Lehrstuhl für Kommunikationsnetze
Rheinisch-Westfälische Technische Hochschule Aachen
Prof. Dr.-Ing. B. Walke

Diplomarbeit

Entwicklung und Leistungsbewertung eines *Selective Repeat-Automatic Repeat Request* (SR-ARQ)-Protokolls für transparenten, mobilen ATM-Zugriff

von

Andreas Hettich

Matr.-Nr. 181475

Aachen, den 17.1.1996

Betreuung durch:
o. Prof. Dr.-Ing. B. Walke
Dipl.-Ing. D. Petras

Ich versichere, daß die vorliegende Arbeit - bis auf die offizielle Betreuung durch den Lehrstuhl - ohne fremde Hilfe von mir durchgeführt wurde. Die verwendete Literatur ist im Literaturverzeichnis vollständig angegeben.

Aachen, den 17.1.1996

(Andreas Hettich)

KAPITEL 1

Einleitung

Die Telekommunikationsindustrie hat in den letzten Jahren trotz Rezession ihren Umsatz steigern können. In Deutschland stieg der Umsatz in den Jahren von 1990 bis 1995 von 11 auf 23 Mrd DM [25]. Mit dem Fall der Telekommunikationsmonopole in Europa im Jahr 1998 wird mit einem zusätzlichen Anstieg der Umsätze gerechnet. Dies verdeutlicht die zunehmende Bedeutung von Telekommunikationsnetzen und -diensten.

Viele zusätzliche neue Dienste im Bereich der Telekommunikation wurden durch digitale Telekommunikationssysteme möglich. In diesem Zusammenhang ist das *ISDN (**I**ntegrated **S**ervices **D**igital **N**etwork)* zu nennen. Das ISDN beseitigt die Vielfalt unterschiedlicher Netz – Benutzer Schnittstellen und bietet alle Sprach-, Text- und Datendienste über eine einheitliche Netzschnittstelle an. Neben digitalen Festnetzen etablierten sich auch neue digitale Mobilfunknetze nach den Standards *GSM (**G**lobal **S**ystem for **M**obile Communication)* und *DCS 1800 (**D**igital **C**ellular **S**ystem 1800)*. In Deutschland arbeitet das D1-Netz der Deutschen Telekom sowie das D2-Netz von Mannesmann Mobilfunk nach dem GSM-Standard. Das E-Netz von E-Plus wurde nach dem DCS-Standard aufgebaut.

Die feste Zuteilung von Bandbreiten von 64 kbit/s im ISDN-Netz und 22,8 kbit/s (brutto) im GSM-System ist auf schmalbandige Sprachübertragung (4 kHz Grenzfrequenz) zugeschnitten und kann die hohen variablen Datenraten, die zukünftige Breitbanddienste wie Videokommunikation, Datenübertragung in Computernetzen, und Multimedia generell erfordern, nicht anbieten. Deshalb wurde das ATM-Übertragungsverfahren (**A**synchronous **T**ransfer **M**ode) entwickelt, das im Breitband-ISDN (B-ISDN) angewendet wird und eine variable Bitrate von 622 Mbit/s und mehr ermöglicht.

Da sich auch der Bedarf an höheren Datenraten im Bereich von Mobilfunknetzen abzuzeichnen beginnt, wurde begonnen, das UMTS-System (**U**niversal **M**obile **T**elecommunication **S**ystem) zu entwickeln. Es stellt Datenraten von bis zu 2 Mbit/s zur Verfügung, bei Verwendung von dienstspezifischen Protokollen.

Ein anderer Weg wird im Rahmen des *MBS*-Projektes *(**M**obile **B**roadband **S**ystem)* gegangen. Es soll ein System entworfen werden, daß der Funktionalität und der Übertragungsqualität des Breitband ISDN im Festnetz möglichst nahe kommt. Als maximale (Brutto-)Datenrate wird 155 Mbit/s angepeilt.

Die für ein Mobilfunksystem sehr hohen Datenraten lassen sich nur mit entsprechend hohen Trägerfrequenzen im Bereich von 17, 40 oder 60 GHz erzielen. In diesen Bereichen befinden sich noch ungenutzte Bandbreiten. Hierzu müssen auf dem Gebiet der digitalen Signalverarbeitung neue leistungsfähige Komponenten entwickelt werden. Bei 17 und 40 GHz kann auf Siliziumtechnik zurückgegriffen werden, während bei der hohen Frequenz von 60 GHz Galliumarsenid verwendet werden muß, das wesentlich teurer ist.

Eine hohe Teilnehmerdichte erreicht man durch kleine Zellgrößen mit Radien von 100–200 m. Kleine Zellgrößen erreicht man durch die hohen Trägerfrequenzen, weil in diesem Bereich eine zusätzlich Resonanzdämpfung der Luft eintritt. Diese liegt etwa bei 14 dB/km

und kann sich bei Regen um noch einmal 10 dB/km erhöhen. Aufgrund der hohen Dämpfung lassen sich die Frequenzen in kleinen Abständen wiederverwenden.

Die Übertragung der Daten erfolgt dabei dienstunabhängig nur unter Angabe der in den ATM-Standards vorgegebenen QoS (**Q**uality **o**f **S**ervice)-Parametern sowie der von ATM vorgegebenen Dienstklassen. Die QoS Parameter, wie auch charakteristische Verkehrsgrößen werden vor dem Verbindungsaufbau von der CAC (**C**onnection **A**dmission **C**ontrol) ausgewertet. Die einmal aufgebaute Verbindung muß auf die Einhaltung der beim Verbindungsaufbau vereinbarten Verkehrsgrößen überwacht werden, denn sonst kann möglicherweise die Güte anderer Verbindungen in Mitleidenschaft gezogen werden.

Ein physikalischer Kanal im MBS kann als Bruttodatenrate 34 Mbit/s tragen. Wenn ein virtueller Kanal eine Datenrate von mehr als 34 Mbit/s hat, hilft nur die Aufteilung auf mehrere Frequenzkanäle (Multilinkübertragung).

Besondere Anforderungen werden an die Protokolle der Sicherungsschicht gestellt, in der neue Konzepte verwirklicht werden müssen. Ein mobiles ATM-Terminal soll in ein ATM-Festnetz unter Beibehaltung der Ende-zu-Ende-Beziehung der ATM-Anpassungsschicht eingesetzt werden. Dabei stellt die schlechte Übertragungsqualität auf der Funkschnittstelle ein besonderes Problem dar, da sie die vom ATM geforderte Bitfehlerrate nicht ohne zusätzliche Maßnahmen zuläßt. Abhilfe schafft eine Vorwärtsfehlerkorrektur (engl.: **F**orward **E**rror **C**orrection (FEC)) in Verbindung mit einem Protokoll zur wiederholten Übertragung von ATM-Zellen (engl.: **A**utomatic **R**epeat Re**Q**uest (ARQ)).

Das Ziel der vorliegenden Arbeit ist, ein neues Sicherungsprotokoll zu entwerfen und simulativ zu bewerten. Dieses Protokoll basiert auf bekannten ARQ-Protokollen und ist den besonderen Anforderungen des MBS angepaßt.

Zunächst wird ein kurzer Überblick über das *Breitband-ISDN* (Kapitel 2) und das *Mobile Broadband System* (Kapitel 3) gegeben. Anschließend wird die für einen mobilen, transparenten ATM-Zugriff konzipierte LLC (**L**ogical **L**ink **C**ontrol)-Schicht vorgestellt und die besonderen Anforderungen des MBS beschrieben (Kapitel 4). Nach einer Einführung in konventionelle ARQ-Protokolle wird das in dieser Arbeit entwickelte und implementierte ASR ARQ-Protokoll vorgestellt (Kapitel 5). In Kapitel 6 wird die Implementierung des Protokolls im Systemsimulator dargestellt. Der für die simulative Leistungsbewertung eingesetzte Simulator SIMCO3++/MBS wird in Kapitel 7 vorgestellt. Die Elemente des Protokolls werden in Kapitel 8 anhand von Simulationen in ihrer Leistungsfähigkeit bewertet. Eine Zusammenfassung und ein Ausblick auf zukünftige Erweiterungen schließen diese Arbeit ab (Kapitel 9).

KAPITEL 2

Breitband-ISDN (B-ISDN)

2.1 Dienste im Breitband-ISDN

Die Integration von neuen breitbandigen Anwendungen in das *Integrated Services Digital Network* (ISDN) erfordert Datenraten von bis zu 155 Mbit/s auf dem Teilnehmeranschluß. Bei derartigen Anwendungen handelt es sich zum einen um Dienste mit kontinuierlichem Datenaufkommen und zum anderen um büschelartige, interaktive Dienste. Zu den Anwendungen, die kontinuierliche Bitströme generieren, gehören z.B. die Sprachübertragung und die Videokonferenz. Interaktive Dienste weisen stark schwankende Anforderungen hinsichtlich ihrer Bitrate auf. So kann eine kurze Anfrage in einer Datenbank eine Antwort mit einer sehr hohen Datenrate zur Folge haben.

- Interaktive Dienste
  - Telefonie
  - Bildtelefonie
  - Breitbandvideokonferenz
- Abrufdienste
  - Zugriff auf Datenbanken
  - Radio, TV, HDTV, Video on Demand
  - elektronische Zeitung
  - Videopost
- Datenkommunikation
  - LAN-Verbindungen
  - Filetransfer
  - CAM-Verbindungen
  - hochauflösende Bildübertragung

Die sehr unterschiedlichen Anforderungen der Breitbanddienste lassen sich mit synchronen Übertragungsverfahren (STDM) nur schwer erfüllen. Eine Überdimensionierung der Übertragungskapazität von synchronen Kanälen verringert zwar die Wartezeiten, führt jedoch zu einer schlechten Kapazitätsausnutzung des Übertragungsmediums. Den Anforderungen ist der *Asynchronous Transfer Mode* (ATM) besser gewachsen.

2.2 Vermittlungstechnik im B-ISDN: *Asynchronous Transfer Mode* (ATM)

Der *Asynchronous Transfer Mode* ist das verbindungsbezogene Paketvermittlungsverfahren des B-ISDN. Er kombiniert die Vorteile der verbindungs- und paketorientierten Vermittlung miteinander. Die zu übertragenden Datenströme werden in kurze Blöcke fester Länge aufgeteilt, die sogenannten ATM-Zellen. Die Zellen von verschiedenen Verbindungen werden zeitlich verschachtelt über einen physikalischen Kanal übertragen. Entsprechend ihrer Datenrate erhalten die Verbindungen unterschiedlich viel Übertragungskapazität dynamisch zugewiesen. Die Zellen werden entsprechend der Reihenfolge ihrer Ankunft übertragen.

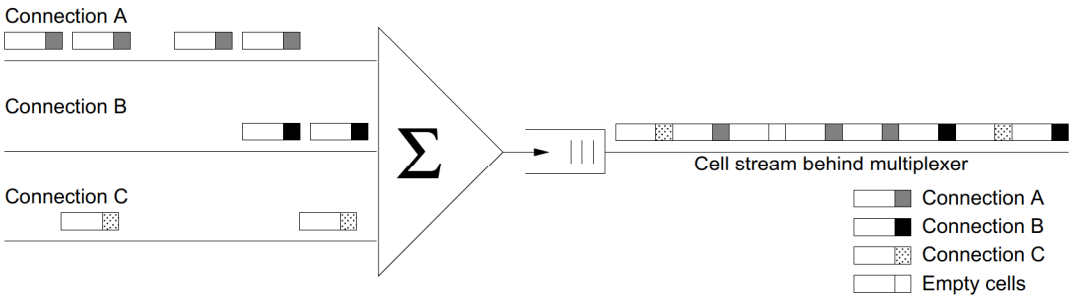


Abbildung 2.1: Statistisches Multiplexen an einem ATM-Anschluß

Der ATM-Multiplexer fügt „Leerzellen“ in den gemeinsamen Datenstrom ein, falls keine der Verbindungen Übertragungskapazität benötigt. Dieses als statistisches Multiplexen bezeichnete Verfahren ist besonders gut geeignet, sich dem dynamischen Kommunikationsverhalten von sehr unterschiedlichen Verbindungen anzupassen.

2.2.1 Aufbau einer ATM-Zelle

Eine ATM-Zelle umfaßt 53 Byte und setzt sich aus einem 5 Byte langen Kopf- und einem 48 Byte langen Informationsfeld zusammen, welches Nutzdaten enthält. Die Vermittlung der Zellen ist verbindungsorientiert. Alle Zellen einer virtuellen Verbindung nehmen den gleichen Übertragungsweg, der beim Verbindungsaufbau durch das Einrichten virtueller Kanäle festgelegt wird. Die Steuerung der Zellen durch das Netz erfolgt anhand der im Kopffeld gespeicherten Routinginformationen. Das Kopffeld enthält folgende Parameter:

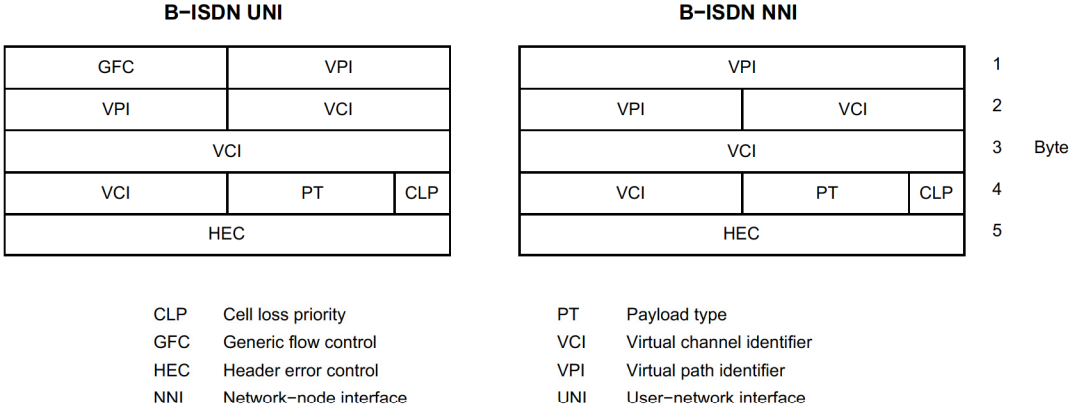


Abbildung 2.2: Kopf einer ATM-Zelle

VCI *Virtual Channnel Identifier*, 2 Bytes
Die Identifizierung des virtuellen Kanals dient der Unterscheidung der verschiedenen, gleichzeitigen Verbindungen und der Zuordnung der Zellen zu diesen Verbindungen. Die virtuelle Kanalnummer wird jeweils nur für einen Vermittlungsabschnitt vergeben.

VPI *Virtual Path Identifier*, 8 oder 12 Bit
Die Kennzeichnung des Bündels erfolgt mit Hilfe der VPI. Es können mehrere Bündel der gleichen Richtung, die jeweils mehrere virtuelle Kanäle beinhalten, auseinander-

gehalten werden. Zellen von Kanälen des gleichen Bündels können auf diese Weise schnell vom Koppelnetz erkannt und entsprechend weitergeleitet werden.

PT *Payload Type*, 3 Bit
Diese Kennzeichnung der Informationsfeldart dient der Unterscheidung von Nutzinformationen und Signalisierungsinformationen. Letztere können beispielsweise Informationen enthalten, die zur Aktualisierung der in den Vermittlungsstellen verwalteten Routingtabellen benötigt werden. Hierzu muß eine Vermittlungstelle neben dem Kopffeld auch das Informationsfeld der ATM-Zelle begutachten. Werden hingegen im Informationsfeld einer ATM-Zelle Nutzdaten übertragen, so bleibt deren Inhalt unbeachtet.

HEC *Header Error Control*, 1 Byte
Da der Kopf einer ATM-Zelle Daten enthält, die für den Transport der Zellen von besonderer Bedeutung sind, wird dieser durch eine gesonderte Prüfsequenz gesichert. Sie ermöglicht das Erkennen von Fehlern und deren Behebung.

CLP *Cell Loss Priority*, 1 Bit
Dieser Parameter kennzeichnet Zellen niedrigerer Priorität, die im Falle eines Warteschlangenüberlaufs in den ATM-Vermittlungsstellen verworfen werden.

2.2.2 ATM-Vermittlungstechnik

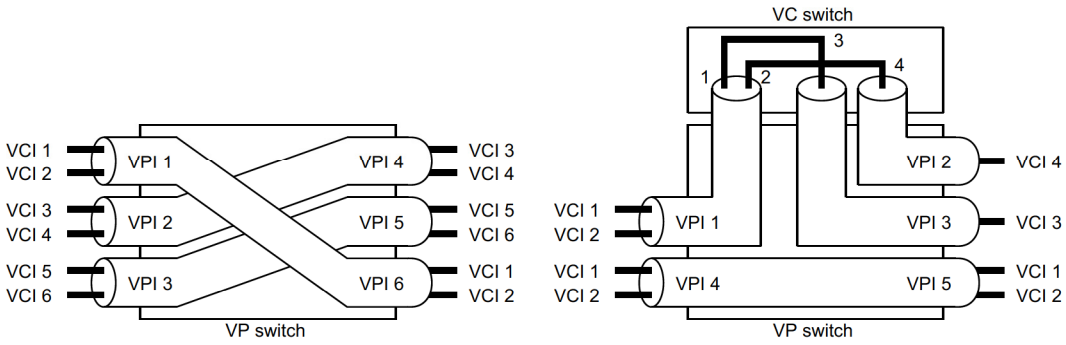


Abbildung 2.3: Virtual Path Switching und Virtual Channel Switching

Die Vermittlung der ATM-Zellen erfolgt, wie bei anderen Paketvermittlungsverfahren, aufgrund der im Zellkopf enthaltenen Routinginformationen. Um diese möglichst gering zu halten und hierdurch den Durchsatz zu erhöhen, wird lediglich beim Verbindungsaufbau die komplette Ursprungs- und Zieladresse versendet. Für die verschiedenen Abschnitte der Verbindung werden weitere Identifizierungen von logischen Kanälen vergeben (VCI, VPI). Die ATM-Vermittlungsstellen richten aufgrund der einlaufenden Steuerinformationen Routingtabellen ein. Diese enthalten eine Eingangs- und Ausgangs-Identifizierung (Leitung + logische Kanal-Identifizierung). Die Vermittlung der Zellen wird auf Grundlage dieser Tabellen vorgenommen und läuft wie folgt ab: Die vermittelnde Koppeleinrichtung entnimmt den eintreffenden Zellen die logischen Kanal-Identifizierungen. Auf Grundlage der in ihrer Routingtabelle enthaltenen Informationen wird anschließend die Kennung des folgenden Verbindungsabschnitts eingetragen und die Zelle zum entsprechenden Ausgang des Koppelnetzes geleitet.

Abbildung 2.3 zeigt die in den Vermittlungsstellen verwendeten Koppelelemente. Man unterscheidet zwischen VP-Switch und VC-Switch. Ein VP-Switch wertet nur die VPI-Werte der Zellen aus, so daß eine schnelle Vermittlung der Zellen möglich ist. Die Einträge

der VCI-Felder bleiben unverändert. Nur beim Durchgang durch einen VC-Switch werden die VCI-Kennungen verändert.

2.2.3 ATM-Referenzmodell

Entsprechend den Empfehlungen des OSI-Referenzmodells kann für ATM ein Referenzmodell bestehend aus vier Schichten angegeben werden. Dies sind die Schichten Physical Layer, ATM Layer, ATM Adaptation Layer und eine Schicht, die die Funktionen der höheren Schichten repräsentiert (Higher Layers). Außerdem sind drei unterschiedliche Ebenen definiert. Die Benutzerebene *(User Plane)*, die Kontrollebene *(Control Plane)* und die Verwaltungsebene (*Management Plane*). Die Management Plane umfaßt die Funktionen zur Ebenenverwaltung *(Plane Management)* und die Funktionen zur Verwaltung der Schichten *(Layer Management)*. Aufgabe des Plane Management ist die Verwaltung des gesamten Systems, während das Layer Management die einzelnen Schichten kontrolliert.

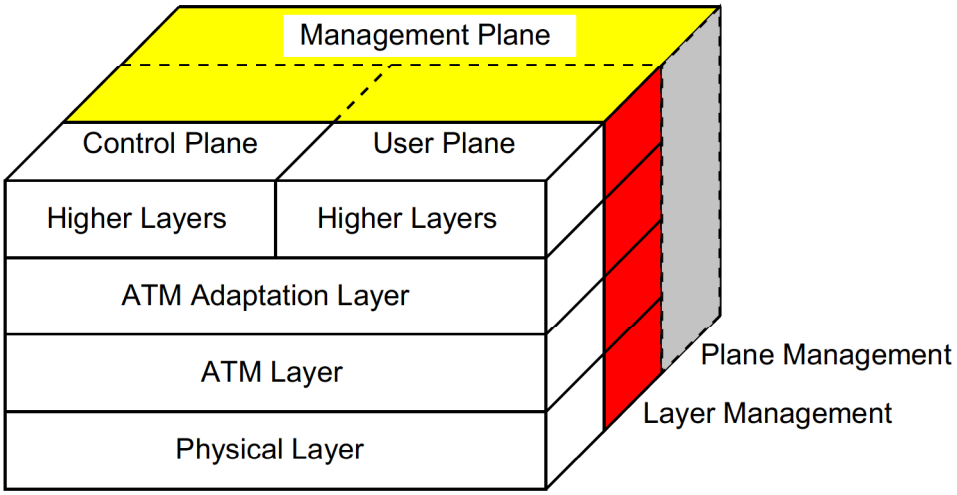


Abbildung 2.4: ATM-Referenzmodell

Der Physical Layer ist vom Übertragungsmedium abhängig und hat die Aufgabe, die Funktionen zum Zusammensetzen und Übertragen der Zellen über das physikalische Medium bereitzustellen. Zu den Aufgaben des ATM-Layer zählen:

- Multiplexen und Demultiplexen der Zellen verschiedener Verbindungen
- Kontrolle der VCI- und VPI-orientierten Funktionen
- Generierung und Extrahierung der Header-Informationen
- Generic Flow Control an der Benutzer-Netzwerk-Schnittstelle (UNI) zur Einrichtung der Verbindung.

Der ATM Adaptation Layer (AAL) dient den übergeordneten Schichten zur Adaptierung an den ATM Layer. Er führt die notwendige Segmentierung von Datenströmen durch und sorgt für eine gesicherte Übertragung. Der AAL Layer wird in zwei Teilschichten unterteilt:

1. Die *Segmentation and Reassembly* (SAR) Schicht, die die Abbildung der Protokolldateneinheiten der höheren Schichten auf die ATM-Zellen und umgekehrt übernimmt.

2. Der *Convergence Sublayer* (CS), der unerwünschte Effekte durch unterschiedliche Zellaufzeiten verschiedener Dienste ausgleicht.

Beispielsweise werden die bei der Sprachübertragung erzeugten kurzen Datenbursts über einen gewissen Zeitraum zusammengefaßt, um die Kapazität einer ATM-Zelle vollständig auszunutzen. Der Empfänger erzeugt aus den eintreffenden ATM-Zellen wieder einen kontinuierlichen Datenstrom.

Unterschiedliche Laufzeiten der Zellen durch das Netz werden durch den ATM Adaptation Layer beim Empfänger ausgeglichen. Dies wird durch die Addition eines konstanten zeitlichen Versatzes erreicht.

2.2.4 ATM Dienstklassen

Um die Anzahl der benötigten Protokolle für den ATM Adaptation Layer gering zu halten, werden die Dienste gemäß den Parametern „Zeitbezug zwischen Quelle und Senke“, Bitrate und Verbindungsart in vier verschiedene Klassen unterteilt, welche in Abbildung 2.5 dargestellt sind.

	Klasse 1	Klasse 2	Klasse 3	Klasse 4
Zeitbezug	zeitkontinuierlich		nicht zeitkontinuierlich	
Bitrate	konstant	variabel		
Verbindungsart	verbindungsorientiert			verbindungslos

Abbildung 2.5: Klassifizierung der Dienste im ATM Adaptation Layer

Den Nutzinformationen werden in Abhängigkeit von der benutzten Dienstklasse zusätzlich noch Steuerdaten angefügt. Diese Steuerdaten dienen dazu, Nutzdaten wiederherstellen zu können, die in mehrere Zellen aufgeteilt worden sind. Die Steuerdaten werden von der SAR Teilschicht beim Aufteilen der Daten in Zellen generiert. Im Empfänger muß die entsprechende SAR Teilschicht die Daten anhand der Steuerdaten wieder in der richtigen Reihenfolge zusammenfügen.

Zur Kennzeichnung der einzelnen Zellen werden diese mit Sequenznummern versehen. Anhand dieser ist es dem Empfänger möglich, den Verlust von Zellen zu erkennen. Während die Sequenznummer bei allen Dienstklassen vorhanden ist, sind zusätzliche Sicherungsdaten und verschiedene Segmenttypen nur bei einigen Klassen vorgesehen. Durch den unterschiedlichen Steuerdatenanteil ergibt sich ein Nutzdatenanteil von 44-48 Byte [36].

KAPITEL 3

Mobile Broadband System

3.1 Überblick

Das RACE II-Forschungsprogramm *(Research and Technology Development in Advanced Communications Technologies in Europe)* hat von 1992 bis 1995 die Entwicklung von Mobilfunksystemen der dritten Generation gefördert, welche die Vereinigung von Systemen wie GSM, DECT *(Digital European Cordless Telecommunications)* und Bündelfunksystemen und ihrer unterschiedlichen Anwendungsbereiche zu einem universellen Mobilfunksystem *(Universal Mobile Telecommunications System – UMTS)* mit Datenraten bis zu 2 Mbit/s ermöglichen sollen. Dabei wurden die Entwicklung einheitlicher Endgeräte und die Erweiterung der Dienste um breitbandige Dienste mit hohen Datenraten angestrebt [8].

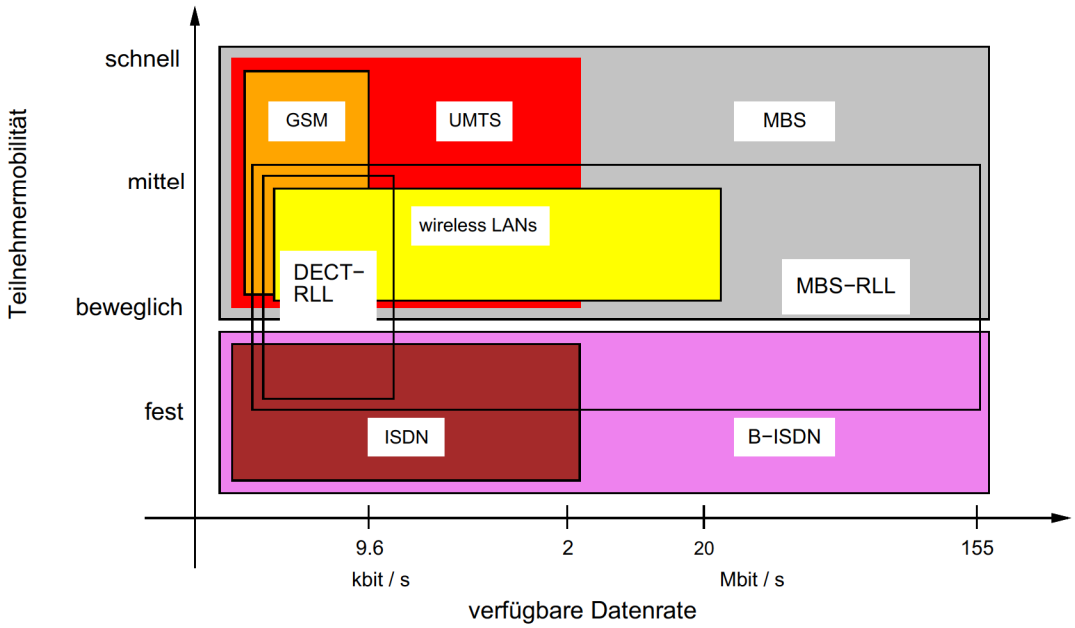


Abbildung 3.1: MBS und andere Datennetze

Das MBS-Projekt, das im Rahmen von RACE II gefördert wurde, beschäftigt sich mit der Anbindung mobiler Teilnehmer an stationäre Breitbandnetze *(Integrated Broadband Communication Network – IBCN)* bei Datenraten bis zu 155 Mbit/s. Darüberhinaus sollen schmalbandige Dienste weiterhin verfügbar sein. Ein Ziel von MBS ist insbesondere die Bereitstellung der Dienste des Breitband-ISDN für mobile Teilnehmer und die Unterstützung der ATM-Übertragung.

Das Konzept von MBS sieht neben der Anbindung an das Breitband-ISDN auch eine mögliche Zusammenarbeit mit anderen Systemen – wie zum Beispiel UMTS – vor. Dabei kann

der Typ des Netzes und der Integrationsgrad von einem privat betriebenen MBS-System mit niedriger Dienstintegration und Mobilität bis hin zum öffentlichen MBS-System mit starker Integration, weitreichender Mobilität und großem Versorgungsbereich variieren [3]. In Abbildung 3.1 wird MBS mit anderen Datennetzen in Beziehung gesetzt. Es wird deutlich, daß durch MBS das breite Dienstspektrum des Breitband-ISDN mit der Mobilität von Mobilfunknetzen kombiniert wird.

Aufgrund der Flexibilität von MBS und der Verfügbarkeit der Dienste des B-ISDN ist eine Vielfalt verschiedener Anwendungen möglich. Diese werden durch die benötigten Datenraten und die Mobilität ihrer Benutzer gekennzeichnet und sind in Abbildung 3.2 grafisch dargestellt.

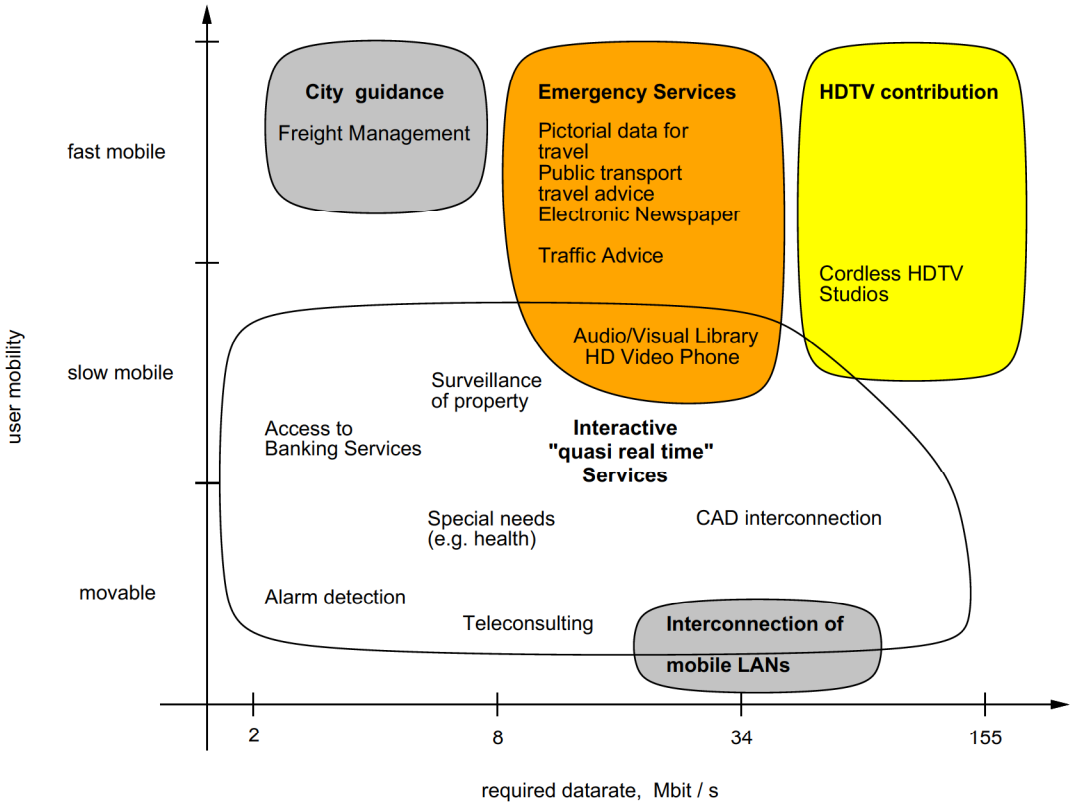


Abbildung 3.2: Anwendungen und Dienste des MBS

Bei der Integration breitbandiger Dienste in Mobilfunksysteme werden an die Funkschnittstelle *(air interface)* besondere Anforderungen gestellt. Diese resultieren aus der Mobilität der Teilnehmer und der Nutzung eines gemeinsamen physikalischen Übertragungsmediums. Für die mobile Anbindung an das ATM-Festnetz bieten sich grundsätzlich zwei unterschiedliche Realisierungsmöglichkeiten an:

1. Bereitstellung dienstspezifischer Protokolle zur Kommunikation über die Funkschnittstelle, bei denen die ATM-Übertragung nur zwischen Basisstation und Festnetz stattfindet.
2. Übertragung von ATM-Zellen über die Funkschnittstelle, wodurch die Mobilstationen transparent auf das ATM-Netz zugreifen können.

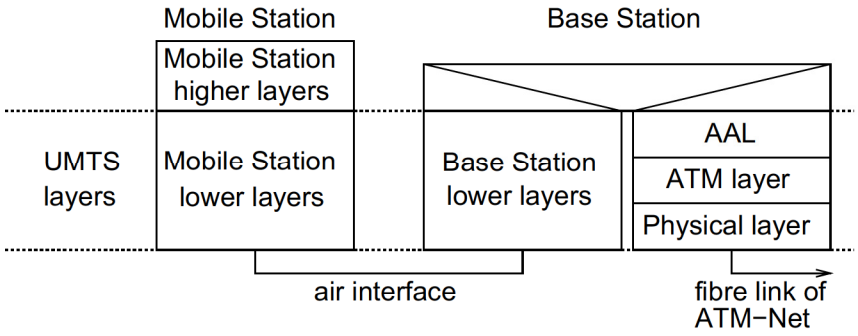
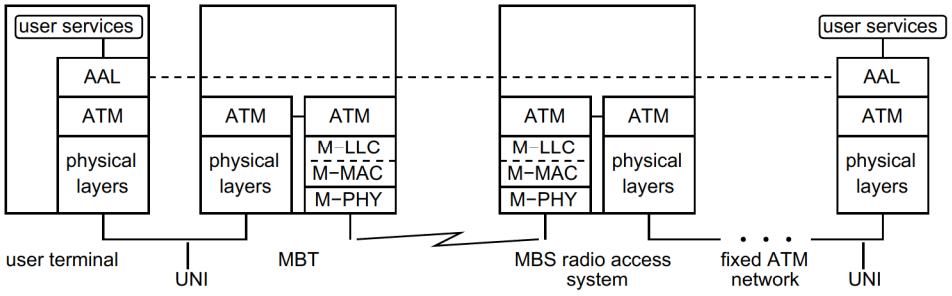


Abbildung 3.3: Dienstspezifische ATM-Übertragung

Das erste Verfahren wird zum Beispiel bei UMTS verwendet und benötigt keine Funktionen zur ATM-Übertragung über die Funkschnittstelle. Aus diesem Grund sind keine Kanalzugriffsverfahren notwendig, die das statistische Multiplexen von ATM unterstützen, und es können bestehende Protokolle an die Anforderungen verschiedener Dienste angepaßt werden. Nachteile dieser Vorgehensweise sind die Notwendigkeit verschiedener, komplexer Kanalzugriffsverfahren, der fehlende transparente ATM-Zugang und die Inflexibilität bezüglich der Integration neuer Dienste (Abbildung 3.3).



AAL	ATM Adaptation Layer	M−PHY	MBS Physical Layer
ATM	Asynchronous Transfer Mode Layer	MBT	Mobile Broadband Termination
M−LLC	MBS Logical Link Control Layer	UNI	User to Network Interface
M−MAC	MBS Medium Access Control Layer		

Abbildung 3.4: Transparenter, mobiler ATM-Zugriff

Transparenter Zugriff bedeutet, daß die ATM-Anpassungsschicht nicht geändert werden muß und auf die MBS-PHY-Schicht zugreifen kann wie auf die ATM-PHY-Schicht eines konventionellen ATM-Netzes. Der zweite Ansatz benötigt deshalb nur ein dienstunabhängiges MAC Scheme (**M***edium* **A***ccess* **C***ontrol*) zur Übertragung im Asynchronous Transfer Mode. Zukünftig hinzukommende ATM-Dienste werden durch diese universelle Lösung ebenfalls unterstützt. Allerdings sind Kanalzugriffsverfahren notwendig, die das statistische Multiplexen des ATM unterstützen. Durch dieses Verfahren kann zwar die gleiche Funktionalität, aber nicht die gleiche Dienstgüte wie im B-ISDN angeboten werden. Dieser Ansatz wurde im MBS-System gewählt (Abbildung 3.4) [15].

3.2 Bitübertragungsschicht im MBS

Die Bitübertragungsschicht *(Physical Layer)* stellt die unterste Schicht im ISO-OSI-Referenzmodell dar und definiert die Protokolle zur Übertragung der Bits auf dem physikalischen Kanal. In MBS ist die Übertragung von hohen Datenraten vorgesehen, für die eine große Bandbreite erforderlich ist. Aus diesem Grund sind Frequenzen im 60 GHz-Frequenzband als Träger gewählt worden. Zum einen sind diese Frequenzen noch nicht anderweitig vergeben, zum anderen ist die Sauerstoffdämpfung der Atmosphäre sehr hoch. Der Radius einer Funkzelle hängt neben den gerade genannten Parametern maßgeblich von der Rauschzahl, dem erforderlichen Signal-zu-Rauschleistungsverhältnis im Empfänger, der Bandbreite und den Kanaleigenschaften ab. Unter Berücksichtigung dieser Faktoren ergeben sich Zellradien in der Größenordnung von $R = 100 - 200$ m. Aufgrund der Dämpfung sind die Wiederverwendbarkeitsabstände einer Frequenz sehr kurz. Dadurch lassen sich hohe Teilnehmerdichten erreichen. Eine Zellplanung, wie sie zum Betrieb von mobilen Kommunikationsnetzen der zweiten Generation nötig ist, kann wegen der mikrozellularen Struktur in MBS nicht mehr durchgeführt werden. Hier müssen Verfahren zur dynamischen Kanalvergabe eingesetzt werden.

Zur Synchronisation ermittelt die Mobilstation den Zeittakt auf dem Downlink, den sie um die Signallaufzeit verzögert empfängt. Dies führt im Uplink zu maximalen Asynchronitäten zwischen Mobilstationen nahe der Basisstation und solchen in großer Entfernung von $T_{asynch} = \frac{2 \cdot R}{c} \approx 0.666\,\mu s$ bis $1.33\,\mu s$ (c ist die Lichtgeschwindigkeit), da das Sendesignal der Mobilstation um die entsprechende Laufzeit verzögert in der Basisstation ankommt. Eine Messung der auch als Schleifen-Laufzeit bezeichneten Größe T_{asynch} durch die Basisstation, wie dies z.B. bei GSM-Netzen erfolgt, ist in MBS nicht vorgesehen.

Frequenzband für den Downlink	62 - 63 GHz
Frequenzband für den Uplink	65 - 66 GHz
Anzahl der Frequenzkanäle	32
Bandbreite je Frequenzkanal	30 MHz
Duplextrennung	2 FDD
Multiplexverfahren	FDMA/TDMA (hybrid)
Slotlänge	21,33 μs
Guard Time	1,33 μs
Modulationsverfahren	QPSK/16 QAM
Datenrate	20 $\frac{MSymbols}{s}$ ($T_{Symbol} = 50\ ns$)
ATM-Zellen/Slot	1 ($\hat{=}$ 424 Bit)
Asynchronität	$T_{asynch} \approx$ 0,666 bis 1,33 μs ($\approx$ 13 bis 26 Symbole)
Funkzellenradius	$100 - 200\ m$

Tabelle 3.1: Systemparameter des MBS-Systems

Grundlage des Systems stellt eine hybride FDMA/TDMA-Kanalstruktur (**F**requency/ **T**ime **D**ivision **M**ultiple **A**ccess) dar. Der Frequenzbereich ist in 32 Kanäle zu je 30 MHz Bandbreite unterteilt, die durch Schutzbänder voneinander getrennt sind. Der Zeitbereich wird in Zeitschlitze *(Slots)* aufgeteilt. Die Dauer eines Slots beträgt $T_{slot} = (20 + 1.33)\,\mu s$. Darin ist auf dem Uplink eine Schutzzeit *(Guard Time)* von $T_{guard} = 1.33\,\mu s$ enthalten, die die zeitliche Überlappung der Pakete verhindern soll, die nacheinander von unterschiedlichen Mobilstationen auf demselben Träger gesendet wurden. Die Länge der Guard Time

entspricht der maximalen Asynchronität T_{asynch}, so daß unabhängig von der Position einer Mobilstation innerhalb einer Zelle die Summe aus Laufzeit und Dauer eines Paketes die Slotdauer nicht übersteigt. Diese 1.33 μs entsprechen der Signallaufzeit von Basis- zu Mobilstation und wieder zurück, wenn die Mobilstation die größtmögliche Entfernung von der Basisstation einnimmt. Auf dem Downlink ist eine Schutzzeit zum Einschwingen des Senders bei Anwendung von Powercontrolverfahren notwendig.

Die Aufteilung eines Slots ist in Abbildung 3.5 dargestellt. Neben den eigentlichen Daten (2 x 168 Symbole) ist in der Mitte jedes Slots eine *Training Sequence* (15 Symbole) enthalten, die dem Equalizer zur Schätzung der Kanalstoßantwort und zur Synchronisation dient. In der Abbildung 3.5 sind drei verschiedene Slotlängen abgebildet, es wird aber nur mit normalen und sehr kurzen Slots gearbeitet. Normale Slots dienen der Datenübertragung, während sehr kurze Slots zur Übertragung dynamischer Parameter und zur Kollisionsauflösung verwendet werden.

Wird QPSK (**Q**uaternary **P**hase **S**hift **K**eying) angewendet, entspricht ein Symbol zwei Bit, im Falle des Modulationsverfahrens 16 QAM (**Q**uadrature **A**mplitude **M**odulation) wäre ein Symbol äquivalent mit 4 Bits. Der Aufwand, wie auch die Fehleranfälligkeit des zweiten Modulationsverfahrens sind deutlich höher einzustufen, so daß in der ersten Implementierungsstufe mit QPSK gearbeitet werden soll.

Im weiteren wird als Modulationsverfahren QPSK angenommen, so daß $168 \cdot 2 \cdot 2$ Bits = 672 Bits = 84 Bytes im Falle eines normalen Slots zur Verfügung stehen. Davon braucht eine ATM Zelle bekanntlich 53 Bytes, die übrigen 31 Bytes teilen sich die MAC und die LLC Schicht als Overhead. Ein Slot sehr kurzer Länge, der von nun an Subslot genannt wird, hat $24 \cdot 2 \cdot 2$ Bits = 96 Bits = 12 Bytes [15].

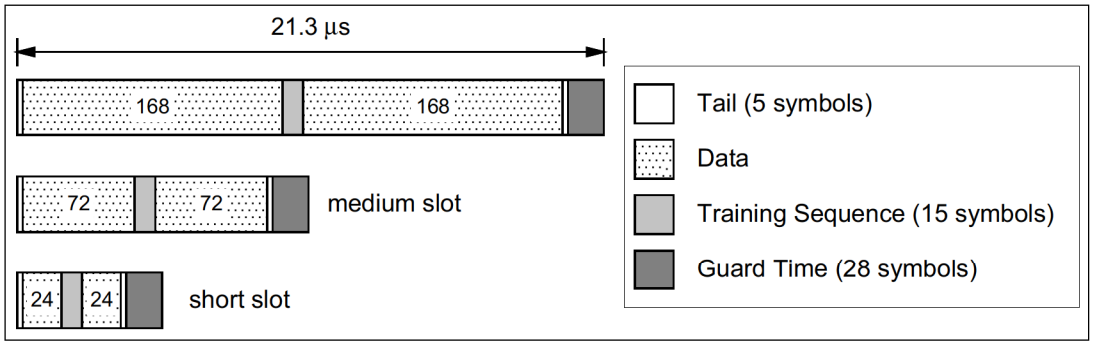


Abbildung 3.5: Aufbau eines Slots

3.3 Zellstruktur des MBS

Die Zellstruktur des MBS ähnelt der des GSM (s. Abbildung 3.6).

Ein **B**ase-**S**tation-**C**ontroller (BSC) hat eine Verbindung zum ATM-Festnetz, an dem auch andere BSCs angeschlossen sind. Die Sicherungsschicht sowie die ATM-Schicht befinden sich innerhalb dieses Controllers. Die eigentlichen Empfangs-/Sendeeinrichtungen (**B**ase-**S**tation-**T**ransceiver (BST)) befinden sich innerhalb einer **B**ase-**T**ransceiver-**S**tation (BTS). Die BTS sind innerhalb des Versorgungsgebietes eines BSC räumlich verteilt und geben den BST eine lokale Zuordnung. An einer *Base-Transceiver-Station* können mehrere *Base-Station-Transceiver* installiert sein. Jeder *Base-Station-Transceiver* sendet dabei auf

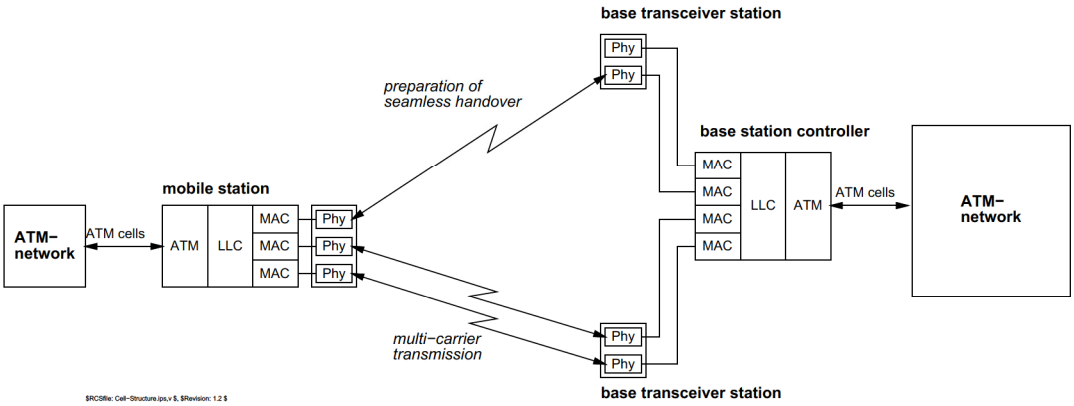


Abbildung 3.6: Zellstruktur des MBS-Systems

einer anderen Frequenz. In den *Base-Transceiver-Stations* sind im Gegensatz zum GSM nur Elemente der physikalischen Schicht vorhanden. Eine Mobilstation (MS) enthält alle MBS-Schichten. An einer Mobilstation kann auch wiederum ein eigenes mobiles ATM-Netz angeschlossen sein, das aus mehreren ATM-Endgeräten zusammengesetzt sein kann.

Um den hohen Datenraten der angeschlossenen Endgeräte gerecht werden zu können, muß die Mobilstation unter Umständen mehrere physikalische Kanäle zum selben oder auch zu einem anderen *Base-Station-Controller* aufbauen (*Multilinkübertragung*). Ein physikalischer Kanal kann inklusive Signalisierung 77000 ATM-Zellen/s $\hat{=}$ 34 Mbit/s übertragen. Das *Mobile-Broadband-System* soll aber Datenrate bis 155 Mbit/s beherrschen können, was fünf gleichzeitig betriebenen Frequenzkanälen entspricht.

Bei schlechter werdender Empfangsqualität muß die Mobilstation in der Lage sein, einen *Handover* zu einem anderen *Base-Station-Transceiver* desselben oder eines anderen *Base-Station-Controller* auszuführen. Dieser *Handover* muß *nahtlos* (engl.: seamless) vor sich gehen, d.h. während des *Handovers* dürfen keine Daten verlorengehen oder verzögert werden. Dies ist nur möglich, wenn rechtzeitig ein zweiter physikalischer Kanal zu einer anderen *Base-Transceiver-Station* aufgebaut wurde und für eine gewisse Zeit beide Kanäle nebeneinander betrieben werden. Da Störungen durch Fading, Abschattung oder ähnliches im 60 GHz-Bereich quasi ohne Vorwarnung auftreten können, müssen ständig zumindest Signalisierungslinks zu anderen BTS verfügbar sein.

KAPITEL 4

Logical Link Control (LLC) im MBS

Die LLC-Schicht ist oberhalb der MAC-Schicht angeordnet und bildet zusammen mit ihr die Sicherungsschicht im ISO/OSI-Referenzmodell.

Das MBS soll einen mobilen, transparenten Zugang zu ATM-Netzen ermöglichen. Transparenz bedeutet hierbei, daß die Protokolle der ATM-Anpassungsschicht vom MBS-System unberührt bleiben. Beim MBS sollen Daten in der gleichen Menge und Qualität wie bei ATM-Festnetzen zur Verfügung stehen. Aufgabe der LLC-Schicht ist es dabei, annähernd so gute Qualitätsstandards wie beim ATM-Festnetz zu ermöglichen.

4.1 Aufgaben und Anforderungen

Das ATM Konzept basiert auf der Annahme, daß eine hohe Übertragungsqualität garantiert ist, was für die derzeitig angewendete optische Glasfasertechnologie sicherlich zutrifft. Die Übertragung über eine Funkschnittstelle ist jedoch weit weniger zuverlässig und stark von den Umgebungsbedingungen abhängig. Da aber auch die Glasfasertechnologie das Auftreten von Übertragungsfehlern nicht vollständig verhindern kann, wird in ATM-Netzen, abhängig vom Diensttyp, innerhalb der ATM Anpassungsschicht (AAL) ein vereinfachtes Verfahren zum End-zu-End Fehlerschutz durchgeführt. Außerdem wird innerhalb der ATM-Schicht eine Header-Error-Control vorgenommen. Es wird im folgenden erläutert, daß mit diesem vereinfachten Fehlerschutzverfahren die geforderte Fehler-Performance nicht zu erreichen ist, wenn ein oder mehrere Glasfaserlinks durch Funklinks ersetzt werden. Stattdessen sind zusätzliche Fehlerkorrekturverfahren erforderlich, die speziell auf die Eigenschaften der Funkübertragung zugeschnitten sind. Aufgrund der geforderten Transparenz für die Protokolle des AAL, sowie der in Abschnitt 4.1.1 aufgeführten Gründe können die zusätzlich erforderlichen Fehlerkorrekturverfahren nur unmittelbar an der Funkschnittstelle ausgeführt werden. Somit liegt der klassische Fall eines LLC Protokolls auf dem Funklink vor.

4.1.1 End-zu-End Fehlerkorrektur innerhalb des AAL

Aufgrund der Vielzahl von Applikationen werden an die End-zu-End Fehlerschutzverfahren innerhalb der verschiedenen AAL-Typen unterschiedliche Anforderungen gestellt.

Bei AAL Type 1 und AAL Type 2 wird ein einfaches Verfahren zum Erkennen von verloren gegangenen oder fehlerhaft eingefügten ATM-Zellen verwendet, so daß die Bitfehlerrate, sowie die Zellverlustrate unverändert an den Dienstbenutzer des AAL weitergegeben wird. Optional können mit Hilfe von FEC einzelne Bitfehler erkannt und ggf. korrigiert werden [19]. Eine aufgrund eines Funklinks stark erhöhte Bitfehlerrate innerhalb der ATM-Schicht kann zu Verletzung der von bestimmten Diensten geforderten Dienstgüte führen, so daß etwa die Wiedergabe eines Sprachdienstes unakzeptabel stark verfälscht werden kann. Daher können bei einer derartigen Konfiguration eines ATM-Netzes bestimmte Dienste

nicht ohne zusätzliche Fehlerkorrekturverfahren an der Funkschnittstelle innerhalb der ATM-Schicht unterstützt werden.

In AAL Type 3/4 sowie AAL Type 5 wird innerhalb des Service-Specific-Convergence-Sublayer (SSCS) ein typisches Transportprotokoll mit Wiederholungsanforderungen (Automatic-Repeat-Request, ARQ) ausgeführt, welches sich auf die Funktionen zur Detektion von Zellverlusten und Bitfehlern der unteren Teilschichten Common-Part-Convergence-Sublayer (CPCS) und Segmentation-and-Reassembly (SAR) stützt [12]. Ein SSCS-Protokoll für AAL Type 3/4 wurde bisher nicht spezifiziert. Für AAL Type 5 wurde das Service-Specific-Connection-Oriented-Protocol (SSCOP) [20] spezifiziert, welches mit sehr langen Sequenznummern arbeitet (24 Bit) und dessen Informationsrahmen eine maximale Länge von 64KByte besitzen. Bei einer angenommenen End-zu-End Bitfehlerrate von 10^{-7} ergibt sich für 1KByte Rahmen eine Rahmenverlustrate von 10^{-3}, womit eine effiziente Ausführung des ARQ-Protokolls möglich ist [13]. Typische Bitfehlerraten auf einer nur mit FEC gesicherten Funkschnittstelle betragen jedoch etwa 10^{-3} bis 10^{-5} [7, 33]. Damit beträgt die Rahmenverlustrate für Rahmen der Größe 1KByte über 8%, womit Fehlerkorrektur durch ein Transportprotokoll sehr viel ineffizienter in bezug auf Overhead und zusätzliche Verzögerungen durch Übertragungswiederholungen wird, als Fehlerkorrektur auf Basis einzelner ATM-Zellen innerhalb eines LLC Protokolls an der Funkschnittstelle.

Zusammenfassend wird deutlich, daß die Algorithmen innerhalb der einzelnen AAL-Typen auf ein bestimmtes Fehlerverhalten der ATM-Schicht zugeschnitten sind. Die geforderten Bitfehlerraten und Zellverlustraten sind dabei nicht nur vom AAL-Type, sondern auch von der speziellen Applikation abhängig, deren Daten über einen Virtuellen Kanal (VC) übertragen werden. Diese Anforderungen sind in den VC-spezifischen QoS Parametern angegeben [18], welche von der ATM-Schicht eingehalten werden müssen. Die Fehlerkorrekturverfahren an der Funkschnittstelle müssen also den Aufwand für Fehlerkorrektur adaptiv an die QoS-Anforderungen einzelner VCs anpassen [27].

Zwei Beispiele sollen den notwendigen Grad an Adaptierung verdeutlichen:

Ein Telefondienst verwendet den AAL Type 1 und wird innerhalb der ATM Schicht auf die CBR Dienstklasse abgebildet. Er ist durch eine relativ niedrige, konstante Datenrate gekennzeichnet und hat sehr hohe Anforderungen an das Echtzeitverhalten, welches dadurch gekennzeichnet ist, daß ATM-Zellen verworfen werden, deren Übertragungsverzögerung einen bestimmten maximalen Wert überschreitet. Des weiteren ist ein Telefondienst relativ unempfindlich gegenüber Übertragungsfehlern, so daß Bitfehlerraten von 10^{-4} nur zu geringen Einbußen in der Sprachqualität führen. Daher kann das ARQ-Protokoll in der LLC-Schicht an der Funkschnittstelle für CBR Dienste relativ einfach aufgebaut sein, und etwa immer nur eine einzige Übertragungswiederholung pro ATM-Zelle ausführen [37]. Bei mehrfacher fehlerhafter Übertragung derselben ATM-Zelle wird diese verworfen.

Ein zeitunkritischer Datendienst (z.B.: Filetransfer) verwendet den AAL Type 3/4 oder AAL Type 5 und wird innerhalb der ATM Schicht auf die ABR Dienstklasse abgebildet. Werte für eine maximale Verzögerung sind nicht definiert. Gegebenenfalls ist eine mittlere Verzögerung mit einem hohen Wert definiert. Die Bitfehlerrate sollte unter 10^{-9} liegen und wird durch den AAL noch weiter reduziert. Für derartige Dienste kann in der LLC-Schicht an der Funkschnittstelle ein HDLC-artiges [16] ARQ-Protokoll ausgeführt werden.

Für echtzeit-orientierte Dienste des AAL Type 2, welche innerhalb der ATM-Schicht auf die VBR Klasse abgebildet werden, muß ein Kompromiß zwischen Rest-Bitfehlerrate und

Übertragungsverzögerungen getroffen werden, wobei die Anzahl von Übertragungswiederholungen entsprechend den Anforderungen eines VC (QoS-Parameter) sowie den momentanen Randbedingungen wie Lastaufkommen innerhalb einer Station, sowie auf dem gemeinsam genutzten Funkressourcen (statistisches Multiplexen) abhängt.

Da das LLC-Protokoll an der Funkschnittstelle unterhalb der ATM-Schicht angesiedelt ist, müssen die unterschiedlichen Anforderungen der AAL-Typen bezüglich des Fehlerverhaltens der ATM-Schicht auf die ATM-Klassen umgesetzt werden. Dabei werden VCs des ATM Type 1 zumeist auf CBR abgebildet, AAL Type 2 auf VBR und AAL Type 3/4 sowie AAL Type 5 auf ABR.

4.2 Anforderungen an ein ARQ-Protokoll

Es stellt sich nun die Frage, ob in der LLC-Schicht je ein ARQ-Protokoll für jede ATM-Klasse spezifiziert werden muß oder ob ein generisches Protokoll für alle Klassen wiederverwendet werden kann. In dieser Arbeit werden die Eigenschaften eines derartigen generischen Protokolls untersucht, welches für jede ATM-Klasse parametrierbar ist. Anschließend wird untersucht, inwiefern sich dieses Protokoll vereinfachen läßt, wenn es auf die Bedürfnisse einer bestimmten ATM-Klasse zugeschnitten wird. Die in einem realen System verwendete Lösung wird je nach den tatsächlichen Randbedingungen wie Bandbreite, Anwenderprofil, Professionalitätsgrad oder Kosten unterschiedlich ausfallen. Die in dieser Arbeit zusammengetragenen Untersuchungen sollen dabei als Entwurfsrichtlinien verwendet werden können.

Wenn zwischen zwei Stationen mehrere VCs parallel verlaufen, kann das ARQ-Protokoll für einzelne VCs ausgeführt werden, oder für ein Bündel von gemultiplexten VCs (Abbildung 4.1). Daraus resultiert eine Struktur von mehreren unabhängigen ARQ-Instanzen, denen je ein VC oder mehrere durch einen VC-Multiplexer gebündelte VCs zugeordnet sind. Der VC Multiplexer ersetzt die Multiplexfunktion innerhalb der ATM-Schicht. Eine ARQ-Instanz überträgt ihre ARQ-Rahmen über einen ihr zugeordneten ARQ-Kanal. Parallele ARQ-Kanäle (zwischen denselben Partnerinstanzen) können durch einen ARQ-Multiplexer auf einen MAC-Kanal gebündelt werden. Für bestimmte ARQ-Kanäle ist aber auch ein direktes Durchreichen an die MAC-Schicht sinnvoll.

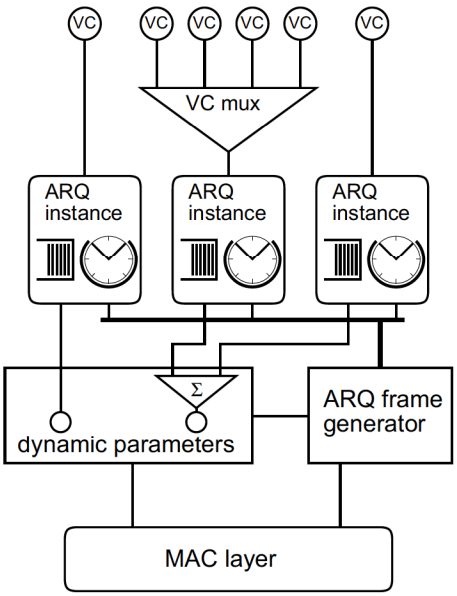


Abbildung 4.1: Verteilung von verschiedenen VCs auf ARQ-Instanzen

Ein wesentlicher Posten, der Einfluß auf die Komplexität und damit die Kosten der LLC-Schicht hat, ist die Anzahl von parallelen ARQ-Instanzen, welche sich aus dem Verfahren zur Zuordnung von VCs zu ARQ-Instanzen ergibt. Da VCs von verschiedenen ATM-Klassen sehr unterschiedliche Anforderungen an die Zellverlustrate oder Übertragungsverzögerungen haben, ist es sinnvoll,

nur VCs mit ähnlichen Anforderungen durch einen VC-Multiplexer oberhalb einer ARQ-Instanz zusammenzufassen.

Parallele ARQ-Instanzen haben den Vorteil, daß ATM-Zellen von zeitkritischen VCs solche von unkritischen VCs überholen können und nicht erst deren Übertragung abwarten müssen. Dadurch werden die Übertragungsverzögerungen aufgrund der Wartezeit auf einen Übertragungszeitschlitz deutlich reduziert.

Desweiteren sollten CBR-artige VCs nicht mit anderen VCs gemultiplext werden, sondern von einer eigenen ARQ-Instanz behandelt werden, wodurch ihre ARQ-Rahmen gesondert an die MAC-Schicht übergeben werden können.

Für echtzeitorientierte, VBR-artige VCs ist in [27] anhand eines Beispiels erläutert, daß die Wartezeiten im Resequencing-Buffer des Empfängers reduziert werden könnten, wenn die Zugehörigkeit von fehlerhaft empfangenen und zur Wiederholungsübertragung angeforderten ARQ-Rahmen zu bestimmten VCs bekannt wäre. Im Falle des Verlustes eines ARQ-Rahmens mit einer ATM-Zelle müssen alle weiteren Zellen mit höherer Sequenz-Nummer im Resequencing-Buffer zwischengespeichert werden, um die Übertragungswiederholung der verlorenen Zelle abzuwarten. Falls die verlorene Zelle jedoch zu einem anderen VC gehört, als die wartenden Zellen, ist das Verzögern der bereits erfolgreich empfangenen Zellen nicht notwendig. Wenn eine ARQ-Instanz nur ATM-Zellen eines einzigen VCs überträgt, ist eine derartige unnötige Vergrößerung der Übertragungsverzögerung ausgeschlossen.

Die Wartezeiten im Resequencing-Buffer können aber auch im Falle von mehreren VCs innerhalb derselben ARQ-Instanz dadurch gesenkt werden, daß eine schnelle Übertragung von positiven oder negativen Quittungen ermöglicht wird. Dies ist etwa bei hochratigen, symmetrischen Kanälen der Fall, die einen großen Teil der Kapazität eines Trägers belegen, so daß Quittungen sehr häufig nach kurzer Wartezeit huckepack auf einem Informationsrahmen der Gegenrichtung übertragen werden können. Da die Übertragungsverzögerungen als Vielfaches der Länge eines Zeitschlitzes auftreten, ist ihre absolute Größe stark von der dem System zugrundeliegenden Bandbreite abhängig und damit eine Definition einer Schranke zur Kennzeichnung hochratiger Kanäle von den jeweiligen Randbedingungen abhängig. Eine schnelle Übertragung von positiven Quittungen kann aber auch durch spezielle Verfahren innerhalb der MAC-Schicht ermöglicht werden, indem mit der Übertragung von ATM-Zellen in ARQ-Rahmen automatisch eine Reservierung für einen kurzen Zeitschlitz auf dem Rückkanal zur Übertragung eines Feedbacks oder einer Quittung erfolgt [29].

Zusammenfassend lassen sich folgende Regeln für das Zuordnen von VCs zu ARQ-Instanzen ableiten:

- Nur für VCs mit ähnlichen Anforderungen an Zellverlustrate und Übertragungsverzögerung ist ein Multiplexen oberhalb einer ARQ-Instanz sinnvoll.
- Alle ABR-artigen VCs sollten von derselben ARQ-Instanz bearbeitet werden.
- Für zeitkritische CBR-artige oder VBR-artige VCs sollte je eine eigene ARQ-Instanz eingerichtet werden.
- Im Falle einer garantierten, sehr schnellen Übertragung von positiven oder negativen Quittungen können auch zeitkritische VCs oberhalb einer ARQ-Instanz gemultiplext werden.

4.3 Zusammenwirken von LLC-Schicht und MAC-Schicht

Bevor in Abschnitt 5.2 detailiert auf die Funktionalität des Protokolls innerhalb einer ARQ-Instanz eingegangen wird, werden in diesem Abschnitt die Algorithmen des ARQ-Multiplexers sowie ihr Zusammenspiel mit den Algorithmen der MAC-Schicht erläutert. Dazu werden zunächst grob die Anforderungen an die MAC-Schicht sowie deren Funktionalität beschrieben. Anschließend werden die von ihr zur Verfügung gestellten Dienste erläutert, welche von der LLC-Schicht vorteilhaft genutzt werden können.

Die Anforderungen an eine transparente Übertragung von ATM-Zellen über die Funkschnittstelle zwischen mehreren Mobilstationen und einer zentralen Basisstation resultieren in einem Kanalzugriffsprotokoll, welches das statistische Multiplexen von ATM-Multiplexern auf das spezielle Szenario erweitert, welches durch den konkurrierenden Vielfachzugriff von nicht einfach zu koordinierenden Mobilstationen charakterisiert ist. Das Protokoll muß dabei das Übertragungsmedium den einzelnen Mobilstationen abhängig von deren momentanen Übertragungsanforderungen zuteilen, wobei eine TDMA Struktur zugrunde gelegt wird, dessen Zeitschlitze für die Übertragung einzelner ATM-Zellen ausgelegt sind. Des weiteren wird Vollduplexübertragung vorausgesetzt. Der Uplink bildet wegen des Wettbewerbs und der damit verbundenen Kollisionen den kritischen Teil für das Kanalzugriffsprotokoll.

In der Literatur finden sich eine Reihe von Protokollen, in denen die Basisstation als zentrale Instanz die Vergabe von Übertragungskapazität an Mobilstationen auf der Basis einzelner Zeitschlitze durchführt [29, 1, 5]. Dabei bildet eine Zelle, bestehend aus mehreren Mobilstationen und einer zentralen Basisstation, einen ATM-Multiplexer, der sich als verteiltes Warteschlangensystem modellieren läßt. Da die Datenrate auf der Funkschnittstelle (34Mbit/s pro Träger in MBS [33]) im Vergleich zum Festnetz relativ niedrig ist, muß die Basisstation bei der Vergabe von Zeitschlitzen sowohl für den Uplink als auch für den Downlink einen Scheduler mit statischen oder dynamischen Prioritäten verwenden [14, 26, 22], damit für ATM-Zellen von echtzeitorientierten VCs die geforderten niedrigen Übertragungsverzögerungen erreicht werden können [28]. Dieser Scheduler befindet sich innerhalb der Basisstation. Aufgrund der besonderen Bedingungen der Funkschnittstelle wird im folgenden von der Verwendung von dynamischen Prioritäten ausgegangen.

Da eine ATM-Zelle in einen ARQ-Rahmen eingebettet ist, der in seinem Header eine Quittung, sowie die aktuellen dynamischen Parameter enthält, ist es nicht sinnvoll, zu versendende ATM-Zellen innerhalb der MAC-Schicht zwischenzuspeichern, um einen Übertragungszeitschlitz abzuwarten. Statt dessen sollte eine ATM-Zelle erst unmittelbar vor ihrer Übertragung vom *ARQ frame generator* in einen ARQ-Rahmen eingebettet und an die MAC-Schicht geliefert werden.

Basierend auf den dynamischen Parametern bestimmt der Scheduler der MAC-Schicht für jeden Zeitschlitz eine Mobilstation, zu der, bzw. von der eine ATM-Zelle übertragen werden darf. Die LLC-Schicht muß daraufhin einen ARQ-Rahmen erzeugen, der neben einer ATM-Zelle zusätzlich im Header eine Quittung und im Falle des Uplinks einen Satz von dynamischen Parametern enthält. Es wird diejenige ATM-Zelle versendet, deren *Dringlichkeit auf Übertragung* am größten ist. Diese Dringlichkeit entspricht ihrer Priorität im Scheduler. Eine mögliche Abarbeitungsstrategie ist G/D/1/FCFS/RU-NONPRE (relative urgency) [38], bei der die ATM-Zellen nach ihren entstandenen Wartezeiten und ihren verbindungsspezifischen Dienstgüteanforderungen zur Übertragung bevorzugt werden. Damit wird erreicht, daß die Schwankung der Verzögerung (delay jitter) jeder ATM-Zelle sowie

die maximale Zellverlustrate der virtuellen Verbindung individuell überwacht und zur Berechnung eines spätmöglichsten Übertragungstermins der Zelle herangezogen wird. Die Wahrscheinlichkeit für eine Verspätung einer Zelle (Terminüberschreitung) wird dabei minimiert [39].

Folgende Eigenschaften der MAC-Schicht können von der LLC-Schicht vorteilhaft ausgenutzt werden:

- ARQ-Rahmen werden nicht innerhalb der MAC-Schicht zwischengespeichert, sondern unmittelbar vor ihrer Übertragung von der LLC-Schicht generiert. Dies resultiert in einer deterministischen Laufzeit der ARQ-Rahmen zwischen den Partner-ARQ-Instanzen in Mobil- und Basisstation.
- Da die MAC-Schicht statistisches Multiplexen auf dem physikalische Kanal realisiert, steht den ARQ-Instanzen Übertragungskapazität entsprechend deren Bedarf zur Verfügung, welche sie dynamisch beanspruchen können.
- Aufgrund der Übertragung von ATM-Zellen haben alle Informationsrahmen dieselbe Länge.

4.4 Struktur der MAC-Schicht

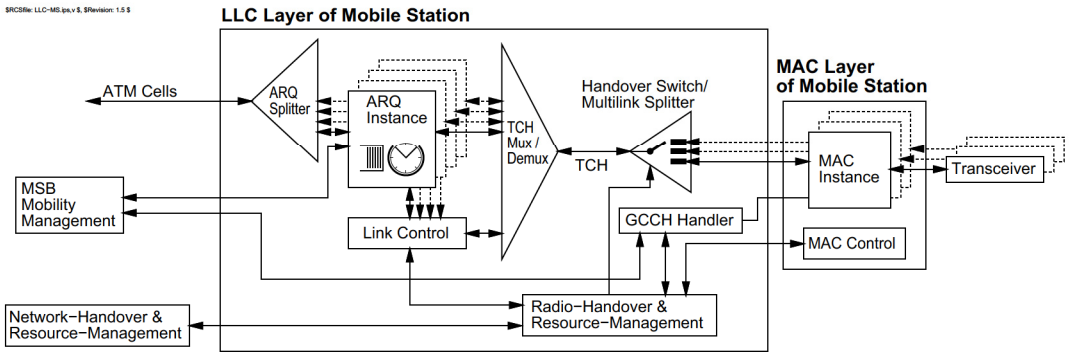


Abbildung 4.2: Aufbau der LLC-Schicht in der Mobilstation

Im rechten Teil der Abbildung 4.2 erkennt man die einzelnen Instanzen der MAC-Schicht. Jeder MAC-Instanz ist ein Transceiver zugeordnet. Der **G***lobal* **C***ontrol* **CH***annel (GCCH) Handler* stellt einen verbindungslosen Dienst zur Verfügung, sobald die Mobilstation auf Frequenzen der Basisstationssende-/empfangseinrichtungen synchronisiert ist. Der *Handover Switch/Multilink Splitter* bietet einen verbindungsorientierten Dienst an. Über ihn werden hauptsächlich Nutzdaten übertragen. Die Signalisierung zum Aufbau dieser Verbindungen geschieht über den GCCH Handler, wenn keine anderen Verbindungen aufgebaut sind. In beiden Fällen wird ein ungesicherter Dienst zur Verfügung gestellt, d.h. es wird keine Gewähr für eine erfolgreich durchgeführte Übertragung übernommen.

4.5 Struktur der LLC-Schicht

In der Abbildung 4.2 ist links neben der MAC-Schicht die LLC-Schicht dargestellt. Die LLC-Schicht kann in einen unteren und einen oberen Teil aufgeteilt werden. Die Grenze

liegt zum einem auf dem mit *TCH* gekennzeichneten **T***raffic* **CH***annel*, zum anderen findet die Teilung zwischen *Radio-Handover & Resource-Management* und *Link Control* statt.

Die untere LLC-Schicht besteht in der Mobilstation im wesentlichen aus dem *GCCH-Handler*, dem *Handover Switch/Multilink Splitter* und dem *Radio-Handover & Resource--Management*. Der *GCCH-Handler* verteilt die Daten, die von MAC-Schicht ankommen, und gibt sie entweder an das *Radio-Handover & Resource-Management* oder an die obere Schicht ab. Der *Handover Switch/Multilink Splitter* teilt die Daten, die er über den *Traffic-Channel* erhält, auf die einzelnen MAC-Instanzen auf. Das *Radio-Handover & Resource-Management* macht Aussagen darüber, auf welchen Verbindung welche Datenmenge versendet werden soll. Die Aufteilung der Datenmengen geschieht im *Radio-Handover & Resource-Management* des *Base-Station-Controller*. Hier sollten auch die CAC Algorithmen angesiedelt sein. Des weiteren hat das *Radio-Handover & Resource-Management* die Aufgabe, den Handover vorzubereiten und die Verbindungen zu verwalten. Es entscheidet über die Einrichtung von Signalisierungsverbindungen zu *Base-Transceiver-Stations* bzw. anderen *Base-Station-Controllern*.

Im linken oberen Teil der Abbildung 4.2 sind die ARQ-Instanzen sichtbar. Nach oben hin werden die ATM-Zellen auf einer einzigen Verbindung zu einem ATM-Zellstrom zur nächsthöheren Schicht im *ARQ-Splitter* zusammengefaßt. Damit die lange Nummer des virtuellen Kanals nicht überall hin übertragen werden muß, findet im *ARQ-Splitter* eine Umwandlung in eine Kurzadresse (LLI =**L***ogical* **L***ink* **I***tentifier*) statt. Nach unten werden die Daten zur Verschickung über den *Traffic Channel* zur unteren Teilschicht durch einen Multiplexer (*TCH Mux/Demux*) zusammengefaßt. Die Auswahl der *ARQ Instance* wird gemäß der Prioritäten der verschiedenen Zellen zum Zeitpunkt des Versendens gesteuert. Es erfolgt keine weitere Zwischenspeicherung der Protokolldateneinheiten in der unteren Teilschicht des LLC, was unweigerlich zu zusätzlichen Verzögerungen und einem weniger dynamischen Verhalten des Protokolls führen würde. Der Auf- und Abbau von ARQ-Instanzen wird durch die *Link Control* sichergestellt. Sie macht Aussagen darüber, welche ARQ-Instanzen welchen virtuellen Kanal zugeordnet werden und wann diese auf- und abgebaut werden müssen. Zur Erfüllung dieser Aufgaben braucht sie Information der Netzschicht, die oberhalb der LLC-Schicht liegt. Für MBS-interne Signalisierung ist eine spezielle ARQ-Instanz abgestellt, deren Dienst an einem speziellen SAP der Netzschicht zur Verfügung gestellt wird.

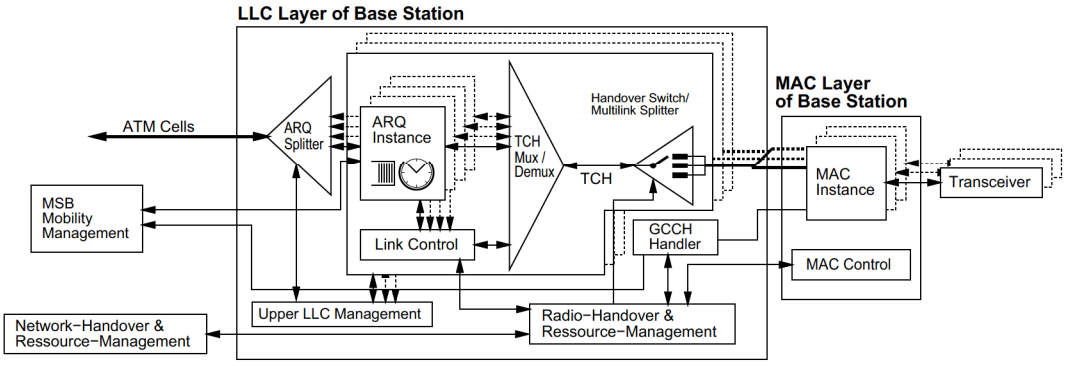


Abbildung 4.3: Aufbau der LLC-Schicht im *Base-Station-Controller*

Die Struktur der LLC-Schicht im *Base-Station-Controller* zeigt Abbildung 4.3. Im oberen Teil der LLC-Schicht befindet sich zu jeder registrierten Mobilstation ein Äquivalent zu den Instanzen in der LLC-Schicht der Mobilstation. D.h. für jede Mobilstation, die Daten

mit dem *Base-Station-Controller* austauscht, gibt es einen *Traffic Channel (TCH)*, an den die untere LLC-Teilschicht die ankommenden Daten weitergibt. Ebenso gibt es für jeden dieser für eine Mobilstation zuständigen Blöcke eine *Link Control* die die einzelnen ARQ-Instanzen verwaltet. Nur der *ARQ-Splitter* muß übergreifend über alle Blöcke arbeiten und die ATM-Zellen nach Mobilstationen und virtuellen Kanälen auf die ARQ-Instanzen aufteilen. Die obere LLC-Schicht wird in der *Base Station* von dem *Upper LLC Management* verwaltet. Sie initiiert den Auf- und Abbau der einzelnen Blöcke.

Äquivalent zu jedem *TCH* gibt es einen *Handover Switch/Multilink Splitter*. Der *GCCH Handler* und das *Radio-Handover & Resource-Management* sind jeweils wie in der Mobilstation nur einmal vorhanden. Deutliche Unterschiede bestehen jedoch in der Schnittstelle zwischen MAC-Schicht und LLC-Schicht. Die Adressierung der Verbindungsendpunkte erfolgt hier hierarchisch in zwei Schritten. Für Details wird auf [6] verwiesen.

KAPITEL 5

Sicherungsprotokolle

Die Aufgabe von Sicherungsprotokollen ist die Beherrschung von Übertragungsfehlern. Sie wurden entwickelt, um folgende Vorgaben zu erfüllen [40]:

- Ein einheitliches Format zur Übertragung von Steuer- und Nachrichtendaten
- Übertragungsfehler werden bis auf eine sehr kleine Restfehlerrate (10^{-6}–10^{-9}) erkannt.
- Datenblöcke gehen weder verloren, noch werden sie mehrfach gesendet oder vertauscht.
- Unterstützung von üblichen Strukturen wie Punkt-zu-Punkt, Mehrpunkt und Ring durch ein einziges Protokoll

Man unterscheidet zwei Klassen von Verfahren, um die Fehlersicherheit eines Übertragungssystems zu erhöhen:

FEC (Forward Error Correction)
: Beim FEC-Verfahren muß der Dekodierer im Empfänger in der Lage sein, nicht nur Fehler zu erkennen, sondern auch die Fehlerstellen exakt lokalisieren zu können, so daß die fehlerhaften Bits korrigiert werden können. Hierzu werden die empfangenen mit den berechneten Prüfbits verglichen.

ARQ (Automatic Repeat ReQuest)
: Beim ARQ-Verfahren muß der Dekodierer im Empfänger lediglich Fehler erkennen können. Beim Auftreten eines Fehlers wird über den Rückkanal automatisch eine Neuanforderung des fehlerhaften Rahmens versendet. Daraufhin wird dieser Rahmen erneut übertragen. Im Gegensatz zum FEC müssen beim ARQ empfangene Rahmen quittiert werden, es entsteht zusätzliche Last auf dem Rückkanal.

Aus der Kodierungstheorie ist bekannt, daß die Anzahl der korrigierbaren Fehler, bei gleicher Anzahl Prüfbits, wesentlich kleiner ist als die Anzahl erkennbarer Fehler. Andererseits belastet das ARQ–Verfahren den Rückkanal mit Quittungen. Durch die Verwendung eines hybriden Verfahrens wird der Kodierungsaufwand für das FEC in Grenzen gehalten und die Anzahl der Neuanforderungen durch das ARQ verringert. Ein solches hybrides Verfahren soll im MBS zur Anwendung kommen.

Ein Mechanismus zum Erkennen von fehlerhaft empfangenen ARQ-Rahmen wird vorausgesetzt. Für eine mögliche Kombination des ARQ-Protokolls mit einem FEC-Verfahren innerhalb der Physikalischen Schicht wird auf die Literatur verwiesen [33].

5.1 Konventionelle ARQ-Protokolle

Im folgenden werden folgende konventionelle ARQ-Protokolle vorgestellt:

1. Stop-and-Wait ARQ

2. Go Back n (continous) ARQ
3. Selective Repeat (SR) ARQ

Dabei stehen zwei Bewertungskriterien im Vordergrund:

Korrektheit
Wie wird bewerkstelligt, daß alle Rahmen in der richtigen Reihenfolge und ohne Fehler und Wiederholungen im Empfänger an die oberen Schichten weitergeleitet werden?

Effizienz
Wieviel Kanalkapazität wird zusätzlich verbraucht bzw. verschwendet?

Es wird vereinfachend angenommen, daß ein Fehlererkennungsalgorithmus vorhanden ist und daß dieser alle auftretenden Fehler erkennt. Die Korrektheit kann nur unter dieser Annahme gezeigt werden. Nicht erkannte fehlerhafte Rahmen können zu fehlerhaften Daten führen. Weiterhin wird vorausgesetzt, daß Rahmen in derselben Reihenfolge empfangen werden, wie sie versendet wurden. Diese Voraussetzung ist im MBS aufgrund des synchronen Kanals erfüllt.

5.1.1 Stop-and-Wait ARQ

Dieses einfachste aller ARQ-Protokolle basiert auf der Idee, daß der korrekte Empfang eines jeden Rahmens sichergestellt wird, bevor der nächste Rahmen übertragen wird.

Um die Eindeutigkeit der Rahmen und Quittungen bei Übertragungsfehlern und -überschneidungen aufrecht zu erhalten, müssen sowohl Rahmen wie auch Quittungen numeriert werden. Diese Nummern heißen Sequenznummer (*sequence number (SN)*) für Rahmen vom Sender und Anforderungsnummer (*request number (RN)*) für Quittungen vom Empfänger.

Der Sender trägt vor dem Übertragen eines Rahmens die aktuelle Sequenznummer ein, startet einen Timer und wartet auf eine Quittung. Läuft der Timer ab, bevor eine Quittung empfangen wurde, wird der bereits gesendete Rahmen wiederholt und der Timer erneut gestartet.

Der Empfänger akzeptiert Rahmen nur, wenn die Sequenznummer des empfangenen Rahmens gleich der Anforderungsnummer ist ($SN = RN$). In diesem Fall wird eine positive, anderenfalls eine negative Quittung verschickt. Quittiert werden Rahmen, indem die Sequenznummer des als nächstes erwarteten Rahmens in die Anforderungsnummer eintragen wird ($RN = SN + 1$). Quittungen können sofort nach Empfang eines Rahmens, verzögert um eine beliebige, aber endliche Zeit oder zusammen mit Nutzdaten in die Gegenrichtung übertragen werden. Das Zusammenfügen von Nutzdaten und Quittungen in einem Rahmen wird *Huckepackverfahren (engl.: piggyback)* genannt. Dabei muß sichergestellt werden, daß die Quittung nach endlicher Zeit übertragen wird. Beim Empfang einer negativen Quittung wiederholt der Sender den Rahmen RN, bei einer positiven Quittung wird nun der neue Rahmen RN gesendet. Es wird also unabhängig von der Quittungsart (positiv oder negativ) mit dem Rahmen RN fortgefahren.

Nach [2] läßt sich mit Hilfe von vollständiger Induktion zeigen, daß der Empfänger nur Rahmen in der richtigen Reihenfolge weitergibt. Voraussetzung ist, daß Sender und Empfänger richtig initialisiert worden sind, z.B. $SN = RN = 0$. Weiterhin läßt sich zeigen, daß bei endlichem Timer im Sender und endlicher Verzögerung im Empfänger es nie zu einem

Deadlock kommt. Zur Unterscheidung der Rahmen und Quittungen reichen jeweils zwei Zustände, so daß SN und RN modulo zwei kodiert werden können. Somit ist auch eine Unterscheidung in positive und negative Quittungen unnötig [9].

Stop-and-Wait ARQ ist nicht besonders effektiv, da viel Kanalkapazität durch das Warten auf Quittungen verschenkt wird. Außerdem muß *jeder* Rahmen einzeln quittiert werden, was zu einer erhöhten Last auf dem Rückkanal führt.

5.1.2 Go Back n (continous) ARQ

Das Go Back n oder continous ARQ Protokoll ist das am weitesten verbreitete ARQ Protokoll und findet in folgenden Standards Anwendung [2]:

- HDLC (**H***igh level* **D***ata* **L***ink* **C***ontrol* procedure) von der ISO
- ADCCP (**A***dvanced* **D***ata* **C***ommunication* **C***ontrol* **P***rocedure*) von der ANSI
- SDLC (**S***ynchronous* **D***ata* **L***ink* **C***ontrol* procedure) von IBM
- LAP-B (**L***ink* **A***ccess* **P***rocedure-***B***alanced*) von der CCITT

HDLC und ADCCP sind in bezug auf das ARQ Protokoll identisch, während SDLC und LAP-B eine eingeschränkte Untermenge der Funktionalität von HDLC darstellen.

Im Gegensatz zum Stop-and-Wait ARQ können beim Go Back n ARQ bis zu n Rahmen nacheinander versendet werden, ohne zwischenzeitlich eine Quittung zu erhalten. Man spricht in diesem Fall von einem Fenstermechanismus mit einer Fenstergröße von n. Zur Kontrolle des Fensters sind nun zwei zusätzliche Sequenznummern nötig. SN_{min} bezeichnet diejenige Sequenznummer, die zuletzt durch den Empfänger angefordert wurde, SN_{max} die Nummer desjenigen Rahmens, der als nächstes versendet werden soll, d.h. $SN_{min} \leq SN_{max} \leq SN_{min} + n$. Nach dem Versenden des Rahmens mit der Nummer $SN_{min}+n-1$ muß auch das Go Back n ARQ auf eine Quittung warten, da jetzt das Fenster geschlossen ist. Nach einem beliebigen, endlichen Timeout werden die Rahmen von SN_{min} bis $SN_{max} - 1$ wiederholt. Beim Empfang einer Quittung mit $SN_{min} \leq RN \leq SN_{max}$ werden alle Rahmen von SN_{min} bis $RN - 1$ quittiert und das Sendefenster so verschoben, daß $SN_{min} = RN$ ist. Beim Empfang einer positiven Quittung wird mit dem Rahmen SN_{max}, bei einer negativen Quittung mit $SN_{min} = RN$ fortgefahren.

Der Empfänger verhält sich genauso wie beim Stop-and-Wait ARQ (Kapitel 5.1.1). Auch der Beweis der Korrektheit verläuft entsprechend. Für Einzelheiten wird jedoch auf die Literatur verwiesen [2].

Nach den Voraussetzungen gilt für die Sequenznummer SN:

$$SN_{min} \leq SN \leq SN_{min} + n - 1 \tag{5.1}$$

Für die Anforderungsnummer RN gilt:

$$SN_{min} \leq RN \leq SN_{min} + n \tag{5.2}$$

Mit (5.1) und (5.2) folgt, daß für eine eindeutige Kodierung der Sequenznummern modulo m gelten muß:

$$m > n \tag{5.3}$$

Bei vorgegebenen Modulus m ist damit die maximale Fenstergröße:

$$n_{max} = m - 1 \tag{5.4}$$

Die Verfahrensweise nach dem Empfang einer Quittung beeinflußt entscheidend die Effektivität des Protokolls. Dabei muß man zwischen Durchsatz und Verzögerung unterscheiden. Wird nach dem Empfang einer Quittung beim Rahmen mit der Nummer SN_{max} fortgefahren, so wird auf fehlende Rahmen erst dann reagiert, wenn das Sendefenster geschlossen ist. Außerdem müssen im Fehlerfall mindestens n Rahmen wiederholt werden. Der Durchsatz sinkt mit p als Rahmenfehlerwahrscheinlichkeit auf:

$$D \leq \frac{1-p}{1+n \cdot p} \tag{5.5}$$

Fährt man hingegen beim Rahmen mit der Nummer SN_{min} fort, so wird einerseits die Verzögerung minimiert, da verlorengegangene Rahmen schnell wiederholt werden, aber andererseits der Durchsatz verringert, da Rahmen wiederholt werden, die sich möglicherweise noch in der Übertragung befinden. Abhilfe schafft hier die Einführung eines *Ignore Timer*, der die Übertragungszeit überwacht (Kapitel 5.2.2).

Ist die Signallaufzeit sehr groß, wie etwa bei Satellitenübertragungen, so muß n entsprechend groß gewählt werden, um Wartezeiten bei geschlossenem Sendefenster weitgehend zu vermeiden. Dadurch wird ebenso wie durch schlechte Übertragungsqualität der Durchsatz verringert (Gleichung (5.5)).

5.1.3 Selective Repeat (SR) ARQ

Die zentrale Idee des SR ARQ ist, daß der Empfänger Rahmen akzeptiert, die er nicht als nächstes erwartet. Diese Rahmen werden zwischengespeichert und nur die fehlenden Rahmen beim Sender angefordert. Sobald die fehlenden Rahmen korrekt empfangen wurden, werden alle zwischengespeicherten Rahmen an die nächsthöhere Schicht weitergegeben. Gelingt es, nur die fehlerhaften Rahmen zu wiederholen, so erreicht man den maximal möglichen Durchsatz von:

$$D = 1 - p \tag{5.6}$$

Durch Einsatz eines *Ignore Timer* wird dieses möglich (Kapitel 5.2.2).

Beim SR ARQ entsteht im Empfänger ebenfalls ein Fenster der Größe n. Im Gegensatz zum Sender (Kapitel 5.1.2) sind im Empfänger keine weiteren Sequenznummern zur Kennzeichnung des Empfangsfensters notwendig. RN bezeichnet jetzt den niedrigsten noch nicht korrekt empfangenen Rahmen. Der Empfänger akzeptiert also alle Rahmen für deren Sequenznummer gilt:

$$RN \leq SN \leq RN + n - 1 \tag{5.7}$$

Unter Berücksichtigung der Gleichungen (5.1) und (5.2) folgt mit Gleichung (5.7) der Bereich der gültigen Sequenznummern:

$$RN - n \leq SN \leq RN + n - 1 \tag{5.8}$$

Aus Gleichung (5.8) folgt, daß für eine eindeutige Kodierung der Sequenznummern modulo m gelten muß:

$$m > 2 \cdot n - 1 \tag{5.9}$$

Bei vorgegebenen Modulus m ist damit die maximale Fenstergröße für SR-ARQ:

$$n_{max} = \frac{m}{2} \tag{5.10}$$

Unter Beachtung von (5.10) ergibt sich die Korrektheit des SR ARQ genauso wie beim Go Back n ARQ (Kapitel 5.1.2).

Um fehlerhafte Rahmen erneut anzufordern, gibt es mehrere Möglichkeiten:

- Im fehlerfreien Fall werden positive Quittungen verwendet. Fehlen ein oder mehrere Rahmen, so werden diese durch eine negative REJ (**REJ***ect*) Quittung angefordert, wobei RN die Nummer angibt, ab der alle Rahmen wiederholt werden sollen. SR ARQ verhält sich in diesem Fall ähnlich wie Go Back n ARQ und bietet kaum Vorteile.
- Bei fehlerfreiem Empfang werden positive Quittungen versendet, woraufhin der Sender sein Fenster verschiebt. Fehlende Rahmen werden mit einer SREJ (**S***elective* **REJ***ect*) Quittung angefordert, wobei RN die Nummer des fehlenden Rahmens angibt. Der Sender wiederholt nur diesen einen Rahmen. Fehlen mehrere Rahmen, so muß jeder einzeln angefordert werden.
- In einer Quittung wird ein Feld gesendet, in dem angegeben ist, welche Rahmen erfolgreich und welche fehlerhaft übertragen wurden. Der Sender kann nun genau die fehlenden Rahmen wiederholen. Je nach Fenstergröße erzeugen diese Quittungen aber einen erheblichen Overhead.

Die einzelnen Quittungsarten schließen sich nicht gegenseitig aus und können so kombiniert werden, daß sich den Anforderungen entsprechend eine hohe Effizienz ergibt.

5.2 Das Adaptive Selective Repeat (ASR) ARQ-Protokoll

Das ASR ARQ-Protokoll [10] basiert auf dem SR ARQ. Die Eigenschaften des ASR ARQ lassen sich in drei Bereiche unterteilen:

1. Elemente aus dem Standard ISO 8802-2
2. Behandlung zeitkritischer ATM-Zellen
3. Vorteile paralleler ARQ Instanzen

Im folgenden werden die einzelnen Eigenschaften detailiert dargestellt.

5.2.1 Elemente des ARQ im Standard ISO 8802-2

Als Basisprotokoll dient das in ISO 8802-2 (IEEE 802.2) spezifizierte ARQ-Protokoll. Hierbei handelt es sich um ein modifiziertes Go Back n ARQ mit folgenden Elementen [17]:

Acknowledgment Timer
: Der Acknowledgment Timer überwacht die Zeit, in der nach dem Versenden von einem oder mehreren Rahmen eine Quittung erwartet wird. Bleibt eine Quittung aus, so beginnt der Sender zu *Pollen*.

P-Bit Timer
: Der P-Bit Timer überwacht die Zeit, in der eine Antwort auf ein *Pollen* erwartet wird. Bleibt eine Antwort aus, so wird erneut gepollt.

Reject Timer
: Der Reject Timer überwacht die Zeit im Empfänger, in der eine Antwort auf eine Neuanforderung erwartet wird. Bleibt eine Antwort aus, so wird die Neuanforderung wiederholt.

N2
: N2 gibt die maximale Anzahl der Wiederholungen als Folge des Ablaufens des Acknowledgment, P-Bit oder Reject Timers an. Beim Überschreiten von N2 wird ein Reset des virtuellen Kanals ausgelöst.

R*eceive* R*eady (RR)* P*rotocol* D*ata* U*nit (PDU)*
: Die RR PDU dient zum Versenden von positiven Quittungen, d.h. es werden alle Rahmen quittiert, für die gilt: $SN < RN$

REJ*ect (REJ) PDU*
: Die REJ PDU dient zum Versenden von negativen Quittungen, d.h. es wird wie beim Empfang einer RR PDU quittiert und zusätzlich alle Rahmen wiederholt, für die gilt: $SN \geq RN$

Weiterhin sieht der Standard Maßnahmen zur *flow control* vor. In der LLC Schicht des MBS ist kein *flow control* vorgesehen, deshalb werden diese Maßnahmen hier vernachlässigt.

Das ASR ARQ erbt alle oben aufgeführten Elemente. Aus Gründen der Stabilität des Protokolls ist lediglich der *Acknowledgment Timer* notwendig. Sowohl der *P-Bit*, wie auch der *Reject Timer* sind Maßnahmen, um die Verzögerung der Rahmen zu verringern. Die Effektivität der drei Timer wird anhand von Simulationen untersucht (Kapitel 8), deshalb wurden sie so implementiert, daß sie zu- und abschaltbar sind. Die *Timer Delays* aller drei Timer sind freie Parameter, die an die jeweiligen Bedingungen angepaßt werden müssen.

Das ASR ARQ ist ein modifiziertes SR ARQ und benutzt eine **S***elective* **REJ***ect (SREJ)* PDU. Mit dieser wird ein einzelner Rahmen erneut angefordert. Wie bei der REJ PDU werden alle Rahmen mit $SN < RN$ quittiert, aber nur der Rahmen mit der Nummer $SN = RN$ erneut versendet. RR PDUs werden beim ASR ARQ ähnlich interpretiert, wie REJ PDUs im ISO 8802-2. Diese Vorgehensweise ist nur im Zusammenhang mit dem *Ignore Timer* sinnvoll. In diesem Fall werden keine REJ PDUs verwendet. Warten im Empfangspuffer weniger Rahmen als in der Empfangsreihenfolge fehlen (Abbildung 5.1, Fall A), so wird eine RR PDU versendet, statt alle fehlenden Rahmen einzeln mit Hilfe von SREJ PDUs anzufordern. Ansonsten werden fehlende Rahmen durch SREJ PDUs einzeln angefordert (Abbildung 5.1, Fall B).

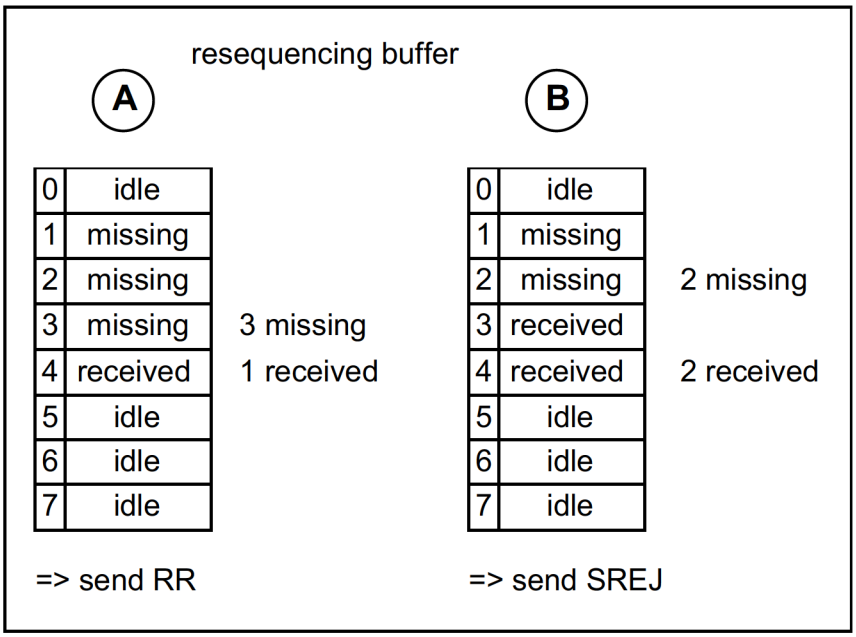


Abbildung 5.1: Beispiel für das Versenden von RR und SREJ

5.2.2 Der *Ignore Timer*

Ein Problem von ARQ-Protokollen ist, daß der Sender beim Empfang einer Quittung nicht weiß, was mit denjenigen Rahmen passiert ist, die während der Übertragungszeit der Quittung versendet wurden. Gerade beim Empfang einer Neuanforderung durch SREJ oder REJ PDUs kann es sein, daß diese angeforderten Rahmen gerade wiederholt wurden, aber noch nicht im Empfänger angekommen waren, als die Quittung generiert wurde. Wenn die Übertragungszeit aber bekannt wäre, könnte man erkennen, welche Rahmen noch nicht beim Empfänger angekommen sind. Diese Übertragungszeit ist im MBS deterministisch. Zum einen werden PDUs vom ARQ-Protokoll unmittelbar vor dem Senden erzeugt, zum anderen ist die Signallaufzeit im MBS aufgrund begrenzter Zellgrößen beschränkt. Die gesamte Übertragungszeit schwankt je nach Versatz von Up- und Downlink zwischen ein und zwei Slots.

Diese Zeit macht sich der Ignore Timer zunutze, sie wird im folgenden *Ignore Delay* genannt. Vor dem Versenden einer PDU wird der Ignore Timer gestartet. Während der Ignore Timer läuft, wird jede Neuanforderung dieses Rahmens ignoriert. Hierdurch werden unnötige Wiederholungen vermieden.

Empfängt der Sender eine positive Quittung mit $RN < SN$ und ist der Ignore Timer des Rahmens SN bereits abgelaufen, so weiß der Sender, daß dieser Rahmen nicht korrekt empfangen wurde und kann ihn wiederholen. Diese Vorgehensweise stellt einen Wissensvorsprung gegenüber dem SR ARQ dar, da beim ASR ARQ der Sender bereits Wiederholungen veranlassen kann, *bevor* Neuanforderungen empfangen werden. Der Vorteil liegt darin, daß der Empfänger erst dann Kenntnis über den Verlust von Rahmen erhält, wenn er einen nachfolgenden Rahmen korrekt empfängt. Vorher führt eine positive Quittung beim SR ARQ nicht zur Wiederholung der verlorengegangenen Rahmen.

Setzt der Empfänger zur Neuanforderung fehlender Rahmen ausschließlich SREJ PDUs ein, so erreicht man mit Hilfe des Ignore Timers den optimalen Durchsatz von $D = 1 - p$.

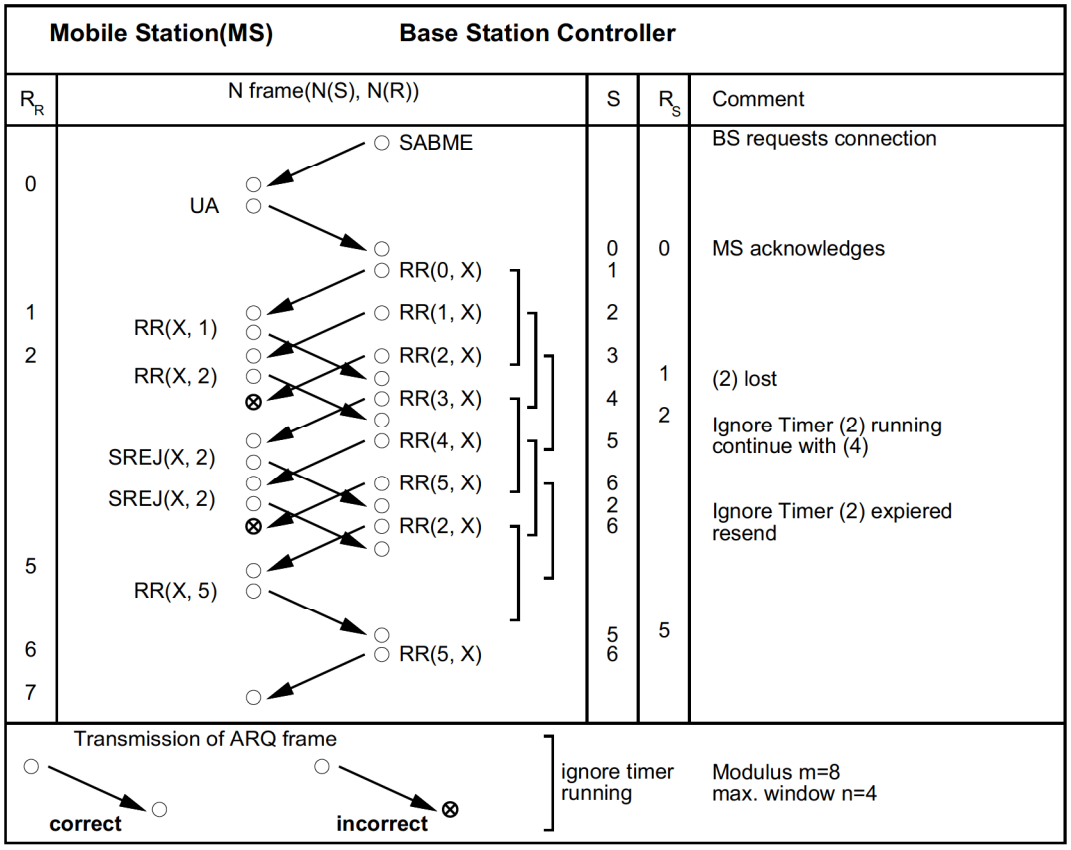


Abbildung 5.2: Vermeidung unnötiger Wiederholungen durch den *Ignore Timer*

Zusammenfassend läßt sich sagen, daß der Ignore Timer folgende zwei Vorteile hat:

1. Vermeidung unnötiger Wiederholungen, Erreichen des optimalen Durchsatzes
2. Schnellere Erkenntnis über verlorengegangene Rahmen, Verminderung der Zellverzögerung

Die Arbeitsweise des Ignore Timer soll nun an zwei Beispielen demonstriert werden. In Abbildung 5.2 wird ein Kanal mit unidirektionalem Verkehr betrachtet. Alle Zellen sind zeitkritisch, so daß eine schnelle Quittierung durch den Empfänger notwendig ist, d.h. nachdem der Empfänger einen Rahmen empfangen hat, schickt er sofort eine Quittung.

Mit einer *SABME (**S**et **A**synchronous **B**alanced **M**ode **E**xtended)* PDU fordert die Basisstation (BS) eine Verbindung an. Diese wird durch die Mobilstation (MS) mittels *UA (**U**nnumbered **A**cknowledgment)* PDU bestätigt. Die BS sendet nun hintereinander die Rahmen 0–2, wobei letzterer verloren geht. Durch den Empfang von *RR(X, 1)* wird der Rahmen 0 bestätigt und die BS verschiebt ihr Sendefenster nach 1. Mit dem Empfang des Rahmens 3 stellt die MS fest, daß Rahmen 2 fehlt. Sie verschickt eine *SREJ(X, 2)* PDU, um den fehlenden Rahmen anzufordern. Beim Empfang des Rahmens 4 verfährt sie ebenso. Da der *Ignore Timer* des Rahmens 2 bereits abgelaufen ist, wird dieser aufgrund der Neuanforderung erneut verschickt und der Timer erneut gestartet. Die zweite Neuanforderung wird ignoriert. Nach dem Erhalt von *RR(X, 5)* kann die BS ihr Fenster auf 5 verschieben und den angeforderten Rahmen übertragen. Danach fährt das Protokoll normal fort.

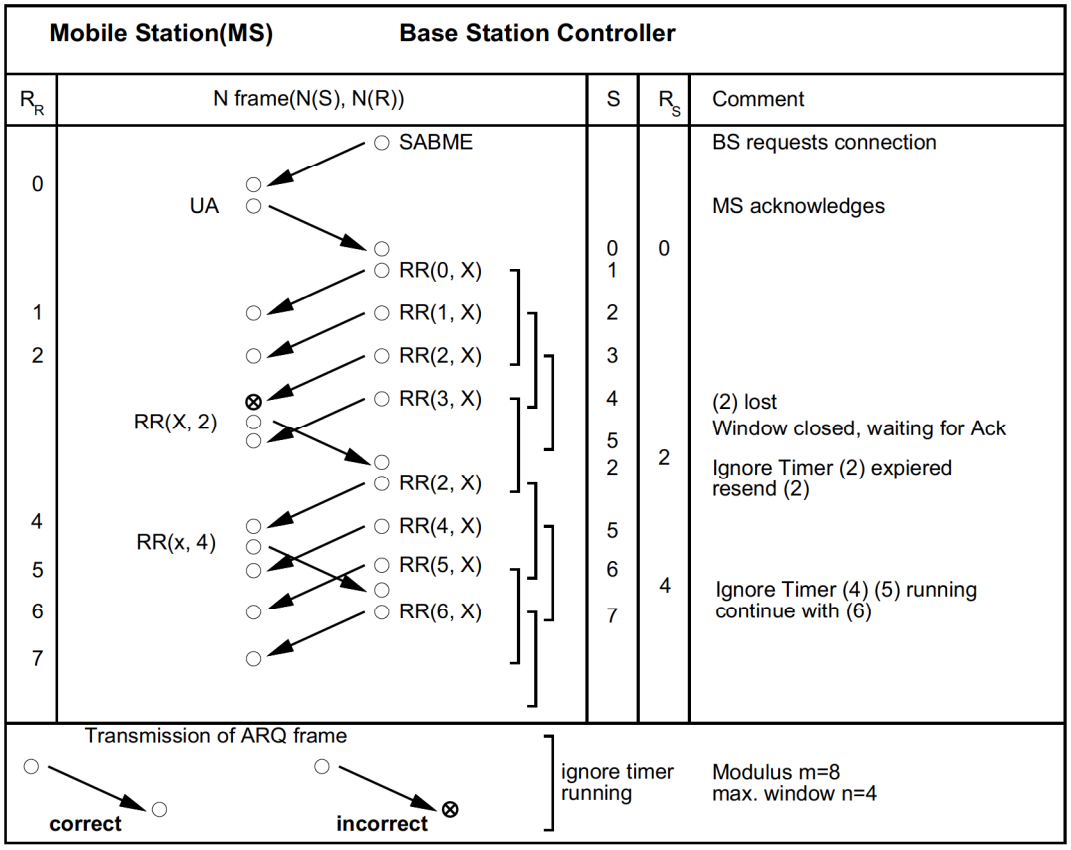


Abbildung 5.3: Schnelle Reaktion auf Verluste durch den *Ignore Timer*

In Abbildung 5.3 wird ebenfalls ein Kanal mit unidirektionalem Verkehr betrachtet. Auch hier handelt es sich um einen zeitkritischen Dienst. Auf dem Rückkanal steht aber nur eingeschränkte Kapazität zur Verfügung, so daß der Empfänger nur alle paar Zeitschlitze eine Quittung verschicken kann.

Mit einer *SABME* PDU fordert die BS eine Verbindung an. Diese wird durch die MS mittels *UA* PDU bestätigt. Die BS sendet nun hintereinander die Rahmen 0–2, wobei letzterer verloren geht. Nach dem Verschicken von Rahmen 3 ist das Sendefenster geschlossen und BS wartet auf eine Quittung. Zu dem Zeitpunkt, wo die MS eine Quittung versenden kann, weiß sie noch nichts vom Verlust von Rahmen 2. Folgerichtig schickt sie ein *RR(X, 2)*. In der BS ist der Ignore Timer bereits abgelaufen, so daß sie beim Eintreffen der Quittung bereits weiß, daß der Rahmen 2 verloren gegangen ist. Die Rahmen 0 und 1 werden quittiert, der Rahmen 2 wird wiederholt, während der Rahmen 3 nicht erneut zum Versenden markiert wird, da hier der Ignore Timer noch läuft. Die BS nimmt richtigerweise an, daß der noch in der Übertragung befindliche Rahmen korrekt empfangen wird. Durch das Überspringen von Rahmen 3 werden als nächstes die Rahmen 4 und 5 verschickt. Nach Empfang der Quittung *RR(X, 4)* kann normal fortgefahren werden.

Ohne den Einsatz des Ignore Timer wäre nach dem Erhalt der *RR(X, 2)* Quittung der Rahmen 2 nicht wiederholt worden. Dieser hätte dann explizit durch eine negative Quittung (SREJ, REJ) angefordert werden müssen. Die begrenzte Kapazität auf dem Rückkanal wäre durch eine weitere Quittung belastet worden. Außerdem wäre aufgrund der zusätzlich notwendigen Quittung die Wahrscheinlichkeit gestiegen, daß sich das Sendefenster

erneut schließt und der Sender untätig auf eine Quittung warten müßte.

5.2.3 Behandlung zeitkritischer ATM-Zellen

ATM sieht für zeitkritische Dienste eine maximale Verzögerung als QoS Parameter vor. Ist das Einhalten dieses Parameters wichtiger als das Einhalten der maximalen Zellverlustrate, so sind im ASR ARQ Mechanismen vorgesehen, um im Falle von Hochlast durch gezieltes Verwerfen von Zellen die maximale Verzögerung einzuhalten. Ein solcher Dienst könnte z.B. eine Videokonferenzverbindung sein, bei der der Verlust einzelner Zellen unkritisch ist, aber aufgrund von Synchronisations- und Echoeffekten eine maximale Verzögerung eingehalten werden muß.

Es sind drei Maßnahmen zum Verwerfen von Zellen vorgesehen:

1. Es werden Zellen vor der ersten Übertragung verworfen, wenn sie die maximale Verzögerung überschritten haben. Diese Zellen werden nicht in das Sendefenster übernommen bzw. aus dem Sendefenster entfernt und durch neuere Zellen ersetzt. Nur in diesem Fall, vor der ersten Übertragung, muß der Empfänger nicht über das Verwerfen informiert werden.
2. Es wird im Sender ein *Delay Timer* eingesetzt, um die Restlebenszeit der Zellen im Sendefenster zu kontrollieren. Läuft die Zeit ab, so wird die Zelle auf jeden Fall verworfen. Befand sie sich bereits im Sendeprozeß, so muß der Empfänger über das Verwerfen dieser Zelle informiert werden. Diese Information wird mit Hilfe einer *Delay* PDU übertragen (Kapitel 5.2.4).
3. Im Empfänger wird die Restlebenszeit derjenigen Zellen überwacht, die zwischengespeichert werden müssen, weil auf fehlende Rahmen in der Sequenz gewartet wird. Läuft die Zeit ab, so wird der Warteprozeß abgebrochen und die bereits empfangenen Zellen an die obere Schicht weitergeleitet. Für diese Maßnahme muß die Restlebenszeit mit übertragen werden, es entsteht also ein erhöhter Overhead. Diese Maßnahme ist nur in Verbindung mit Maßnahme 2 sinnvoll.

Die drei Maßnahmen bauen aufeinander auf, d.h. daß für Maßnahme 2, Maßnahme 1 notwendig ist. Nur durch alle drei Maßnahmen zusammen erreicht man die strikte Einhaltung der maximalen Verzögerung, während sonst einzelne Zellen diese überschreiten können. Die Komplexität des Protokolls erhöht sich durch diese Maßnahmen erheblich. Die Korrektheit ist nicht mehr ohne weiteres gegeben, es müssen Ausnahmefälle beachtet werden, da es sonst unter Umständen zu einem Deadlock kommen kann.

5.2.4 Die *Delay PDU*

Die Delay PDU wird dazu eingesetzt, den Empfänger über das Verwerfen einer Zelle zu informieren. Sie wird nur dann versendet, wenn der Empfänger eine verworfene Zelle angefordert hat (mit RR oder SREJ). In diesem Fall wird statt einer Quittung in die Gegenrichtung die Delay PDU mit $RN = SN$ übertragen, wobei SN die höchste Nummer aller verworfenen Zellen ist. Dabei wird vorausgesetzt, daß der Sender die Zellen in der richtigen Reihenfolge verwirft, d.h. daß es keine gültige (nicht verworfene) Zelle mit niedrigerer Sequenznummer vorhanden ist. Aus diesem Grund hat die Nummer der neuesten verworfenen Zelle den größten Informationswert und wird unter Umständen anstatt der angeforderten verschickt.

Erhält der Empfänger eine Delay PDU, so bricht er das Warten auf die Zellen ab, für deren Nummer gilt: $N \leq RN$. Er verschiebt daraufhin sein Fenster und quittiert dieses entsprechend. Bei bidirektionalem Verkehr ist darauf zu achten, daß die Delay PDU mit Quittungen konkurriert. Es muß sichergestellt werden, daß sowohl Quittungen, wie auch Delay PDUs in endlicher Zeit übertragen werden.

Hat die Delay PDU Priorität gegenüber den Quittungen, so kann es zu der in Abbildung 5.4 dargestellten Verklemmung kommen:

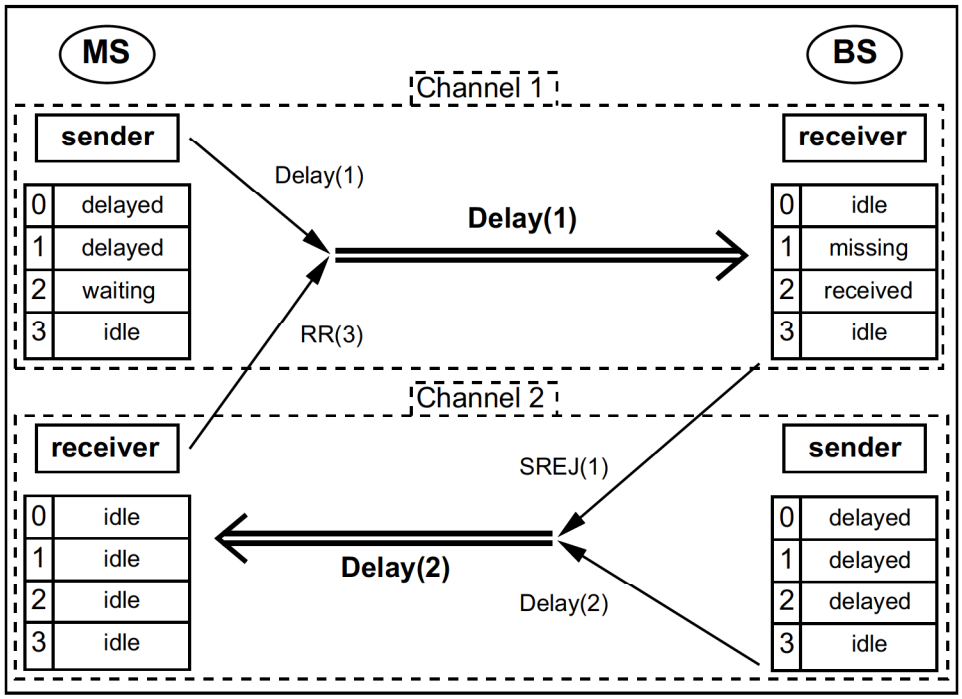


Abbildung 5.4: Verklemmung durch Priorisierung von Delay PDUs

Im Kanal 1 wartet der Empfänger in der BS auf Rahmen (1) und möchte deshalb ein SREJ(1) versenden. Da aber der Sender in der BS (Kanal 2) ein Delay(2) mit höherer Priorität hat, wird dieses gesendet. Der Sender in der MS (Kanal 1) kann aber erst dann fortfahren, wenn er eine Quittung erhält. Ebenso ergeht es dem Sender in der BS (Kanal 2), der ebenfalls erst sein Fenster verschieben kann, wenn er eine Quittung erhält. Übertragen die Sender permanent ihre Delay PDUs, so kommt es zur Verklemmung, die sich erst dann auflöst, wenn aufgrund eines Timeouts ein Reset ausgelöst wird.

Hat als anderes Extrem die Quittung Priorität gegenüber der Delay PDU, so kommt es ebenfalls zur Verklemmung, wie Abbildung 5.5 zeigt:

In diesem Fall fordert im Kanal 1 der Empfänger in der BS mittels SREJ(1) den Rahmen (1) an. Dieser ist verworfen worden, doch da der Empfänger in der MS (Kanal 2) ebenfalls Rahmen mittels RR(1) anfordert, können die Benachrichtigungen über verworfene Zellen (Delay PDUs) nicht übertragen werden. In dieser Situation können weder Sender noch Empfänger ihre Fenster verschieben. Verharren die Empfänger in ihrem Verhalten so kommt es zum Deadlock, der erst durch einen Reset aufgelöst werden kann.

Es muß also gewährleistet sein, daß sich die Prioritäten dynamisch so ändern, daß sowohl Quittungen wie auch Delay PDUs in endlicher Zeit übertragen werden. Im einfachsten Fall

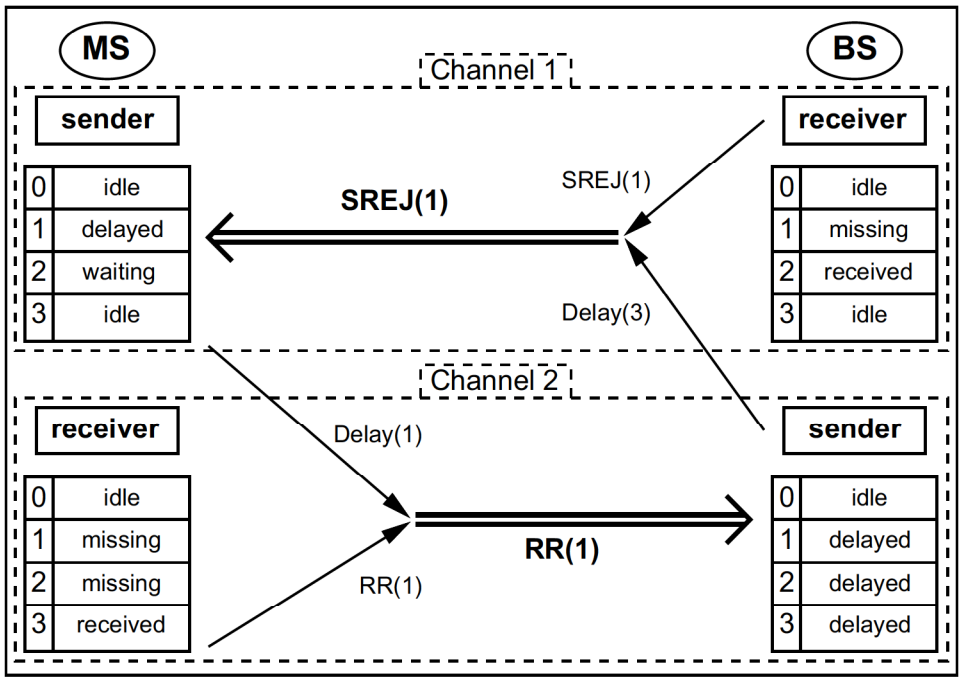


Abbildung 5.5: Verklemmung durch Priorisierung von Quittungen

kann zwischen Quittung und Delay PDU hin- und hergeschaltet werden (engl.: *toggle*), d.h. nach dem Versenden einer Quittung wird bei Bedarf als nächstes eine Delay PDU verschickt und umgekehrt.

Weiterhin muß beachtet werden, daß im Sender entweder der Platz im Sendefenster, der durch verworfene Zellen belegt wird, freigegeben werden kann, oder daß nach dem Verwerfen der Acknowledgment Timer läuft. Sonst kann es passieren, daß verworfene Zellen das Sendefenster komplett belegen, der Empfänger keine Zellen erwartet und der Sender nicht pollt, da kein Timer ablaufen kann. Es kommt zum Deadlock, der auch nicht von einem Reset des virtuellen Kanals beendet wird.

Bricht der Empfänger das Warten auf fehlende Zellen ab, so verschiebt er sein Fenster. Werden danach die fehlenden Zelle empfangen, so müssen diese komplett verworfen werden. Sie sind daran zu erkennen, daß ihre Sequenznummer außerhalb des Empfangsfensters liegen (auch aus diesem Grund darf die Fenstergröße maximal den halben Modulus betragen.).

Alle drei Maßnahmen zur Behandlung von zeitkritischen ATM-Zellen sind Gegenstand der Simulationen (Kapitel 8).

5.2.5 Parallele ARQ-Instanzen

In der oberen LLC-Schicht des MBS werden parallele ARQ-Instanzen aufgebaut, um angemessen auf die unterschiedlichen QoS Parameter der VCs reagieren zu können. Im Gegensatz zum ATM, das Zellen nach dem FCFS (**F**irst **C**ome **F**irst **S**erve) Prinzip bedient, werden beim ASR ARQ zeitkritische Zellen bevorzugt. Hierzu dient ein Prioritätenalgorithmus, der die Dringlichkeit der einzelnen Zellen berechnet. Um die Vorteile von parallelen ARQ-Instanzen voll nutzen zu können, werden erweiterte PDU-Formate verwendet.

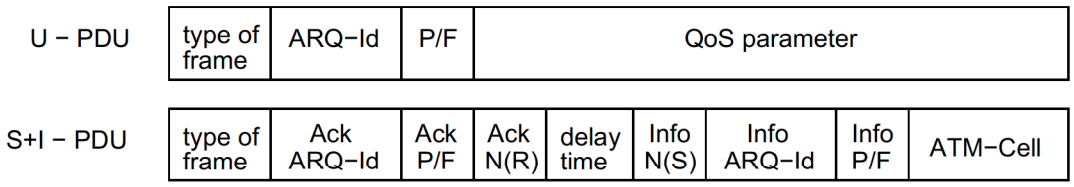


Abbildung 5.6: Rahmenstruktur des ASR ARQ

Wie Abbildung 5.6 zeigt, werden Quittungen und Nutzdaten in einem Rahmen versendet. Dieses Verfahren wird Huckepack (engl.: piggyback) Verfahren genannt. Nachteilig bei der ursprünglichen Form dieses Verfahrens ist, daß Quittungen auf Nutzdaten angewiesen sind. Die Priorität der Quittung ist an die Priorität der verbundenen Nutzdaten gekoppelt. Gerade bei asymmetrischen Verkehrsangebot ist aber die Quittungscharakteristik innerhalb eines Kanals anders als das Nutzdatenaufkommen. Quittungen werden in diesem Fall verspätet oder gar nicht geschickt, bzw. es muß ein Prioritätenalgorithmus entwickelt werden, der diese Problematik berücksichtigt.

Das ASR ARQ beschreitet einen anderen Weg, indem der Rahmenstruktur eine zweite *VC-Id* (Kanalkennzeichnung) hinzugefügt wird. Hierdurch ist es möglich, Nutzdaten von Kanal **A** mit einer Quittung von Kanal **B** zu kombinieren (Abbildung 5.7). Innerhalb einer Mobilstation können somit Quittungen und Nutzdaten beliebiger Kanäle zusammengefügt werden. Die Berechnung der dringlichsten Quittung kann nun von einem anderen Algorithmus unabhängig vom Nutzdatenaufkommen vorgenommen werden. Die Vorteile zeigen sich am deutlichsten bei stark asymmetrischen Verkehr innerhalb einer Mobilstation. Nachteilig ist die Erzeugung zusätzlichen Overheads durch die Felder *Ack ARQ-Id* und *Ack P/F* (Abbildung 5.6). Das zweite *P/F* Feld wurde eingefügt, um ein möglichst schnelles Pollen zu ermöglichen.

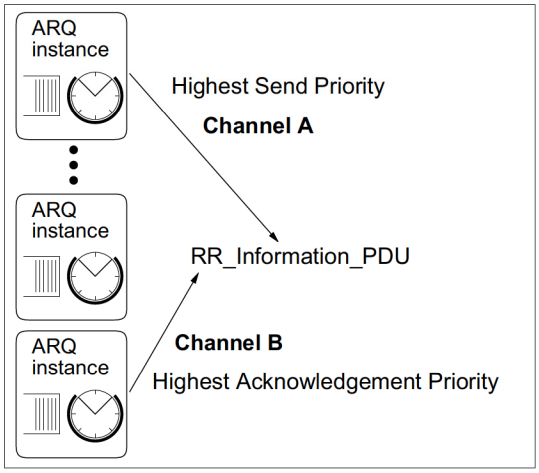


Abbildung 5.7: Generierung von Huckepack PDUs

5.2.6 Quittierungsverfahren

Im folgendem soll erläutert werden, wann der Empfänger eine Quittung verschicken will. Dazu gibt es grundsätzlich verschiedene Ansätze:

Zeitorientiert

Bei diesem Ansatz verschickt der Empfänger in äquidistanten Zeitintervallen eine Quittung. Hierdurch wird sichergestellt, daß der Sender nach konstanten Zeitabschnitten über den Zustand des Empfängers informiert wird. Andererseits ensteht gerade bei burstartigem Verkehr ein hoher Overhead, da viele Quittungen unnötig gewesen wären.

Bedarfsorientiert

Es wird ausschließlich auf Ereignisse reagiert, die eine Quittung notwendig machen. Es wird also genau dann quittiert, wenn der Empfänger die Notwendigkeit dazu erkennt. Der Overhead verringert sich gegenüber dem zeitorientierten Ansatz erheblich, wohingegen nicht mehr sichergestellt ist, daß der Sender nach einer festen Zeitschranke informiert wird.

Im folgendem wird die bedarfsorientierte Strategie betrachtet und erläutert, welches die auslösenden Ereignisse für Quittungen sind (vgl. Abbildung 5.8).

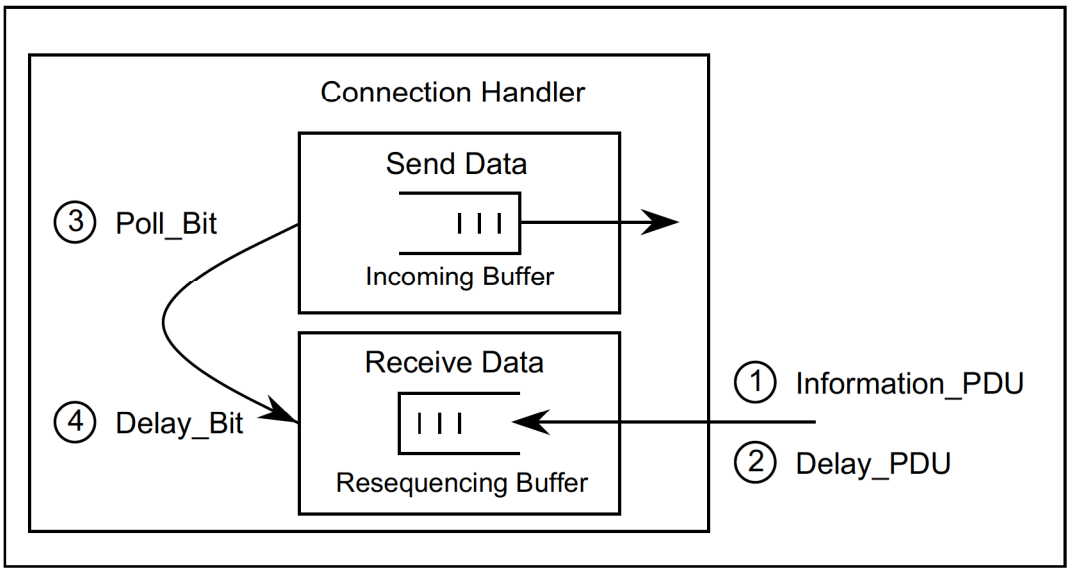


Abbildung 5.8: Ereignisse zum Auslösen von Quittungen

1. Empfang einer *Information_PDU*
2. Empfang einer *Delay_PDU*
3. Setzen des *Poll_Bit* durch den Sender in der zugehörigen *ARQ-Instanz*. Dieser Sender setzt das *Poll_Bit*, das eigentlich seinen Nutzdaten zugeordnet ist, dann, wenn er keine Daten versenden darf (z.B. aufgrund eines geschlossenen Fensters), aber dennoch pollen möchte. Zu diesem Zweck ist in den PDUs ein zweites *Poll_Bit* vorgesehen.
4. Setzen des *Delay_Bit* durch den zugehörigen Sender in derselben *ARQ-Instanz*. Der Sender setzt das *Delay_Bit*, wenn er eine *Delay_PDU* zu versenden hat. *Delay_PDUs* werden vom Protokoll als eine Art Quittung behandelt, die den Empfänger über den veränderten Zustand des Senders informiert.

Sind mehrere Quittungswünsche vorhanden, so müssen sie mit einer Priorität versehen werden. Diese kann wie folgt berechnet werden:

- Die Quittungen werden anhand ihres Zeitstempels in der Art ausgewählt, daß die Älteste versendet wird.
- Das Alter der Quittung wird zusätzlich mit dem *maximalen Delay* des Senders gewichtet.
- Die Quittungen werden nach der Anzahl der quittierten PDUs gewichtet, wobei sich das Gewicht bei *RR*– und *SREJ*–PDUs nach der Anzahl der empfangenen und fehlenden PDUs richtet.

- Die Art der Quittungen wird in der folgenden Reihenfolge bearbeitet:
  1. gepollte Quittung
  2. Delay
  3. Selective Reject (SREJ)
  4. Receive Ready (RR)

Zur Minimierung des Protokolloverheads können Quittungen gesammelt werden und so die absolute Anzahl der Quittungen verringert werden. Quittungen können auch zeitlich verzögert werden, um sie zusammen mit ATM-Zellen verschicken zu können. Zu diesem Zwecke wird im Protokoll eine Prioritätsschwelle definiert. Erst wenn Quittungen eine höhere Priorität als die Schwelle erreicht haben, werden sie auch ohne ATM Zellen in einem leeren Informationsrahmen verschickt.

KAPITEL 6

Implementierung des ASR ARQ-Protokolls

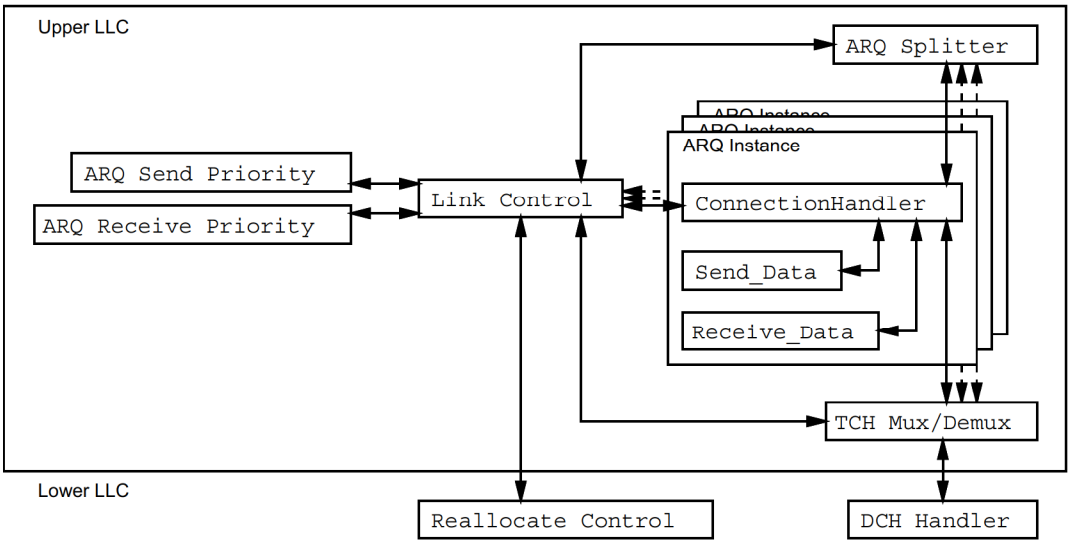


Abbildung 6.1: Struktur der oberen LLC Schicht

Abbildung 6.1 zeigt die Struktur der oberen LLC Schicht im MBS Simulator. Diese Schicht stellt innerhalb des Protokollstapels des Simulators einen austauschbaren Baustein dar und kann weitgehend unabhängig von anderen Bausteinen betrieben werden. Die genaue Konfiguration ist im Kapitel 7 beschrieben.

In der Initialisierungsphase einer Station werden die `Link Control`, der `ARQ Splitter`, der `TCH Mux/Demux` sowie die Prioritätenobjekte aufgebaut und sind während der gesamten Betriebszeit präsent. Die ARQ-Instanzen werden dagegen nach Bedarf auf- und abgebaut.

Zentrales Objekt in der oberen LLC Schicht ist die `Link Control`. Sie baut Kanäle auf und ab, indem sie ARQ-Instanzen auf- und abbaut. Die Zuordnung der ARQ-Instanzen zu den virtuellen Kanälen wird ebenfalls in der `Link Control` verwaltet. Der gesamte schichtinterne Verwaltungsverkehr läuft über sie, und alle anderen Objekte haben Zugriff auf die `Link Control`. Erhält sie eine Verbindungsanforderung von der oberen Schicht, so gibt sie diese Anforderung zur Kapazitätenprüfung an die `Reallocate Control` weiter und baut bei positiver Antwort eine ARQ-Instanz mit den angeforderten QoS Parametern auf. Der neue Kanal wird bei den Prioritätenobjekten angemeldet. Bei einer Verbindungsauslösung wird entsprechend umgekehrt vorgegangen.

Der `ARQ Splitter` teilt den ATM-Zellenstrom auf die einzelnen ARQ-Instanzen auf. Dazu wertet er die VC-Kennung und in der Basisstation zusätzlich die Mob (Mobile Station) Kennung aus und fragt mit diesen Daten die `Link Control` nach der zugehörigen ARQ-Instanz. Ist diese vorhanden, so werden die Daten an den `ConnectionHandler` dieser Instanz weitergegeben. Daten vom `ConnectionHandler` werden an die höhere Schicht durchgereicht.

Der `TCH Mux/Demux` verhält sich mit Daten vom `DCH Handler` wie der `ARQ Splitter` und gibt diese an den `ConnectionHandler` der zuständigen ARQ Instanz weiter. Zur Ermittlung des momentanen Zustandes werden die entsprechenden Funktion der Prioritäten Objekte via `Link Control` aktiviert. Zum Generieren einer PDU wird zunächst diejenige ARQ-Instanz ermittelt, die Nutzdaten am dringendsten zu versenden hat (`ARQ Send Priority`) und dann der `ConnectionHandler` dieser Instanz beauftragt, eine Information PDU zu generieren. Anschließend wird diejenige ARQ-Instanz ermittelt, die am dringendsten eine Quittung übermitteln muß (`ARQ Receive Priority`). Diese Instanz setzt ggf. aus der Information PDU und der Quittung eine Huckepack PDU zusammen. Der `TCH Mux/Demux` gibt diese an den `DCH Handler` zurück. Wird eine Verbindungsanforderung empfangen, so wird diese an die `Link Control` weitergereicht. Diese baut eine ARQ Instanz auf, ohne die `Reallocate Control` fragen zu müssen, da diese den Kanal bereits bei der Anforderung in der Partnerinstanz freigegeben hat. Die Verbindungsanforderung wird dann an den `ConnectionHandler` der neuen Instanz zur weiteren Behandlung übergeben. Bei der Aufforderung zur Verbindungsauslösung wird umgekehrt vorgegangen, d.h. zunächst wird die Anforderung im `ConnectionHandler` behandelt und dann auf Anfrage des `TCH Mux/Demux` die ARQ-Instanz von der `Link Control` abgebaut.

Die Prioritätenalgorithmen `ARQ Send Priority` und `ARQ Receive Priority` haben zwei Aufgaben:

Statusberechnung der ARQ-Instanzen
: Berechnet werden die Anzahl der Sendewünsche (`Length`), das älteste Datum (`First_Gen_Time`) und die kürzeste Restlebensdauer (`Remain_Life_Time`). Zur Ermittlung der Daten wird via `Link Control` auf die ARQ-Instanzen zugegriffen. Obwohl die Algorithmen für die Sende- und Empfangsseite sehr unterschiedlich sein können, ist die Schnittstelle einheitlich ausgeführt, um die Implementierung neuer Algorithmen zu vereinfachen.

Berechnung der Sendepriorität
: Hier wird nun diejenige ARQ-Instanz ermittelt, die jetzt senden möchte. Der Algorithmus zur Ermittlung der höchsten Priorität sollte dabei nach den gleichen Spielregeln handeln wie bei der Ermittlung des Status. Aus diesem Grund wurden diese beiden Aufgaben in den Prioritätenobjekten zusammengefaßt. Im Gegensatz zum Sendealgorithmus kann der Empfangsalgorithmus auch mitteilen, daß keine Instanz zu bevorzugen ist. In diesem Fall wird die Quittung derjenigen Instanz übermittelt, die Nutzdaten versendet.

Als `ARQ Send Priority` wurde der `Short_Remain`-Algorithmus implementiert. Diejenige Zelle mit der kürzesten `Remain_Life_Time` erhält die höchste Priorität. `Length` und `First_Gen_Time` finden in diesem Algorithmus keine Beachtung.

Als `ARQ Receive Priority` wurden drei Algorithmen zum Vergleich implementiert:

oldest_first Bei dieser Variante wird die `First_Gen_Time` der Quittungen betrachtet und die Ältesten mit der höchsten Priorität versehen. Zusätzlich wird die Anzahl der zu quittierenden Rahmen beachtet.

answer_poll_first Antworten auf Pollen und Delay PDUs werden sofort mit einer hohen Priorität versehen, andere Quittungen werden wie bei **oldest_first** behandelt.

prefer_poll Diese Variante unterscheidet nach der Art der Quittung und der Anzahl der zu quittierenden Rahmen im Verhältnis zur maximalen Fenstergröße. Bevorzugt werden Antworten im Poll-Zyklus und Delay_PDUs behandelt. Die Prioritäten werden in `Remain_Life_Time` umgerechnet, um sie mit der Send Priority vergleichen zu können.

`ARQ Send Priority` und `ARQ Receive Priority` müssen aufeinander abgestimmt werden, damit sie vergleichbare Prioritäten liefern, so daß auch Quittungen gegenüber Nutzdaten Priorität erhalten können. Das ist insbesondere bei asymmetrischen Verkehrsaufkommen notwendig.

6.1 Der `ConnectionHandler`

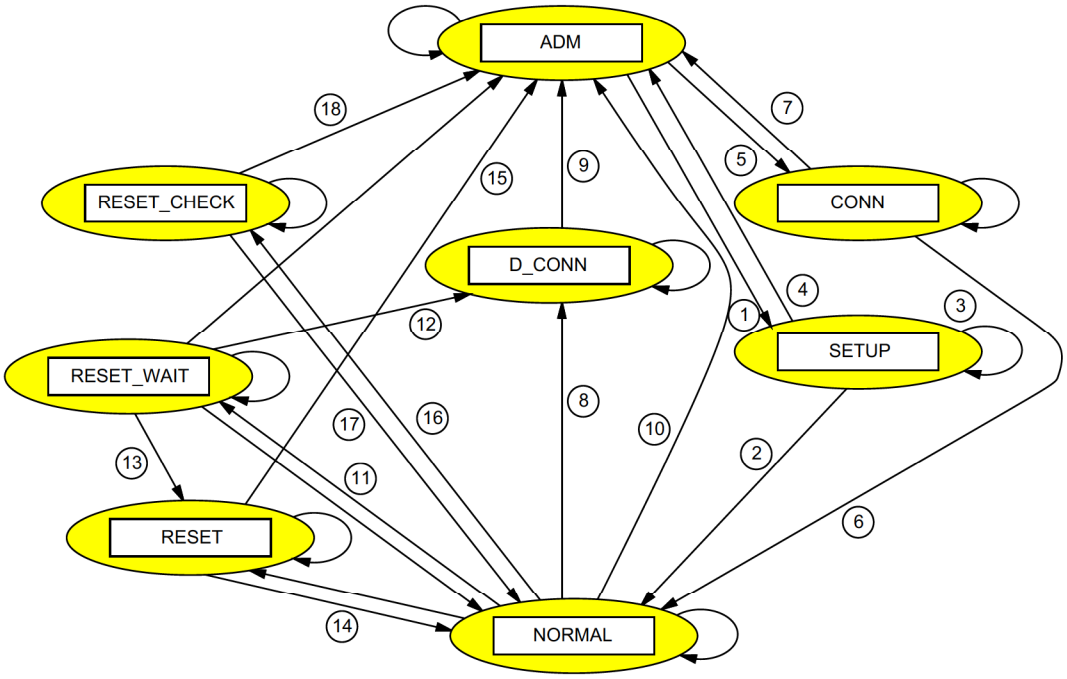


Abbildung 6.2: Zustände des `ConnectionHandler`

Der `ConnectionHandler` ist das zentrale Objekt der ARQ-Instanz. Er ist als endlicher Zustandsautomat implementiert und kann die in Abbildung 6.2 gezeigten Zustände annehmen. In bezug auf Verbindungsaufbau, -abbau und Resetbehandlung ist der Zustandsautomat gemäß ISO 8802-2 implementiert [17].

Zunächst befindet sich der `ConnectionHandler` im inaktiven Zustand ADM (Asynchronous Disconnected Mode). Die Zustandsübergänge lassen sich dann in drei Gruppen einteilen:

Verbindungsaufbau

Wird von der eigenen Station ein Verbindungswunsch geäußert, so verschickt er eine *SABME PDU* und geht in den Zustand SETUP über (1). Hier wird auf eine Antwort gewartet. Ist diese positiv, so wechselt der `ConnectionHandler` in den Zustand NORMAL und kann mit der Nutzdatenübertragung beginnen (2). Bleibt eine Antwort aus, so wird nach einem Timeout die *SABME PDU* wiederholt (3). Nach N2 Versuchen oder bei einer negativen Antwort wird in den Zustand ADM zurückgekehrt (4). Wird im Zustand ADM eine *SABME PDU* empfangen, so wird die nächsthöhere Schicht benachrichtigt und im Zustand CONN (CONNect) auf eine Antwort dieser Schicht gewartet (5). Bei einer positiven Antwort wird in den Zustand NORMAL (6), bei einer negativen in den Zustand ADM gegangen (7).

Verbindungsabbau
Trifft im Zustand NORMAL ein Verbindungsauslösungswunsch ein, so wird eine *DISC (DISConnect) PDU* versendet und im Zustand D_CONN (DisCONNect) auf eine Antwort gewartet (8). Es wird in jedem Fall in den Zustand ADM gewechselt (9). Die Partnerinstanz zeigt der höheren Schicht die Verbindungsauslösung an, versendet eine *UA PDU* und kehrt direkt in den Zustand ADM zurück (10).

Resetbehandlung
Wird im Zustand NORMAL ein lokaler Reset ausgelöst, so wird die höhere Schicht darüber informiert und im Zustand RESET_WAIT auf eine Antwort gewartet (11). Soll die Verbindung ausgelöst werden, so wird in den Zustand D_CONN gewechselt und wie beschrieben verfahren (12). Soll hingegen ein Reset ausgeführt werden, so wird die Verbindung zurückgesetzt, eine *SABME PDU* versendet und im Zustand RESET auf die Antwort der Partnerinstanz gewartet (13). Bei positiver Antwort wird in den Zustand NORMAL (14), bei negativer nach ADM gegangen (15). Empfängt die Partnerinstanz die *SABME PDU*, so fragt sie die höhere Schicht nach der Vorgehensweise und geht in den Zustand RESET_CHECK über (16). Wird der Reset angenommen, so wird auch auf dieser Seite die Verbindung zurückgesetzt, mittels *UA PDU* bestätigt und in den Zustand NORMAL zurückgekehrt (17). Bei negativer Antwort sendet sie eine *DM (Disconnect Mode) PDU* und geht in den Zustand ADM über (18).

Im Zustand NORMAL wird die Kontrolle an die Objekte `Send_Data` und `Receive_Data` abgegeben, die beim Verbindungsaufbau vom `ConnectionHandler` erzeugt werden.

6.2 Send Data

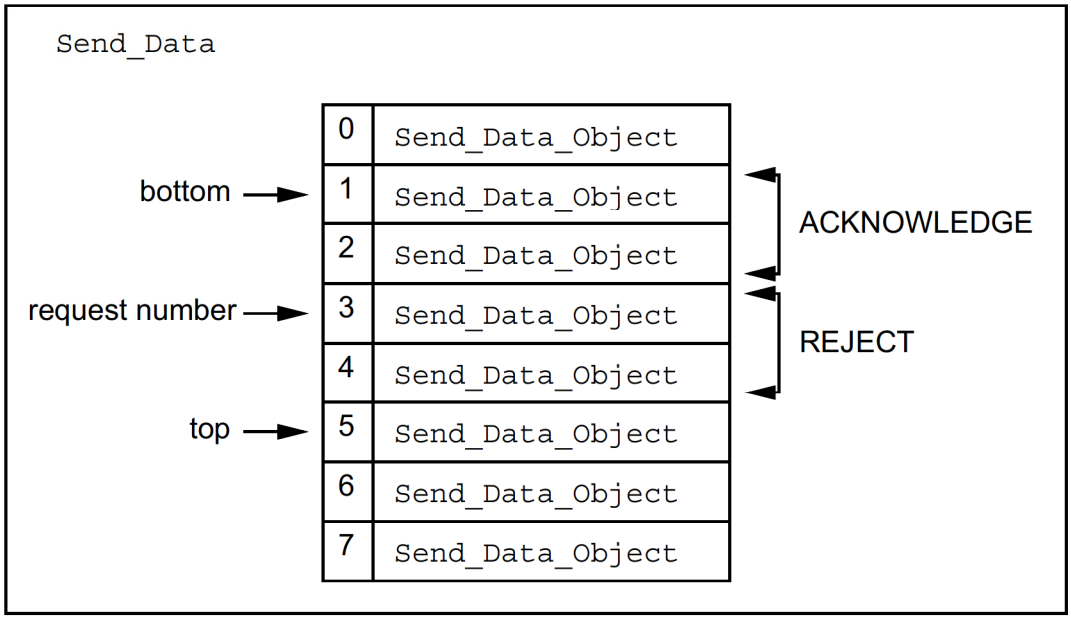


Abbildung 6.3: Ereignisse beim Empfang einer Quittung

`Send_Data` verwaltet das Sendefenster des ASR ARQ-Protokolls. Für jeden Fensterplatz wird ein `Send_Data_Object` eingerichtet, das diesen eigenständig verwaltet. `Send_Data`

kann Ereignisse an die `Send_Data_Objects` senden, ohne auf deren Status achten zu müssen, da diese selbstständig erkennen, ob das Ereignis für sie selbst sinnvoll ist. Dies hat den Vorteil, daß beim Empfang einer Quittung unabhängig vom internen Zustand immer in der gleichen Weise vorgegangen werden kann. Zur Verwaltung des Sendefensters besitzt `Send_Data` drei Zeiger:

bottom zeigt immer auf die unterste Fensterposition. Er zeigt also auf die älteste noch nicht quittierte Zelle.

top zeigt immer auf die Position, an der eine neue Zelle gespeichert würde. Dies ist nur möglich, wenn $top - bottom < n$ ist (n ist die maximale Fenstergröße, Kapitel 5).

send_number zeigt auf die Zelle, die als nächstes versendet werden soll.

Der Quittungsmechanismus soll anhand von Abbildung 6.3 erläutert werden. Dargestellt ist eine `Send_Data` Instanz mit acht Instanzen von `Send_Data_Objects`. Die maximale Fenstergröße ist $n = 4$. Das Fenster ist geschlossen, da $top - bottom = n$ ist. `Send_Data` empfängt nun eine *RR PDU* mit der Anforderungsnummer 3, d.h. der Empfänger erwartet als nächstes die Nummer 3. Daraufhin werden die Plätze *request number* $- 1$ bis *bottom* quittiert (ACKNOWLEDGE), in diesem Fall also die Plätze 2 und 1 (in dieser Reihenfolge). Danach werden alle Plätze *request number* bis $top-1$ zurückgewiesen (REJECT), hier 3 und 4. Wie die Quittung und die Zurückweisung behandelt werden, ist allein Aufgabe der `Send_Data_Objects`.

Nun müssen die Zeiger *bottom* und *send_number* aktualisiert werden. Der Zeiger *bottom* wird solange erhöht, wie der Zustand des `Send_Data_Object` von *bottom* IDLE ist und $bottom < top$ gilt. Der Zeiger *send_number* wird solange erhöht, wie der Zustand des `Send_Data_Object` von *send_number* ungleich SEND ist und $send\_number < top$ gilt. Ist das Sendefenster leer so gilt:

$$bottom = send\_number = top \tag{6.1}$$

Es gilt aber immer:

$$bottom \leq send\_number \leq top \tag{6.2}$$

Der Zeiger *top* wird erhöht, wenn eine neue Zelle gespeichert wird und muß nur dann aktualisiert werden, wenn Plätze im Falle eines Refreshs unbesetzt bleiben (s. Kapitel 6.2.1).

Zusätzlich zu diesen Aufgaben muß `Send_Data` den Acknowledgment Timer verwalten und die Empfangsinstanz über das Setzen des Poll Bits und das Versenden von Delay PDUs informieren.

6.2.1 `Send_Data_Object`

`Send_Data_Object` verwaltet einen Sendefensterplatz. Es ist als endlicher Zustandsautomat implementiert und kann die in Abbildung 6.4 dargestellten Zustände annehmen:

IDLE

Nach der Initialisierung befindet sich `Send_Data_Object` im Zustand IDLE. Trifft ein *DATA.request* ein, so wird die SDU (**S**ervice **D**ata **U**nit) gespeichert, der Delay_Timer gestartet und $count = 0$ gesetzt. `Send_Data_Object` geht in den Zustand SEND (1).

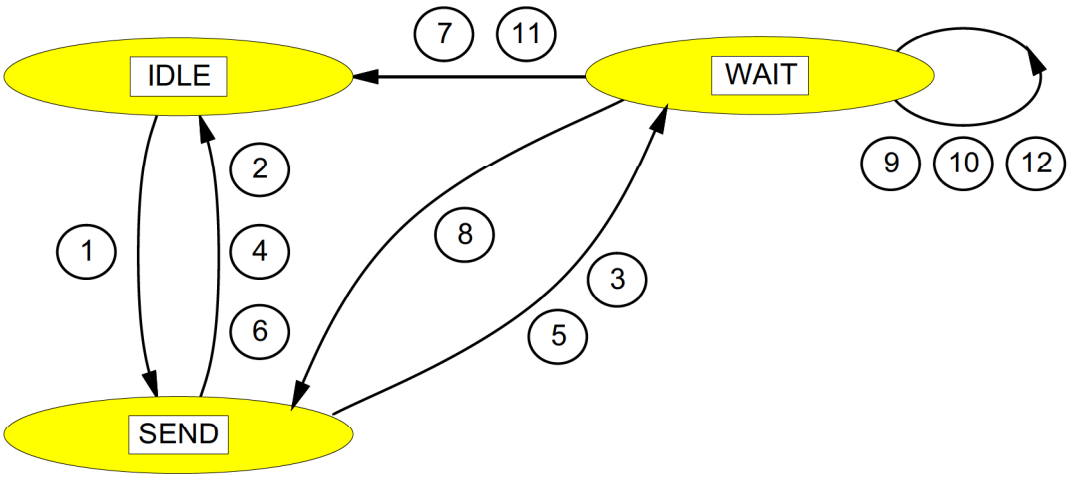


Abbildung 6.4: Zustände des `Send_Data_Object`

SEND

Im Zustand SEND wird auf das Versenden der SDU gewartet. Wurde die SDU bereits gesendet ($count > 0$) und trifft ein *ACKNOWLEDGE* ein, so wird die SDU gelöscht und die Timer gestoppt (2). Beim *TRANSMISSION.indication* muß unterschieden werden, ob die SDU bereits gesendet wurde ($count > 0$) und ob die Restlebenszeit noch nicht abgelaufen ist ($delay\_ok$). Soll die SDU das erste Mal gesendet werden und ist ihre Restlebenszeit abgelaufen, so kann sie verworfen und durch eine Neuere ersetzt werden. Dazu wird die SDU gelöscht, in den Zustand IDLE gewechselt und ein Refresh gestartet (4). Ansonsten wird die SDU zum Senden freigegeben, der ACK_ bzw. der Ignore_Timer gestartet und in den Zustand WAIT übergegangen (3). Läuft im Zustand SEND der *DELAY_Timer* ab, so wird die SDU gelöscht und in den Zustand WAIT gegangen, um den Empfänger über das Verwerfen der SDU zu informieren (5). Ist zusätzlich $count == 0$, so muß der Empfänger nicht informiert werden, es wird in Zustand IDLE gewechselt und ein Refresh initiiert (6).

WAIT

Im Zustand WAIT wird auf eine Quittung gewartet. Trifft eine positive Quittung ein (*ACKNOWLEDGE*), so wird die SDU gelöscht, die Timer gestoppt, ein Reset_Delayed initiiert und zurück in den Zustand IDLE gegangen (7). Bei einer negativen Quittung (*REJECT*) passiert nur etwas, wenn der Ignore_Timer nicht mehr läuft ($NOT\ Ignore$). Ist eine SDU vorhanden ($SDU \neq NIL$), so wird der ACK_Timer gestoppt und zum erneuten Versenden in den Zustand SEND gewechselt (8). Wurde die SDU verworfen ($SDU == NIL$), so wird eine Pollanforderung explizit zurückgenommen (Reset_Poll), eine Delay PDU angefordert (Set_Delayed) und im Zustand WAIT verharrt (9). Läuft der *ACK_Timer* ab, so ist die Anzahl der Übertragungsversuche zu beachten. Ist $count < N2$, so wird der ACK_Timer erneut gestartet, *count* um eins erhöht, das Poll Bit gesetzt und in WAIT weiter auf eine Quittung gewartet (10). Ist $count \geq N2$, so wird die SDU gelöscht, der Delay_Timer gestoppt, ein lokaler Reset ausgelöst und in den Zustand IDLE zurückgekehrt (11). Läuft der *Delay_Timer* ab, wird die SDU gelöscht und im Zustand WAIT verharrt (12).

In Tabelle 6.1 werden Funktionsaufrufe verwendet, hinter denen sich eine erhöhte Komplexität verbirgt. Diese Funktionen werden kurz erläutert:

	Istzustand	Ereignis	Aktion	Folgezustand
1	IDLE	DATA.request	SDU speichern DELAY_Timer starten $count = 0$	SEND
2	SEND	ACKNOWLEDGE $count > 0$	SDU löschen ACK_Timer stoppen DELAY_Timer stoppen	IDLE
3		TRANSMISSION indication $delay\_ok$ or $count > 0$	SDU senden ACK_Timer starten Ignore_Timer starten $count = count + 1$	WAIT
4		TRANSMISSION indication $NOT\ delay\_ok$ $count == 0$	SDU löschen Refresh starten	IDLE
5		DELAY_Timer abgelaufen $count > 0$	SDU löschen	WAIT
6		DELAY_Timer abgelaufen $count == 0$	SDU löschen Refresh starten	IDLE
7	WAIT	ACKNOWLEDGE	SDU löschen ACK_Timer stoppen DELAY_Timer stoppen Reset_Delayed	IDLE
8		REJECT $NOT\ Ignore$ $SDU \neq NIL$	ACK_Timer stoppen	SEND
9		REJECT $NOT\ Ignore$ $SDU == NIL$	Reset_Poll Set_Delayed	WAIT
10		ACK_Timer abgelaufen $count < N2$	ACK_Timer starten $count = count + 1$ Set_Poll	WAIT
11		ACK_Timer abgelaufen $count \geq N2$	SDU löschen DELAY_Timer stoppen Reset auslösen	IDLE
12		DELAY_Timer abgelaufen	SDU löschen	WAIT

Tabelle 6.1: Zustandsübergänge des `Send_Data_Object`

ACK_Timer stoppen Der Acknowledgment Timer ist in `Send_Data` einmal für alle `Send_Data_Objects` implementiert. Stoppen bedeutet in diesem Fall, daß der Timer gestoppt wird, falls dieses Objekt das einzige ist, das den Timer laufen hatte. Es wird also von `Send_Data` festgestellt, ob der Acknowledgment Timer gestoppt wird. Gleichzeitig wird auch ein **Reset_Poll** ausgeführt.

ACK_Timer starten Der Acknowledgment Timer wird nur dann neu gestartet, wenn er noch nicht läuft.

Refresh starten Zunächst ist es wichtig, daß *vor* dem Start des Refresh in den Zustand IDLE gewechselt wird. Der Refresh wird von `Send_Data` ausgeführt und aktualisiert das Sendefenster nach dem Verwerfen einer Zelle vor dem ersten Versenden. Alle nachfolgenden Zellen werden im Sendefenster einen Platz nach vorne verschoben, so daß wieder eine geschlossene Sequenz entsteht. Während der Ausführung können weitere Refresh ausgelöst werden, so daß es zu einer Rekursion kommt.

Set_Delayed `Send_Data` wird angezeigt, daß diese Zelle verworfen wurde und der Empfänger darüber benachrichtigt werden muß.

Reset_Delayed `Send_Data` wird benachrichtigt, daß das Versenden einer Delay PDU für diese Zelle nicht mehr notwendig ist.

Set_Poll `Send_Data` wird zum Setzen des Poll Bits aufgefordert, falls dieses noch nicht geschehen ist.

Reset_Poll Wegen dieser Zelle muß nicht mehr gepollt werden.

Reset auslösen Es wird ein Reset dieses VC ausgelöst, der vom `Connection_Handler` ausgeführt wird.

6.3 `Receive_Data`

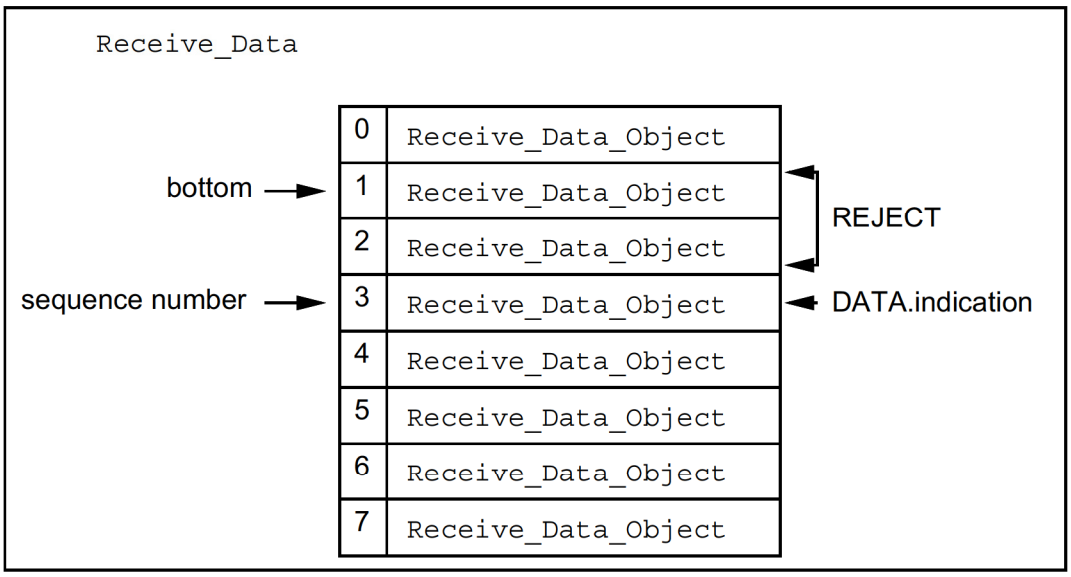


Abbildung 6.5: Ereignisse beim Empfang eines Rahmens

`Receive_Data` verwaltet das Empfangsfenster des ASR ARQ-Protokolls. Für jeden Fensterplatz wird ein `Receive_Data_Object` eingerichtet, das diesen eigenständig verwaltet. `Receive_Data` kann Ereignisse an die `Receive_Data_Objects` senden, ohne auf deren Status achten zu müssen, da diese selbstständig erkennen, ob das Ereignis für sie selbst sinnvoll ist. Dies hat den Vorteil, daß beim Empfang eines Rahmens unabhängig vom internen Zustand immer in der gleichen Weise vorgegangen werden kann. Zur Verwaltung des Empfangsfensters besitzt `Receive_Data` zwei Zeiger:

bottom zeigt immer auf die unterste Fensterposition. Er zeigt also auf die älteste noch nicht empfangene Zelle.

reject_number zeigt auf die Zelle, die als nächstes mittels SREJ angefordert werden soll.

Der Empfangsmechanismus soll anhand von Abbildung 6.5 erläutert werden. Dargestellt ist eine `Receive_Data` Instanz mit acht Instanzen von `Receive_Data_Objects`. Die maximale Fenstergröße ist $n = 4$. `Receive_Data` empfängt nun eine *RR PDU* mit der Sequenznummer 3. Daraufhin werden die Plätze *sequence number* − 1 bis *bottom* zurückgewiesen, in diesem Fall also die Plätze 2 und 1 (in dieser Reihenfolge). Danach wird die Zelle am Platz 3 gespeichert. Wie die Zurückweisung behandelt wird, ist allein Aufgabe der `Receive_Data_Objects`.

Nun müssen die Zeiger *bottom* und *reject_number* aktualisiert werden. Vom Zeiger *bottom* aus wird solange ein ACKNOWLEDGE an `Receive_Data_Objects` von *bottom* geschickt und *bottom* erhöht, wie deren Zustand RECEIVED ist. Der Zeiger *reject_number* wird nur dann verändert, wenn sich die Anzahl und Position der fehlenden Rahmen verändert hat. Dies wird durch die einzelnen `Receive_Data_Objects` angezeigt. Zusätzlich zu diesen Aufgaben muß `Receive_Data` den Reject Timer verwalten.

6.3.1 `Receive_Data_Object`

`Receive_Data_Object` verwaltet einen Empfangsfensterplatz. Es ist als endlicher Zustandsautomat implementiert und kann die in Abbildung 6.6 dargestellten Zustände annehmen:

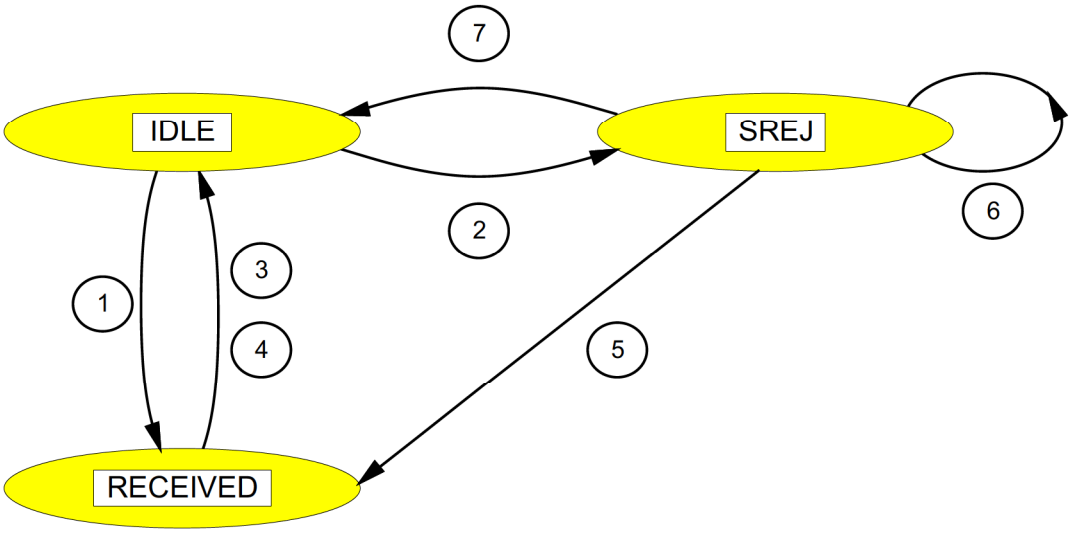


Abbildung 6.6: Zustände des `Receive_Data_Object`

IDLE

Nach der Initialisierung befindet sich `Receive_Data_Object` im Zustand IDLE. Trifft ein *DATA.indication* ein, so wird die SDU gespeichert, der FORWARD_Timer gestartet und in den Zustand RECEIVED übergegangen (1). Fehlt eine SDU in der Empfangssequenz (*REJECT*), so wird der REJECT_Timer gestartet und *reject_count* = 0 gesetzt. `Receive_Data_Object` geht in den Zustand SREJ über (2).

	Istzustand	Ereignis	Aktion	Folgezustand
1	IDLE	DATA indication	SDU speichern FORWARD_Timer starten	RECEIVED
2		REJECT	REJECT_Timer starten $reject\_count = 0$	SREJ
3	RECEIVED	ACKNOWLEDGE	SDU weitergeben FORWARD_Timer stoppen	IDLE
4		FORWARD_Timer abgelaufen	Start_Forwarding	IDLE
5	SREJ	DATA indication	SDU speichern REJECT_Timer stoppen FORWARD_Timer starten	RECEIVED
6		REJECT_Timer abgelaufen $reject\_count < N2$	$reject\_count++$ REJECT_Timer starten	SREJ
7		REJECT_Timer abgelaufen $reject\_count \geq N2$	Reset auslösen	IDLE

Tabelle 6.2: Zustandsübergänge des `Receive_Data_Object`

RECEIVED

Im Zustand RECEIVED wird auf das Eintreffen fehlender SDUs anderer `Receive_Data_Objects` gewartet. Wurden diese SDUs empfangen, so trifft ein *ACKNOWLEDGE* ein. Die SDU wird an die obere Schicht weitergegeben, der FORWARD_Timer gestoppt und in den Zustand IDLE zurückgekehrt (3). Läuft der *FORWARD_Timer* ab, so wird das Forwarden gestartet und in den Zustand IDLE gegangen (4).

SREJ

Im Zustand SREJ wird auf eine fehlende SDU gewartet. Trifft diese ein (*DATA.indication*), so wird die SDU gespeichert, der REJECT_Timer gestoppt, der FORWARD_Timer gestartet und in den Zustand RECEIVED übergegangen (5). Läuft der *REJECT_Timer* ab, so ist die Anzahl der Neuanforderungen zu beachten. Ist $reject\_count < N2$, so wird der REJECT_Timer erneut gestartet, $reject\_count$ um eins erhöht und weiter auf die fehlende SDU gewartet (Zustand SREJ) (6). Ist $count \geq N2$, so wird ein lokaler Reset ausgelöst und in den Zustand IDLE zurückgekehrt (7).

In Tabelle 6.2 werden Funktionsaufrufe verwendet, hinter denen sich eine erhöhte Komplexität verbirgt. Für **Reject_Timer starten** und **Reject_Timer stoppen** gilt das für den **Acknowledgment_Timer** gesagte entsprechend (Kapitel 6.2.1). Ebenso verhält es sich für **Reset auslösen**. Bei **Start_Forwarding** wird `Receive_Data` angezeigt, daß nicht weiter auf den Empfang von fehlenden SDUs gewartet werden soll. `Receive_Data` gibt daraufhin die zwischengespeicherten SDUs an die obere Schicht weiter und verschiebt sein Fenster entsprechend.

KAPITEL 7

Der Simulator SIMCO3++/MBS

7.1 Überblick

Die Implementierung des Simulators erfolgt in der objektorientierten Programmiersprache C++. Der Simulator arbeitet nach dem Prinzip der ereignisgesteuerten Simulation. Die hierzu benötigten Klassen werden von der am Lehrstuhl für Kommunikationsnetze entwickelten Klassenbibliothek *Communication Networks Class Library* (CNCL) bereitgestellt.

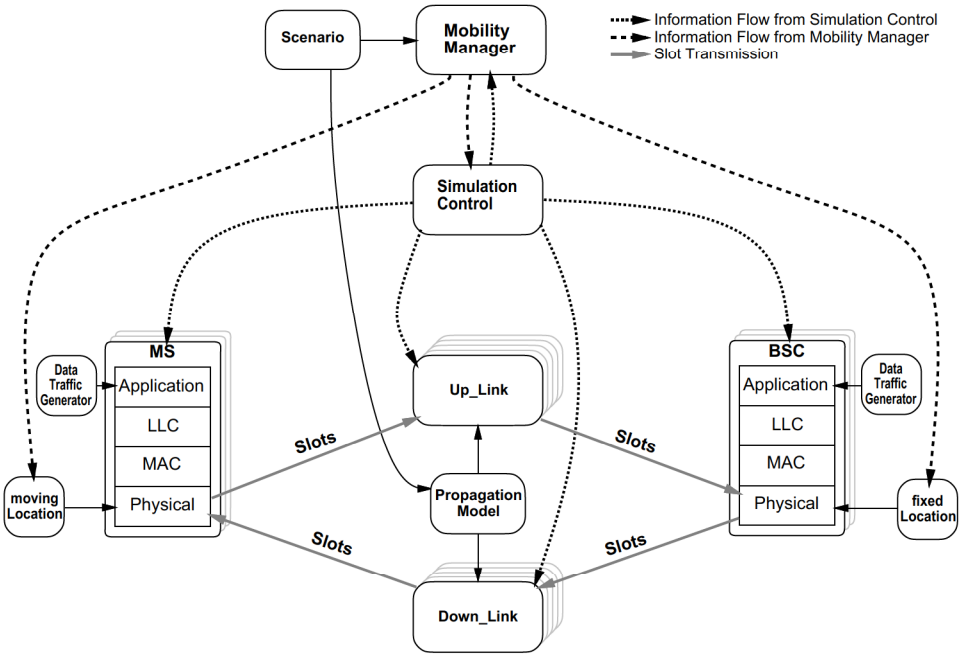


Abbildung 7.1: Module des Simulators

Der Aufbau des Simulators und der Informationsfluß zwischen den einzelnen Modulen geht aus Abbildung 7.1 hervor. Der Simulator verknüpft Kommunikationsobjekte mit ihrer Umgebung. Kommunikationsobjekte sind der *Base-Station-Controller* und die Mobilstationen. Die Umgebung wird durch die Mobilität der Kommunikationsobjekte und die physikalische Ausbreitung der Funksignale modelliert. Ein einheitliches Szenario, das gleichermaßen die physikalische Funkausbreitung und die Mobilität beeinflußt, beschreibt die Simulationsumgebung.

Die Module, welche die Mobilität bzw. die Funkausbreitung beinhalten, sind vom restlichen System durch ihre objektorientierte Implementierung strukturell getrennt. Die zunächst sehr vereinfachten Beschreibungsansätze können zu einem späteren Zeitpunkt problemlos gegen verschiedene, wesentlich komplexere Modellbeschreibungen ausgetauscht werden.

Ein vergleichbarer Implementierungsgrundsatz wurde auch beim Aufbau der Protokollstapel in den *Base-Station-Controllern* und Mobilstationen verfolgt. Die objektorientierte Unterteilung der verschiedenen Schichten durch Definition klarer Schnittstellen fördert die Austauschbarkeit und Wiederverwendbarkeit der spezifizierten Protokolle. Die Protokollstapel sind gemäß dem ISO/OSI-Referenzmodell aufgebaut. Zur Kommunikation zwischen den Schichten werden in der C++-Implementierung Klassen der SIMCO-Library verwendet. Die wichtigsten in Abbildung 7.1 dargestellten Objekte sind:

Simulation_Control Die Steuerung der Simulation wird von der zentralen Kontrolleinheit `Sim_Control` übernommen. Sie ist für die Erzeugung und das Löschen der *Base-Station-Controller* `BSC` und Mobilstationen `MS` verantwortlich. `BSC` und `MS` erzeugen wiederum die einzelnen Protokollschichten des Simulators.

Mobility_Manager Die Mobilitätssteuerung ermöglicht sowohl zentral als auch dezentral gesteuerte Mobilität. Die Funktionalität wird dabei einerseits auf dynamisch erzeugte und den Kommunikationsobjekten zugeordnete `Location`-Objekte verteilt, andererseits existiert aber auch eine zentrale Kontrolleinheit. Abhängig vom implementierten Mobilitätsmodell steckt mehr Funktionalität in der zentralen Einheit als in den dynamischen Objekten.

MS und BSC In den Klassen `MS` und `BSC` werden die einzelnen Schichten des Simulators instantiiert und gelöscht. Jedes Objekt der obigen Klassen steht dabei für eine Mobilstation bzw. einen *Base-Station-Controller*.

`.Application_Type`	`test_LLC`	Testen der LLC-Schicht
	`stat_APP`	Leistungsbewertung der LLC-Schicht
`.LLC_Type`	`LLC`	Aufteilung in `.Upper_LLC` und `.Lower_LLC`
	`LLC_stat_MAC`	Leistungsbewertung der MAC-Schicht
`.Upper_LLC_Type`	`Upper_LLC`	volle Funktionalität
	`Stat_LLC_Lower`	Leistungsbewertung von `Lower_LLC`
`.Lower_LLC_Type`	`Lower_LLC`	volle Funktionalität
	`Test_Upper_LLC`	Test von `Upper_LLC`
`.MAC_Type`	`MAC_DSA`	DSA-Protokoll
	`MAC_DSA++`	DSA-Protokoll++
	`MAC_Test_LLC`	Test von `Lower_LLC` und `Upper_LLC`

Tabelle 7.1: Protokollvarianten der verschiedenen Schichten in `.sim_defaults`

Der Protokollstapel ist entsprechend dem ISO/OSI-Referenzmodell in verschiedene Schichten mit verschiedenen Aufgabenbereichen aufgebaut. Da es verschiedene Möglichkeiten gibt, die Aufgaben einer Schicht bzw. Teilschicht zu erfüllen, muß es möglich sein, zur Simulation verschiedener Protokolle, das Protokoll einer Schicht gegen ein anderes Protokoll der gleichen Schicht auszutauschen. Die Auswahl von verschieden Schichttypen kann beim Start einer Simulation vonstatten gehen. Diese Aufgabe wird mit Hilfe der Datei `.sim_defaults` (Kapitel 7.5) in den Objekten der Klassen `MS` und `BSC` durchgeführt. Die in der Datei `.sim_defaults` eingetragenen Schichttypen werden instantiiert, initialisiert und danach gestartet. Am Schluß des Programmablaufs werden die einzelnen Schichten im Destruktor der Klassen `MS` und `BSC` gelöscht. Die Vielzahl von verschiedenen Protokollen und deren Bedeutung kann aus Tabelle 7.1 entnommen werden. Beim LLC-Typ *LLC* wird die LLC-Schicht nochmals in zwei Teilschichten unterteilt, die unabhängig voneinander konfiguriert werden können.

Für die simulative Bewertung des ASR-ARQ Protokolls sind der `stat_APP`, die Quellen zur Lasterzeugung und der `MAC_Test_LLC` von besonderer Bedeutung.

7.2 Der Application Layer `stat APP`

Der *Application Layer* repräsentiert die Schichten oberhalb der LLC-Schicht. Seine Aufgabe ist die Generierung von Verkehrslast für spezifizierte virtuelle Kanäle. Um möglichst realistische Szenarios simulieren zu können muß der *Application Layer* folgende Eigenschaften besitzen:

- Kommunikation über den Dienstzugangspunkt im LLC mittels *Service Primitives*
- Verbindung der Application Instanzen mit ausgehandelten Verbindungsendpunkten
- Sinnvolle Reaktion auf Verbindungsauf-, abbau- und Resetaufforderungen
- Generierung von Nutzdaten nach verschiedenen Quellenmodellen
- Anforderung von spezifizierten virtuellen Kanälen
- Flexibilität in bezug auf Stations-, Kanalanzahl und Quellenarten. Insbesondere sollen unterschiedliche Stationen mit unterschiedlicher Kanalcharakteristika erzeugt werden können.

Diese Anforderungen lassen sich einteilen in die Erzeugung von Nutzdaten in den *Quellen* (Kapitel 7.3) und die Verwaltung der virtuellen Kanäle.

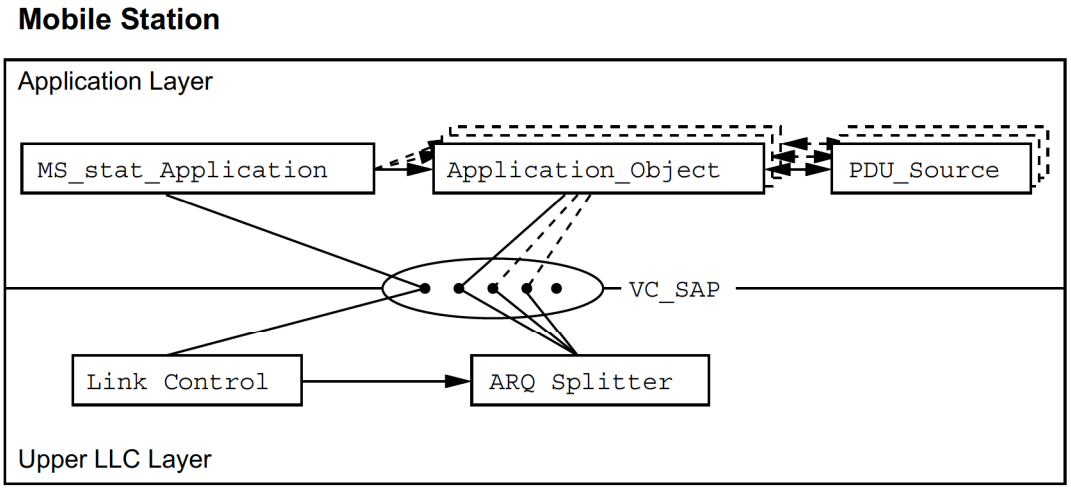


Abbildung 7.2: Application Layer in der Mobilstation

In Abbildung 7.2 ist zu erkennen, daß die gesamte Kommunikation zwischen Application und Upper LLC Layer über den `VC-SAP` (Virtual Channel-Service Access Point) geschieht. Tabelle 7.2 gibt Auskunft über die verwendbaren SPs und deren Bedeutung.

Für jede Mobilstation wird ein `MS_stat_Application`–Objekt erzeugt, das den Aufbau aller spezifizierten virtuellen Kanäle initiiert. Aufgrund des *Resource Managements* in der unteren LLC-Schicht kann nur die Mobilstation einen Kanalaufbau initiieren [6]. Dementsprechend muß `MS_stat_Application` auch den Aufbau derjenigen Kanäle anstoßen, die ausschließlich auf dem Downlink senden. Um einen Kanal aufbauen zu können, muß der Application Layer die *Mean Rate*, *Peak Rate* und das *maximale Delay* sowohl auf dem Up-,

Service Primitive	erzeugende Schicht	Bedeutung
L_Connect_Request	Application	Verbindungsanforderung
L_Connect_Indication	LLC	Anzeige eines Verbindungswunsches
L_Connect_Response	Application	Bestätigung eines Verbindungswunsches
L_Connect_Confirm	LLC	Bestätigung einer Verbindungsanforderung
L_Disconnect_Request	Application	Verbindungsauslösung
L_Disconnect_Indication	LLC	Anzeige einer Verbindungsauslösung
L_Reset_Request	Application	Rücksetzungsanforderung
L_Reset_Indication	LLC	Anzeige eines Rücksetzungswunsches
L_Reset_Response	Application	Bestätigung eines Rücksetzungswunsches
L_Reset_Confirm	LLC	Bestätigung einer Rücksetzungsanforderung
L_Data_Request	Application	Übergabe von Nutzdaten zur Übertragung
L_Data_Indication	LLC	Übergabe von empfangenen Nutzdaten

Tabelle 7.2: Service Primitive des Application Layer

wie auch auf dem Downlink wissen. Mit diesen Daten kann dann ein `L_Connect_Request` SP an die `Link Control` geschickt werden. Der Verbindungsaufbau wird über den Verbindungsendpunkt 0 verhandelt. In der LLC-Schicht ist hier die `Link Control` angeschlossen, die auch den Aufbau der zugehörigen ARQ-Instanzen veranlaßt (Kapitel 6).

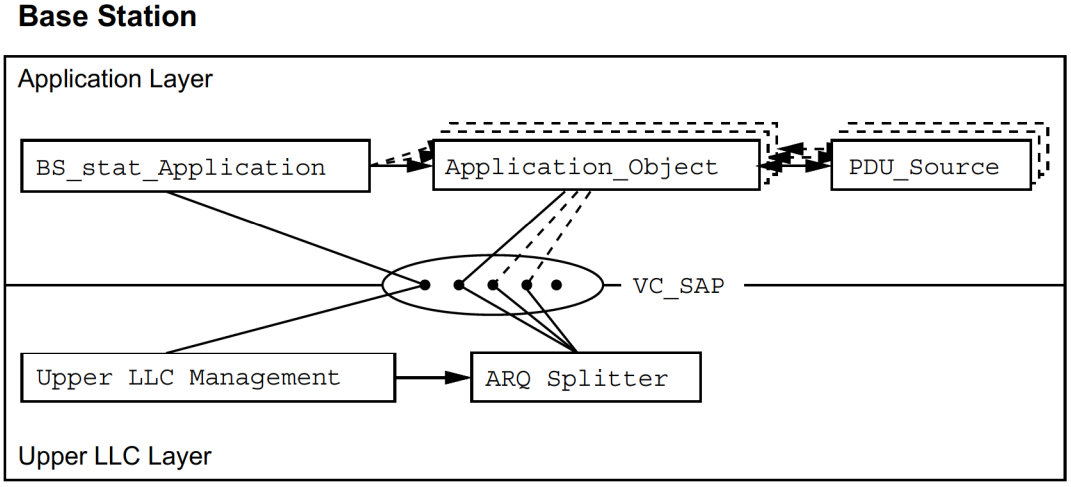


Abbildung 7.3: Application Layer in der Basisstation

Nach Erhalt eines `L_Connect_Indication` SP mit positiver Antwort oder eines `L_Connect_Confirm` SP, baut `MS_stat_Application` ein `Application_Object` auf, das mit dem übermittelten CEP im `VC_SAP` verbunden wird. Dieses Objekt baut die angegebene Quelle auf und beginnt mit der Nutzdatengenerierung. Empfangene Nutzdaten werden im Application Layer statistisch ausgewertet und vernichtet. Nach dem Aufbau eines VC wird die Kommunikation für diesen Kanal ausschließlich zwischen `Application_Object` und `ARQ Splitter` geführt.

In jeder Basisstation wird ein `BS_stat_Application`–Objekt erzeugt (Abbildung 7.3), das auf Verbindungsauf- und abbauanforderungen entsprechend reagiert. Im Gegensatz zur Mobilstation kann `BS_stat_Application` nur Verbindungsauslösungen initiieren. Sein An-

sprechpartner ist in der Basisstation das `Upper LLC Management`, das SPs an die zuständigen `Link Control` Objekte weiterreicht. Nur das `Upper LLC Management` besitzt in der Basisstation genügend Informationen über den zuständigen Upper LLC Block.

Während in der Mobilstation die ARQ-Kurzadresse des virtuellen Kanals als CEP-Adresse verwendet wird, muß in der Basisstation eine freie Verbindungsendpunktadresse vom *ARQ-Splitter* ermittelt und dem `BS_stat_Application` mitgeteilt werden. Dies ist notwendig, da in der Basisstation die ARQ-Kurzadressen nur in Verbindung mit der Mobilstationskennung eindeutig sind, d.h. verschiedene Mobilstationen können dieselben ARQ-Adressen verwenden.

7.3 Quellen

Als Quellentypen wurden folgende implementiert:

- Poisson
- LAN
- Video
- CBR (**C**onstant **B**it **R**ate)

Die Quellen berechnen entsprechend ihrer Charakteristika die Zwischenankunftszeit der nächsten PDU und veranlassen die Generierung einer solchen nach der berechneten Zeit.

7.3.1 Poisson Quelle

Bei diesem Quellentyp sind die Zwischenankunftszeiten negativ exponentiell verteilt. Daraus ergibt sich, daß die einzelnen Zeiten vollkommen unkorreliert sind, d.h. die Zwischenankunftszeit der nächsten Zelle ist unabhängig von der vorherigen. Man spricht von einer gedächtnislosen Quelle. Aus diesem Grund kann auch keine *Peak Rate* angegeben werden. Die Poisson Quelle wird durch die mittlere Datenrate eindeutig beschrieben.

Auch wenn davon auszugehen ist, daß das Datenaufkommen im MBS *nicht* negativ exponentiell verteilt sein wird, so stellt sie doch sehr allgemeine Anforderungen an das Protokoll, da ihre Zwischenankunftszeiten unkorreliert sind.

7.3.2 LAN Quelle

Die LAN Quelle modelliert den Datenverkehr in einem **L**ocal **A**rea **N**etwork. Die folgenden Daten basieren auf Messungen an bestehenden Rechnernetzen [35]. Es werden vier verschiedene Diensttypen unterschieden:

Terminal Input bezeichnet die Eingaben des Benutzers an einem Terminal.
Terminal Response ist die Antwort eines Rechners auf die Anfrage eines Terminals.
File Transfer beschreibt die Übertragung ganzer Dateien.
Paging ist das Ein- und Auslagern von Hauptspeicherseiten bei Workstations ohne eigenen Massenspeicher.

Weiterhin gibt es fünf verschiedene Stationstypen:

Workstation beschreibt einen leistungsfähigen Arbeitsplatzrechner. Er generiert Dienste vom Typ Terminal Response und File Transfer.

Diskless Workstation bezeichnet einen leistungsfähigen Arbeitsplatzrechner ohne eigene Festplatte. Er erzeugt Dienste vom Typ Terminal Input und Paging.

Department Host dient im Szenario als File-Server und generiert die Diensttypen Terminal Response, File Transfer und Paging.

Gateway stellt die Verbindung zu anderen Rechnernetzen dar. Von ihm werden Terminal Response und File Transfer erzeugt.

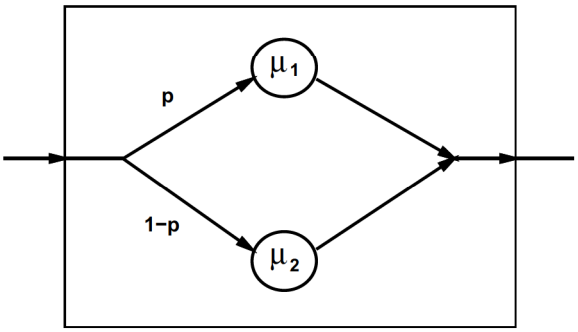
Central Host ist ein leistungsfähiger Großrechner. Er generiert die Dienste Terminal Response und File Transfer.

Diese Stationstypen können die verschiedenen Dienste mit unterschiedlichen Packetlängen erzeugen. Die Packetlänge ist hyperexponentiell nach einer H_2 Verteilung mit folgenden Mittelwerten und relativen Varianzen verteilt:

	Terminal Input	**Terminal Response**	**File Transfer**	**Paging**
Mittelwert $\overline{x}$ [Bytes]	12	180	40000	8192
relative Varianz c	1.3	6.1	6.0	0

Tabelle 7.3: Mittelwert und Varianz nach Dienstklassen

Mit den Werten aus Tabelle 7.3 ergibt sich für die Werte in dem Modell der H_2 -Verteilung (Abbildung 7.4):



$$p = \frac{1}{2} + \frac{1}{2} \cdot \sqrt{1 - \frac{1}{1+c^2}} \tag{7.1}$$

$$\mu_1 = \frac{2 \cdot p}{\overline{x}} \tag{7.2}$$

$$\mu_2 = \frac{2 \cdot (1-p)}{\overline{x}} \tag{7.3}$$

Abbildung 7.4: Modell der H_2 -Verteilung

Mit den Gleichungen (7.1), (7.2) und (7.3) gilt für die Packetlängen:

$$P[X \leq x] = 1 - p \cdot e^{-\mu_1 \cdot x} - (1-p) \cdot e^{-\mu_2 \cdot x} \tag{7.4}$$

Tabelle 7.4 gibt den Mittelwert der Anzahl der Dienste pro Stunde nach Stationstypen an. Die Zwischenankunftszeit zwischen zwei Diensten einer Klasse ist negativ exponentiell verteilt. Die Zellen eines Paketes werden mit der Rate 1 erzeugt, d.h. in jedem Slot eine. Die gesamte mittlere Datenrate wird der Quelle übergeben und die einzelnen Datenraten werden dementsprechend angepaßt.

7.3.3 Video Quelle

Diese Quelle simuliert Videotelefonie in hoher Qualität und basiert auf Vorschlägen aus [4].

Station Type	Terminal Input	Terminal Response	File Transfer	Paging
Workstation	0	2500	60	0
Diskless	750	0	0	360
Department Host	0	31500	296	1800
Gateway	0	21250	128	0
Central Host	0	55000	832	0

Tabelle 7.4: Mittelwert der Anzahl der Dienste pro Stunde

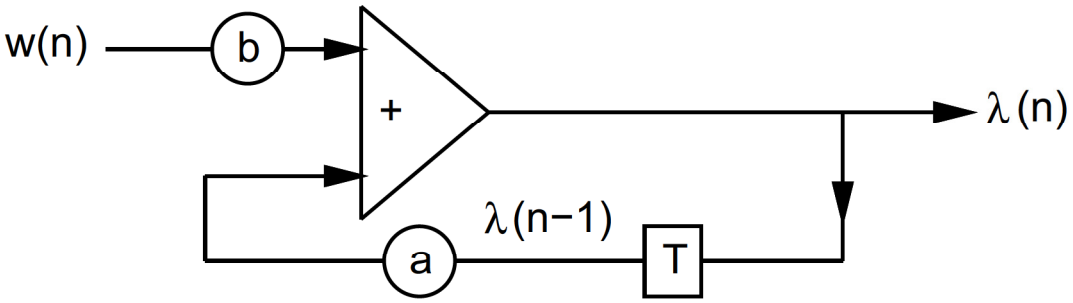


Abbildung 7.5: Autoregressiver Prozeß

Zur Modellierung der Anzahl der Pakete pro Bild dient ein autoregressiver Prozeß 1. Ordnung (Abbildung 7.5). Die Anzahl der Bits pro Pixel für das n-te Bild ist gegeben durch:

$$\lambda(n) = max\left\{a \cdot \lambda(n-1) + b \cdot w(n), 0\right\} \tag{7.5}$$

mit $\lambda(0) = 0.52$, $a = 0.878$, $b = 0.111$ und $w(n)$ einer Gaußverteilung mit $\eta = 0.572$ und $\sigma^2 = 0.0536$.

Bei 30 Bildern pro Sekunde und 250.000 Pixel Auflösung pro Bild, produziert diese Quelle im Mittel 170 Zellen pro Bild, das entspricht 1.9 Mbit/s. Der Abstand zwischen zwei Bildern beträgt 1548 Slots. Diese Quelle wird wiederum auf eine gesamte mittlere Datenrate normiert. Alle Zellen eines Bildes werden quasi gleichzeitig erzeugt.

7.3.4 CBR Quelle

Die Zwischenankunftszeit bei dieser Quelle ist konstant gegeben durch:

$$t = \frac{1}{mean\ rate} \quad [\tau_{slot}] \tag{7.6}$$

Sie ist deterministisch vorhersagbar. Sie stellt somit geringe Ansprüche an das ASR ARQ-Protokoll und wird nur im Quellenmix zusammen mit den anderen vorgestellten Quellen verwendet.

7.4 Der MAC Layer MAC_Test_LLC

Der MAC_Test_LLC bildet für die LLC-Schicht alle unteren Schichten dar, also die MAC und die Physikalische Schicht.

Um die Auswirkungen von Teilaspekten des Protokolls zu bestimmen, wurde der Einfluß eines konkreten MAC-Protokolls eliminiert. Stattdessen wird die MAC-Schicht durch einen idealen Scheduler mit G/D/1/FCFS/RU-NONPRE Strategie modelliert. Dabei wird im Scheduler eine ideale Kenntnis der Kapazitätsanforderungen angenommen. Die Übertragung von kurzen Supervisory-Rahmen ist nicht möglich und Quittungen werden nur huckepack zu Informationsrahmen übertragen, gegebenenfalls mit leerem Informationsfeld. Die gemessenen Verzögerungen sind daher stets niedriger als in einem realen System. Das für die Bewertung von Protokolloptionen wichtige Verhältnis zwischen Verteilungen von Übertragungsverzögerungen bleibt jedoch erhalten. Das Kanalmodell entspricht einem Indoor-Szenario mit vernachlässigbaren Signallaufzeiten.

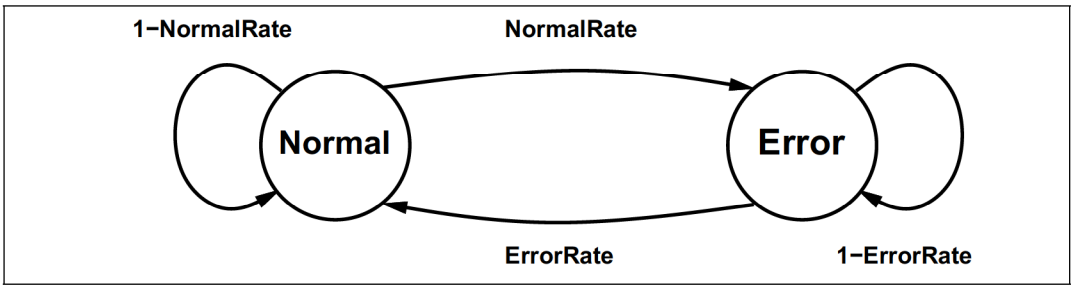


Abbildung 7.6: Gilbert Modell

Die Bitfehlerraten werden durch ein Gilbert Modell mit zwei Zuständen nachgebildet (Abbildung 7.6). Im Zustand **Normal** treten keine Bitfehler auf. Im Zustand **Error** tritt ein Bitfehler auf, der zu der in PDUErrorRate festgelegten Rahmenverlustrate führt. Die mittlere Fehlerburstlänge wird in StateErrorRate in $\frac{1}{\tau_{slot}}$ festgelegt. Mit der Angabe der mittleren Länge von fehlerfreien Perioden in StateNormalRate wird die mittlere Fehlerrate festgelegt:

$$mean\ error\ rate = \frac{StateNormalRate}{StateErrorRate} \cdot PDUErrorRate \tag{7.7}$$

Mit diesem Modell läßt sich ein Bitfehler parametrieren, der für einen Breitband-Kanal typischen ist [7, 33] (mittlere Fehlerburstlänge = 50 τ_{slot}, Bitfehlerrate im schlechten Zustand = 10^{-3}, mittlere Bitfehlerrate = 10^{-4}). Aufgrund der Vollduplexübertragung auf zwei unterschiedlichen FDM Kanälen werden Uplink und Downlink als unkorreliert angenommen. Weiterhin werden in dieser Schicht alle Fehler als erkennbar angenommen, so daß sich Bitfehler durch das Verwerfen des kompletten Rahmens äußern und keine fehlerhaften Rahmen an die LLC-Schicht weitergeleitet werden.

Zusammenfassend hat der MAC_Test_LLC folgende Funktionalität:

- Kapazität wird der Station mit der höchsten Priorität zur Verfügung gestellt.
- Es tritt eine konstante Übertragungsverzögerung von $0.9\tau_{slot}$ auf.
- Fehlerhafte Rahmen werden komplett verworfen.
- Die Fehlerrate und ihre Korrelation ist parametrierbar.

7.5 Die Konfigurationdatei `.sim_defaults`

`Sim_Length:`	Simulationsdauer, angegeben in der Einheit Slot
`Sim_No:`	dient zur Kennzeichnung und Unterscheidung der Ausgabedateien
`Output.directory:`	Verzeichnis, in das die Ausgabedateien geschrieben werden
`Parameter`	
`.LLC`	
`.ConSetup_TimerDelay:`	Zeit bis der Timer im `ConnectionHandler` abläuft
`.N2:`	Anzahl der maximalen Übertragungsversuche im `ConnectionHandler`
`.CheckTime:`	Zeit bis Quellen im *BSC* gestartet werden
`.GIST_Delay:`	Zeit bis zum Start des GIST
`.output_interval:`	Zeit zwischen zwei Dateiausgaben
`.max_ms_in_cell:`	maximale Anzahl der Mobilstationen, die eine MAC-Instanz im *BSC* aufnehmen kann
`.max_vc_in_ms:`	Anzahl der virtuellen Kanäle, die die LLC-Schicht aufnehmen kann
`.max_ms_in_bs:`	Anzahl der Mobilstationen, die die LLC-Schicht aufnehmen kann
`.MacMaxCEP:`	Anzahl der Verbindungsendpunkte in der Mobilstation
`.stat_Application:`	
`.NoPDUs:`	Anzahl der erzeugten PDUs pro VC
`.Nr_Uplink_VC:`	Maximale Anzahl von VCs in einer MS
`LLC`	
`.VC_Send_Prio_method:`	Algorithmus für `ARQ Send Priority`
`.VC_Receive_Prio_method:`	Algorithmus für `ARQ Receive Priority`
`.MaxQueueLength:`	maximale Anzahl von Warteschlangeplätzen
`MAC.Test_LLC`	
`.PDUErrorRate:`	Fehlerrate im Zustand Error
`.StateErrorRate:`	Rate zum Verlassen von Error
`.StateNormalRate:`	Rate zum Verlassen von Normal
`Mobility.MS_Density:`	Anzahl des Mobilstationen
`debug_LLC_up:`	Debuginformationen für die obere LLC-Schicht

Tabelle 7.5: Parameter in der Konfigurationdatei `.sim_defaults`

Die Datei `.sim_defaults` enthält eine Reihe von Simulationsparametern. Sie werden zu Beginn der Simulation eingelesen und verarbeitet. Dadurch ist es möglich, verschiedene Simulationen mit verschiedenen Parametern und verschiedenen Schichtprotokollen durchzuführen, ohne erneut kompilieren zu müssen. In der Tabelle 7.5 sind die für die obere

LLC-Schicht wichtigen Parameter aufgeführt.

`Uplink_VC`	
`.VC`*x*	*x = Nummer des virtuellen Kanals*
`.MobId:`	Kennung der Mobilstation
`.VcId:`	Kennung des virtuellen Kanals
`.Windowsize:`	maximale Fenstergröße im Sender und Empfänger
`.max_sequence_number:`	Modulus zur Kodierung der Sequenznummern
`.Ack_Threshold:`	Prioritätenschwelle, ab der leere Rahmen gesendet werden
`.ARQ_TimerDelay:`	Delay für Acknowledgment Timer
`.RejectTimer_used:`	Reject Timer benutzen
`.Reject_TimerDelay:`	Delay für Reject Timer
`.PTimer_used:`	Poll Bit Timer benutzen
`.P_TimerDelay:`	Delay für Poll Bit Timer
`.DelayTimer_used:`	Delay Timer und Delay PDU benutzen
`.ForwardTimer_used:`	Forward Timer benutzen, um das Resequencing im Empfänger abzubrechen
`.IgnoreTimer_used:`	Ignore Timer benutzen
`.N2:`	Anzahl der Übertragungsversuche bis zum Reset
`.Mean_Rate:`	mittlere Datenrate der Quelle
`.Source_Type:`	Quellentyp
`.Delay:`	maximale Verzögerung für ATM Zellen
`.LRE_used:`	LRE zur statistischen Auswertung benutzen

Tabelle 7.6: Parameter in der Konfigurationdatei `.uplink_vc_llc`

Aus Gründen der Übersichtlichkeit wurden die Parameter zur Spezifikation der einzelnen virtuellen Kanäle in die Dateien `.uplink_vc_llc` und `.downlink_vc_llc` ausgelagert. In diesen Dateien können für jeden virtuellen Kanal die in Tabelle 7.6 aufgeführten Parameter eingestellt werden. Außerdem müssen in `Uplink_VC.Nr_of_VCs_used` die Anzahl der verwendeten virtuellen Kanäle spezifiziert werden. In `.downlink_vc_llc` sind alle Parameter durch `Downlink_VC` statt durch `Uplink_VC` einzuleiten. Diese beiden Dateien werden durch `VC_Base` ausgewertet und verwaltet, so daß unter Angabe der `MobId` und der `VcId` die entsprechenden Werte abgefragt werden können.

7.6 Der graphische Debugger

Protokollabläufe in der LLC-Schicht können mit dem Programm GIST (**G***raphical* **I***nteractive* **S***imulation result* **T***ool*) sichtbar gemacht werden. GIST ermöglicht es, die zeitlichen Änderungen der Zustände der Simulation aufzuzeichnen und dann wie mit einem Videorecorder vorwärts und rückwärts wieder abzuspielen, um so das Protokoll schrittweise demonstrieren zu können bzw. das Auffinden von Fehlern zu ermöglichen.

Speziell für die Visualisierung des ARQ-Protokolls in der LLC-Schicht wurde in [10] die Schnittstelle `LLC-Graphic` entwickelt, die alle für das Protokoll wichtigen Abläufe darstellen kann. Eine Auflistung der in dieser Schnittstelle definierten Funktionen befindet sich in [34].

Abbildung 7.7 zeigt beispielhaft die Bildschirmanzeige im GIST.

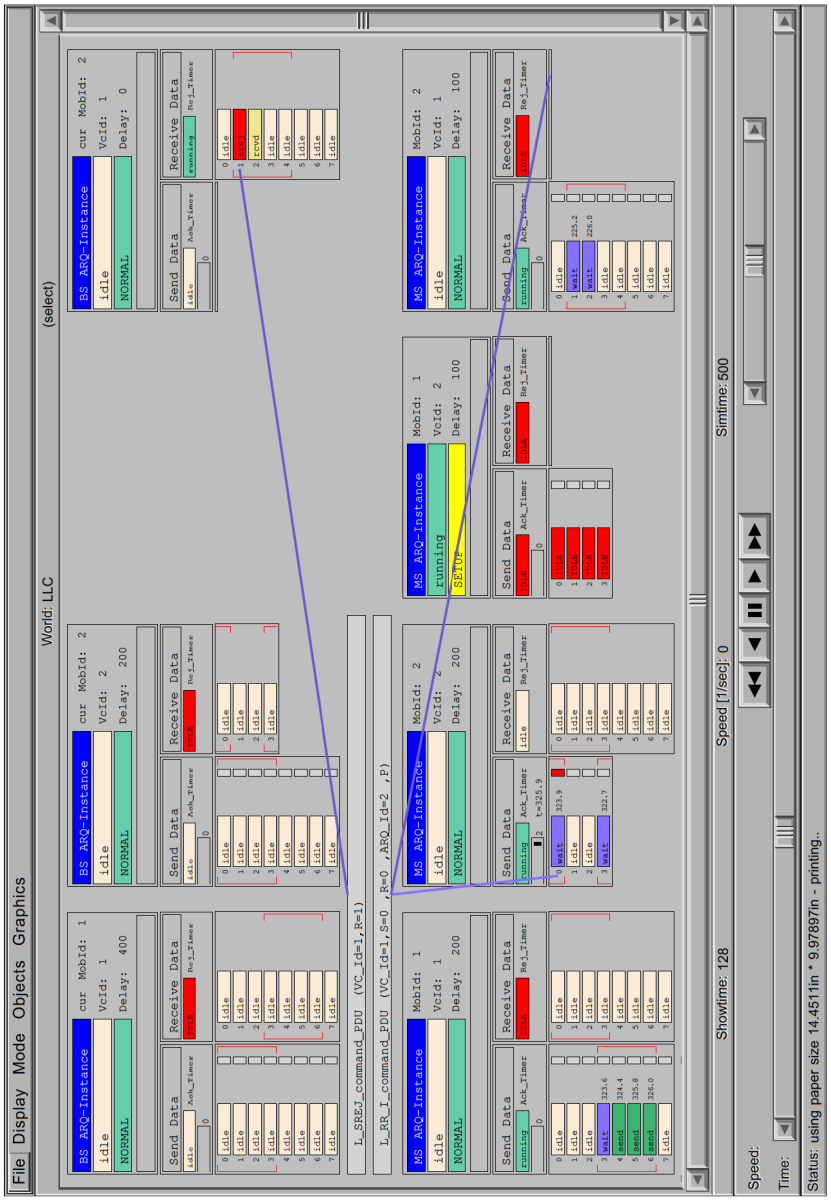


Abbildung 7.7: Bildschirmanzeige im GIST

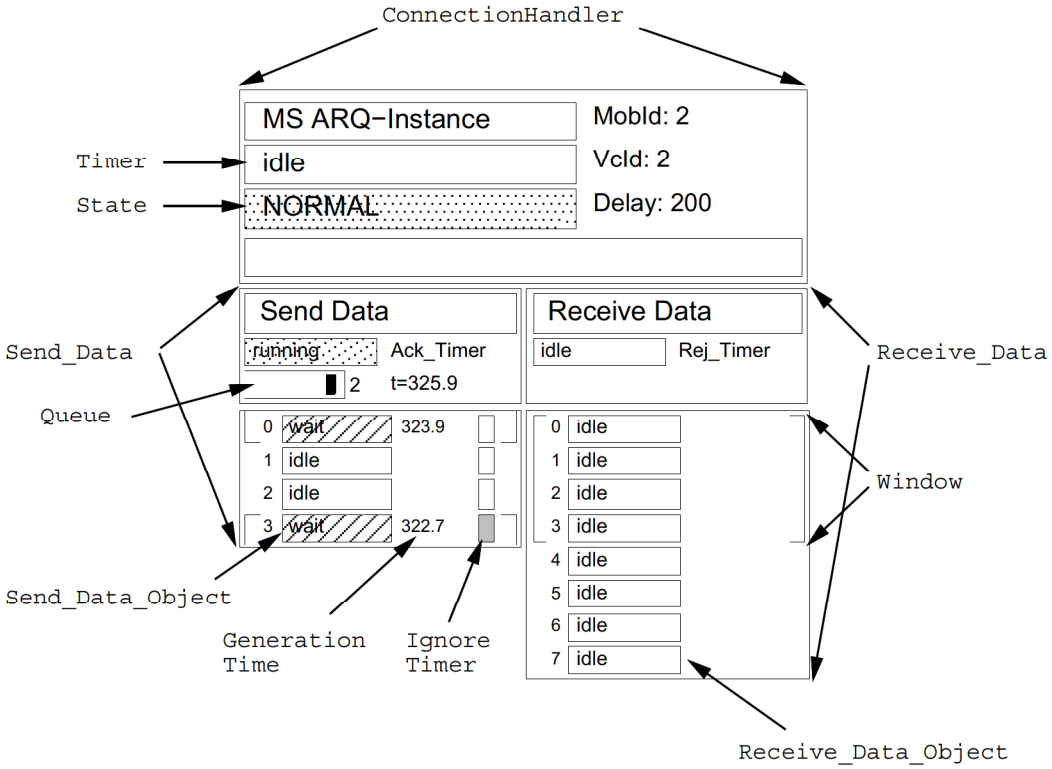


Abbildung 7.8: Elemente im GIST

Folgende Elemente können im GIST dargestellt werden (vgl. Abbildung 7.8):

- Der `ConnectionHandler` mit seinem Zustand und Timer
- `Send_Data` mit der Position des Sendefensters, mit dem Zustand der Warteschlange und dem Zustand des Acknowledgment Timers
- `Send_Data_Object` mit seinem Zustand, der Entstehungszeit einer gespeicherten Zelle und dem Zustand des Ignore Timers
- `Receive_Data` mit der Position des Empfangsfensters und dem Zustand des Reject Timers
- `Receive_Data_Object` mit seinem Zustand und der Entstehungszeit einer gespeicherten Zelle
- Art und Herkunft von übertragenen ARQ-Rahmen

Für Implementierungsdetails wird auf [34] verwiesen.

Simulationsergebnisse

8.1 Einführung

Ziel der Simulationen ist es, die Leistungsfähigkeit des ASR ARQ gegenüber herkömmlichen ARQ-Protokollen und gegenüber einem idealisierten ATM Multiplexer zu bewerten. Dementsprechend steht die relative Leistungsfähigkeit als Bewertungskriterium im Vordergrund. Es können auch nur relative Aussagen sinnvoll getroffen werden, da ausschließlich mit einer idealisierten MAC Schicht simuliert wurde. Dadurch wird aber andererseits erreicht, daß die Ursachen unterschiedlicher Meßergebnisse ausschließlich innerhalb der LLC Schicht liegen.

8.1.1 Meßgrößen und Leistungsparameter

ITU-T I.356 sieht für ATM folgende Leistungsparameter vor [18]:

CER Cell Error Ratio

$$CER = \frac{total\ errored\ cells}{total\ successfully\ transferred\ + total\ errored\ cells}$$

CLR Cell Loss Ratio

$$CLR = \frac{total\ lost\ cells}{total\ transmitted\ cells}$$

CMR Cell Misinsertion Rate

$$CMR = \frac{total\ number\ of\ misinserted\ cells\ in\ time\ interval}{time\ interval\ duration}$$

SECBR Severely Errored Cell Block Ratio

$$SECBR = \frac{total\ severely\ errored\ cell\ blocks}{total\ cell\ blocks}$$

CTD Cell Transfer Delay

$$CTD = time\ between\ arrival\ in\ sender\ and\ department\ in\ receiver$$

Da es im MBS-Simulator kein realitätsnahes Modell zur Erzeugung fehlerhafter Zellen gibt, können *CER*, *CMR* und *SECBR* nicht gemessen werden.

Gemessene Größen sind neben der *CTD* unter anderen:

discard ratio

$$\frac{Anzahl\ verworfener\ Zellen}{Anzahl\ erzeugter\ Zellen}$$

throughput

$$\frac{Anzahl\ weitergeleiteter\ Zellen}{Anzahl\ versendeter\ PDUs}$$

multitransmission ratio

$$\frac{Anzahl\ empfangener\ Information\ PDUs\ -\ Anzahl\ weitergeleiteter\ Zellen}{Anzahl\ weitergeleiteter\ Zellen}$$

Aus der *discard ratio* und der Verteilungsfunktion der Zellverzögerungen läßt sich die *CLR* ermitteln. *CLR* und *CTD* stellen in den Simulationen als QoS Parameter die einzuhaltenden Bedingungen an die VCs dar. Aus der *CTD* werden die *MCTD* (Mean Cell Transfer Delay) als arithmetisches Mittel und die *CDV* (Cell Delay Variation) ermittelt.

8.1.2 Statistische Aussagefähigkeit

Für jede während eines Simulationslaufes zu bestimmende Zufallsgröße gibt es eine endliche Sequenz von Meßwerten, die nach einem statistischen Verfahren zu bewerten sind. Typische Gegebenheiten in den durchgeführten Simulationen sind:

- Der Zufallsprozeß ist stationär
- Gemessene Werte sind untereinander korreliert
- Der Typ des Zufallsprozesses ist unbekannt.
- Anzahl der Meßwerte in der Größenordnung von 10^6-10^7

Aufgrund dieser Gegebenheiten soll das statistische Auswerteverfahren folgende Aussagen machen:

- empirische Verteilungsfunktion
- empirische Momente
- Korrelationsaussagen
- objektives Fehlermaß

Beim konventionellen Verfahren nach Batch Means wird die Zufallssequenz in gleichgroße Teilsequenzen unterteilt, die als quasi unabhängig betrachtet werden. Dies setzt implizit voraus, daß die Summen-Zufallswerte quasi normalverteilt sind. Dies ist aber nur bei unkorrelierten Zufallssequenzen der Fall. Auch die als Fehlermaß abgeleiteten Konfidenzintervalle basieren auf priori Annahmen und sind deshalb nur bedingt anwendbar.

Die Anwendung der Bayes-Laplace-Statistik umgeht diese Probleme [11]. Diese wird im LRE (**L**imited **R**elative **E**rror) Algorithmus vorteilhaft angewendet. Mit Hilfe des LRE wird die Simulationsdauer anhand des relativen Fehlers überwacht. Für eine genaue Einführung in die Bayes-Laplace-Statistik und die verschiedenen LRE-Algorithmen wird auf [11] verwiesen.

Zur Überwachung der Simulationsdauer und des relativen Fehlers wurde der in der CNCL Bibliothek implementierte LRE-Algorithmus angewendet. In den folgenden Simulationen beträgt der relative Fehler im angezeigten Bereich weniger als 1%.

8.1.3 Simulationsparameter des ASR ARQ

Zwischen den einzelnen Parametern in Tabelle 8.1 gibt es folgenden kausalen Zusammenhang: Die Größe des *Modulus* ist ein Systemparameter, der beim Entwurf des Systems festgelegt wird und nicht dynamisch verändert werden kann. Mit dem *Modulus* liegt auch die maximale Fenstergröße fest (Kapitel 5.1.3). Im folgenden wird daher $Modulus = 8$ und $Windowsize = 4$ gesetzt.

Klasse	Bezeichnung	Bedeutung
Timer	Acknowledgment	Warten auf Quittung
	Reject	Warten auf neuangeforderten Rahmen
	Poll	Warten auf Antwort auf ein pollen
	Delay	Restlebenszeitüberwachung im Sender
	Forward	Restlebenszeitüberwachung im Empfänger
	Ignore	Ausnutzung von τ_{loop}
Fenster	Modulus	Feldgröße im PDU Rahmen für SN, RN
	Windowsize	maximale Fenstergröße
Kontrolle	N2	Anzahl der Übertragungswiederholungen
Prioritäten-algorithmus	Send_Prio	Algorithmus zur Ermittlung des sendende VC
	Receive_Prio	Algorithmus zur Ermittlung des quittierenden VC
	Ack_Threshold	Prioritätenschwelle

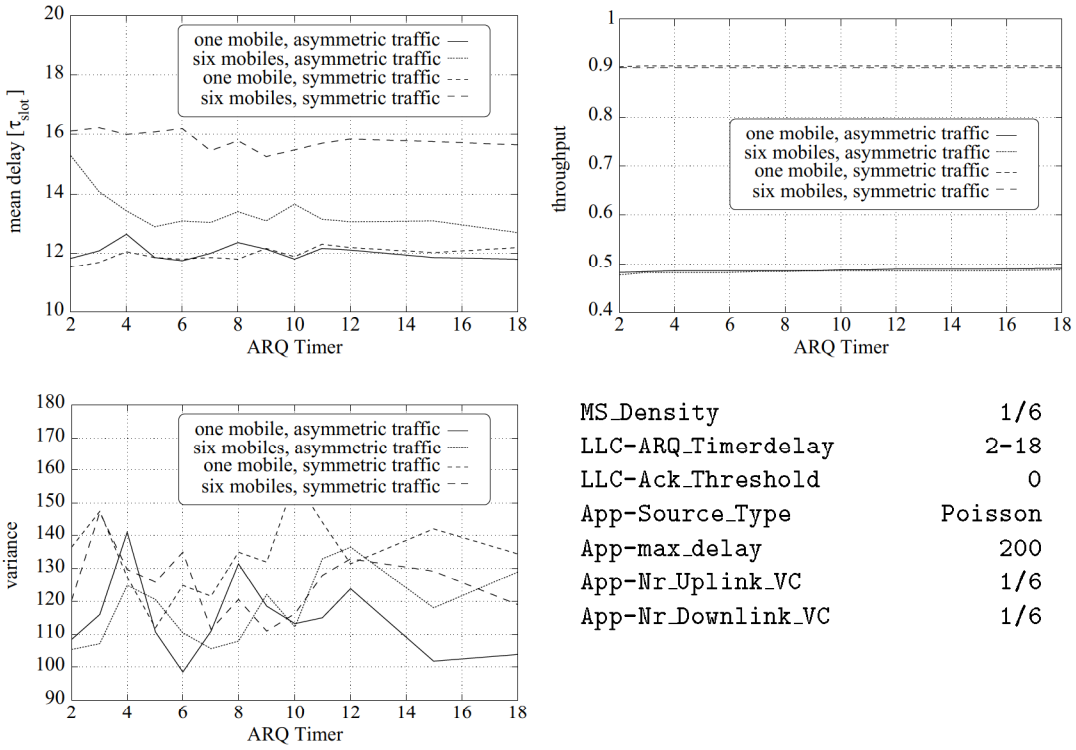
Tabelle 8.1: Parameter des ASR ARQ im Überblick

N2 wird immer so gesetzt, daß es zu keinen Resets kommen sollte. Kommt es dennoch zum Reset, so ist es sehr wahrscheinlich, daß ein Fehler im Protokoll zu einer Verklemmung geführt hat (Beispiele siehe Kapitel 5.2.4). Für *Send_Prio* wurde ausschließlich `Short_Remain` verwendet (vgl. Kapitel 6). Als variable Simulationsparameter bleiben für das ASR ARQ somit die Timer, *Receive_Prio* und damit zusammenhängend *Ack_Threshold*.

Im folgenden soll untersucht werden, wie diese Parameter voneinander und von der Fenstergröße abhängen.

8.2 Optimierung der Parameter *ARQ Timer Delay* und *Ack_Threshold*

8.2.1 ARQ Timer Delay



MS_Density	1/6
LLC-ARQ_Timerdelay	2-18
LLC-Ack_Threshold	0
App-Source_Type	Poisson
App-max_delay	200
App-Nr_Uplink_VC	1/6
App-Nr_Downlink_VC	1/6

Abbildung 8.1: Mittlere Verzögerung, Varianz und Durchsatz bei Variation von ARQ Timer Delay

Zunächst soll untersucht werden, welchen Einfluß der Parameter *ARQ Timer Delay* in verschiedenen Szenarios hat. Dazu werden alle anderen Timer ausgeschaltet und mit der *oldest_first* Quittungsstrategie ohne *Ack_Threshold* simuliert. Die gesamte Verkehrslast beträgt in allen Simulationen 90%, d.h. der Kanal ist bei einer unkorrelierten Rahmenfehlerrate von 5% hoch ausgelastet. Es werden vier verschiedene Szenarios betrachtet:

one mobile, asymmetric traffic In einer Mobilstation werden sechs VCs mit jeweils 15% Last auf dem Uplink betrieben.

six mobiles, asymmetric traffic In sechs Mobilstationen werden jeweils ein VC mit 15% Last auf dem Uplink betrieben.

one mobile, symmetric traffic In einer Mobilstation werden sechs VCs mit jeweils 15% Last auf dem Up- und Downlink betrieben.

six mobiles, symmetric traffic In sechs Mobilstationen werden jeweils ein VC mit 15% Last auf dem Up- und Downlink betrieben.

Die Diagramme in Abbildung 8.1 lassen nur den Schluß zu, daß der Parameter *ARQ Timer Delay* keinen entscheidenden Einfluß auf die Simulationen hat, bzw. daß sich kein System erkennen läßt, wie dieses Delay am günstigsten gewählt werden sollte. Es zeigt sich nur, daß ein zu kurz gewähltes Delay in manchen Szenarios zu einem zusätzlichen Verkehr führt.

Offensichtlich kommt das Pollen aufgrund des Ablaufs des *ARQ Timers* so selten, bzw. wird das zusätzliche Pollen so oft huckepack übertragen, daß es keinen entscheidenden Einfluß hat. Dennoch will ich eine aus der Erfahrung gewonnene Regel zur Dimensionierung des *ARQ Timer Delay* angeben:

$$ARQ\ Timer\ Delay = 2 \cdot \frac{0.9}{mean\ rate} \tag{8.1}$$

8.2.2 Ack_Threshold

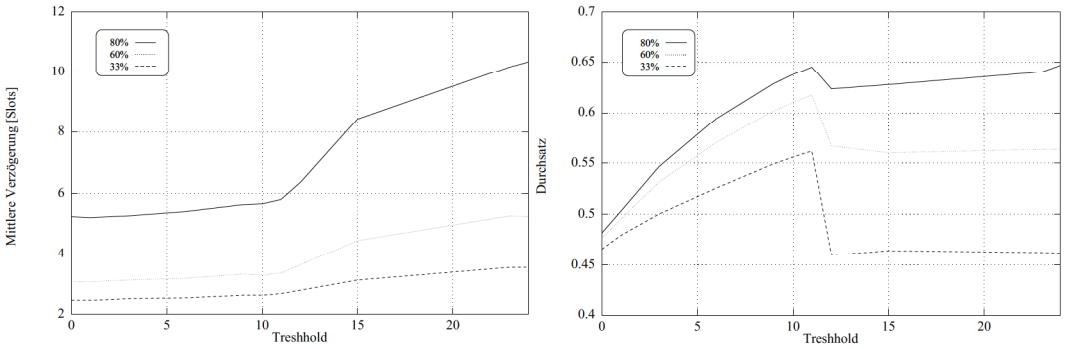


Abbildung 8.2: Mittlere Verzögerung und Durchsatz bei einseitigem Verkehrsaufkommen

Ziel der Simulation ist die Untersuchung der Kosten für eine Bündelung von Quittungen. Dadurch soll der Durchsatz erhöht werden, ohne die Verzögerung zu stark zu verschlechtern. Außerdem soll die Abhängigkeit des `Ack_Threshold` vom `ARQ Timer Delay` untersucht werden.

`MS_Density`	6
`LLC-ARQ_Timerdelay`	12
`LLC-Ack_Threshold`	0-24
`App-Source_Type`	Poisson
`App-max_delay`	200
`App-Nr_Uplink_VC`	1
`App-Nr_Downlink_VC`	0

Tabelle 8.2: Simulationsparameter

Wie Tabelle 8.2 zeigt, wird hierzu ein Szenario mit sechs Mobilstationen mit je einem VC betrachtet. Nutzdaten werden ausschließlich auf dem Uplink übertragen, so daß der Downlink für Quittungen frei ist. Um eine Vergleichbarkeit zu erreichen, wird *ARQ Timer Delay* = 12 fest vorgegeben und das `Acknowledge Threshold` im Bereich von 0-24 variiert. Das Gesamtangebot beträgt 33%, 60% und 80% und wird gleichmäßig auf die Stationen aufgeteilt. Als Lastgeneratoren fungieren Poisson Quellen.

Aus Abbildung 8.2 wird ersichtlich, daß bis zu einer Schwelle (*engl. threshold*) von 11 die mittlere Verzögerung nur leicht ansteigt, der Durchsatz sich aber erheblich erhöht. Bei einer Schwelle von 12 bricht der Durchsatz ein und steigt auch nur noch im Fall hoher Auslastung an. Die mittlere Verzögerung steigt nun merklich an.

Bei einem `Acknowledge Threshold` ≥ 12 erhalten die Sender die Quittungen nicht rechtzeitig und pollen den Empfänger. Dieses Pollen kann bei hohem Verkehrsaufkommen wesentlich öfter huckepack übertragen werden, als bei geringem. Deshalb ist der Einbruch des Durchsatzes bei geringer Last wesentlich größer als bei hoher. Bei geringer Last wird dann auch der Gewinn durch Bündelung der Quittungen komplett durch das Pollen vernichtet.

Die mittlere Verzögerung steigt bei hoher Last am stärksten an, weil hier das Pollen mit den Nutzdaten am stärksten um den Kanal konkuriert und den Nutzdaten Kapazität wegnimmt. Außerdem steigt mit wachsendem `Acknowledge Threshold` die Wahrscheinlichkeit, daß sich das Sendefenster schließt, und es somit zu weiteren Verzögerungen kommen kann.

Für das `Acknowledge Threshold` gilt somit bei einseitigem Verkehrsaufkommen:

$$\texttt{Acknowledge Threshold} = \texttt{ARQ Timer Delay} - 1 \tag{8.2}$$

Eine weitere Steigerung des Durchsatzes erreicht man durch ein größeres `ARQ Timer Delay`, auf Kosten der mittleren Verzögerung, oder durch ein größeres Fenster, auf Kosten des Overheads.

Es wird nun dieselbe Simulation mit beidseitigem Verkehrsaufkommen betrachtet, d.h. es wird mit den Parametern aus Tabelle 8.2 und mit der gleichen Last auf dem Up- und Downlink simuliert.

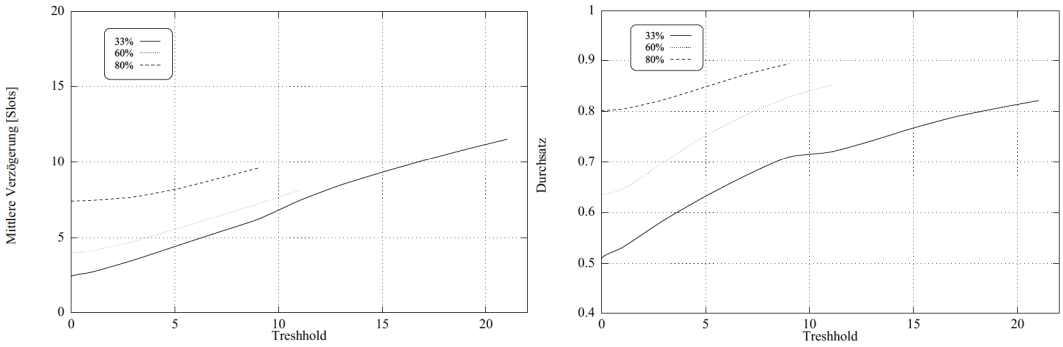


Abbildung 8.3: Mittlere Verzögerung und Durchsatz bei beidseitigem Verkehrsaufkommen

In den Diagrammen aus Abbildung 8.3 ist zu erkennen, daß mit zunehmenden *Threshold* die mittlere Verzögerung zunächst abnimmt. In Abhängigkeit von der Gesamtauslastung steigt sie dann spätestens bei einem *Threshold* von 24 wieder an. Der Durchsatz steigt dagegen kontinuierlich an, zuerst sehr stark, später nur noch leicht.

Im Unterschied zum einseitigen Verkehr wirkt *Threshold* jetzt auch auf das Pollen, das nun ebenfalls verzögert wird. Dadurch treten die negativen Einflüsse des Pollens verspätet und stark abgeschwächt auf, da viele Pollanforderungen huckepack übertragen werden können. Die Verringerung der mittleren Verzögerung durch das Sammeln von Quittungen scheint auf den ersten Blick widersprüchlich zu sein, doch zum einen nehmen reine Quittungen dann nur selten den ATM Zellen Kanalkapazität weg, zum anderen kann eine verzögerte Quittung mit Hilfe des *Ignore Timers* vorteilhaft ausgewertet werden (Kapitel 5.2.2). Bei beidseitigem Verkehrsaufkommen bietet sich also folgende Dimensionierungsregel an:

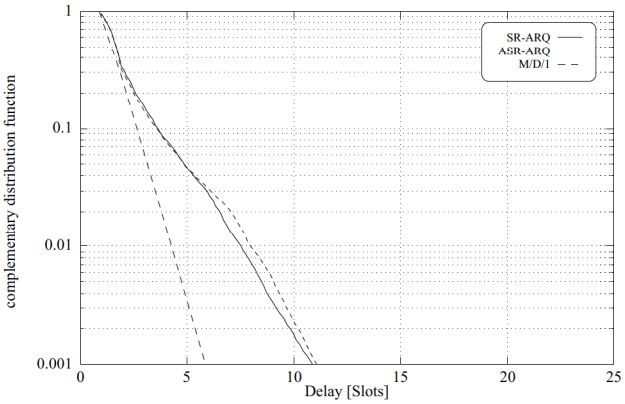
$$\texttt{Acknowledge Threshold} \approx 2 \cdot \texttt{ARQ Timer Delay} \tag{8.3}$$

Abschließend möchte ich anhand des Durchsatzes bei 33% Kanalauslastung auf die Bedeutung des *Threshold* hinweisen. Durch die Steigerung des Durchsatzes von 52% auf 80% erreicht man eine Einsparung an Sendeleistung von $1 - \frac{52\%}{80\%} = 35\%$!

8.3 Untersuchung der Ursachen für Zellverzögerungen

8.3.1 Einfluß des *Ignore Timers*

Um den Einfluß des *Ignore Timers* auf die Zellverzögerung zu verstehen, wurde das Protokoll mit und ohne *Ignore Timer* in sehr einfachen Szenarios untereinander und mit einem idealisierten ATM Multiplexer verglichen. In diesen einfachen Szenarios betreibt eine MS einen VC mit einem Angebot von 35%, 70% und 90% auf dem Uplink unter Verwendung einer Poisson Quelle. Die maximale Verzögerung beträgt 200 τ_{slot} und das *ARQ Timer Delay* ist auf 2 τ_{slot} (minimaler Wert) gesetzt.



Ignore Timer	Mean	Variance
YES	2.09	1.98
NO	2.10	1.83
ATM Mux	1.38	0.48

Abbildung 8.4: Zellverzögerung bei 35% Angebot

Bei einem Angebot von nur 35% verhalten sich beide Varianten sehr ähnlich. Während mit *Ignore Timer* eine kleinere mittlere Verzögerung auftritt, ist die Varianz der Verzögerung ohne *Ignore Timer* geringer (Tabelle in Abbildung 8.4). Die unterschiedliche Varianz zeigt sich im Diagramm in Abbildung 8.4 im Auseinanderlaufen der Kurven bei mehr als 6 τ_{slot} Zellverzögerung.

Die Abweichung der ARQ-Protokolle vom ATM Multiplexer ist protokollbedingt, bzw. liegt in der Simulation des ATM Multiplexers. Im ATM Multiplexer wird die Paketfehlerrate dadurch simuliert, daß in Höhe der Fehlerrate keine Aufträge bedient werden. Die Verzögerung beträgt hierbei 1 τ_{slot}. Beim ARQ-Protokoll wird der Verlust eines Rahmens hingegen erst dann festgestellt, wenn ein weiterer Rahmen erfolgreich empfangen wird. Zusätzlich muß dann noch der Empfänger den Sender über den Verlust informieren. Die Verzögerung beträgt also mindestens 3 τ_{slot}. Da in der Zwischenzeit weitere Rahmen übertragen werden können, kommt es zu keinem Sprung in den dargestellten Kurven.

Bei einem Angebot von 70% zeigen sich schon deutliche Unterschiede in den Protokollvarianten. Während sich die Leistungsfähigkeit im Verhältnis zum ATM Multiplexer ohne Anwendung des *Ignore Timers* leicht verschlechtert, verbessert sich das Verhalten des Protokolls mit diesem. Drei Effekte spielen hierbei eine Rolle:

1. Durch das erhöhte Angebot ist es den ARQ Protokollen häufiger möglich, während der Behandlung eines Rahmenverlustes weitere Rahmen erfolgreich zu übertragen. Der Verlust eines Rahmens wirkt sich somit nicht so stark aus wie bei nur 35% Angebot. Aus diesem Grund knicken die Kurven in Abbildung 8.5 nicht ab.
2. Das schlechtere Verhalten des Protokolls ohne *Ignore Timer* resultiert daraus, daß es bei Verlust eines Rahmens stottert. Stottern bedeutet, daß durch wiederholte

Ignore Timer	Mean	Variance
YES	3.47	6.18
NO	3.83	7.98
ATM Mux	2.59	3.80

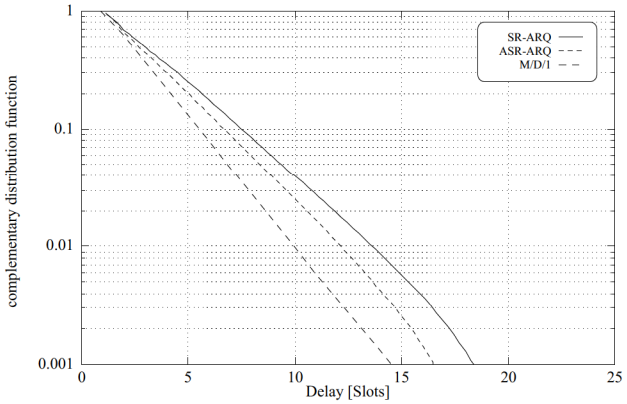


Abbildung 8.5: Zellverzögerung bei 70% Angebot

Anforderung eines Rahmens durch den Empfänger mittels SREJ, der Sender diesen Rahmen mehrfach (in diesem Fall zweifach) sendet, obwohl ein erneuter Verlust sehr unwahrscheinlich ist. Aus diesem Grunde ist insbesondere ohne Anwendung des *Ignore Timer* ein höheres *ARQ Timer Delay* zu wählen. An der Rate der Mehrfachübertragungen zeigt sich das Stottern. Während mit *Ignore Timer* eine Rate von weniger als 0.2% auftritt, ist diese ohne *Ignore Timer* mit 3,9% im Bereich der Paketfehlerrate. In 80% der Fälle kommt es somit zum Stottern.

3. Mit Hilfe des *Ignore Timer* kann aus positiven Quittungen Informationen über verlorengegangene Rahmen ableiten werden. Dieses erlaubt eine schnellere Reaktion auf Verluste. Erkennbar wird dieser Effekt beim LLC-Overhead. Während ohne *Ignore Timer* ein Overhead von 13,4% entsteht, liegt er mit *Ignore Timer* mit 6,1% nur wenig über der Paketfehlerrate von 5%.

Die Tabelle in Abbildung 8.5 zeigt die Auswirkung der beschriebenen Verhaltensweisen auf die mittlere Verzögerung und die Varianz der Verzögerung.

Bei einem Angebot von 90% bewegt sich das ARQ Protokoll ohne *Ignore Timer* bereits am Rande seiner Leistungsfähigkeit. In dem simulierten Szenario kommt es schon zu geringen Verlusten, d.h. einzelne Zellen haben die maximale Verzögerung überschritten und wurden verworfen. Die Limitierung der Verzögerung auf maximal 200 τ_{slot} erklärt auch das Abknicken der Kurve in Abbildung 8.6. Eine Leistungssteigerung ohne *Ignore Timer* kann nur noch über ein größeres Fenster erreicht werden.

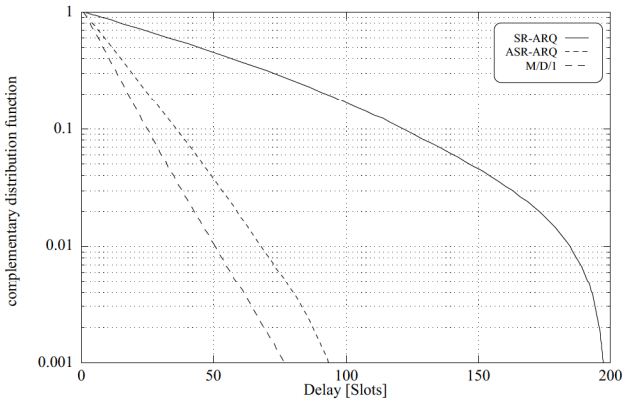


Abbildung 8.6: Zellverzögerung bei 90% Angebot

Mit *Ignore Timer* verhält sich das ARQ Protokoll nur wenig schlechter als der ATM Multiplexer. In der Tabelle 8.3 werden die Gründe für die unterschiedliche Leistungsfähigkeit deutlich.

Unter Verwendung des *Ignore Timer* hat das ARQ Protokoll eine um den Faktor 10–

Ignore Timer	Mean Delay	Variance	Multitransmission	Overhead
YES	16.22	227.16	0,2%	5,4%
NO	55.66	1998.28	3,9%	8,9%
ATM Mux	11.03	112.79	-	-

Tabelle 8.3: Mittlere Verzögerung, Varianz, Mehrfachübertragungsrate und Overhead bei 90% Angebot

20 kleinere Mehrfachübertragungsrate und dadurch einen deutlich geringeren Protokoll Overhead. Dieser Overhead setzt sich aus drei Anteilen zusammen:

1. Mehrfachübertragungen in Höhe der Paketfehlerrate. Diese sind unbedingt notwendig.
2. Mehrfachübertragungen aufgrund des Protokollverhaltens. Diese sind überflüssig und belasten den Kanal.
3. Übertragung von reinen Supervisory Rahmen ohne Nutzdatenanteil. Diese sind notwendig, um bei geschlossenem Sendefenster den Empfänger pollen zu können. Sie stellen ebenfalls eine Belastung des Kanals dar, die vermieden werden sollte.

Abschließend soll die Leistungsfähigkeit des *Ignore Timers* (τ_{loop} *Timer*) in einem realistischeren Szenario demonstriert werden.

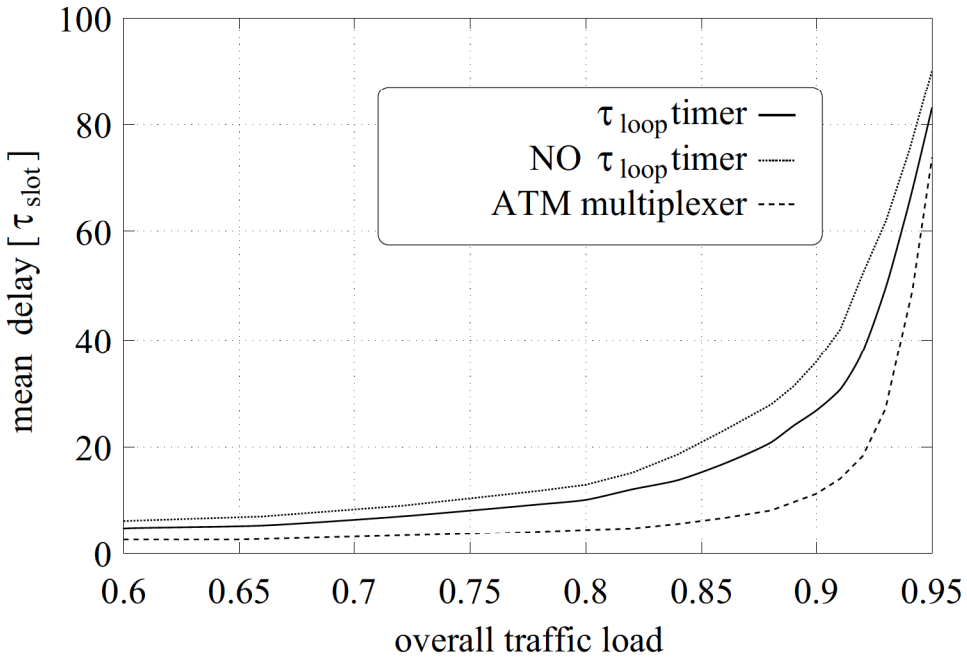


Abbildung 8.7: Mittlere Verzögerung über Gesamtangebot bei Verwendung des *Ignore Timers*

Es wird jetzt ein Szenario mit 6 MS verwendet, welche jeweils einen VC mit symmetrischer Last betreiben. Die Last wird durch Poissonquellen generiert. Im Diagramm der Abbildung 8.7 sind die Übertragungsverzögerungen über das Angebot mit und ohne Berücksichtigung vom *Ignore Timer* dargestellt. Als untere Grenze ist der idealisierte ATM Multiplexer

eingetragen. Es ist zu erkennen, daß sich unter Verwendung des *Ignore Timers* die mittlere Verzögerung spürbar verringert.

8.3.2 Einfluß des Fehlermodells

Die Übertragungsverzögerung der ATM Zellen hat zwei Ursachen:

1. Einzelne ATM-Zellen werden aufgrund von mehrfacher Übertragung stark verzögert.
2. Alle ATM-Zellen werden dadurch verzögert, daß durch die mehrfachfache Übertragung einzelner fehlerhafter Zellen eine zusätzlich Last entsteht.

Im eingesetzten Prioritätenalgorithmus werden Übertragungswiederholungen von ATM-Zellen deshalb bevorzugt behandelt, weil sie sich näher an ihrer Deadline befinden als Neuübertragungen. Aus diesem Grunde hat die zweite Ursache den entscheidenden Einfluß auf die Zellverzögerung. Um dieses Verhalten simulativ zu demonstrieren, wird ein Szenario mit fünf Mobilstationen mit je einem VC mit Verkehr ausschließlich auf dem Uplink betrachtet. Erzeugt wird die Last von Videoquellen mit einer mittleren Datenrate von 14% der Kanalkapazität (das entspricht 5Mbit/s bei einem 34Mbit/s Kanal). Das Gesamtangebot beträgt somit 70% der Kanalkapazität. Es werden drei verschiedene Fehlermodelle betrachtet:

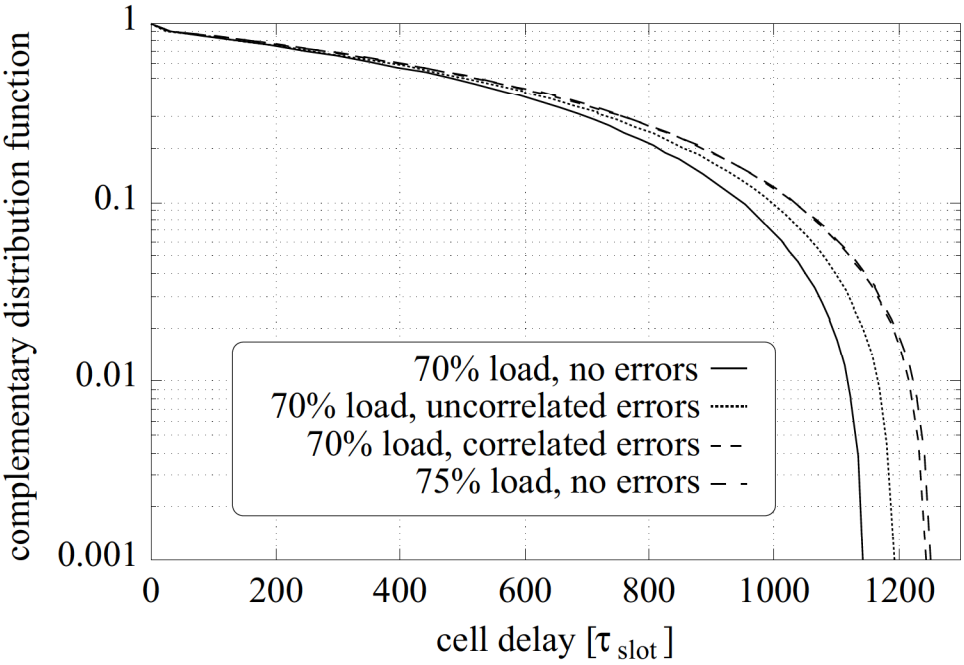


Abbildung 8.8: Zellverzögerung bei unterschiedlichen Fehlermodellen

- keine Bitfehler
- unkorrelierte Bitfehler mit einer Rate von $10^{-4} \equiv 5\%$ Rahmenfehlerrate
- korrelierte Bitfehler nach dem Gilbert Modell (Kapitel 7.4), mit einer Rate von 10^{-3} im Zustand **Error**, das entspricht einer mittleren Rahmenfehlerrate von 5%

Abbildung 8.8 zeigt, daß die Verzögerung weniger als 5% bei unkorreliertem Fehler und weniger als 10% bei korreliertem Fehler ansteigt. Vergleicht man diese Kurven mit einer, bei der im fehlerfreien Modell die Gesamtlast um 5% erhöht wird, so sieht man, daß die Verzögerung ziemlich genau derjenigen entspricht, die bei korreliertem Fehler aufgetreten ist. Dieses Ergebnis war deshalb zu erwarten, weil die zussätzliche Last durch die Videoquellen ebenso korreliert ist wie die zusätzliche Last, die durch den korrelierten Fehler hervorgerufen wird.

8.4 Untersuchung des Aufwandes zum Verwerfen von Zellen

Um die maximale Zellverzögerung der ATM-Zellen exakt einhalten zu können, muß ein erheblicher Aufwand betrieben werden, um Zellen verwerfen zu können, die ihre Restlebenszeit überschritten haben. Im einzelnen sind das:

1. Einsatz eines *Delay* Timers zur Überwachung der Restlebenszeit im Sender
2. Einsatz eines *Forward* Timers zur Überwachung der Restlebenszeit im Empfänger. Nach Ablauf dieses Timers wird die entsprechende Zelle an den oberen Layer weitergegeben, ohne weiter auf fehlende Zellen zu warten.
3. Einsatz von *Delay* PDUs, um den Empfänger über das Verwerfen von Zellen zu informieren.
4. Verwendung des *delay time* Feldes (Abbildung 5.6) zum Übertragen der Restlebenszeit.

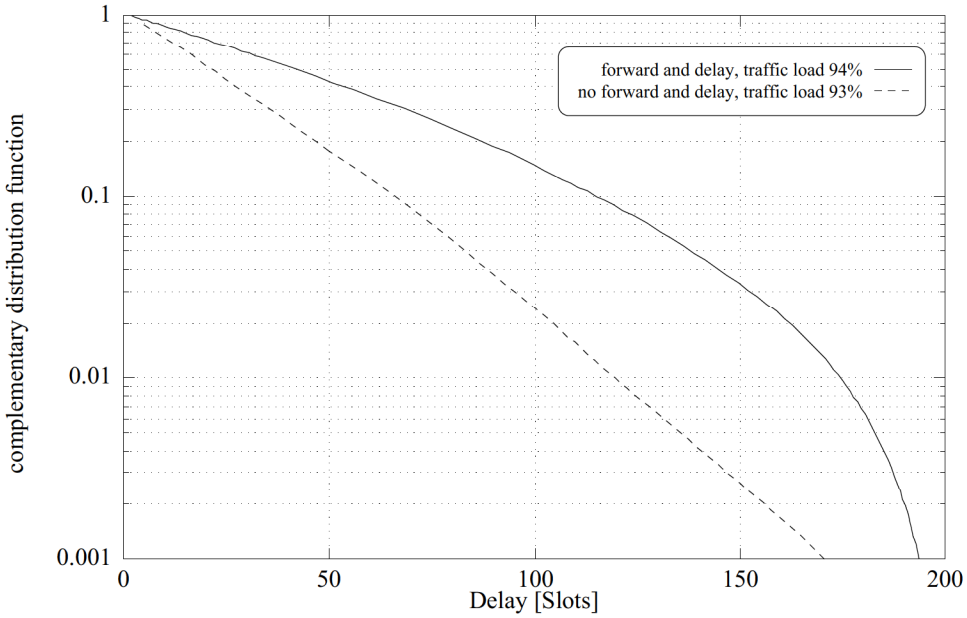


Abbildung 8.9: Benachrichtigung des Empfängers über verspätete Zellen

Eine vereinfachte Vorgehensweise würde folgendermaßen aussehen:

1. Vor der Übernahme von Zellen in das Sendefenster wird die Restlebenszeit überprüft. Besteht keine Chance die Zelle innerhalb ihrer Restlebenszeit zu übertragen, wird diese verworfen.

2. Läuft die Restlebenszeit einer Zelle im Sendefenster ab, so wird diese nicht verworfen und trotzdem übertragen. Da das Sendefenster beschränkt ist (4-8 Plätze), sind auch dem Überschreiten des Delay enge Grenzen gesetzt.

Zunächst wird untersucht, ob eine Kodierung der Restlebenszeit in 6-8 Bit sinnvoll erscheint, oder ob der zusätzliche Overhead zu groß ist. Dazu wird ein Szenario mit einer MS mit einem VC mit Nutzdaten ausschließlich auf dem Uplink betrachtet. Das Angebot beträgt 94% der Kanalkapazität. Eine Kodierung in 6 Bit entspricht einem Overhead von 1%. Es wird nun der zusätzliche Overhead derart berücksichtigt, daß beim Protokoll ohne Restlebenszeitübertragung das Angebot um 1% verringert wird.

In Abbildung 8.9 erkennt man sofort, daß der zusätzliche Overhead zu einem deutlich schlechteren Ergebnis führt. Zumindest in diesem nicht sehr realistischen Szenario erscheint die Übertragung der Restlebenszeit nicht sinnvoll.

In einem realistischerem Szenario sollen nun die in Kapitel 5.2.3 vorgestellten Methoden zur Behandlung von zeitkritischen ATM-Zellen untersucht werden. Diese sind:

1. Keine Beachtung der maximalen Verzögerung
2. Verwerfen vor der ersten Übertragung
3. Verwerfen von allen verspäteten Zellen und Einsatz der Delay PDU
4. Vorzeitiges Abbrechen des Resequencing im Empfänger

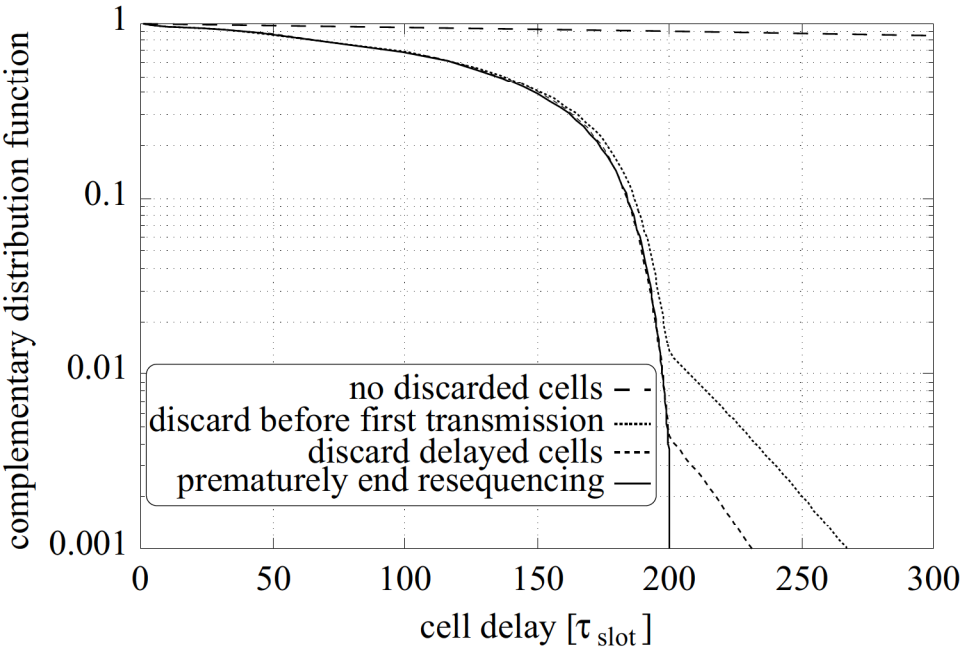


Abbildung 8.10: Zellverzögerung bei verschiedenen Strategien zum Verwerfen von Zellen

Hierzu wird ein Szenario mit sechs Mobilstationen mit je einem VC betrachtet. Jeder VC betreibt je eine Poissonquelle für den Up- und Downlink. Das Gesamtangebot beträgt 95% der Kanalkapazität, was bei einer mittleren Rahmenfehlerrate von 5% einen Überlastfall darstellt. Diese Simulation kann als kurzfristige Überlastsituation in einem sonst mit einem geringeren Gesamtangebot betriebenen System interpretiert werden.

procedures	label in Figure 8.10	% discarded cells	% lost cell
-	no discarded cells	0	90.1
1	discard before first transmission	1.85	3.07
1-2	discard delayed cells	2.31	2.72
1-3	prematurely end resequencing	1.76	1.76

Tabelle 8.4: Prozentsatz von verworfenen und verlorenen Zellen (vgl. Abbildung 8.10)

Die Kurven in Abbildung 8.10 gehören zu den in Tabelle 8.4 aufgeführten Strategien. Zusätzlich sind in dieser Tabelle die *discard ratio* und die *CLR* aufgetragen.

Werden keine ATM-Zellen verworfen, so überschreiten 90% der Zellen die maximale Verzögerung von 200 τ_{slot}. Je mehr Zellen verworfen werden, desto geringer ist die Zellverzögerung der erfolgreich übertragenen Zellen. Wenn ATM-Zellen die maximale Verzögerung überschreiten, werden sie als verloren gezählt. In Tabelle 8.4 erkennt man, daß mit zunehmender Komplexität des Protokolls die Zellverlustrate abnimmt.

Bei Verwendung der Methoden 1-3 hat der Empfänger eine genauere Kenntnis über die Notwendigkeit von Quittungen, deshalb werden in diesem Fall weniger Zellen verworfen. Sobald er das Resequencing abbricht, versendet der Empfänger eine Quittung und macht somit, im Gegensatz zur Verwendung nur der Methoden 1-2, die Übertragung von Delay PDUs häufig überflüssig.

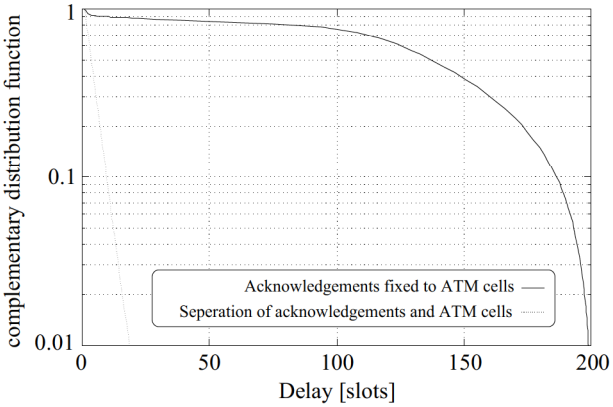
Die Steilheit der Äste der Kurven beim Überschreiten der maximalen Verzögerung hängen davon ab, welche Priorität verspätete Zellen bzw. Delay PDUs erhalten. Es handelt sich um Geraden, weil in diesem Bereich keine Zellen verworfen werden. Bei der Verwendung der Methoden 1-2 befinden sich in diesem Ast der Kurve diejenigen Zellen, die im Empfänger auf verspätete Zellen warten. Die Steilheit der Kurve hängt hier ausschließlich von der Priorität der Delay PDUs ab, die in diesem Szenario sehr hoch gewählt wurde. Bezahlt wird diese hohe Priorität mit einer erhöhten *discard ratio*, weil Delay PDUs den Nutzdaten unter Umständen Kapazität wegnehmen. Bei Verwendung der Methode 1 hängt die Steilheit des Astes ausschließlich von der Priorität von verspäteten Zellen ab, die beim eingesetzten Scheduler sehr hoch ist, da allein anhand der Deadline die Priorität berechnet wird.

Gelingt es, die Restlebenszeit der Zellen in wenigen oder sogar nur einem Bit zu kodieren, so wäre die Anwendung aller drei Methoden vorteilhaft. Dazu müßten entsprechende Untersuchungen gemacht werden.

8.5 Trennung von Quittungen und Nutzdaten

In dieser Simulation soll der Einfluß des Auftrennens von Quittungen und Nutzdaten gezeigt werden. Dazu wird ein Szenario mit einer Mobilstation mit vier virtuellen Kanälen betrachtet. Jeder dieser Kanäle ist bidirektional und wird beidseitig von Poisson Quellen gespeist. Zwei dieser Kanäle haben eine mittlere Datenrate von 30% auf dem Uplink und 10% auf dem Downlink, die anderen beiden Kanäle haben ein genau entgegengesetztes Verkehrsaufkommen. In einem Fall können Nutzdaten nur Quittungen vom selben Kanal transportieren, im anderen von einem beliebigen Kanal.

Abbildung 8.11 zeigt sehr deutlich, daß im Fall der Bindung von Quittungen an den selben Kanal wie Nutzdaten das Protokoll nicht mehr in der Lage ist, die Deadline einzuhalten.



MS_Density	1
LLC-ARQ_Timerdelay	12
LLC-Ack_Threshold	0
App-Source_Type	Poisson
App-max_delay	200
App-Mean_Rate	0.1/0.3
App-Nr_Uplink_VC	4
App-Nr_Downlink_VC	4

Abbildung 8.11: Zellverzögerung beim Verbinden bzw. Trennen von Quittungen und Nutzdaten

Es kommt zu deutlichen Verlusten ($> 10\%$), obwohl Uplink wie auch Downlink nur mit insgesamt 80% belastet sind.

Kann bei der Übertragung von ATM Zellen der Quittungskanal frei gewählt werden, so werden immer die dringlichsten Quittungen versendet. Dies führt zu einem problemlosen Protokollablauf mit geringen Verzögerungen.

Nur bei beidseitig symmetrischen Angebot können mehrere parallele VCs in einer Mobilstation bei hoher Kanalauslastung betrieben werden, ohne Quittungen von Nutzdaten zu trennen. Diese Bedingung schränkt aber das Dienstangebot erheblich ein.

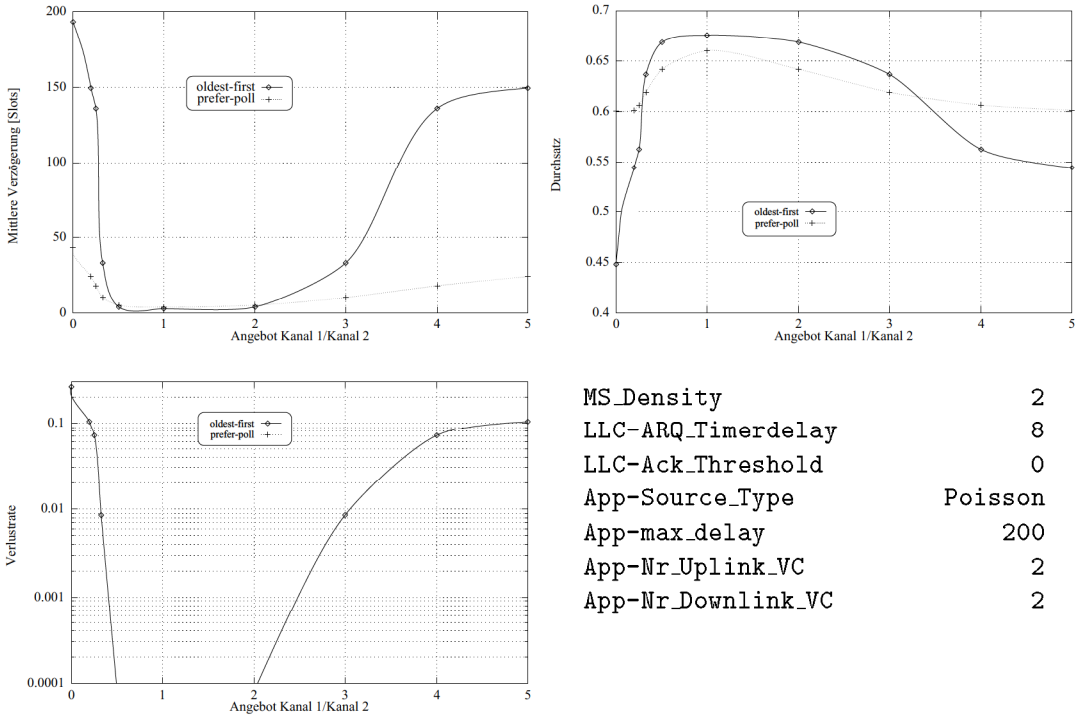
8.6 Asymmetrisches Verkehrsaufkommen

Es wird nun der Einfluß der Prioritätenalgorithmen für Quittungen (`ARQ Receive Priority`) auf die mittlere Verzögerung, den Durchsatz und die Verlustrate untersucht. Es wird ein Szenario mit zwei Mobilstationen betrachtet. Jede Mobilstation betreibt zwei virtuelle Kanäle, wobei jeweils ein Kanal auf dem Uplink und einer auf dem Downlink Nutzdatenaufkommen hat. Insgesamt beträgt dieses Aufkommen 60% unter Verwendung von Poisson Quellen. Variiert wird die Verteilung dieses Gesamtangebotes auf Up- und Downlink im Bereich von 0–5. Dabei werden die beiden Stationen gegenläufig in der Art belastet, daß Station 1 auf dem Uplink die Hauptlast hat, wenn Station 2 auf dem Downlink den Hauptverkehr trägt.

In diesem Szenario ergibt sich folgende Problematik für das Quittierungsverfahren:

1. Für den hochlastigen Kanal steht kein ausreichendes Angebot auf dem Rückkanal zur Verfügung, um Quittungen huckepack übertragen zu können.
2. Reine Quittungen (ohne Nutzdatenanteil) konkurrieren mit den Nutzdaten des anderen hochlastigen Kanals um das Medium.
3. Wird der hochlastige Kanal nicht rechtzeitig quittiert, so schließt sich sein Sendefenster und er fängt an zu Pollen; er belastet damit zusätzlich den Kanal.

Die beiden untersuchten Prioritätsalgorithmen unterscheiden sich in der Methode zur Erhöhung der Priorität der Quittungen:



MS_Density	2
LLC-ARQ_Timerdelay	8
LLC-Ack_Threshold	0
App-Source_Type	Poisson
App-max_delay	200
App-Nr_Uplink_VC	2
App-Nr_Downlink_VC	2

Abbildung 8.12: Mittlere Verzögerung, Durchsatz und Verlustrate bei asymmetrischem Verkehrsaufkommen

oldest first Dieser Algorithmus erhöht die Priorität hauptsächlich nach der Wartezeit der Quittungen.

prefer_poll Antworten auf Pollen und SREJ Quittungen erhalten sofort eine hohe Priorität. RR Quittungen erhöhen ihre Priorität mit der Anzahl der zu quittierenden Rahmen sehr schnell.

Im Mittel haben die Quittungen bei **prefer_poll** eine wesentlich höhere Priorität als bei **oldest_first** und können sich dadurch häufiger gegen Nutzdaten durchsetzen.

Abbildung 8.12 zeigt, daß sich **oldest_first** bei geringer Asymmetrie der Kanäle sowohl in bezug auf Verzögerung, als auch auf Durchsatz günstiger verhält als **prefer_poll**. Es entstehen keine Verluste. Bei größerer Asymmetrie ist lediglich **prefer_poll** in der Lage, den minimalen Durchsatz von 60% zu garantieren, damit es zu keinen Verlusten kommt. Mit dem Algorithmus **oldest_first** bricht in dieser Einstellung das Protokoll ein; es kommt zu erheblichen Verlusten.

Anhand der Ergebnisse für absolut asymmetrisches Angebot (der zweite Kanal hat hier keine Nutzlast) sollen die Ursachen für den Einbruch erläutert werden. Bei einer Last von 60% und einer Fehlerrate von 5% bleiben 35% Kapazität, um reine Quittungen zu übertragen, d.h. daß bei voller Ausnutzung dieser Kapazität im Mittel jeder zweite Rahmen quittiert werden könnte. Bei einer Fenstergröße von vier sollte damit ein stabiler Ablauf möglich sein.

Tabelle 8.5 zeigt, daß die Nutzdaten mit einer großen Menge Quittungen konkurrieren. Allein 4.7% der Kapazität wird durch reines Polling verbraucht. Die geringe mittlere Wartezeit der reinen Quittungen zeigt, daß diese übertragen werden, wenn keine Nutzdaten übertragen werden können. Da andererseits die mittlere Verzögerung der ATM-Zellen im

	Station 1		Station 2	
	Uplink	Downlink	Uplink	Downlink
Angebot	60%	–	–	60%
genutzte Kapazität	56.6%	36.5%	36.5%	56.5%
Nutzdaten	51.8%			51.8%
reine Quittungen	4.7%	36.5%	36.5%	4.7%
mittlere Wartezeit reiner Quittungen	1.05	1.57	1.57	1.05

Tabelle 8.5: Meßergebnisse für **oldest_first**

Vergleich sehr hoch ist, tritt diese Situation immer dann auf, wenn das Sendefenster geschlossen ist. Der Einbruch des Protokolls ist also dadurch zu erklären, daß die Kanäle gegenseitig die Übertragung von Quittungen bei geschlossenem Fenster blockieren.

Solange kein Fehler auftritt, schließen sich beide Sendefenster gleichzeitig und ermöglichen das Versenden von Quittungen. Tritt dagegen ein Fehler auf, so wird ein Fenster später geschlossen und dieser Kanal erhält erst dann eine Quittung, wenn sich das andere Fenster schließt oder die Quittung lange genug gewartet hat. Das Sendefenster schließt sich in diesem Fall aber nur sehr selten, da der Rückkanal nun für Quittungen frei ist.

Es soll nun untersucht werden, ob eine Bündelung der Quittungen nach Gleichung 8.2 (Kapitel 8.2.2) zusammen mit einem optimierten `ARQ Timer Delay` nach Gleichung 8.1 (Kapitel 8.2.1) zu einer Verbesserung des Ergebnisses führt.

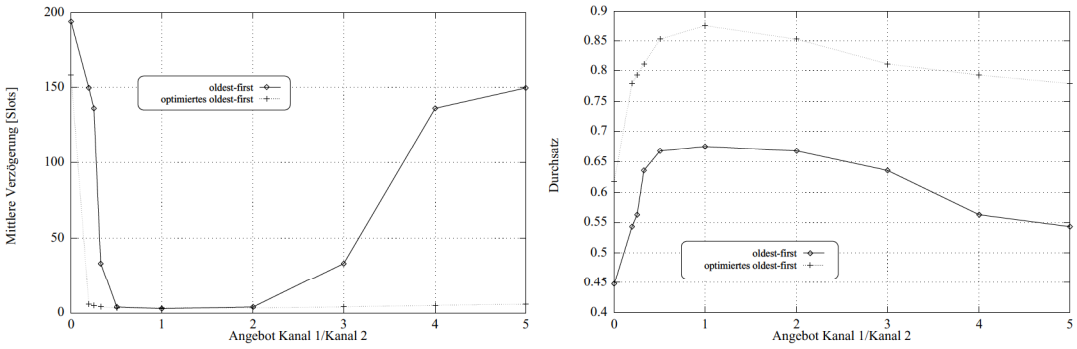


Abbildung 8.13: Mittlere Verzögerung und Durchsatz des optimierten **oldest_first**

Abbildung 8.13 zeigt, daß die Optimierung der Parameter zu einem erheblich günstigeren Verhalten des Protokolls führt. Die mittlere Verzögerung bleibt auch bei zunehmender Asymmetrie gering und der Durchsatz erhöht sich im gesamten Bereich erheblich, im Mittel um 20% absolut.

Trotz eines Durchsatzes von über 60% kommt es bei totaler Asymmetrie zu erheblichen Verlusten (13%), das Protokoll bricht ein. Die Gründe für diesen Einbruch sind aus Tabelle 8.6 ersichtlich. 11% der Kapazität wird durch Pollen belegt, ein eindeutiges Indiz dafür, daß zum einen Quittungen nicht rechtzeitig übertragen werden, zum anderen erst nach mehrmaligen Pollen eine Antwort verschickt wird.

Es soll nun untersucht werden, ob ein hybrider Ansatz zum Erfolg führt. Dazu wird **oldest_first** dahingehend erweitert, daß Antworten aufs Pollen sofort eine hohe Priorität

	Station 1		Station 2	
	Uplink	Downlink	Uplink	Downlink
Angebot	60%	–	–	60%
genutzte Kapazität	63.6%	17.4%	17.3%	63.6%
Nutzdaten	52.6%			52.7%
reine Quittungen	11.0%	17.4%	17.3%	11.0%
mittlere Wartezeit reiner Quittungen	1.05	4.23	4.24	1.05

Tabelle 8.6: Meßergebnisse für optimiertes **oldest_first**

erhalten. Der neue Algorithmus wird **answer_poll_first** genannt und soll folgende Eigenschaften besitzen:

- Verzögerung und Durchsatz vom optimierten **oldest_first**
- Stabilität von **prefer_poll**

Abbildung 8.14 zeigt, daß diese Anforderungen von **answer_poll_first** sehr gut erfüllt werden. Dieser Algorithmus erzeugt in allen Meßpunkten die kleinste mittlere Zellverzögerung und liegt beim Durchsatz nur wenig schlechter als das optimierte **oldest_first**. Der verringerte Durchsatz resultiert aus dem Versenden von reinen Quittungen als Antwort aufs Pollen. Deshalb ist auch die mittlere Zellverzögerung am geringsten.

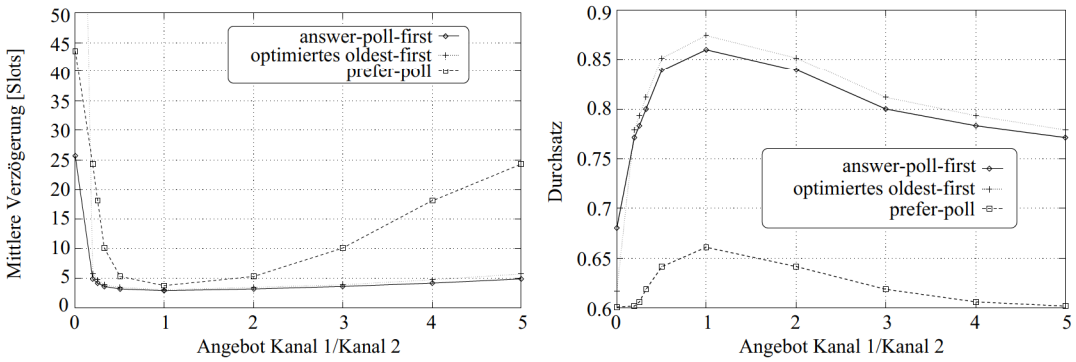


Abbildung 8.14: Mittlere Verzögerung und Durchsatz von **answer_poll_first**

Eine endgültige Optimierung der Quittierungsalgorithmen kann nur im Zusammenhang mit einem realen MAC-Protokoll erfolgen. Dabei ist dann auch auf die Besonderheiten des jeweiligen MAC-Protokolls einzugehen. Es bleibt aber festzuhalten, daß die Bündelung normaler Quittungen im Zusammenspiel mit der hohen Priorität für Antworten aufs Pollen und für Delay PDUs (vgl. Kapitel 8.4) zu den günstigsten Ergebnissen unter den idealisierten Bedingungen führt. Dieses sollte bei zukünftigen Weiterentwicklungen Beachtung finden.

KAPITEL 9

Zusammenfassung und Ausblick

In dieser Arbeit wurde das ASR ARQ-Protokoll weiterentwickelt und in seiner Leistungsfähigkeit bewertet. Nach einer kurzen Einführung in MBS und ATM wurde die Notwendigkeit eines Fehlerkorrekturverfahrens an der Funkschnittstelle herausgearbeitet. Anschließend wurden die Anforderungen an ein ARQ-Protokoll und die daraus resultierende Struktur der LLC-Schicht dargestellt. Weiterhin wurden diejenigen Eigenschaften der MAC-Schicht herausgestellt, die vom ARQ-Protokoll vorteilhaft genutzt werden können. Diese sind vor allem die dynamische Kapazitätsvergabe, die deterministische Übertragungsverzögerung und die Tatsache, daß ARQ Rahmen nicht in der MAC-Schicht zwischengespeichert werden.

Nach einem Überblick über konventionelle ARQ-Protokolle wurden die Erweiterungen gegenüber standardisierten Protokollen dargestellt. Hier sind vor allem der Betrieb paralleler ARQ-Instanzen, das Trennen von Nutzdaten und Quittungen, die Verwendung des *Ignore Timers* und die besondere Behandlung zeitkritischer ATM-Zellen zu nennen.

Das ASR ARQ-Protokoll wurde als endlicher Zustandsautomat konzipiert, in C++ implementiert und die sich ergebene Klassenhierarchie dargelegt. Zur Leistungsbewertung wurden weiterhin eine modifizierte Application- und MAC-Schicht implementiert, die die Dienste ihrer Schichten idealisiert zur Verfügung stellen, um bei der Leistungsbewertung externe Einflüsse zu eliminieren.

Das ASR ARQ-Protokoll wurde anhand der ATM Leistungsparameter in verschiedenen Szenarios simulativ bewertet. Der Einsatz des *Ignore Timers* erwies sich hierbei als nützlich. In weitergehenden Untersuchungen ist festzustellen, inwieweit die angenommene konstante Übertragungsverzögerung in einem realen System vorhanden ist.

Der Betrieb paralleler ARQ-Instanzen für virtuelle Kanäle unterschiedlicher Charakteristik und die Trennung von Quittungen und Nutzdaten stellten sich als großer Gewinn heraus. Trotz eines erhöhten Overheads stellt sich eine erhebliche Verbesserung der Zellverzögerung ein.

Es wurden verschiedene Methoden entwickelt, um zeitkritische ATM Zellen nach dem Überschreiten der maximalen Verzögerung verwerfen zu können. In den Simulationen konnte gezeigt werden, daß ein Verwerfen von Zellen zur Behebung von kurzfristigen Überlastsituationen sehr sinnvoll ist. Das ASR ARQ-Protokoll ist in seiner Komplexität in bezug auf die Methoden des Verwerfens parametrierbar, so daß je nach vorhandenen Kapazitäten verschiedene Verfahrensweisen möglich sind. Die Frage nach der Kodierung der Restlebenszeit ist aber noch genauso offen wie die Frage nach der Stabilität des erweiterten Protokolls. Eine analytische Untersuchung des entwickelten Protokolls erscheint sinnvoll zu sein.

Schließlich wurde die Notwendigkeit von intelligenten Prioritätenalgorithmen für das Versenden von Quittungen aufgezeigt. Bei der Weiterentwicklung solcher Algorithmen muß auf die Besonderheiten der eingesetzten MAC-Schicht eingegangen werden.

Das ASR ARQ-Protokoll ist an die bestehenden und in Entwicklung befindlichen MAC-Protokolle anzupassen. Insbesondere ist eine erneute Leistungsbewertung notwendig, um die getroffenen Annahmen verifizieren zu können. Die Verwendung kurzer Signalisierungsslots im DSA++ Protokoll bietet dem ASR ARQ-Protokoll die Möglichkeit, Quittungen auch ohne ARQ-Rahmen zu versenden. Das Protokoll ist in dieser Hinsicht zu modifizieren.

Das ASR ARQ-Protokoll eignet sich hervorragend zur Behandlung von Diensten der Klassen VBR und ABR. Für die Dienstklasse CBR wäre ein alternatives, einfaches Protokoll möglich, das etwa nach dem zweiten Übertragungsversuch einer Zelle diese verwirft. Es ist zu untersuchen, wie ein solches Protokoll vorteilhaft auszusehen hat und ob dieses Protokoll durch entsprechende Parametereinstellungen durch das ASR ARQ-Protokoll bereits abgedeckt wird. Ist dies nicht der Fall, so muß untersucht werden, ob die Implementierung zweier unabhängiger Protokolle gerechtfertigt erscheint oder ob der größere Aufwand des ASR ARQ-Protokolls auch für CBR-Dienste betrieben wird, um eine einheitliche Protokollstruktur zu erhalten.

Abkürzungsverzeichnis

AAL	ATM Adaptation Layer
ABR	Available Bit Rate
ADCCP	Advanced Data Communication Control Procedure
ADM	Asynchronous Disconnected Mode
ARQ	Automatic Repeat ReQuest
ASR	Adaptive Selective Repeat
ATM	Asynchronous Transfer Mode
B-ISDN	Breitband-ISDN
BSC	Base Station Controller
BST	Base Station Transceiver
BTS	Base Transceiver Station
CAC	Connection Admission Control
CBR	Constant Bit Rate
CDV	Cell Delay Variation
CEP	Connection End Point
CER	Cell Error Ratio
CLP	Cell Loss Priority
CLR	Cell Loss Ratio
CMR	Cell Misinsertion Rate
CNCL	Communication Networks Class Library
CPCS	Common Part Convergence Sublayer
CS	Convergence Sublayer
CTD	Cell Transfer Delay
DCS 1800	Digital Cellular System 1800
DECT	Digital European Cordless Telecommunications
DM	Disconnect Mode
FCFS	First Come First Serve
FDM	Frequency Division Multiplexing
FDMA	Frequency Division Multiple Access
FEC	Forward Error Correction
GCCH	Global Control CHannel
GFC	Generic Flow Control
GIST	Graphical Interactive Simulation result Tool
GSM	Global System for Mobile Communication
HDLC	High level Data Link Control
HEC	Header Error Control
IBCN	Integrated Broadband Communication Network
ISDN	Integrated Services Digital Network
LAP-B	Link Access Procedure-Balanced
LLC	Logical Link Control
LLI	Logical Link Itentifier
LRE	Limited Relative Error

MAC	**M**edium **A**ccess **C**ontrol
MBS	**M**obile **B**roadband **S**ystem
MBT	**M**obile **B**roadband **T**ermination
MCTD	**M**ean **C**ell **T**ransfer **D**elay
MS	**M**obile **S**tation
NNI	**N**etwork **N**ode **I**nterface
PDU	**P**rotocol **D**ata **U**nit
PT	**P**ayload **T**ype
QAM	**Q**uadrature **A**mplitude **M**odulation
QPSK	**Q**uaternary **P**hase **S**hift **K**eying
QoS	**Q**uality **o**f **S**ervice
RACE	**R**esearch and technology development in **A**dvanced **C**ommunications technologies in **E**urope
REJ	**REJ**ect
RN	**R**equest **N**umber
RR	**R**eceive **R**eady
SABME	**S**et **A**synchronous **B**alanced **M**ode **E**xtended
SAP	**S**ervice **A**ccess **P**oint
SAR	**S**egmentation **A**nd **R**eassembly
SDLC	**S**ynchronous **D**ata **L**ink **C**ontrol
SDU	**S**ervice **D**ata **U**nit
SECBR	**S**everely **E**rrored **C**ell **B**lock **R**atio
SN	**S**equence **N**umber
SP	**S**ervice **P**rimitive
SREJ	**S**elective **REJ**ect
SR	**S**elective **R**epeat
SSCOP	**S**ervice **S**pecific **C**onnection **O**riented **P**rotocol
SSCS	**S**ervice **S**pecific **C**onvergence **S**ublayer
STDM	**S**ynchronnous **T**ime **D**ivision **M**ultiplexing
TCH	**T**raffic **CH**annel
TDMA	**T**ime **D**ivision **M**ultiple **A**ccess
UA	**U**nnumbered **A**cknowledgment
UMTS	**U**niversal **M**obile **T**elecommunication **S**ystem
UNI	**U**ser **N**etwork **I**nterface
VBR	**V**ariable **B**it **R**ate
VCI	**V**irtual **C**hannnel **I**dentifier
VC	**V**irtual **C**hannel
VPI	**V**irtual **P**ath **I**dentifier

ABBILDUNGSVERZEICHNIS

TABELLENVERZEICHNIS

[1] C. Apostolas, R. Tafazolli, B. G. Evans. *Wireless ATM LAN.* In *PIMRC'95*, pp. 773–777, Toronto, Canada, September 1995.

[2] D. Bertsekas, R. Gallager. *Data Networks.* Prentice-Hall, Englewood Cliffs, NJ, 1987.

[3] J. Brázio, C. Belo, S. Sveat, B. Langen, D. Plassmann, P. Roman, A. Cimmino. *MBS Scenarios.* In *RACE Mobile Workshop*, pp. 194–197, Metz, F, June 1993.

[4] J. Brázio, J. Sobrinho, Correia A. *Multiple access methods for packet video.* R2067 MBS/WP2.2.2/IST032.2, IST, February 1994.

[5] O. Casals, J. Garcia, C. Blondia. *A Medium Access Control Protocol for an ATM Access Network.* In *Proc. of 5th Int. Conf. on Data Comm. Syst. and their Performance*, Raleigh, North Carolina (USA), October 1993.

[6] B. Deselaers. *Struktur und Protokolle der Sicherungsschicht für eine mobile Erweiterung von ATM–Netzen unter Berücksichtigung von Multilink-Übertragung und Handover.* Diplomarbeit, Kommunikationsnetze, RWTH Aachen, 95.

[7] R. Dinis, A. Gusmão. *Adaptive Serial OQAM-Type Receivers for Mobile Broadband Communications.* In *IEEE 45th Vehicular Technology Conference*, pp. 200–205, July 1995.

[8] L. Fernandes. *Developing a System Concept and Technologies for Mobile Broadband Communications.* IEEE Personal Communications, Vol. 2, No. 1, pp. 54–59, February 1995.

[9] Galagher. *Networks.* Addison Wesley, 1993.

[10] C. Garburg. *Entwicklung und Leistungsbewertung eines ARQ-Protokolls für eine mobile Erweiterung von ATM–Netzen unter Berücksichtigung von Multichannel-Übertragung und Macro-Diversity.* Diplomarbeit, Kommunikationsnetze, RWTH Aachen, March 95.

[11] Schreiber F. Görg C. *Stochastische Simulationstechnik.* Vorlesung an der RWTH Aachen, Aachen, 1995.

[12] R. Händel, M.-N. Huber, S. Schröder. *ATM Networks: Concepts, Protocols, Applications.* Addison–Wesley, 1994.

[13] T. Henderson. *Design priciples and performance analysis of SSCOP: a new ATM Adaptation protocol.* Computer Communication Review, Vol. 25, No. 2, pp. 47–59, April 1995.

[14] J. Hyman, A. Lazar, G. Pacifici. *Real-Time Scheduling with Quality of Service Constraints.* IEEE Journal on Selected Areas in Communications, Vol. 9, No. 7, pp. 1052–1063, September 1991.

[15] J. Immonen, J-M. Romann, D. Plassmann. *Requirements and Protocol Architecture for MBS access to ATM network.* In *RACE Mobile Telecommunications Summit*, pp. 324–328, Cascais, P, November 1995.

[16] ISO. *Standard ISO 6256: HDLC – high level data link control.*

[17] ISO. *ISO 8802-2, Information processing systems-Local area networks-Part2:Logical link control.* International standard, ISO, 1990.

[18] ITU-T. *Recommendation I.356: B-ISDN ATM Layer Cell Transfer Performance*, 1993.

[19] ITU-T. *Recommendation I.363: B-ISDN ATM Adaptation Layer Specification*, 1993.

[20] ITU-T. *Recommendation Q.2110: B-ISDN ATM Adaptation Layer - Service Specific Connection Oriented Protocol (SSCOP)*, 1993.

[21] H. Kerner. *Rechnernetze nach OSI.* Addison-Wesley, München, 1993.

[22] Leonard Kleinrock. *Queueing Systems — Computer Applications*, Vol. 2. Wiley-Interscience, New York, NY, 1976.

[23] H. Kopka. *LATEX: eine Einführung.* Addison–Wesley, Bonn, 4. edition, 1993.

[24] J.-Y. Le Boudec. *The Asynchronous Transfer Mode: a tutorial.* In *Computer Networks and ISDN Systems 24*, pp. 279 – 309, North-Holland, 1992.

[25] H. Meineke. *Wettbewerber warten auf den Fall der Telekom-Monopole.* VDE-dialog, Vol. 1, No. 1, pp. 4–5, 1995.

[26] S. Nogami. *A Study of Quality Control at the Cell Level in the ATM Network – Simplicity of Control Mechanisms vs. Efficient Utilization of Resources.* In *Proc. of the 13th Int. Teletraffic Congress*, pp. 987–992, Copenhagen, DK, June 1991.

[27] D. Petras. *Functionality of the ASR-ARQ Protocol for MBS.* In *RACE Mobile Telecommunications Summit*, pp. 225–229, Cascais, P, November 1995. available at http://www.comnets.rwth-aachen.de/~petras.

[28] D. Petras. *Medium Access Control Protocol for transparent ATM access in MBS.* In *RACE Mobile Telecommunications Summit*, pp. 218–224, Cascais, P, November 1995. available at http://www.comnets.rwth-aachen.de/~petras.

[29] D. Petras. *Medium Access Control Protocol for wireless, transparent ATM access.* In *IEEE Wireless Communication Systems Symposium*, pp. 79–84, Long Island, NY, November 1995. available at http://www.comnets.rwth-aachen.de/~petras.

[30] D. Petras, A. Hettich. *Performance evaluation of the ASR-ARQ Protocol for wireless ATM.* In *IEEE Wireless Communication Systems Symposium*, pp. 71–77, Long Island, NY, November 1995. available at http://www.comnets.rwth-aachen.de/~petras.

[31] D. Petras, A. Hettich, B. Walke. *Design Principles and Performance Evaluation of an LLC Protocol for an ATM Air Interface.* In *IEEE Journal on Selected Areas in Communications*, Long Island, NY, January 1996.

[32] S. Prata. *C++: Einführung in die objektorientierte Programmierung.* te-wi Verl., München, 3. edition, 1992.

[33] M. Prögler. *MBS Air Interface Principles.* In *RACE Mobile Telecommunications Summit*, Cascais, P, November 1995.

[34] F. Reif. *C++ Implementierung eines graphischen Protokolldebuggers für eine ATM-Funkschnittstelle.* Studienarbeit, Kommunikationsnetze, RWTH Aachen, January 96.

[35] L. Schmickler. *Einsatzmöglichkeiten der Optimalstrategie SRPT in Lokalen Rechnernetzen.* PhD thesis, Kommunikationsnetze, RWTH Aachen, 91.

[36] G. Siegmund. *ATM – Die Technik des Breitband-ISDN.* v. Decker, Heidelberg, 1993.

[37] P.F.M. Smulder. *Broadband Wireless LANs: A Feasibility Study.* PhD thesis, Eindhoven University of Technology, 1995.

[38] B. Walke. *Improved bounds and an approxination for a dynamic priority queue.* In *3rd Int. Comp. Sympos. Modell & Perform. Evaluation of Computer Systems, Bonn, G, Oct. 1977.* North-Holland Publishing Company, 1977.

[39] B. Walke. *Waiting-Time Distributions for deadline-oriented Serving.* In M. Arato, A. Butrimenko, E. Gelenbe, editors, *Performance of Computer Systems*, pp. 241 – 260. North-Holland Publishing Company, 1979.

[40] B. Walke. *Kommunikationsnetze und Verkehrstheorie I.* Vorlesung an der RWTH Aachen, Aachen, 1993.

[41] B. Walke. *Kommunikationsnetze und Verkehrstheorie II.* Vorlesung an der RWTH Aachen, Aachen, 1993.

[42] B. Walke. *Mobilfunknetze und ihre Protokolle.* Vorlesung an der RWTH Aachen, Aachen, 1993.