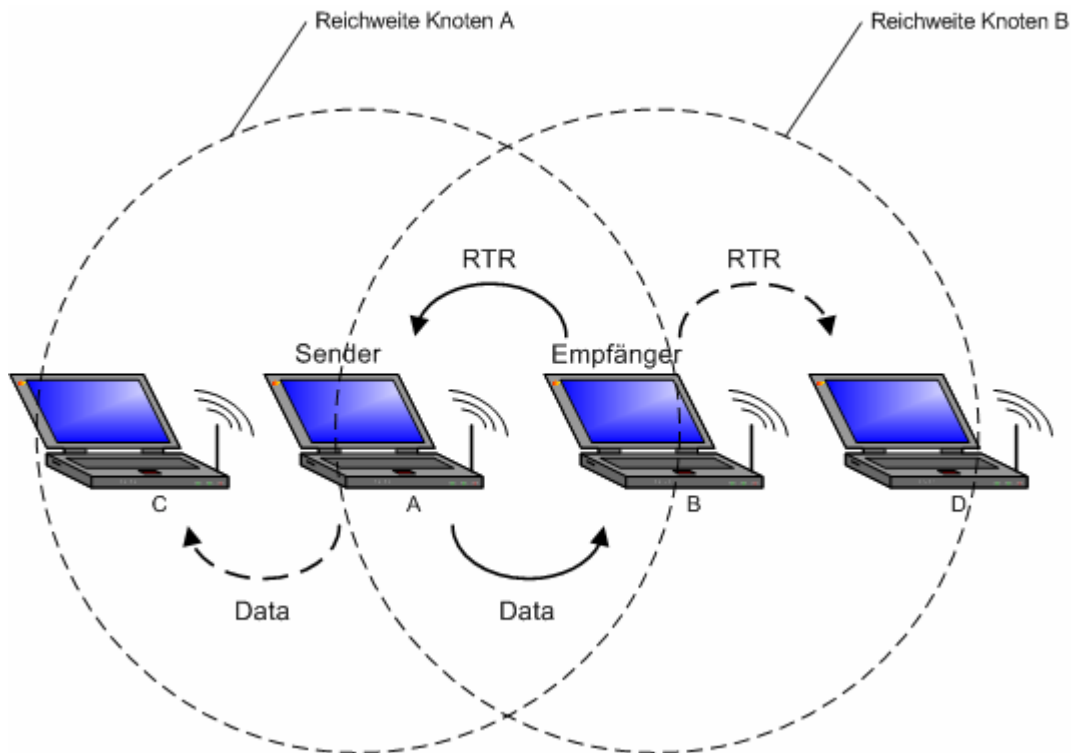


Abbildung 3-24: Empfängerinitiiertes MAC Protokoll

Beim empfangeninitiierten MAC Protokoll geht die Kommunikationsaufforderung, wie der Name schon sagt, vom Empfänger aus. Dieser teilt allen Nachbarknoten, die sich in seinem Sendebereich befinden, durch ein sogenanntes *Ready-To-Receive* (RTR) Signal mit, dass er bereit ist, Daten zu empfangen.

Die Knoten, die dieses RTR-Signal erhalten, müssen nicht unbedingt Daten zum Senden zur Verfügung haben und deswegen handelt es sich beim empfangeninitiierten MAC Protokoll um eine Art Polling (Sendeaufwurf).

Im Vergleich zum senderinitiierten MAC Protokoll wird hier nur ein Steuerungssignal benötigt, nämlich RTR. Abbildung 3-24 zeigt ein Beispiel für eine empfangeninitiierte Kommunikation zwischen einigen Netzwerkknoten.

3.2.5.3.1 MACA-BI

MACA-BI steht für „*MACA by Invitation*“ und stammt aus der Gruppe der empfangeninitiierten MAC Protokolle.

Der Kommunikationsaufbau zwischen 2 Knoten erfolgt anstelle eines 3-Wege-Handshakes (RTS-CTS-Daten) nun mit einem 2-Wege-Handshake (RTR-Daten).

RTR steht hierbei, wie anfangs schon erwähnt, für „*Request to Receive*“. Das heißt, der Empfänger sendet an den Sender das RTR-Signal wenn er bereit ist, Daten zu empfangen. Er „lädt“ den Sender sozusagen zur Datenübertragung ein.

Dieses heißt aber nicht, dass der Empfänger hundertprozentig wissen muss, dass der Sender Daten zum Senden bereithält. Er rät eigentlich eher mehr, ob Daten zur Verfügung stehen.

Abbildung 3-25 und Abbildung 3-26 zeigen, wie solch ein 2-Wege-Handshake mit dem MACA-BI Protokoll aussehen könnte. Im ersten Schritt wird das RTR-Signal vom Empfänger

gesendet und blockiert dadurch auch Geräte im unmittelbaren Umfeld, die nicht an der Kommunikation teilhaben werden. Im zweiten und letzten Schritt überträgt der Sendeknoten an seine Nachbarknoten, also auch an den Empfänger, die angeforderten Daten.

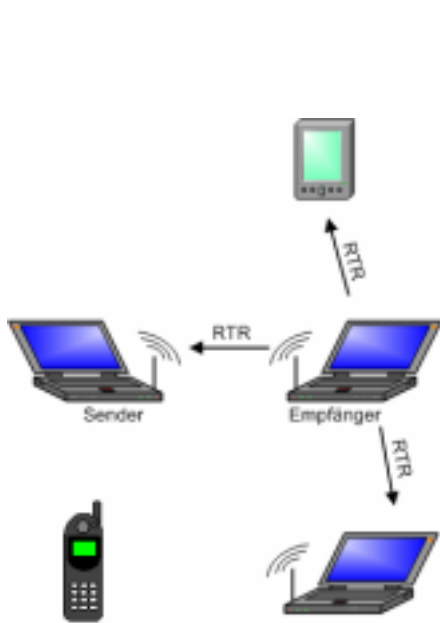


Abbildung 3-25: MACA-BI Handshake (RTR)

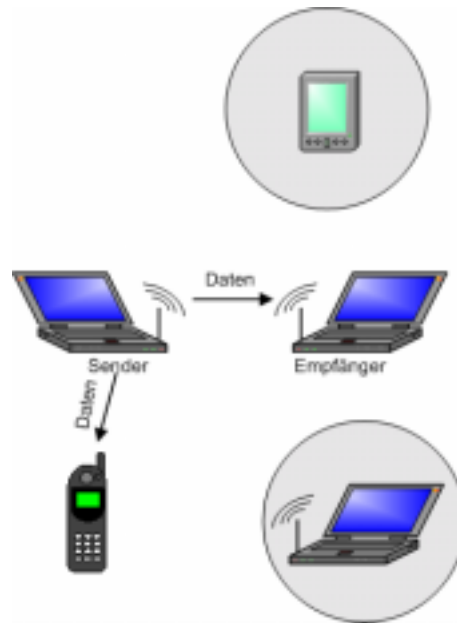


Abbildung 3-26: MACA-BI Handshake (Daten)

Damit nicht andauernd Knoten RTR-Einladungen an andere Knoten versenden und damit die reibungslose Kommunikation behindern, bietet es sich an, dass man eine Abschätzung der durchschnittlichen Transferdauer zwischen zwei Knoten vornimmt und somit eine gewisse Vorstellung hat, wie lange eine durchschnittliche Kommunikation dauert.

Damit hat man einen ungefähren Wert, in welchen Zeittakten man neue RTR-Signale versenden kann.

Bei konstantem Verkehr in einem Netzwerk, ist MACA-BI als Protokoll eine gute Wahl, da es dort eine sehr hohe Performance hat. Zudem ist es im Vergleich zu MACA, im Bereich der Kontrollsignale (RTR, RTS, CTS) nicht so kollisionsgefährdet, da es nur über die Hälfte der Anzahl der Signale von MACA verfügt.

Als Besonderheit, um die Performance zu erhöhen, bietet MACA-BI noch die Möglichkeit an, wieder zu MACA zu werden. Wenn ein Knoten viele Daten versenden möchte, aber nicht mehr die Zeit hat, auf ein RTR Signal zu warten (z.B. Grenzwertüberschreitung), dann kann er selbst ein RTS Signal senden, um eine Datenübertragung einzuleiten.

Abbildung 3-27 verdeutlicht die Unterschiede von MACA und MACA-BI beim Aufbau einer Kommunikation zwischen zwei Knoten. Hier sieht man auch deutlich, dass ein einzelnes RTR-Signal eine komplette RTS-CTS Kommunikation vollständig ersetzt.

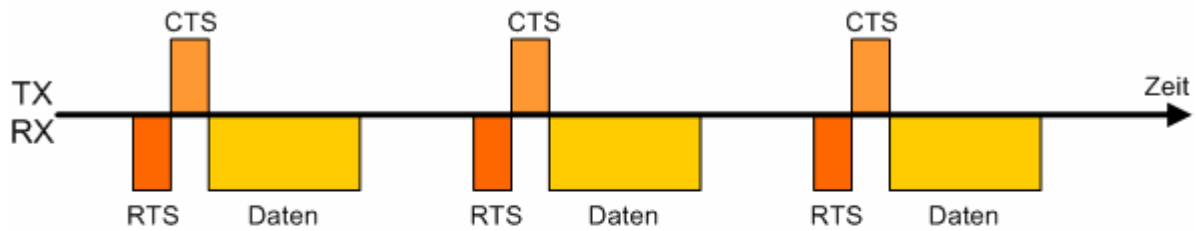
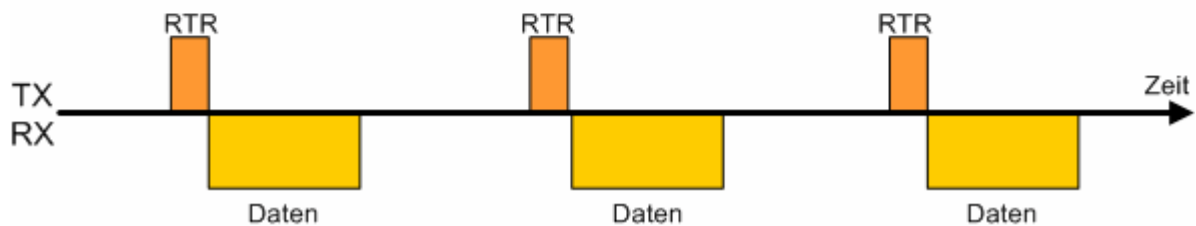
MACAMACA-BI

Abbildung 3-27: Signalunterschiede von MACA und MACA-BI

3.2.5.3.2 MARCH

MARCH steht für "*Media Access with Reduced Handshake*" und ist das zweite wichtige MAC Protokoll in der Kategorie der empfangenerinitiierten Protokolle.

Bei der Beschreibung des PAMAS Protokoll haben wir gesehen, dass das „Belauschen“ von Kontrollsignalen zu unnötiger Energieverschwendung führen kann.

MARCH hingegen macht sich diesen „Belauschungs-Effekt“, der bei der Verwendung ungerichtete Antennen auftritt, zur Nutze. Das geschieht dadurch, dass MARCH fast die Hälfte der Handshake-Signale einspart, indem es Knoten durch das Mithören von CTS-Signalen davon in Kenntnis setzt, dass Daten beim Nachbarknoten zur Verfügung stehen. Man erspart sich dadurch das erneute Versenden von RTS-Signalen.

Am Beispiel der Abbildung 3-28 kann man sich gut veranschaulichen wie so eine Kommunikation zwischen einigen Knoten aussehen kann, wenn man das MARCH Protokoll verwendet:

1. Der Knoten A möchte Daten an den Knoten B weitergeben und sendet aus diesem Grund ein RTS Signal
2. B ist mit einer Kommunikation einverstanden und bestätigt A seine Bereitschaft durch ein CTS
3. Dieses CTS wird vom Knoten C mitgehört, wodurch nun dieser weiß, dass in einiger Zeit Daten beim Knoten B vorhanden sein werden
4. A sendet die Daten an Knoten B
5. Knoten C, der durch das CTS-Signal von B von Daten bei B weiß, sendet nun ebenfalls ein CTS-Signal an B, um die Daten zu erhalten
6. Dieses CTS-Signal erreicht B, aber auch Knoten D belauscht dieses Signal und weiß nun von Daten an Knoten C
7. Knoten B sendet die Daten an C

8. Knoten D fordert nun die Daten von C, durch ein CTS-Signal, an.

Usw.

Wie man sieht, kann sich diese Kommunikation über beliebig viele Knoten hinweg ziehen und man kommt ganz ohne erneute RTS-Signale aus und sparte dadurch eine Menge an Handshake-Signalen.

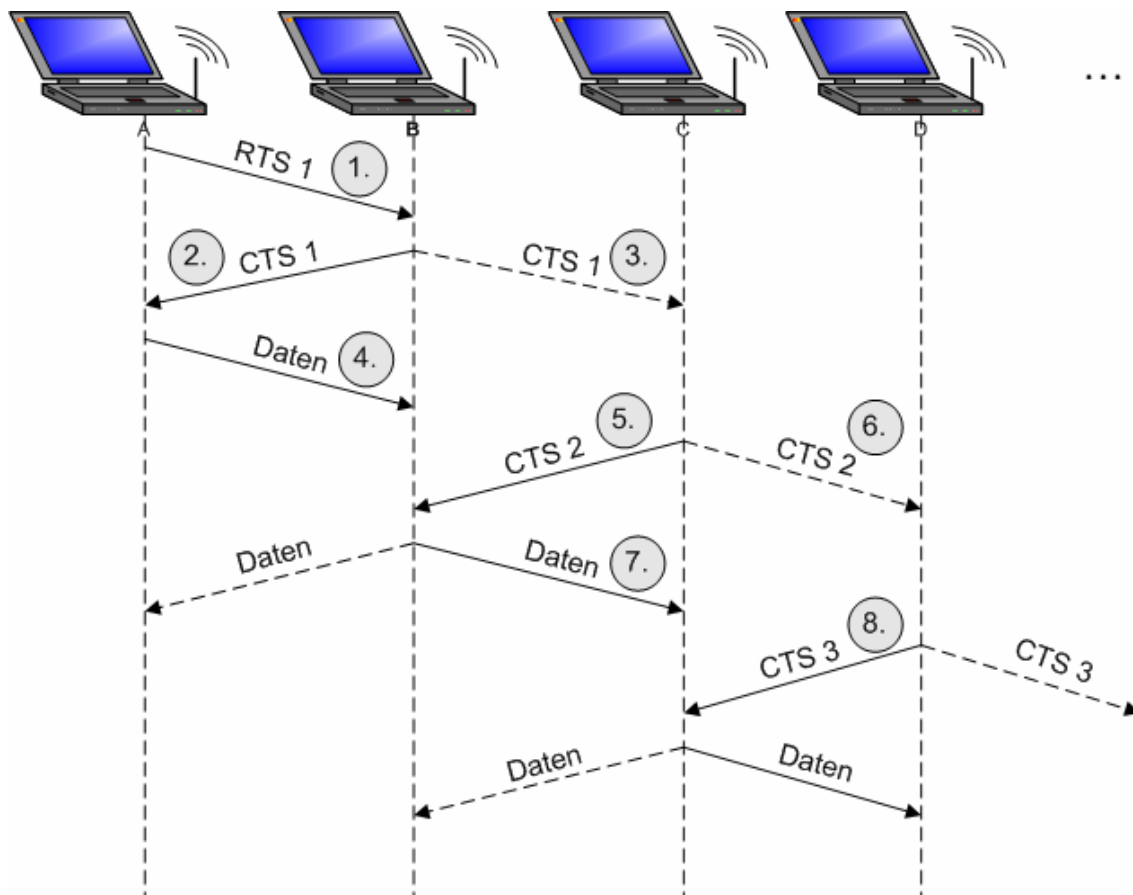


Abbildung 3-28: MARCH Handshake

4 Bluetooth

4.1 Einführung

Seit 1998 entwickeln IBM, INTEL, Ericsson, Nokia und Toshiba als "Bluetooth Special Interest Group" unter dem Bluetooth (Blauzahn) eine Technologie für die drahtlose Übermittlung von Sprache und Daten per kurzer Radiowellen. Die Bluetooth-Technik nutzt das frei verfügbare Funknetz ISM (Industrial Scientific Medical), das mit 2.45-GHz arbeitet. Die Übertragungsleistung soll bis zu 1 MBit pro Sekunde bei einer Reichweite von 12 Metern betragen. Leistungsverstärkt soll sogar eine Reichweite bis zu 100 Meter möglich sein. Im Höchstfall können 127 Geräte miteinander verbunden werden.

4.2 Architektur

4.2.1 Piconet

Das Piconet ist eine Ansammlung von mobilen Endgeräten die ein gemeinsames örtlich begrenztes Netzwerk bilden. Ein Piconet beginnt mit zwei verbundenen Geräten z.B. Notebook und Mobiltelefon, und kann bis auf acht Endgeräte ausgedehnt werden. Alle Bluetooth Geräte sind gleichberechtigte Einheiten im Netz, dennoch operiert ein Gerät als Master und alle anderen Geräte als Slave. Sind noch zusätzliche Geräte in Reichweite sind diese entweder nicht aktiv, oder werden als *parked* deklariert. Die einzelnen Slaves haben untereinander keine Verbindung, sondern sind nur mit dem Master verbunden. Kommunikation unter Slaves ist nur möglich wenn die Pakete vom Master weitergeleitet werden.

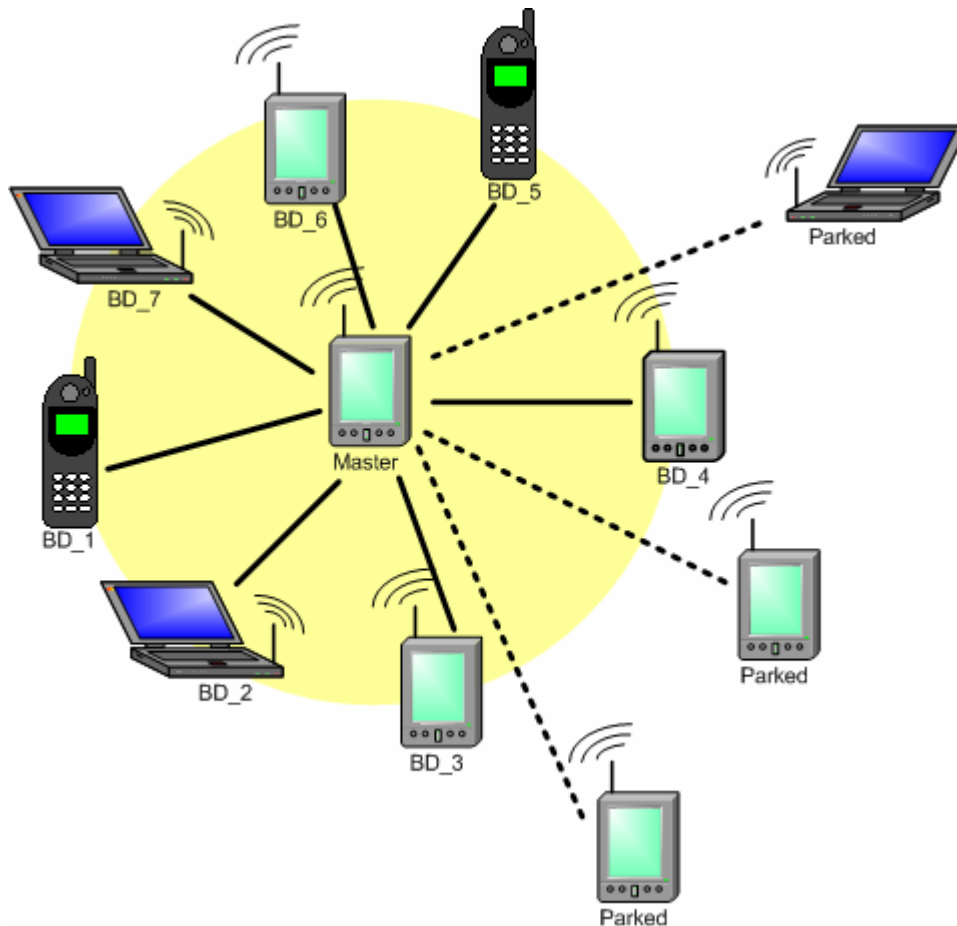


Abbildung 4-1: Bluetooth Piconet

4.2.2 Scatternet

Ein Piconet deckt, wie wir oben gesehen haben, einen bestimmten Sendebereich ab, in dem sich die Geräte aufhalten können, die mit dem MASTER kommunizieren wollen.

Was passiert aber, wenn sich ein Gerät im Senderadius von zwei unterschiedlichen Piconets befindet? Dann bildet sich ein sogenanntes *Scatternet*.

Geräte, die sich in beiden Piconets befinden, müssen sich die Zeit, die sie in den jeweiligen Netzen verbringen, durch die Benutzung von Zeitfenstern (englisch: timeslots) einteilen. Das heißt, dass ein Gerät erst eine gewisse Zeit in dem einen Piconet verbringt und mit diesem kommuniziert und dann wieder für eine Zeitspanne in das andere Netz wechselt.

Die Unterscheidung zwischen MASTER und SLAVE ist hier auch klar geregelt. Wenn ein Gerät die MASTER-Rolle in einem Piconet übernommen hat, kann es nicht gleichzeitig MASTER für das andere Piconet sein. Dafür ist es aber möglich, dass dieser MASTER gleichzeitig SLAVE im anderen Piconet ist. An Abbildung 4-2 ist dieses sehr gut zu erkennen. Der PDA aus dem rechten Piconet, in dem er MASTER ist, fungiert als SLAVE im linken Piconet.

Es ist auch möglich, dass ein SLAVE-Gerät SLAVE für zwei verschiedene Piconets ist. Dieses Gerät bildet, weil es sich ja im äußeren Bereich des Radius befindet, eine „Brücke“ (Gateway) zwischen den Netzen und leitet Datenpakete zwischen diesen weiter.

Unter diesen Hin- und Herschalten von Geräten leidet natürlich auch die Geschwindigkeit mit der in den Netzen gesendet wird. Zur Zeit ist aber die Unterteilung in Pico- und Scatternets die einzige bei der Bluetooth-Technologie.

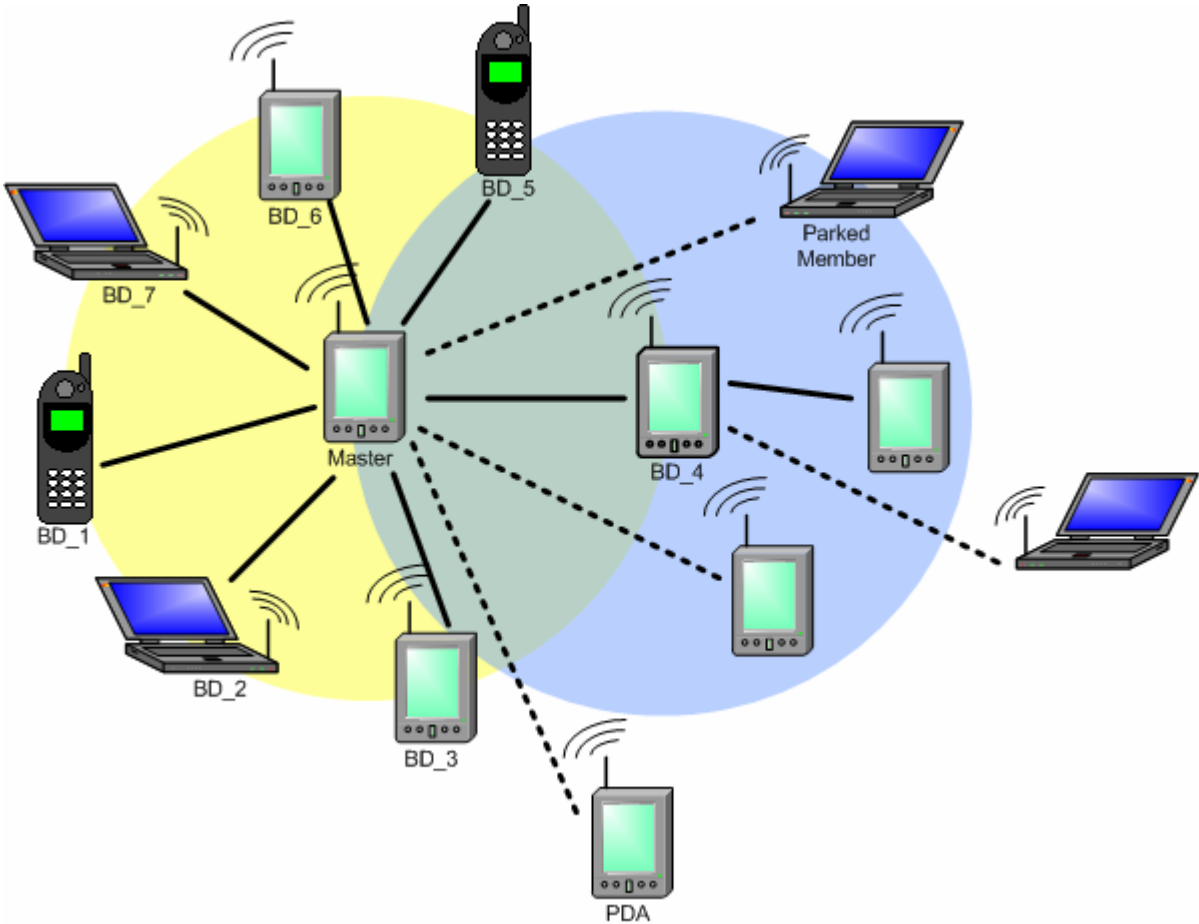


Abbildung 4-2: Bluetooth Scatternet

4.3 Protokolle

4.3.1 Protokollstack

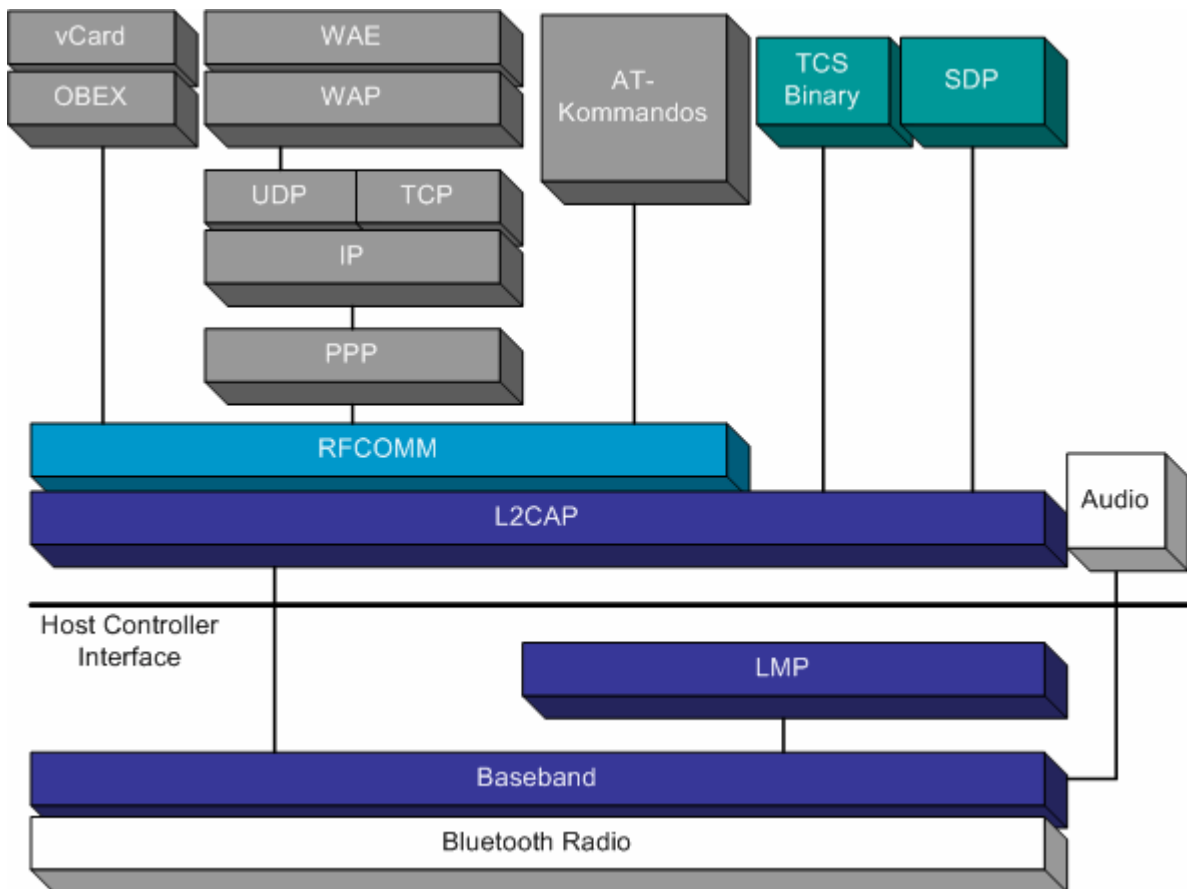


Abbildung 4-3: Bluetooth Protokollstack

Der Bluetooth Protokollstack verwendet, wie man in Abbildung 4-3 sehen kann, viele alt bekannte Protokolle wie zum Beispiel PPP, IP oder TCP. An diese Kompatibilität wurde extra beim Spezifizieren gedacht, damit schon bestehende Anwendungen, die andere Übertragungsverfahren außer Bluetooth nutzen, leicht wiederverwendet werden können. So könnten zum Beispiel Anwendungen, die OBEX (*Object Exchange Protocol*) benutzen, schnell und einfach für Bluetooth nutzbar gemacht werden.

Bluetooth-spezifisch und neu hinzugekommen sind die Protokollschichten L2CAP (*Logical Link Control and Adaptation Protocol*) und LMP (*Link Manager Protocol*). Diese und die anderen Schichten werden weiter unten im Kapitel näher beschrieben.

4.3.2 Bluetooth Kernprotokolle

Bluetooth Kernprotokolle sind Protokolle, die extra für Bluetooth entwickelt wurden. Zu diesen zählen das Baseband, der Link Manager und das Logical Link and Adaption Protokoll.

4.3.2.1 Baseband

Das Basisband realisiert die physikalische HF-Verbindung (Hochfrequenz⁴) zwischen den einzelnen Stationen in einem Piconet. Die Übertragung findet paketvermittelt statt, wobei in definierten Zeitschlitzen Daten, über sich nach festgelegtem Muster ändernden Frequenzen, gesendet werden (*Frequenz Hopping*). Die Sprungsequenzen müssen natürlich festgelegt und synchronisiert werden.

Es existieren zwei verschiedene Übertragungsvarianten: synchron verbindungsorientiert (SCO) und asynchron verbindungslos (ACL).

Unterschiedliche MASTER-SLAVE Paare in einem Piconet können zwischen diesen beiden Varianten frei wählen. Sie können sogar während einer Session die Übertragungsart wechseln.

SCO Verbindungen werden meistens dazu genutzt, um Audiodaten zu übertragen, da hier eine feste Bandbreite durch eine Punkt-zu-Punkt Verbindung garantiert wird. Das heißt, es wird ein fester Zeit-Slot bestimmt, in dem der Slave senden darf. Dazu schickt der MASTER dem SLAVE ein Paket, das der SLAVE im nächsten Slot beantworten darf. SCO Pakete werden ohne Prüfsummen verschickt. Innerhalb eines Piconets können bis zu drei Audio-(Sprach-) Verbindungen gleichzeitig bestehen. Die Bandbreite beträgt 64 kbit/s.

ACL Verbindungen hingegen werden hauptsächlich für die Übertragung von Datenpaketen benutzt. Dabei werden asymmetrische sowie symmetrische Point-to-Multipoint Verbindungen unterstützt.

Falls keine automatische Fehlerkorrektur genutzt wird, können Multi-Slot Pakete (asymmetrische Verbindung) eine Übertragungsgeschwindigkeit von bis zu 721 Kbit/s in die eine und bis zu 56 Kbit/s in die Gegenrichtung erreichen. Bei einer symmetrischen Verbindung wird eine Geschwindigkeit von maximal 444 Kbit/s in beide Richtungen erreicht.

Der MASTER kontrolliert die Symmetrie des Traffics und die Bandbreite im Netz, wobei beim Letzteren der MASTER auch noch entscheidet, wie viel Bandbreite ein SLAVE im Piconet nutzen darf.

ACL Verbindungen unterstützen zudem auch das Versenden von Broadcast Paketen, zum Beispiel vom MASTER zu allen SLAVES im Piconet.

Die Fehlerkorrektur findet mit dem sogenannten ARQ-Schema statt (*Automatic Retransmission Query*). Bei jedem Empfang eines Datenpaketes wird mittels eines Prüfsummenchecks (CRC) überprüft, ob alles korrekt angekommen ist. Falls dies nicht der Fall ist, wird in dem Antwortpaket sofort eine Benachrichtigung zurückgesendet und es erfolgt eine erneute Übertragung des Datenpaketes. So entstehen nur Zeitverzögerungen von einem Slot bei fehlerhaften Paketen.

Dieses ARQ-Schema ist natürlich nicht optimal für die Übertragung von Sprache, da hier die Verzögerungen die Qualität erheblich behindern würden. Aus diesem Grund wird bei SCO-Verbindungen ein Sprach-Kodierungs-Schema benutzt, welches sehr resistent gegenüber Bitfehlern ist. Bitfehler, die nicht korrigiert werden können machen sich durch Hintergrundgeräusche in der Sprache bemerkbar.

⁴ Frequenzbereich von 10Kz-300Ghz

Baseband Zustände:

Zustand	Status des Gerätes
StandBy	Wartet darauf, einem Piconet beizutreten.
Inquiry	Versucht andere Geräte zu finden, mit denen es eine Verbindung aufbauen kann.
Page	Versucht mit einem bestimmten anderen Gerät, eine Verbindung herzustellen.
Connected	Ist aktiv in einem Piconet.
Park/Hold/Sniff	Mit anderem Gerät verbunden und befindet sich im Stromsparmodus.

Tabelle 4-1: Baseband Modi

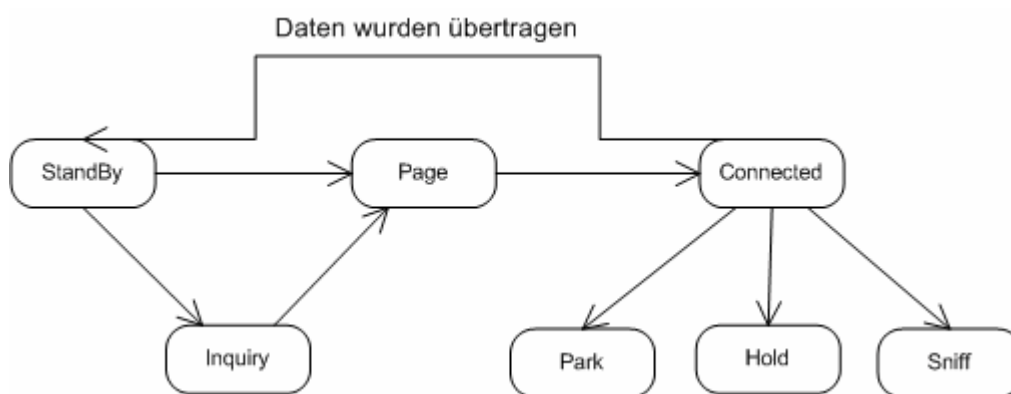


Abbildung 4-4: Bluetooth Zustandsübergangsdiagramm

4.3.2.2 Link Manager Protokoll (LMP)

Das Link Manager Protokoll zählt, wie wir oben schon gesehen haben, zu den Bluetooth-spezifischen Protokollen. Es ist hauptsächlich für den Verbindungsaufbau zwischen Bluetooth-Einheiten verantwortlich, übernimmt aber auch andere wichtige Aufgaben.

Zu diesen zählen zum Beispiel die Kontrolle der Paketgrößen während einer Datenübertragung, die Verwaltung der Energiemodi und des Energieverbrauches. Zudem ist das Link Manager Protokoll für den Status der Geräte im Piconet verantwortlich und übernimmt auch sicherheitsrelevante Aufgaben, wie zum Beispiel die Kontrolle von Schlüsseln für die Verschlüsselung von Daten.

4.3.2.3 Logical Link Control und Adaptation Protokoll (L2CAP)

Die Basis der Kernprotokolle bildet das sogenannte *Logical Link and Adaption* Protokoll (L2CAP). Es verbindet die aufgesetzten Protokolle mit den unteren Protokollschichten. Er ist zuständig für das Zerteilen und Zusammenfügen von Paketen. Da der Transportlayer nur kleine Pakete (max. 64KB) versenden kann, die aufgesetzten Protokolle aber mit größeren Paketen arbeiten, werden diese vom L2CAP geteilt. Empfangene Pakete werden wieder zusammengesetzt bevor sie an die höheren Protokolle weitergeleitet werden.

Des Weiteren werden verbindungsorientierte und verbindungslose (Loopback) Verbindungen für die höheren Protokollschichten ermöglicht. Das L2CA Protokoll unterstützt nur ACL Datenverbindungen, und keine SCO-Verbindungen.

4.3.3 Host Controller Interface (HCI)

Das Host Controller Interface dient der Ansteuerung der Schichten Link Manager und Baseband. Die Schnittstelle wurde entwickelt, um die einheitliche Ansteuerung von Bluetooth-Chipsets zu ermöglichen. Somit ist es Applikationen, die auf den oberen Protokollschichten aufbauen, möglich, mittels standardisierter HCI-Ansteuerung auf Bluetooth-Endgeräte von unterschiedlichen Herstellern zuzugreifen.

Aus diesem Grund spielt die HCI-Schnittstelle auch bei Qualitätstests eine wichtige Rolle. [17]

4.3.4 Bluetooth Service Discovery Protokoll (SDP)

Das Service Discovery Protokoll ist eine wichtige Komponente im Protokollstack, da es für die Umsetzung des Client-/Serverprinzips bei Bluetooth verantwortlich ist. Somit ist es möglich, dass Geräte untereinander auf Dienste (Services) des jeweils anderen Gerätes zugreifen können.

Jedes im Netz befindliche Bluetooth-Gerät verfügt über einen sogenannten SDP-Server, der Informationen über die angebotenen Services für andere Netzteilnehmer bereitstellt. Diese Informationen beinhalten die Art des Dienstes und ggf. erforderliche Parameter und Eigenschaften. Der SDP-Server selbst ist als Datenbank aus sogenannten „*Service Records*“ aufgebaut, die diese Daten enthalten.

Für den Austausch der Service-Informationen ist eine bestehende L2CAP-Verbindung notwendig und auch für die Nutzung des jeweiligen Dienstes wird eine L2CAP-Verbindung benötigt. Abbildung 4-5 zeigt, welche Protokollschichten insgesamt in einen Austausch von Diensten verwickelt sind.

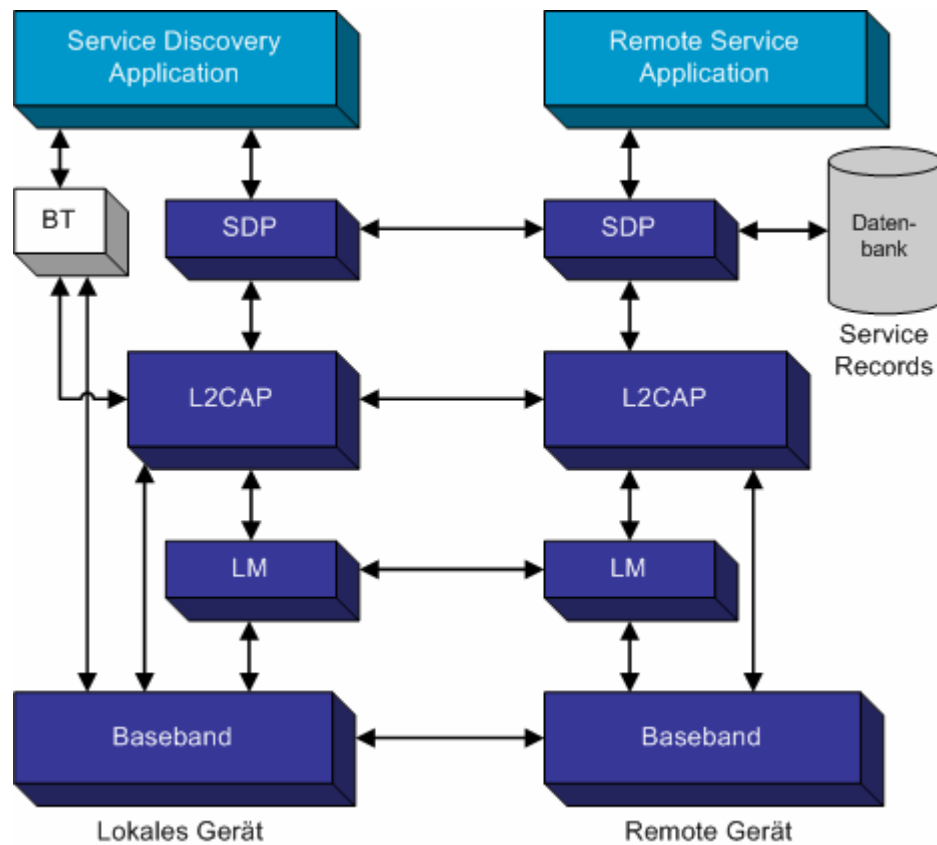


Abbildung 4-5: Bluetooth Service Discovery

Falls ein Gerät aus Gründen der Privatsphäre oder aus Energiegründen keine Dienste oder nur ausgewählte Dienste anbieten möchte bzw. kann, ist es auch möglich, diese Dienste vor anderen Netzteilnehmern zu „verstecken“.

4.3.5 RFCOMM

In Anlehnung an die zur seriellen Kommunikation via Kabel verwendeten COM-Ports wurde die sogenannte RFCOMM-Schnittstelle in den Bluetooth-Protokollstack aufgenommen. Die Integration dieser Schnittstelle soll Anwendungsentwicklern die Portierung bereits bestehender Anwendungen auf Bluetooth-basierte Hardware erleichtern. RFCOMM bietet Signalkompatibilität zum RS-323 Standard und erlaubt das Multiplexing von bis zu 60 seriellen Ports.

4.3.6 TCS Binary

Die TCS (*Telephony Control Protokoll Spezifikation*) beschreibt wie Telefonanrufe über eine Bluetooth-Verbindung gesendet werden können. Das Protokoll unterstützt *CallControl* (CC), *GroupManagement* (GM), und *ConnectionlessSignaling* (CTCS) um damit Ton und Sprache zu übertragen. Call Control ist für den Ruf Auf- und Abbau zuständig. Es unterstützt *Point-to-Point* und *Point-to-Multipoint* Verbindungen. Group Management verwaltet Gruppen von Geräten, und unterstützt die Suche nach verbindungsreifen Geräten. Gruppen von Geräten die über das TCS Protokoll kommunizieren nennt man *Wireless User Groups* (WUG). Alle Sprechverbindungen werden in Bluetooth über SCO gesendet. Da bei Übertragungsfehlern die

Daten nicht wiederholt gesendet werden können wird ein Sprachkodierungsschema eingesetzt, das auf CVSD-Modulation (*Continuous Variable Slope Delta Modulation*) basiert. Dieses Schema folgt dem analogen Signal und ist weitgehend unempfindlich gegenüber Bitfehlern. Diese werden als Hintergrundrauschen wahrgenommen, das sich mit Fehlerhäufigkeit verstärkt.

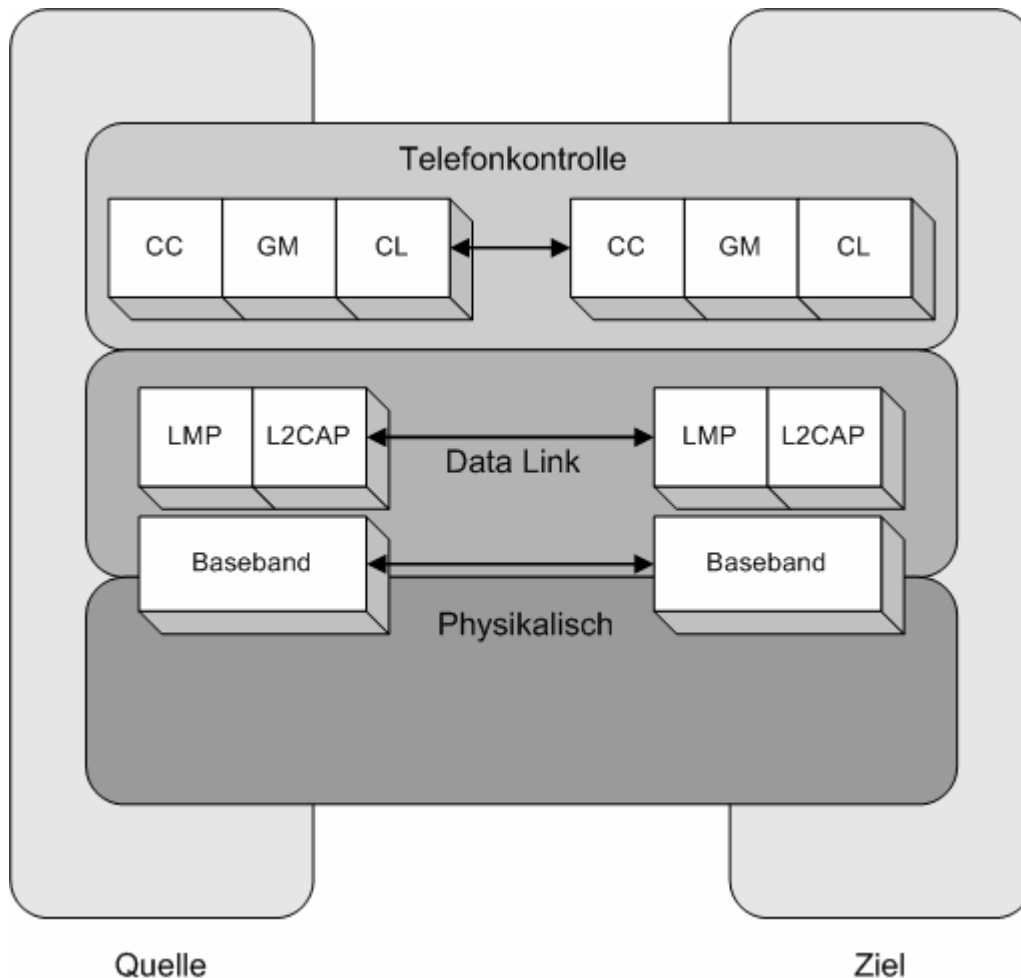


Abbildung 4-6: TCS Ablauf

4.3.7 Aufgesetzte Protokolle

4.3.7.1 PPP

Das *Point-to-Point* Protokoll baut auf der RFCOMM Schnittstelle auf, um Point-to-Point Verbindungen mit Bluetooth möglich zu machen.

PPP ist ein paketorientiertes Protokoll und muss aus diesem Grund einen Mechanismus zur Serialisierung benutzen, um den Datenstrom in einer seriellen Strom zu verwandeln.

4.3.7.2 TCP/UDP/IP

TCP (*Transport Control Protocol*), UDP (*User Datagram Protocol*) und IP (*Internet Protocol*) wurden als Standards bei Bluetooth mitdefiniert, um es Bluetooth-Einheiten zu ermöglichen,

mit anderen LAN- und WLAN-Einheiten, wie beispielsweise dem Internet, zu kommunizieren. Deshalb können Bluetooth-Geräte auch als Gateways ins Internet dienen. Hierbei wird vor allen Dingen die Kombination mit PPP genutzt⁵.

4.3.7.3 OBEX

OBEX steht für *Object Exchange Protokoll* und ist ein, für den einfachen Austausch von Objekten zwischen zwei Geräten, entwickeltes Protokoll. OBEX benutzt das Client-/Servermodell und ist unabhängig von der benutzten Transportart und der Transport-API.

OBEX beinhaltet zudem ein Objekt zum Auflisten von Verzeichnissen, so dass damit auf einfache Art und Weise Verzeichnisse des Remote-Gerätes ausgelesen werden können. Als Haupttransportschicht wird RFCOMM von OBEX genutzt.

4.3.7.4 Inhaltsformate vCard und vCalendar

Bei vCard und vCalendar handelt es sich um zwei vom Internet Mail Consortium (IMC) herausgegebene Spezifikationen, um einfach und schnell Visitenkarten bzw. Termine auszutauschen.

Das Format vCard ist für viele Geräte nutzbar wie zum Beispiel PDAs, PIMs (Personal Information Manager), SmartCards usw.. Eine vCard kann einfach Textdaten, aber auch Bilderdaten enthalten.

VCalendar bietet die Möglichkeit Termin- und Aufgabenplanungen zwischen Bluetooth-Geräten zu übertragen und ist somit für Arbeit in Gruppen nützlich.

4.3.7.5 WAP

WAP steht für *Wireless Application* Protokoll und ist eine Spezifikation, welche für die unterschiedlichsten drahtlosen Geräte nutzbar ist, um Verbindung mit dem Internet herzustellen.

Bluetooth kann wie viele andere drahtlose Netze dazu genutzt werden, Daten zwischen WAP-Client und Server zu übertragen und bietet zudem durch seine Ad Hoc Fähigkeiten mehr Möglichkeiten als andere Netze.

4.4 Bluetooth Audio

Bluetooth Audio bietet die Möglichkeit schnell und unkompliziert Audiodaten mit Bluetooth zu übertragen. Es werden dafür SCO (Synchronous Connection Oriented) Kanäle verwendet, da es sich hierbei um eine zeitkritische Anwendung handelt. Beim Verbindungsaufbau wird nicht über den L2CAP Layer gegangen, sondern die Daten werden nach Verbindungsaufbau direkt zwischen den Bluetooth-Einheiten übertragen.

⁵ Bei Bluetooth 1.0 und zukünftigen OBEX-Versionen.

4.5 Bluetooth MAC

Bei Bluetooth ist der Kanal in Zeitschlitz (Time Slots) geteilt, wobei jeder eine Länge von $625\mu\text{s}$ besitzt. Die Zeitschlitz werden durch die Bluetooth-Clock des Piconet Masters nummeriert. Die Nummerierung reicht von 0 bis 2^{27} und ist zyklisch, mit einer Länge von 2^{27} . In diesen Zeitschlitz können Master und Slave Pakete übermitteln. Dazu wird ein Time-Division-Duplex-Schema (TDD) genutzt. Der Master beginnt mit seiner Übertragung nur in gerade nummerierten Zeitschlitz und der Slave nur in ungeraden. Der Paketkopf sollte auf den Anfang des Slots ausgerichtet sein. Die Pakete können sich über bis zu 5 Slots erstrecken. Für ein Einzelpaket wird die benötigte Slot-Nummer aus dem Bluetooth-Clock-Wert gewonnen. Ein Multi-Slot-Paket erhält seine RF-Sprungfrequenz aus dem Bluetooth-Clock-Wert des ersten Slots des Pakets.

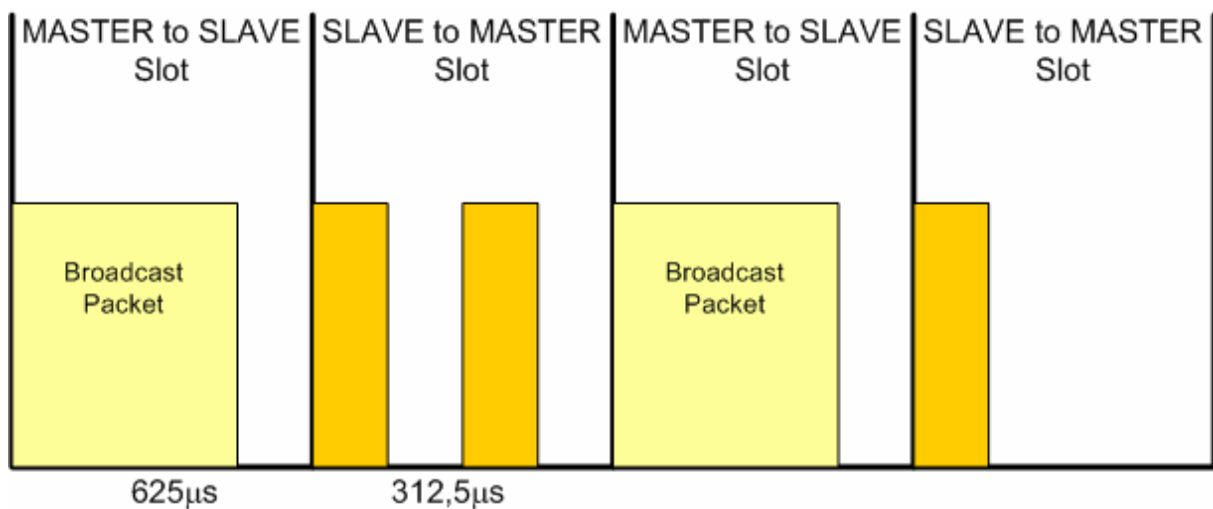


Abbildung 4-7: Bluetooth Zeitfenster (time slots)

4.6 Paketstruktur

Die Bluetooth-Paketstruktur besteht aus dem Access Code (68 oder 72 Bit), einem 54 Bit langen Header, und den eigentlichen Daten (0 bis 2745 Bit). Der Access Code wird benutzt um die Pakete zu unterscheiden. Der Header enthält Kontrollinformationen (Adressierung, Verbindung etc). Im Paket-Typ des Headers stehen Informationen über die Art der Datenpakete und welche Fehlerkorrektur im Datenteil verwendet wird. Das Flow-Signal wird verwendet um anzuzeigen wenn der Empfangspuffer voll ist.

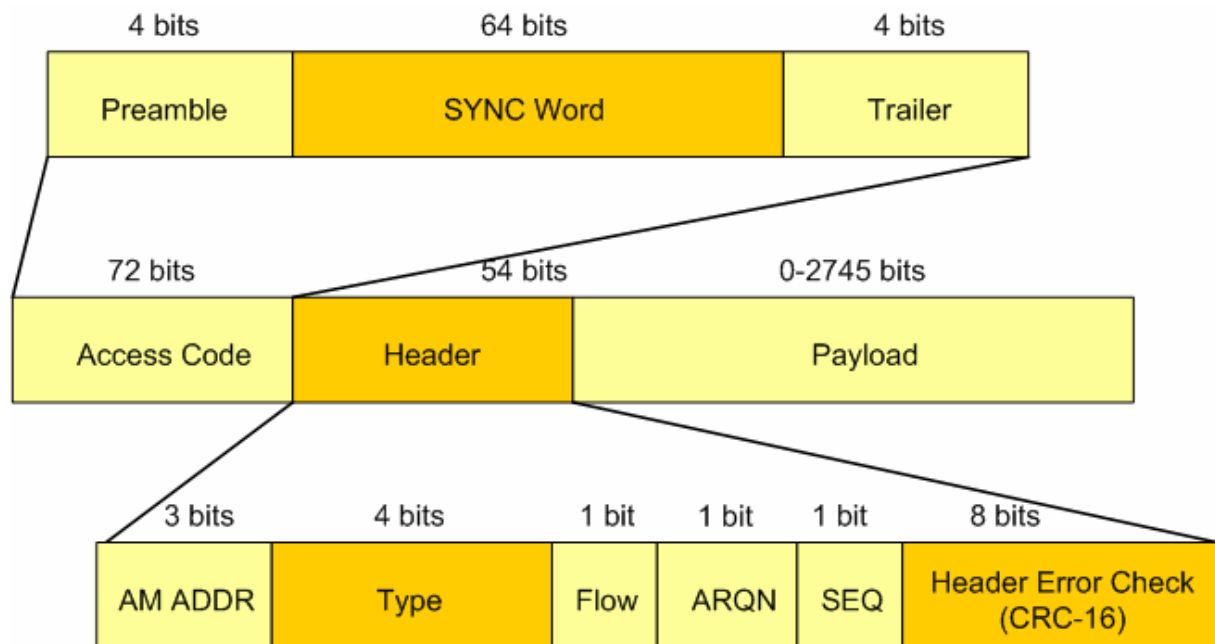


Abbildung 4-8: Bluetooth Paketstruktur

4.7 Adressierung

Jedes Bluetooth Gerät hat eine 48-bit IEEE MAC Adresse. Diese ist geteilt in den Non-significant- Adress Part (NAP), den Upper-Adress-Part (UAP), und den Lower-Adress-Part (LAP). Der NAP wird gebraucht um den Entschlüsselungsalgorithmus zu initialisieren. Mit dem Upper-Adress-Part werden Berechnungen für HEC,CRC, und Frequency Hopping durchgeführt. Der Lower-Address-Part wird auch zur Berechnung des Frequency Hoppings gebraucht und zusätzlich zur Sync-Word-Generierung.

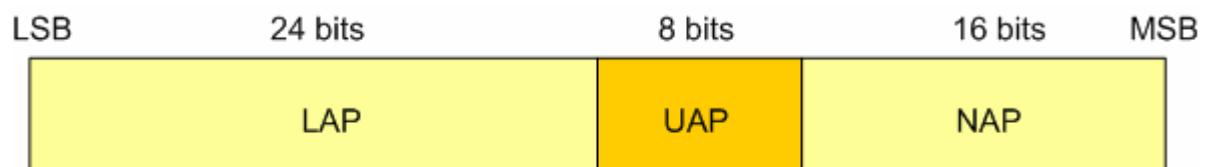


Abbildung 4-9: Bluetooth 48-bit MAC Adresse

5 Infrarot

5.1 Einführung

Der Infrarot-Standard wurde von der *Infrared Data Association* (IrDA) entwickelt und zeichnet sich vor allen Dingen durch den geringen Energieverbrauch aus, was auch notwendig ist, da Infrarotgeräte fast ausschließlich nur im mobilen Bereich zum Einsatz kommen.

Man unterscheidet bei Infrarot zwei Kommunikationsstandards, IrDA Data und IrDA Control. Den Letzteren werden wir in diesem Kapitel zwar erwähnen und kurz beschreiben, aber der Schwerpunkt liegt eindeutig auf IrDA Data, damit diesem Standard die wirklich interessanten Anwendungen möglich sind.

Die Entfernungen, die man mit Infrarotübertragungen zurücklegen kann, bewegen sich zwischen 1cm und maximal 8m, wobei Punkt-zu-Punkt Verbindungen nur in einem Ausstrahlungsbereich von bis zu 30° möglich sind (siehe Abbildung 5-1).

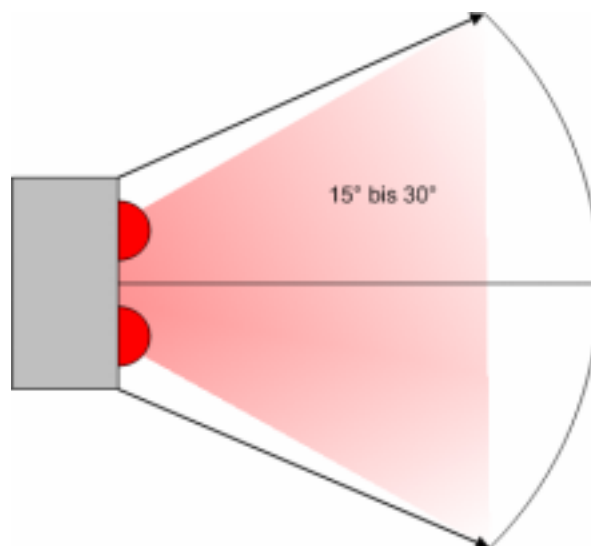


Abbildung 5-1: Infrarot-Ausstrahlungsbereich

5.2 IrDA Control

IrDA Control ist eine sogenannte Kommando- und Kontrollarchitektur (*command and control*), die es kabellosen Peripheriegeräten wie Tastaturen, Mäusen, GamePads und Joysticks erlaubt, über eine durchschnittliche Entfernung von 7 Metern, mit Hosts zu interagieren. Hosts können zum Beispiel PCs, Haushaltsgeräte, Spielkonsolen, Fernseher oder Videorekorder sein.

Da nur einfache Kontrollaufgaben von IrDA Control übernommen werden, ist es auch nicht notwendig große Datenpakete zu übertragen, sondern es reichen kleine Kontrollpakete zwischen dem Host und dem Remote-Gerät aus.

Ein Host kann bis zu 8 Geräte verwalten, wobei jedes Gerät durchnummeriert ist und somit auch eine Priorität erhält. 4 Geräte können gleichzeitig mit dem Host kommunizieren und bei Nichtbenutzung auch in einen Wartemodus gefahren werden, um anderen Geräten eine Chance zur Kommunikation zu geben.

5.3 IrDA Data

IrDA Data definiert einen Standard für einen vollständig kompatiblen Port zur Datenübertragung mittels Infrarot. Die Technologie existiert seit 1994 und steckt mittlerweile in über 300 Millionen unterschiedlichsten Geräten, wie zum Beispiel Digitalkameras, Mobiltelefonen, Laptops, Uhren und vielen mehr.

IrDA Data ist besonders für hohe Datenübertragungen auf sehr kurze Entfernungen geeignet und eignet sich aufgrund der beschränkten Reichweite meistens auch nur zur Anwendung in den oben genannten Geräten.

5.3.1 IrDA Data Protokollstack

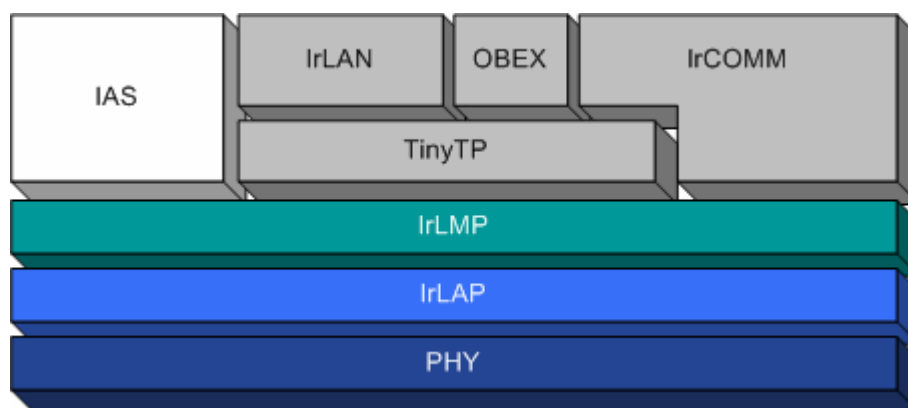


Abbildung 5-2: IrDA Data Protokollstack

5.3.2 Zwingend erforderliche Protokolle

5.3.2.1 PHY (Physical Signalling Layer)

Die Reichweite von Infrarotübertragungen zwischen Sender- und Empfängergerät geht bis zu normalerweise einen Meter weit. Der Abstand sollte nicht über maximal zwei Meter gehen, da dann eine korrekte Übertragung nicht mehr gewährleistet ist.

Bei Geräten, die über begrenzte Energie verfügen, wird eine stromsparendere Variante benutzt, welche aber auch nur Entfernungen von 20cm bis 30cm überbrücken kann.

Die Kommunikation zwischen den Geräten verläuft bei Infrarot bidirektional und mit einer Geschwindigkeit von bis zu 4MB/s. Für die sichere und fehlerfreie Datenübertragung wird die CRC-Methode (Prüfsumme) verwendet⁶.

5.3.2.2 IrLAP (Link Access Protocol)

Das IrDA Link Access Protokoll ist für den Verbindungsaufbau zwischen den Geräten verantwortlich und stellt für sichere Datenübertragungen u.a. Fehlerkorrekturen und eine Low-Level Fluss-Kontrolle zur Verfügung.

⁶ CRC-16 bei langsamer Übertragung, CRC-32 bei höheren Geschwindigkeiten (4MB/s).

Man unterscheidet bei IrDA Systemen zwischen Master- und Slave-Geräten, wobei man häufig auch die Master als die *Primary*- und die Slaves als die *Secondary*-Stationen bezeichnet.

Die Primary-Stationen sind verantwortlich für den Verbindungsaufbau, den Datentransfer, die Fluss-Kontrolle und die Fehlerbehandlung. Primary Geräte sind beispielsweise PCs, Digitalkameras und jedes Gerät, das drucken möchte.

Secondary-Stationen, wie zum Beispiel Drucker, sind passiv und senden nur, wenn sie von einem anderen Gerät angesprochen werden.

Während einer Verbindung darf jede Station jeweils nur 500ms lang senden, bevor wieder ein Wechsel stattfindet. Diese Zeitspanne gilt auch, wenn eine Seite keine Daten zu übertragen hat.

Man unterscheidet bei IrLAP zwischen zwei Verbindungsmodi, dem *Normal Disconnected Mode* (NDM) und dem *Normal Response Mode* (NRM). Der Erstere gilt, wie man am Namen schon erkennen kann, für Geräte, die gerade nicht verbunden sind. In diesem Modus überprüft das Gerät alle 500ms, ob im Umfeld gerade eine Übertragung stattfindet.

Der NRM Zustand gilt für alle Geräte, die sich in einer Verbindung mit einem anderen Gerät befinden.

Bei IrLAP existieren sogenannte Primitives, um die Kommunikation zu realisieren. Primitives sind die Signale REQUEST, INDICATION, RESPONSE und CONFIRM. REQUEST ist für das starten eines Dienstes zuständig und wird z.B. von höheren Schichten gesendet. INDICATION teilt höheren Protokollschichten Anfragen mit und liefert Statusinformationen. RESPONSE ist eine Art Acknowledgement von einem höheren Protokoll und CONFIRM teilt einem Sender die Bestätigung vom Empfänger mit.

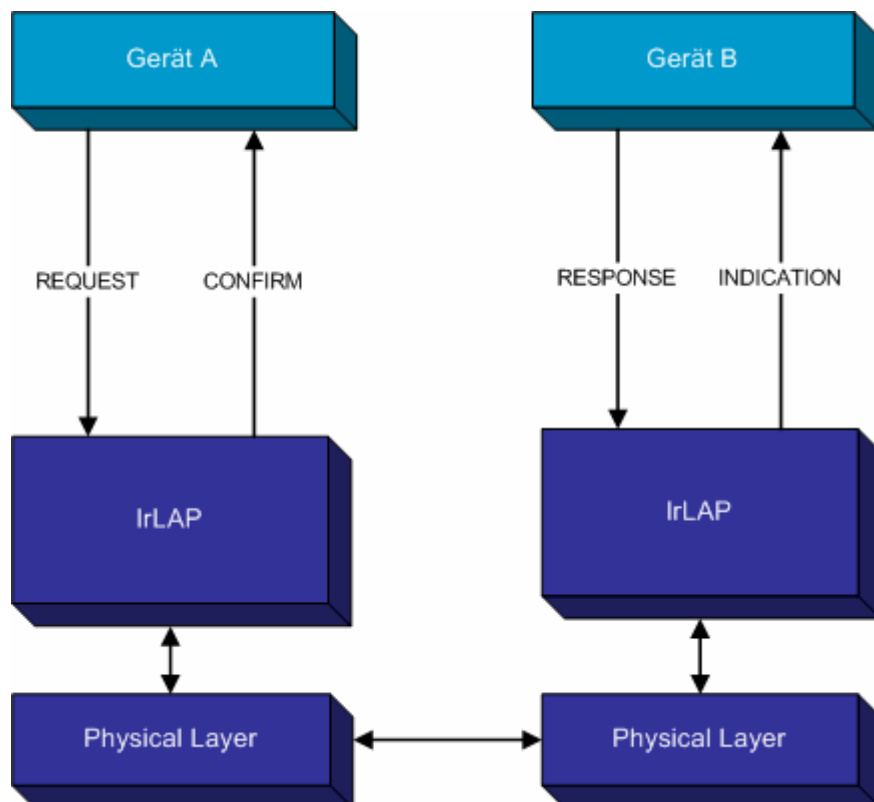


Abbildung 5-3: IrLAP Verbindungsaufbau mit Primitives [12]

Als Adresse bekommt jedes Gerät von IrLAP 32-bit Adressen bei Verbindungsaufbau. Da die Geräte nicht wie in statischen Netzwerken feste Adressen bekommen können, werden den Netzteilnehmern 32-bit Zufallsadressen zugewiesen. Falls durch Zufall eine Adresse doppelt verteilt wird, kommt eine Routine zum Einsatz, die diese Kollision auflöst.

5.3.2.3 IrLMP (Link Management Protocol)

Das Link Management Protokoll besteht aus zwei Komponenten. Dem *Information Access Service* (IAS) und dem *Link Management Multiplexer* (MUX). Obwohl IrLMP für den IrDA-Standard zwingend vorgeschrieben ist, müssen nicht alle Fähigkeiten von IrLMP genutzt werden.

Der erste Bestandteil von IrLMP, der Information Access Service, ist eine Datenbank, welche Informationen über jedes Gerät in Form von Datenobjekten enthält. Diese Daten werden gebraucht, damit ein Sendegerät über ein anderes Gerät Informationen erhalten kann, die Aufschluss darüber geben, welche Dienste auf dem Empfängergerät zur Verfügung stehen. Die Datenobjekte erhalten jeweils einige Attribute mit Informationen. Beispielsweise hätte ein Objekt „Gerät“ die Attribute „ObjektName“ (ASCII-String) und das Attribut „IrLMPSupport“ (IrLMP Versionsnummer, IAS Support, MUX Support).

Der zweite Bestandteil von IrLMP ist der Link Management Multiplexer. Dieser erlaubt es, dass mehrere IrLMP Verbindungen auf einem IrDA Kanal laufen können. Dafür gibt es bei IrLMP sogenannte *Logical Service Access Punkte* (LSAP) und den *LSAP-Selector* (LSAP-SEL). Über die LSAPs ist es möglich, Dienste oder Anwendungen innerhalb des IrLMP anzusprechen. Der LSAP-SEL ist ein 1 Byte großer Zeiger, der auf den LSAP verweist. Dieses Byte kann Werte zwischen 0x00 und 0x6F annehmen. Wobei 0x00 für den IAS steht und 0x01 bis 0x6F für die LMP Verbindungen.

5.3.3 Optionale Protokolle

Neben den zwingend erforderlichen Protokollen wie sie in Kapitel 5.3.2 erwähnt worden, gibt es noch eine Reihe optionaler Protokolle im IrDA Data Protokollstack. Wir werden die beiden Protokolle Tiny TP und IrCOMM ein wenig näher betrachten und die anderen kurz erwähnen.

5.3.3.1 Tiny TP

Obwohl Tiny TP vom IrDA Data Standard nur optional gefordert wird, ist es trotzdem sinnvoll, wenn Tiny TP immer vorhanden ist. Tiny TP bietet nämlich die Möglichkeit einer Fluss-Kontrolle (flow-control) und ist auch für die Segmentierung und das Zusammensetzen von Datenpaketen zuständig (SAR, *Segmentation and Reassembly*).

Die Fluss-Kontrolle wird eingesetzt, wenn über LMP im Multiplexing-Modus (MUX) mehrere Kanäle aktiv sind. Für jeden einzelnen Kanal wird dann zur Fehlervermeidung eine sogenannte Credit-Fluss-Kontrolle benutzt, da hier herkömmliche Fluss-Kontrolle über IrLAP zu Problemen führen würde.

Dabei wird vor jedem Verbindungsaufbau dem jeweils anderem Teilnehmer Credits in Form von Pufferspeicher zugewiesen. Ein Credit entspricht der Erlaubnis ein LMP Paket zu senden,

wobei die maximale Anzahl von Credits 127 betragen darf. Bei jedem Empfang von Daten wird die Credits-Anzahl um eins dekrementiert.

Der Empfänger kann während der Kommunikation weitere Credits verteilen und die alleinige Kontrolle über Anzahl der Credits. Wenn daher zu wenige Credits verteilt werden, wirkt sich das auf den Datendurchsatz schnell negativ aus.

Credit-Pakete selbst unterliegen nicht der Fluss-Kontrolle und können auch ohne vorher zugesicherten Pufferspeicher gesendet werden.

Segmentation und Reassembly ist für die Aufteilung von Datenpaketen und die spätere Zusammensetzung beim Empfänger zuständig. Bei der ersten LMP Verbindung wird zwischen den Teilnehmern die Größe einer Service Data Unit (SDU) ausgehandelt, d.h. die Größe eines vollständig, nicht segmentierten Datenpakets.

5.3.3.2 IrCOMM

Da bei der Entwicklung des IrDA Data Standards eine große Anzahl der am Markt befindlichen Peripherie-Geräte und PCs über eine serielle bzw. eine parallele Schnittstelle verfügten, bot sich die Möglichkeit an, dass in den Protokollstack ein Protokoll ausgenommen wird, was die Zusammenarbeit mit diesen Schnittstellen unterstützt.

Hierfür wurde daher das IrCOMM Protokoll entwickelt, welches insgesamt vier unterschiedliche Varianten definiert, um unterschiedlichste Schnittstellen zu unterstützen:

1. *3 Draht Raw Data* (IrLPT), welches eine parallele oder serielle Schnittstelle emuliert. Hier gibt es keine Möglichkeit zum Multiplexing und es wurde auch jegliche Kontrollinformationen weggelassen. Zudem setzt IrLPT direkt auf dem IrLMP auf und nicht auf Tiny TP.
2. *3 Draht*, das auf Tiny TP aufsetzt und eine einfache serielle oder parallele Schnittstelle nach dem V.24 Standard⁷ emuliert. Hier werden auch einige Kontrollsignale eingesetzt.
3. Die *9 Draht* Variante emuliert eine vollständige V.24 Schnittstelle und wird nur für serielle Schnittstellen verwendet. Sie setzt auch auf Tiny TP auf und benutzt einige Kontrollinformationen für den Status der V.24 Steuersignale.
4. Die letzte Variante ist *Centronics* und wird zur Emulierung von parallelen Schnittstellen verwendet. Sie setzt ebenfalls auch TinyTP auf und verwendet Kontrollinformationen für den Status der Centronics Steuersignale.

IrCOMM stellt zudem zur Emulierung der Handshake Signale wie beispielsweise DTR, DSR, usw. sogenannte Kontrollservices zur Verfügung.

5.3.3.3 IrOBEX

IrOBEX steht für *Infrared Object Exchange Protocol* (kurz: OBEX) und ist ein effizientes, binäres Protokoll zum Austausch von Daten, das ebenfalls auf dem Tiny TP-Protokoll aufsetzt.

OBEX übernimmt ähnliche Aufgaben wie das Hypertext Transfer Protokoll (HTTP), welches für den Datentransfer im World Wide Web (WWW) zuständig ist.

⁷ Meist verwendete Universalschnittstelle zwischen Datenendgerät und Modem.

Der Schwerpunkt bei OBEX liegt bei den sogenannten „Push“- und „Pull“-Applikationen. Diese Art von Applikationen sind nur für kurzfristige Datenübertragungen da, das heißt sie verbinden sich kurz mit einem Gerät und senden bzw. empfangen Daten. Beispielsweise gehört eine Digitalkamera, die ein Bild an ein Notebook sendet und dann die Verbindung abbaut zu den „Push“-Applikationen.

5.3.3.4 IrLAN

IrLAN steht für *LAN Access Extensions for Link Management Protocol* und ist ein Protokoll für die Anbindung von Infrarot-Geräten an LANs.

IrLAN definiert ein 2-Kanal-Interface zwischen einem Protokoll-Client und einem Protokoll-Server. Da der IrLAN-Provider passiv arbeitet, liegt es am Client, einen passenden Provider zu finden und zu kontaktieren, um Daten zu empfangen bzw. zu senden.

Mit IrLAN sind auch Peer-to-Peer Verbindungen möglich, wobei in diesem Modus jede Infrarot-Station ihren eigenen Client und Server besitzt. Um einen Datenkanal zu öffnen, müssen bei Peer-to-Peer die Geräte miteinander konkurrieren, um den Kanal zu erhalten.

Beim IrLAN-Verbindungsaufbau fordert der Client einen Datenobjekt beim IAS (Information Access Service) an, welches einen IrLMP LSAP (siehe Kapitel 5.3.2.3) für einen „Kontrollkanal“ liefert. Der Client stellt eine Verbindung mit dem Kontrollkanal her und versucht weitere Informationen über einen verfügbaren Datenkanal zu erhalten.

Falls ein Datenkanal gefunden wird, wird dieser über den Kontrollkanal konfiguriert. Über den Datenkanal selber werden nur die Datenpakete übertragen und für das LAN formatiert.

Auch IrLAN setzt auf Tiny TP auf, das heißt Tiny TP ist für die Segmentierung und das spätere Zusammenfügen von Datenpaketen zuständig und übernimmt gleichzeitig die Fehlerkorrektur.

6 Zusammenfassung

In den letzten fünf Kapiteln wurden einige Aspekte von Ad Hoc Protokollen bezüglich der drei wichtigen Ad Hoc Netzwerktechnologien IEEE 802.11, Bluetooth und Infrarot vorgestellt. Wir haben den Fokus in der Ausarbeitung auf drahtlose Ad Hoc Netze gelegt, da hier Probleme auftreten können, die bei verdrahteten Netzwerken, nicht vorkommen. Ein großes Problem waren die Kollisionen, die entweder in Form von Signal-Kollisionen (RTS, CTS, ...) oder Datenkollisionen vorkommen konnten. Dieses Problem war bei IEEE 802.11 der Schwerpunkt in der Ausarbeitung, weswegen hier auch verschiedene Lösungen in Form von Media Access Control Protokollen (MAC), wie beispielsweise MACA, DBTMA oder MARCH aufgezeigt wurden. Durch die Trennung von Master und Slave bei der Bluetooth Technologie hatte man das Problem der Kollisionen nicht, das hier die Kommunikation über den Master geregelt wurde.

Neben der Problematik der Kollisionen musste man bei drahtlosen Ad Hoc Netzen auch noch auf eine effiziente Energieverwaltung achten, da hier oft Geräte im Einsatz sind, die über eine begrenzte Energiekapazität verfügen (Handy, Palm, etc.). Zur Lösung dieses Problems haben wir bei IEEE 802.11 zum Beispiel das MAC Protokoll PAMAS kennen gelernt, das durch Abschalten von Knoten energiesparender arbeiten kann.

Bei den beiden Technologien Bluetooth und Infrarot haben wir einen detaillierten Einblick in die Protokolle geworfen und gesehen, dass ein Schwerpunkt auf dem Teilen von Services (Dienste) lag. Beide Technologien unterstützten das „Sharing“ von Diensten über speziell dafür ausgelegte Protokollschichten. Beim Betrachten der Protokollstacks wurde auch gezeigt, dass hier beim Definieren der einzelnen Schichten auf Kompatibilität zu alt bewährten Protokollen geachtet wurde. So konnte man mit Bluetooth oder Infrarot ohne weiteres Anwendungen nutzen, die beispielsweise auf dem OBEX Protokoll zum Austausch von Datenobjekten aufbauen.

Zusammenfassend kann man sagen, dass mit der Verwendung von Ad Hoc Netzen viele neue Probleme, aber auch viele neue Möglichkeiten entstehen werden. Die derzeitigen Ad Hoc Protokolle sind für einen Einsatz schon sehr gut geeignet und werden sicherlich in vielen Geräten zum Einsatz kommen.

Quellenverzeichnis

- [1] Pablo Brenner (1996). *A Technical Tutorial on the IEEE 802.11 Protocol*.
http://www.sss-mag.com/pdf/802_11tut.pdf
- [2] R. Bruno, M. Conti, E. Gregori (2001). *WLAN technologies for Mobile ad hoc Networks*. In: *Proceedings of the 24th Hawaii International Conference on System Sciences 2001*.
<http://dlib2.computer.org/conferen/hicss/0981/pdf/09819003.pdf>.
- [3] Ajay Chandra V. Gumalla, John O. Limb (2000). *Wireless Medium Access Control Protocols*. Georgia Institute of Technology.
<http://www.comsoc.org/livepubs/surveys/public/2q00issue/gummalla.html>.
- [4] Jacqueline Hewitt (2002). *WAN Technologies and Techniques*.
<http://www.csd.uwo.ca/courses/CS457a/reports/handin/jhewitt/A2/protocols.htm>.
- [5] Zhuochuan Huang und Chien-Chung Shen. *A Busy-Tone Based Directional MAC Protocol for Ad Hoc Networks*.
<http://www.scalable-networks.com/customers/QUP/dbtmada.pdf>.
- [6] Internet Mail Consortium. *vCard and vCalendar*. <http://www.imc.org/pdi/>.
- [7] Infrared Data Association (1999). *IrDA Object Exchange Protocol IrOBEX*.
<http://cooltown.hp.com/dev/reference/coolbase/esquirt/Win32Docs/IrOBEX12.pdf>
- [8] Infrared Data Association. *IrDA Serial Infrared Data Link Standard Specifications*.
<http://www.irda.org/standards/specifications.asp>.
- [9] Infrared Data Association (1997). *LAN Access Extensions for Link Management Protocol*. <http://www.irda.org/standards/pubs/IrLAN.PDF>
- [10] Dr. Keming, W. Yeh und Dr. Lichen Wang. *An Introduction to the IrDA Standard and System Implementation*. <http://www.actisys.com/article.html>.
- [11] Prof. Dr. W. Kowalk (2002). *Rechnernetze*.
<http://einstein.offis.uni-oldenburg.de/rechnernetze/Default.htm>
- [12] Rudi Latuske (1998). *IrDA – ein Protokoll zur Datenübertragung mit Infrarotlicht*.
<http://www.ars2000.com/pdf/IrDA-WhitePaper.pdf>.
- [13] Daniel L. Lough, T. Keith Blankenship und Kevin J. Krizman (1997). *A Short Tutorial on Wireless LANs and IEEE 802.11*. The IEEE Computer Society's Student Newsletter Summer 1997, Volume 5, No. 2.
<http://www.computer.org/students/looking/summer97/ieee802.htm>.
- [14] Christian Mormann (2001). *Bluetooth. Vom mobilen zum ubiquitären System*.
http://www.informatik.uni-stuttgart.de/ipvr/vs/lehre/ss01/Seminare/Seminar_MobUbi/Ausarbeitungen/mormann.pdf.

- [15] Jyrki Oraskari (2000). *Bluetooth versus WLAN IEEE 802.11x*.
<http://www.hut.fi/~joraskur/>.
- [16] C.S. Raghavendra und Suresh Singh (1997). *PAMAS – Power Aware Multi-Access protocol with Signalling for Ad Hoc Networks*. <http://pacman.usc.edu/pamas.ps>.
- [17] Peter Riedler und Ralf Wenninger (2002). *Bluetooth aus Sicht der Protokolle*. Funkschau 02/2002 vom 25. Januar 2002.
<http://www.funkschau-handel.de/heftarchiv/pdf/2002/fs0202/fs0202054.pdf>.
- [18] Axel Sikora (2001). *802.11: Standard für drahtlose Netze*. TecChannel.
<http://www.tecchannel.de/hardware/680/index.html>
- [19] C.K. Toh (2001). *Ad Hoc Mobile Wireless Networks: Protocols and Systems*. S. 27-56, S. 243-255.
- [20] Yu-Cheng Tseng, Chih-Shun Hsu, Ten-Yueng Hsieh (2002). *Power-Saving Protocols for IEEE 802.11-Based Multi-Hop Ad Hoc Networks*.
<http://www.ieee-infocom.org/2002/papers/217.pdf>.
- [21] Praveen Yalagandula (2000). *A Survey on Security Issues in Wireless Networks*.
http://www.cs.utexas.edu/users/ypraveen/surveys/wlan_security/survey.html.
- [22] ZDNet Deutschland (2000). *Datenübertragung per Infrarot-Port*.
http://www.zdnet.de/mobile/artikel/techreport/mobile-business/mobile-business03_00-wc.html.

Abbildungsverzeichnis

Abbildung 3-1: Vereinfachter IEEE 802.11 Protokollstack.....	6
Abbildung 3-2: A sendet Daten an B	8
Abbildung 3-3: C sendet Daten an B... Kollision!	8
Abbildung 3-4: Hidden Terminal Handshake 1. Schritt (RTS)	8
Abbildung 3-5: Hidden Terminal Handshake 2. Schritt (CTS)	8
Abbildung 3-6: Hidden Terminal Handshake 3. Schritt (Daten)	8
Abbildung 3-7: Hidden Terminal Handshake 4. Schritt (ACK)	8
Abbildung 3-8: Probleme der RTS-CTS Methode.....	9
Abbildung 3-9: RTS-CTS Problem (Datenkollision).....	10
Abbildung 3-10: Exposed Node Problem	10
Abbildung 3-11: Exposed Node (Blockierungen bei ungerichteten Antennen)	11
Abbildung 3-12: Exposed Node Lösung (Gerichtete Antennen).....	11
Abbildung 3-13: Senderinitiiertes MAC Protokoll	12
Abbildung 3-14: MACA Handshake (RTS).....	13
Abbildung 3-15: MACA Handshake (CTS).....	13
Abbildung 3-16: MACA Handshake (Daten)	13
Abbildung 3-17: MACAW Handshake (RTS).....	14
Abbildung 3-18: MACAW Handshake (CTS).....	14
Abbildung 3-19: MACAW Handshake (Daten).....	14
Abbildung 3-20: MACAW Handshake (ACK).....	14
Abbildung 3-21: Energieverschwendung ohne PAMAS	15
Abbildung 3-22: PAMAS Zustandsübergangsdiagramm [16].....	16
Abbildung 3-23: Dual Busy Ton beim DBTMA Protokoll.....	16
Abbildung 3-24: Empfängerinitiiertes MAC Protokoll	17
Abbildung 3-25: MACA-BI Handshake (RTR).....	18
Abbildung 3-26: MACA-BI Handshake (Daten)	18
Abbildung 3-27: Signalunterschiede von MACA und MACA-BI.....	19
Abbildung 3-28: MARCH Handshake	20
Abbildung 4-1: Bluetooth Piconet.....	22
Abbildung 4-2: Bluetooth Scatternet.....	23
Abbildung 4-3: Bluetooth Protokollstack	24
Abbildung 4-4: Bluetooth Zustandsübergangsdiagramm	26
Abbildung 4-5: Bluetooth Service Discovery	28
Abbildung 4-6: TCS Ablauf	29
Abbildung 4-7: Bluetooth Zeitfenster (time slots)	31
Abbildung 4-8: Bluetooth Paketstruktur	32
Abbildung 4-9: Bluetooth 48-bit MAC Adresse	32
Abbildung 5-1: Infrarot-Ausstrahlungsbereich	33
Abbildung 5-2: IrDA Data Protokollstack	34
Abbildung 5-3: IrLAP Verbindungsaufbau mit Primitives [12]	35

Index

- ACK8
- ACL.....25
- asynchron25
- Baseband Modi.....26
- Belauschungs-Effekt19
- Bluetooth 21, 22, 24, 25, 26, 27, 28, 29, 30,
31, 32, 41
- Bluetooth-Einheiten26, 29
- CallControl*.....28
- Clear To Send.....8
- ConnectionlessSignaling*28
- CSMA/CA.....7
- CTS..8, 9, 12, 13, 14, 15, 16, 17, 18, 19, 20
- CTS-Signal.....8, 9, 13, 14, 19, 20
- CVSD-Modulation29
- Direct Sequence Spread Spectrum*6
- Dual Busy Tone Multiple Access.....16
- Energieverschwendung19
- Ericsson21
- Ethernet7
- Exposed Node10, 11, 12, 13, 14
- FDMA5
- Frequency Hopping Spread Spectrum*6
- Frequenz Hopping25
- Frequenzmultiplex.....5
- GroupManagement*.....28
- Handshake12, 17, 19, 20
- Header31
- Hidden Terminal8, 9, 12, 16
- HTTP.....37
- IAS36, 38
- IAS Support.....36
- IBM21
- IEEE3, 4, 5, 6, 7, 8, 32, 39, 40, 41
- IEEE 802.113, 4, 5, 6, 7, 8, 39, 40, 41
- Infrared Data Association*33, 40
- INTEL21
- IP7, 24, 29
- IrCOMM.....37
- IrDA Control33
- IrDA Data34
- IrLAN38, 40
- IrLAP.....34, 36
- IrLMP.....36, 38
- IrOBEX37, 40
- ISM.....6, 21
- ISM-Band.....6
- Kernprotokolle25, 26
- Kollisionen5, 8, 9
- Kommunikationsaufforderung..... 17
- Kontrollnachrichten 8
- Kontrollsignale 18
- Link Manager Protokoll..... 26
- LMP 24, 26
- Lower-Address-Part..... 32
- Master 5, 21, 31
- Media Access Control..... 3, 5, 6, 7, 39
- Media Access Protokolle* 7
- Multiple Access with Collision Avoidance*
..... 12
- MUX..... 36
- Netzwerkprotokolle 7
- Nokia..... 21
- OBEX 24, 30, 37, 38
- packet forwarding 4
- PAMAS..... 14, 15, 16, 19, 39, 41
- Physical Layer* 6
- Physical Layer Convergence Protocol* 6
- Physical Media Dependent 6
- Piconet 21, 22, 25, 26, 31
- PLCP..... 6
- Point-to-Point..... 28, 29
- Power-Aware Multi-Access Protokoll... 15
- PPP..... 24, 29, 30
- Protokollschichten 24, 27
- Protokollstack 24, 27, 28, 34, 36
- Prüfsumme 34
- receive-data* 15
- Request To Send..... 8
- RFCOMM..... 28, 29, 30
- RTR..... 17, 18
- RTR-Einladungen 18
- RTS..... 8, 9, 12, 14, 15, 16, 17, 18, 19, 20
- RTS Signal..... 9, 15, 18
- Scatternet* 22, 23
- SCO..... 25, 28
- SDP-Server 27
- Serialisierung 29
- Service Discovery Protokoll..... 27
- Service-Provider 4
- Slave 21, 25, 31
- TCP..... 7, 24, 29
- TDMA..... 5
- Time Slots..... 31
- Time-Division-Duplex-Schema..... 31
- Tiny TP..... 36, 37, 38
- Toshiba 21
- Übertragungsbandbreiten..... 6

UDP	29	vCard.....	30, 40
ungerichtete Antennen.....	11, 19	<i>wait-for-CTS</i>	15
Unterschiede von MACA und MACA-BI		WAP-Client	30
.....	18	<i>Wireless User Groups</i>	28
Upper-Address-Part.....	32	Zeitmultiplex.....	5
vCalendar	30, 40		