Informatik-Spektrum (1993) 16: 143–158

Informatik
© Springer-Verlag 1993 Spektrum

Das DQDB-Zugriffsprotokoll in Hochgeschwindigkeitsnetzen und der IEEE-Standard 802.6

Ein Überblick

R. Dittmann, T. Stock und P. Tran-Gia

Universität Würzburg

Zusammenfassung. Glasfaserbasierte lokale und regionale Hochgeschwindigkeitsnetze (HS-LANs und -MANs: High-Speed Local and Metropolitan Area Networks) und die zugehörigen Übertragungsprotokolle gehören zu den momentan vieldiskutierten Themen in der Kommunikationstechnik. Von den vorgeschlagenen Kommunikationsprotokollen für Systeme mit hoher Kapazität könnte das DQDB-Protokoll (Distributed Queue Dual Bus) einer der meistakzeptierten Standards der Zukunft werden. Ziel dieses Beitrages ist, eine Übersicht über dieses Protokoll zu geben. Nach einer kurzen Abhandlung über strukturelle Systemvoraussetzungen wird auf die dem DQDB-Zugriffsmechanismus zugrundeliegenden Prinzipien eingegangen. Die diversen Phasen in der Entstehung des nunmehr verabschiedeten IEEE-Standards 802.6 werden aufgezeigt und verschiedene Ansätze zur Leistungsanalyse geschildert. Es schließt sich die Präsentation einiger Leistungsmerkmale an. Schließlich werden mögliche Erweiterungen des Standards und deren Einsatzmöglichkeiten in anderen Systemarchitekturen vorgestellt.

Schlüsselwörter: LAN, MAN, Hochgeschwindigkeitsnetze, Zugriffsprotokolle, DQDB, Modellierung, Leistungsbewertung

Summary. High-speed local and metropolitan area networks (HSLAN, HSMAN) based on optical fibers and the corresponding access protocols are under intensive discussion in the field of telecommunication technology. Among the proposed protocols, the DQDB system (Distributed Queue Dual Bus) has a good chance of gaining wide acceptance in the near future. The objective of this paper is to provide an overview of this protocol. A short outline of the architectural system requirements is followed by a description of the underlying principles on which the DQDB access mechanism is based. Various phases in the evolution of the IEEE standard 802.6 are discussed and some analytical approaches to the study of system performance are addressed. Several performance measures are then presented. Finally, some modifications and extensions to the standard and their potential use in other system architectures are described.

Key words: High-speed LAN and MAN access schemes, Protocols, DQDB, Modeling, Performance analysis

Computing Reviews Classification: C.2, C.4

1. Einleitung

Die Leistungsfähigkeit von Rechnernetzen in lokalen und regionalen Bereichen wird im wesentlichen von zwei Faktoren bestimmt: der Übertragungskapazität des physikalischen Mediums und den Übertragungsprotokollen. Getragen von den Möglichkeiten moderner Übertragungstechnik und bedingt durch die Verfügbarkeit der Glasfasertechnologie sowie deren zunehmende Wirtschaftlichkeit und Zuverlässigkeit, stehen Übertragungskapazitäten von 100 Mbps (Megabit pro Sekunde) bis zu 2,4 Gbps (Gigabit pro Sekunde) bereit, die noch mit entsprechenden Protokollarchitekturen ausgestattet werden müssen, um die angestrebten Leistungen erbringen zu können. In der jüngeren Literatur über Kommunikationsprotokolle und ihre Analyse finden sich zahlreiche Veröffentlichungen zu übergreifenden regionalen Netzen (MAN – Metropolitan Area Network) und besonders zu einem mittlerweile standardisierten Zugriffsprotokoll: DQDB (Distributed Queue Dual Bus).

Ziel dieses Artikels ist es, kurz in die Problematik der regionalen Netze sowie etwas tiefer in die DQDB-Materie einzuführen und den aktuellen Stand der Entwicklung des IEEE-DQDB-Standards 802.6 aufzuzeigen. Im Anschluß sollen Analyseansätze aus der Literatur geschildert und die so gewonnenen Maße zur Leistungsbewertung präsentiert werden. Es sei angemerkt, daß das beigefügte Literaturverzeichnis selbst für den speziellen Themenkreis DQDB nur eine Auswahl der verfügbaren Literatur enthalten kann.

Im Rahmen eines Ausblicks soll dann noch auf die Bedeutung des IEEE-DQDB-Standards 802.6 für das Internetworking und als Zubringersystem für künftige ATM-Netze (Asynchronous Transfer Mode, geplante Übertragungsart zukünftiger Breitband-Weitverkehrsnetze, siehe z.B. [26], wo einige einführende Artikel zu B-ISDN und ATM zu fin-

den sind) sowie auf verschiedene Möglichkeiten der Modifikation des DQDB-Protokolls eingegangen werden.

1.1. LAN, MAN und WAN

Zu Beginn der 80er Jahre wurden mit dem Aufkommen von LANs (Local Area Network, z. B. Ethernet, Token Ring mit 10 Mbps bzw. 4 Mbps) erstmals in größerem Umfang Rechner innerhalb eines beschränkten Areals – typischerweise in einem Gebäude – miteinander verknüpft (siehe z. B. [48]). Damit war eine Infrastruktur geschaffen, mit deren Hilfe Dienste verwirklicht werden konnten, die wesentlich mehr Bandbreite benötigten, als die bis dahin üblichen Terminalverbindungen mit einer Geschwindigkeit von maximal einigen Kilobyte pro Sekunde zur Verfügung stellen konnten.

Schon Mitte der 80er Jahre wurde die Notwendigkeit zur Verknüpfung der so entstandenen LAN-Inseln mit schnellen Backbone-Systemen immer deutlicher. Die zu diesem Zweck schon frühzeitig verwendete WAN-Technologie (WAN – Wide Area Network), die im wesentlichen auf X.25 und verwandten Protokollen aufbaut, ermöglicht zwar Verbindungen auf nationaler wie auch auf internationaler Ebene, setzt aber in bezug auf die verfügbare Bandbreite sehr enge Grenzen,[1] so daß neue Konzepte aufgegriffen werden mußten. Verschiedene Entwicklungen waren auch schon bis Anfang der 90er Jahre abgeschlossen und zum Teil – was besonders wichtig war – von einflußreichen Gremien standardisiert worden, z. B. FDDI von ANSI und ISO. Ein Überblick zu FDDI findet sich in [39]. Derzeitige Systeme arbeiten mit einer Geschwindigkeit von bis zu 100 Mbps und decken Regionen bis zu einem Durchmesser von etwa 100 km ab, bedienen also den typischen MAN-Bereich. Parallel dazu verliefen die ersten – zunächst prototypischen – Installationen schneller WAN-Systeme (z. B. auf Basis von X.25-basierten Protokollen und gemieteten Leitungen mit 2 oder 34 Mbps in Europa bzw. Leitungen mit 1,5 oder 44 Mbps in Nordamerika).

All diesen Ansätzen war gemeinsam, daß sie zumindest anfangs im wesentlichen nur reinen Datenverkehr unterstützten. Da aber neue Dienste mit völlig anderen Anforderungen an das Übertragungssystem immer mehr an Bedeutung gewannen – man denke an Bildtelefon, Videoconferencing, Joint Editing, entfernte Prozeßsteuerung etc. –, war auch hier ein Umdenken notwendig. Zwar versprach die Einführung der ATM-Technologie (s. [26]) in der öffentlichen Vermittlungstechnik eine Basis für derartige Dienste bis in den WAN-Bereich, aber zu Beginn der 90er Jahre waren bei weitem noch nicht alle damit verbundenen Probleme gelöst (vgl. [6]), und es wird noch geraume Zeit vergehen, bis eine breitere Verfügbarkeit erreicht ist. Um die erwähnten Dienste dennoch schon jetzt einführen zu können, plant man als Zwischenstufe auf regionaler Ebene sog. hybride MANs, z. B. FDDI-II (ANSI/ISO), DQDB (Australian Telecom, IEEE 802.6) und Orwell (British Telecom) sowie andere Systeme, die aber zum großen Teil das Prototypstadium nicht erreicht oder gar überschritten haben (vgl. [8, 24, 28]). Diese Systeme integrieren – im Gegensatz z. B. zu FDDI-I, das nur Datenverkehr überträgt, oder zu privaten Nebenstellenanlagen, die nur synchrone Verbindungen zur Verfügung stellen – verschiedenste Dienste, seien es asynchrone wie Dateitransfer oder synchrone (bzw. isochrone) wie Sprachübermittlung. Die Protokolle zielen in der Regel zunächst auf Kapazitäten von 100 bis 155 Mbps ab. Es gibt aber auch einige Spezifikationen und Prototypen für 34 und 44 Mbps, um vorhandene PCM-Strecken nutzen zu können. Nach Einführung ATM-basierter Fernverkehrsnetze können diese Systeme dann als Zubringernetze für das ATM-Netz dienen. In Abb. 1 ist als Beispiel eine MAN-Konfiguration dargestellt.

[1] Ein Datex-P-Hauptanschluß beispielsweise stellt eine Kapazität von 64 kbps zur Verfügung.

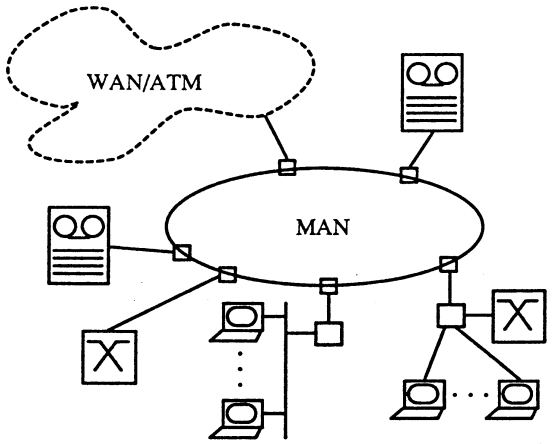

Abb. 1. MAN-Konfiguration

1.2. Grundlegende Probleme beim Design von MAN-Systemen

Das entscheidende Problem beim Design von Zugriffsprotokollen für Hochgeschwindigkeits-MAN-Systeme ist die im Vergleich zu den bekannten LANs wesentlich größere *logische Länge*, d. h. die Länge des Übertragungssystems gemessen in bit. Mit der physikalischen Länge l, der Signalausbreitungsgeschwindigkeit c und der Übertragungskapazität C erhält man die logische Länge als den Quotienten $l \cdot \frac{C}{c}$. Heute weit verbreitete Netze wie Ethernet (maximale Länge etwa 2,5 km, Übertragungsrate 10 Mbps) erreichen eine logische Länge von etwa 100 bit (zuzüglich der Verzögerungen, die sich z. B. durch die Verwendung von Repeatern ergeben). Mit einigen 100 bis etwa 10000 bit liegen die typischen Paketlängen also in der Regel deutlich über der logischen Länge des Übertragungsmediums.

DQDB dagegen hat, wenn man eine Übertragungsrate von 150 Mbps zugrunde legt, eine logische Länge von 1500 bit bei einer 2,5-km-LAN-Konfiguration bzw. von 120000 bit bei einer 200-km-MAN-Konfiguration. Die logische Länge des Systems ist also deutlich größer als die üblichen mittleren Paketlängen. Im folgenden soll kurz gezeigt werden, weshalb bei einer derartigen Konfiguration die bei LANs gebräuchlichen Protokolle nicht mehr anwendbar sind.

Bisher übliche Zugriffsstrategien:

- Wahlfreier Zugriff auf das Medium ohne zusätzlichen Austausch von Statusinformationen zwischen den angeschlossenen Stationen wie bei Ethernet. In diesem Fall ist ein Algorithmus zur Erkennung und Auflösung von Kollisionen auf dem Medium notwendig, der bei jeder Kollision Verzögerungen in der Größenordnung von etwa 2 Buslängen mit sich bringt. Ein solches Protokoll kann in einem –

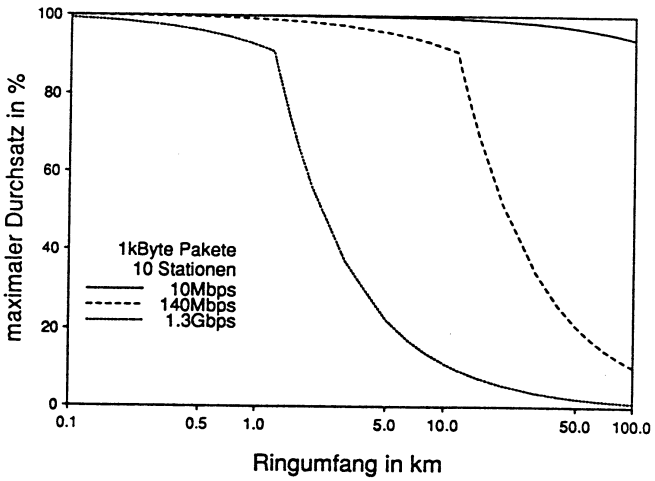

Abb. 2. Maximaler Durchsatz von Token-Ring-Systemen in Abhängigkeit vom Umfang des Ringes

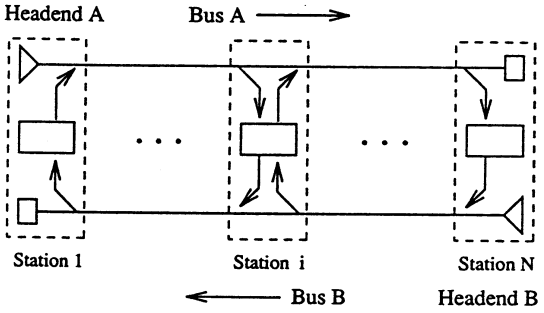

Abb. 3. Dual-Bus-Konfiguration

annähernd – saturierten Fall natürlich dann nicht mehr effizient arbeiten, wenn die logische Länge des Übertragungskanals die Länge der zu sendenden Pakete um Zehnerpotenzen überschreitet. Es sei noch angemerkt, daß die spezielle Funktionsweise des CSMA/CD-Verfahrens eine minimale Paketlänge voraussetzt, die von der logischen Länge des Systems abhängt. Dies hat zur Folge, daß der CSMA/CD-Mechanismus bei großen logischen Buslängen nicht nur ineffizient, sondern nicht praktikabel ist.

- Zyklischer Zugriff, wobei eine Marke (Token), die die Sendeberechtigung anzeigt, von einer Station zur nächsten weitergereicht wird, wie bei Token-Ring- oder Token-Bus-Systemen. In diesem Fall geht pro Zyklus eine Systemlänge an Übertragungskapazität verloren (neben anderen zusätzlichen Verzögerungen). Der damit verbundene Leistungsabfall ist zunächst zwar nicht so gravierend wie bei CSMA/CD, ist aber nur bis zu einer bestimmten logischen Ringlänge tolerierbar (vgl. dazu Abb. 2, Kurven für den maximalen Durchsatz eines Token-Ring-Protokolls bei verschiedenen Übertragungsraten in Abhängigkeit vom – der Anschaulichkeit halber realen – Ringumfang).

1.3. Fairneß von Zugriffsmechanismen

In vielen Veröffentlichungen, die sich in jüngster Zeit mit Hochgeschwindigkeitsnetzen auf lokaler oder regionaler Ebene befassen, erscheint „Fairneß" als zentraler Begriff in der Diskussion um die Qualität eines Zugriffsprotokolls. Fairneß, als Metrik verstanden, steht dabei aber nicht immer für denselben Sachverhalt – in der Regel wird er so gedeutet, daß sich das eigene Produkt/Protokoll als anderen überlegen darstellt.

Die beiden gebräuchlichsten Definitionen sind:

- Fairneß in bezug auf die Zugriffsverzögerungen, die einzelne Datenpakete erfahren. In der Literatur wird zwischen zwei Zugriffsverzögerungen – die häufig mit demselben Begriff „Medium Access Delay" belegt werden – unterschieden: Zum einen mißt man die komplette Wartezeit von der Ankunft eines Paketes bis zu seiner Übertragung auf das Medium. Im anderen Fall beschränkt man sich auf den Zeitraum vom Erreichen des Anfangs der lokalen Warteschlange einer Station bis zur Übertragung. (Im saturierten Fall bezeichnet dies genau das Intervall vom Ende der Übertragung eines Paketes bis zum Ende der Übertragung des darauf folgenden.)
- Fairneß bezogen auf den Anteil an Bandbreite, den die angeschlossenen Stationen – innerhalb eines bestimmten Intervalls – für sich nutzen können. Hier stellt sich natürlich die Frage, über welchen Zeitraum gemessen werden soll, wenn man nach diesem Kriterium bewerten will.

Denkbar wäre z. B. aber auch ein Kriterium, das sich an der Varianz der Wartezeiten orientiert, um so z. B. Jitter-Effekte zu bewerten, die bei Videoübertragungen minimiert werden sollten.

Es stellt sich die Frage, welche Art von Fairneß-Kriterium nun die richtige ist. Leider kann es auch hier keine allgemeingültige Antwort geben, obwohl in manchen Veröffentlichungen der gegenteilige Eindruck erweckt wird. Die Anforderungen der einzelnen Dienste, über die derzeit diskutiert wird und die in Zukunft unterstützt werden sollen, sind zum Teil gegensätzlicher Natur, so daß man entweder Kompromisse bei der Definition von Fairneß eingehen oder sich auf spezielle Typen von Diensten beschränken muß.

2. Dual-Bus-Systeme und Verteiltes Warten

Ein Dual-Bus-System basiert auf einem Paar unidirektionaler, entgegengesetzt gerichteter Busse A und B (vgl. Abb. 3), an die die einzelnen Stationen angeschlossen sind.[2] Logisch gesehen, können die Knoten alle von oberhalb[3] ankommenden Daten lesen und, wenn sie von dem noch zu erläuternden Zugriffsmechanismus dazu berechtigt werden, mit Hilfe eines ODER-Gliedes überschreiben. Ein solches System wird physikalisch wohl immer unter Verwendung von Punkt-zu-Punkt-Verbindungen aufgebaut werden, da wegen der bei einer passiven Implementierung beschränkten Zahl von Stationen ohnehin eine aktive Auslegung, eventuell in Ver-

[2] Die physikalische Realisierung erfolgt dabei in der Regel in Form eines Ringes, um so eine schnelle Rekonfiguration des Systems zu gewährleisten, falls eine Verbindungsstrecke zwischen zwei Stationen ausfällt [2]. Dies impliziert, daß jede Station die volle Funktionalität eines Zell- (oder Slot-) Generators haben muß, wenn sie als Head-Of-Bus arbeiten soll.

[3] Begriffe wie oberhalb, unterhalb, stromauf oder stromab sind hier und im folgenden immer relativ zur Übertragungsrichtung des betrachteten (Nutzdaten-) Busses zu sehen.

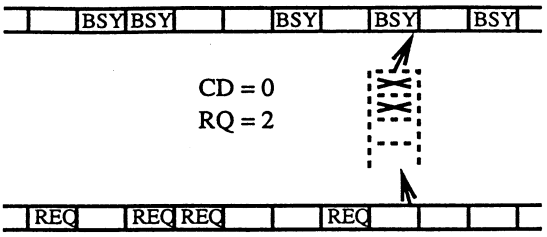

Abb. 4a. Übertragungsbeispiel, t = 0

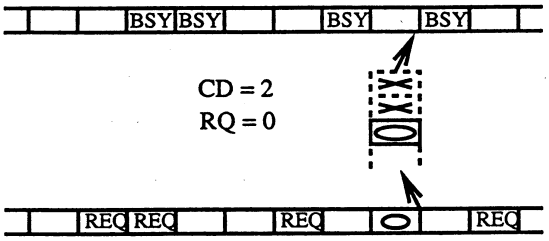

Abb. 4b. Übertragungsbeispiel, t = 1

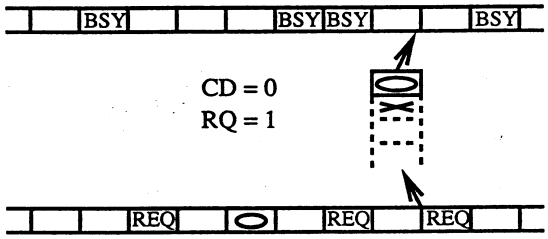

Abb. 4c. Übertragungsbeispiel, t = 4

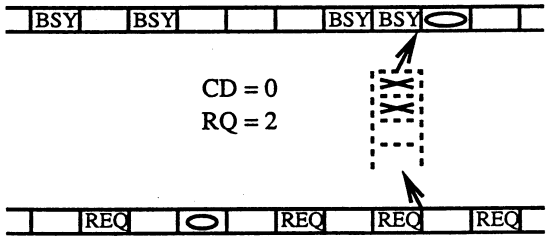

Abb. 4d. Übertragungsbeispiel, t = 5

bindung mit einer Bypass-Schaltung für den Fehlerfall, überlegen ist.

Der Datenaustausch erfolgt über Minipakete fester Länge (sog. Zellen im BISDN-Sprachgebrauch oder Slots im IEEE-Bereich), die von den beiden Zellgeneratoren HOB A und HOB B[4] erzeugt und am jeweiligen unteren Ende des Busses vernichtet werden. Da eine Station zunächst alle freien Zellen benutzen und somit die unterhalb liegenden Stationen jedweder Übertragungsbandbreite berauben könnte, wird nun ein Zugriffsprotokoll eingeführt, dessen Prinzip als verteiltes Warten bezeichnet wird (s. auch [33]). Dieses Prinzip soll im folgenden kurz erläutert werden.

Die Grundidee ist, den Stationen am Netz Informationen über den Gesamtzustand des Systems zur Verfügung zu stellen, die es ermöglichen, die Position des eigenen Datensegments im verteilten Wartesystem zu bestimmen. Eine Station verfährt dabei wie folgt:

- Wenn ein Knoten ein Datensegment auf Bus A senden will, so muß er alle oberhalb gelegenen Stationen davon in Kenntnis setzen, indem er einen Request für ein solches Segment auf Bus B absetzt.
- Gleichzeitig notiert er alle Requests, die von unterhalb gelegenen Stationen gesendet wurden, d. h. er beobachtet alle Requests auf Bus B.
- Ein Knoten muß mit der Übertragung eines Segments so lange warten, bis alle Requests bedient sind, die vor der Ankunft seines Segmentes eingetroffen waren, d. h. er muß eine entsprechende Anzahl von leeren Zellen auf Bus A passieren lassen.
- Pro Knoten darf sich immer nur ein einziges Segment in diesem globalen Wartesystem befinden.

Gleiches gilt völlig analog für die Übertragung von Nutzdaten in entgegengesetzter Richtung auf Bus B.

Unter diesen Voraussetzungen kann man das globale Wartesystem innerhalb der einzelnen Knoten mit Hilfe zweier Zähler[5] für jede Übertragungsrichtung realisieren:

1. Countdown-Counter (CD): zählt diejenigen Requests, die noch vor dem eigenen Segment bedient werden müssen. Dieser Zähler ist nur aktiv, wenn der Knoten auch ein Paket zu senden hat; er wird dann für jede freie Zelle, die den Knoten passiert, um eins heruntergezählt.
2. Request-Counter (RQ): zählt alle neu eingetroffenen Requests. RQ wird um eins erhöht, wenn von unten ein Request eintrifft und, sofern CD nicht aktiv ist, um eins erniedrigt, wenn von oben eine leere Zelle vorbeiläuft.

Die Funktionsweise des Protokolls soll mit folgendem Beispiel erläutert werden: Zum Zeitpunkt t = 0 habe die betrachtete Station nichts zu senden, aber noch zwei Requests zu bedienen, d. h. in der virtuellen globalen Warteschlange, die die Station sieht, befinden sich noch zwei Anforderungen, die von Requests unterhalb liegender Stationen herrühren, also CD = 0, RQ = 2 (vgl. Abb. 4a).

Zum Zeitpunkt t = 1 treffe an der Station ein Datensegment ein, das sofort in die virtuelle Warteschlange eingereiht wird (großes Oval). Dies bewirkt, daß der Inhalt des Request-Counters in den Countdown-Counter übertragen (CD: = RQ) und der Zähler selbst zurückgesetzt wird (RQ: = 0). Parallel dazu wird der zugehörige Request auf den unteren Bus abgesetzt (kleines Oval, vgl. Abb. 4b).

Drei Zelldauern später haben zwei leere Zellen die Station auf dem oberen Bus passiert, die zwei vor dem lokalen Datensegment befindlichen Requests sind also bedient worden, daher CD = 0. Währenddessen sei ein weiterer Request auf dem unteren Bus eingetroffen. Der Request-Counter wird deswegen um eins erhöht (RQ = 1, vgl. Abb. 4c).

Bei t = 5 hat die Station die nächste freie Zelle zur Übertragung benutzt und, da keine weiteren lokalen Segmente

[4] HOB – Head(end) Of Bus

[5] In der standardisierten Version [23, 24] kommt noch ein weiterer Zähler hinzu, der sog. Request-Queue-Counter, der alle noch nicht abgesetzten eigenen Requests zählt. Dieser Zähler ist notwendig, weil das Senden der Requests über das Setzen von sog. Request-Bits im ACF (Access Control Field – Zugriffskontrollfeld, s. Abb. 5) im Header einer Zelle realisiert wird, diese aber schon von weiter unten liegenden Stationen benutzt worden sein können, der Knoten also unter Umständen mit dem Senden seiner eigenen Requests warten muß.

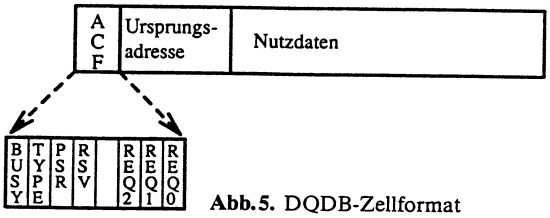

Abb. 5. DQDB-Zellformat

vorhanden sind, den Countdown-Counter wieder deaktiviert (vgl. Abb. 4d).

3. Phasen des DQDB-Standardisierungsprozesses

Vom ersten Vorschlag bis zum fertigen Standard war es ein weiter Weg. Dies zeigt die stattliche Zahl von 16 veröffentlichten Entwürfen [17] in knapp drei Jahren. Bei der Entwicklung von DQDB wurde der Mechanismus der verteilten Warteschlange in Teilen deutlich verändert [20] und erweitert [21], um die Fairneß des Zugriffsprotokolls zu verbessern.

3.1. Die Zugriffsmechanismen

Das DQDB-Protokoll unterscheidet zwei Zugriffsmechanismen, einen für isochronen und einen für asynchronen Verkehr. Beim isochronen, verbindungsorientierten Zugriff wird nach einem Verbindungsaufbau vom Headend periodisch eine Anzahl von Zellen reserviert,[6] die von der benötigten Bandbreite abhängt. Der Nutzdatenbereich dieser Zellen kann nur für isochronen Verkehr verwendet werden. Die nichtreservierten Zellen werden für den asynchronen Verkehr genutzt. Hierbei wird der Zugriff nach dem Prinzip des verteilten Wartens von den Stationen selbst kontrolliert.[7]

Die für den Zugriff benötigten Informationen stehen im Zugriffskontrollfeld (ACF) einer Zelle (s. Abb. 5). Das Busy-Bit gibt an, ob die Zelle bereits benutzt ist. Das Type-Bit wird vom Headend zusammen mit dem Busy-Bit bei allen für isochrone Dienste reservierten Zellen gesetzt. Das PSR-Bit („previous segment received") gibt an, ob die Nutzdaten der vorangegangenen Zellen bereits empfangen wurde. Eine solche Zelle könnte für eine erneute Übertragung wieder freigegeben werden (s. Abschnitt 6.2). Das reservierte (RSV) Bit wird, wie später noch gezeigt wird, bei einer Erweiterung des Protokolls benötigt.[8] Mit den Request-Bits werden die verteilten Warteschlangen für die verschiedenen Übertragungsprioritäten aufgebaut (s. die folgenden Abschnitte). Eine Station kann hierzu auch die Request-Bits von isochronen Zellen nutzen.

Da beim isochronen Verkehr alles vom Headend kontrolliert ist, betrachtet die nachfolgende Analyse (Abschnitt 4) nur den verteilten Zugriff für den asynchronen Verkehr. Sie

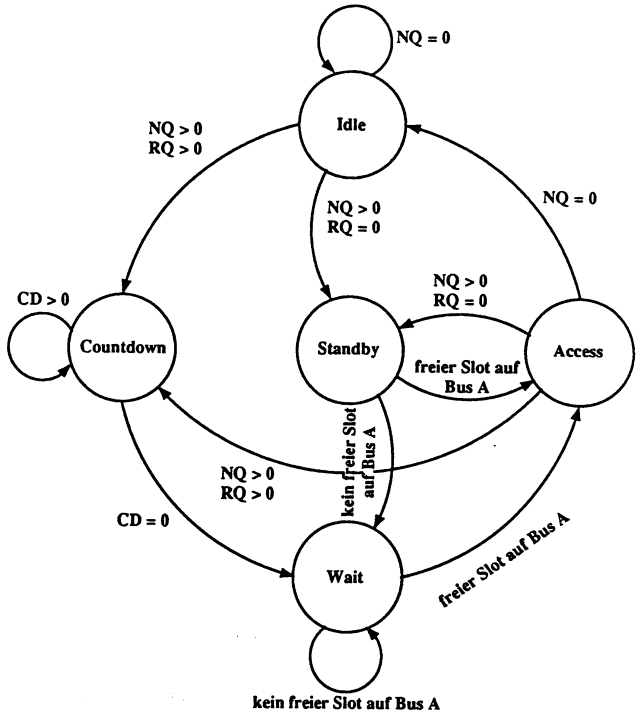

Abb. 6. Transitionsdiagramm des QPSX-Protokolls (NQ: Anzahl der Datensegmente in der lokalen Warteschlange)

berücksichtigt aber eine (konstante) isochrone Grundlast. In den folgenden Abschnitten wird die (historische) Entwicklung des verteilten Wartens in DQDB beschrieben. Dabei wird der Mechanismus für nur einen Bus (Bus A) gezeigt, da die Aktionen auf Bus B genau spiegelverkehrt ablaufen.

3.2. QPSX

Das erste Dual-Bus-Protokoll, das auf dem Prinzip der verteilten Warteschlange basierte, war QPSX (queued packet and synchronous circuit exchange) und wurde an der University of Western Australia entwickelt [2, 33, 34]. Diese Realisierung der verteilten Warteschlange arbeitete noch mit nur einer Priorität, und ihr Zustandsautomat hatte im Vergleich zu den in den Standard aufgenommenen Vorschlägen noch fünf Zustände (s. Abb. 6, [40]).

Eine Station ohne eigene Datensegmente befindet sich im Idle-Zustand. Hier verändert sie ihren Request-Zähler (RQ) entsprechend dem Prinzip des verteilten Wartens. Will sie ein Datensegment übertragen und ist ihr Request-Zähler größer null, so geht sie in den Countdown-Zustand und veranlaßt das Senden eines Requests. Nachdem der Countdown-Zähler (CD) bis null dekrementiert wurde, wechselt die Station in den Wait-Zustand. Wenn die nächste freie Zelle auf Bus A eintrifft, erreicht sie schließlich den Access-Zustand, in dem sie das Datensegment auf das Medium schreibt.

Will eine Station ein Datensegment übertragen und ist ihr Request-Zähler null, so geht sie in den Standby-Zustand. Dort wartet sie auf die nächste Zelle auf Bus A, ohne das Senden eines Requests zu veranlassen. Ist diese Zelle frei, so wechselt sie sofort in den Access-Zustand, anderenfalls ver-

[6] Sie werden im Original daher „Pre-Arbitrated (PA) slots" (vorherbestimmte Zellen) genannt.
[7] Diese Zellen werden im Standard deshalb „Queued Arbitrated (QA) slots" genannt.
[8] Im aktuellen Standard wurde das REQ3-Bit durch ein weiteres reserviertes Bit ersetzt, da einige Erweiterungen zwei Bits benötigen (z. B. [15]).

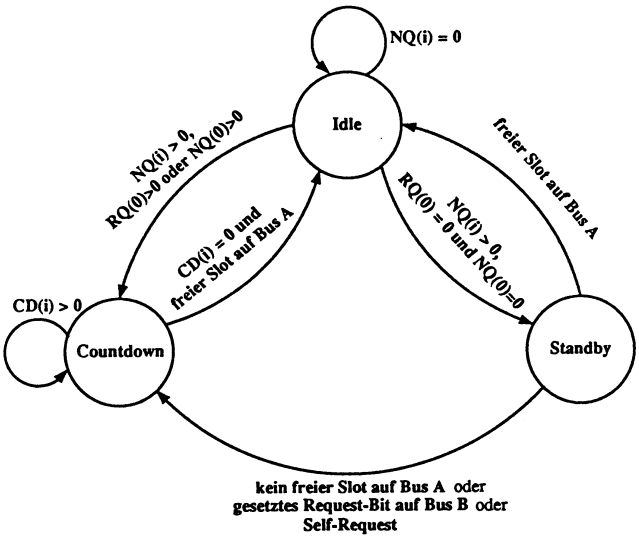

Abb. 7. Transitionsdiagramm des DQDB-Protokolls mit Standby-Zustand (NQ(i): Anzahl der Datensegmente in der lokalen Warteschlange der Priorität i)

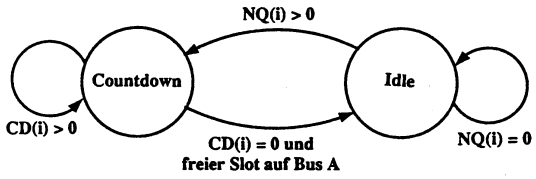

Abb. 8. Transitionsdiagramm des DQDB-Protokolls ohne „Standby"-Zustand

anlaßt sie zuerst das Senden eines Requests. Der Standby-Zustand soll unnötige Requests verhindern, die Stationen oberhalb der Sendenden zwingen würden, Zellen freizulassen, die schon nicht mehr benötigt werden.

Eine genaue Beschreibung des Zugriffsmechanismus von QPSX ist in [33] zu finden. Über Leistungsaspekte geben [16, 40, 54] Auskunft.

3.3. IEEE Standard 802.6 Draft 0-6

Im ersten Entwurf des Standards [18] wurden der Wait-Zustand und der Access-Zustand mit dem Countdown-Zustand vereint. Im Gegensatz zu einigen Varianten von QPSX [40] sind die Übertragung eines Datensegments und des zugehörigen Requests nur noch lose gekoppelt. Zudem wurde das Protokoll so erweitert, daß es nun vier Übertragungsprioritäten unterstützt. Dazu wird pro Priorität eine verteilte Warteschlange gebildet, d. h. jede Station hat – für jede Übertragungsrichtung – einen Request-Zähler RQ(i), einen Countdown-Zähler CD(i) und einen Zustandsautomaten pro Priorität i (s. Abb. 7, [40]).

Bei Ankunft eines Requests auf Bus B mit Priorität i wird der zugehörige Request-Zähler, RQ(i), inkrementiert. Zudem werden alle Request-Zähler, RQ(j), bzw. Countdown-Zähler, CD(j), für die niedrigeren Prioritäten $j < i$ erhöht, je nachdem, ob die Station Datensegmente der Priorität j zu übertragen hat.

Reiht eine Station ein Datensegment in eine der verteilten Warteschlangen ein, so veranlaßt sie das Senden eines Requests derselben Priorität auf Bus B. Um sicherzustellen, daß die Datensegmente einer Station entsprechend ihrer Priorität gesendet werden, muß sie dies auch den anderen Zustandsautomaten mitteilen. Dazu löst sie einen sog. Selbst-Request mit der Priorität des Datensegments aus, was dazu führt, daß innerhalb der Station die Request-Zähler bzw. Countdown-Zähler für niedrigere Prioritäten inkrementiert werden. Die Zähler RQ(i) bzw. CD(i) werden dekrementiert, wenn eine freie Zelle auf Bus A ankommt, die der Knoten nicht für Priorität i (aber evtl. für eine höhere) nutzen kann.

Die Zustandsübergänge (s. Abb. 7) verlaufen prinzipiell wie beim QPSX-Protokoll. Für mehrere Prioritäten lautet die Bedingung für den Übergang in den Standby-Zustand, daß alle Request-Zähler einer Station null sind und auch keine eigenen Datensegmente auf ihre Übertragung warten. Es genügt aber, nur RQ(0) und NQ(0) zu betrachten, da RQ(0) die Requests aller Prioritäten [$\Rightarrow$RQ (0) $\geq$ RQ(i)] und die Selbst-Requests der höheren Prioritäten (= Anzahl der Datensegmente höherer Priorität, die von der Station in die entsprechende verteilte Warteschlange eingereiht wurden) zählt.

Der Standby-Zustand soll auch hier unnötige Requests verhindern. Bei hoher Last verursacht er aber eine Verschlechterung des ohnehin unfairen Zugriffsverhaltens, da er die Stationen in der Nähe des Headends in Abhängigkeit vom Abstand zu anderen gleichzeitig aktiven Stationen begünstigt [53] (s. Abschnitt 5.1).

3.4. Der standardisierte DQDB-Zugriffsmechanismus

Eine wesentliche Änderung im Vergleich zu den ersten Protokollvorschlägen ist die Abschaffung des Standby-Zustands mit Draft D7. Der Zustandsautomat hat jetzt nur noch zwei Zustände (s. Abb. 8, [40]).

Die Zähler arbeiten weiterhin so, wie im vorhergehenden Abschnitt beschrieben. Eine Station, die ein Datensegment in die seiner Priorität entsprechende verteilte Warteschlange einreihen möchte, geht jetzt aber direkt in den Countdown-Zustand über. Sie löst dabei intern einen Selbst-Request aus und veranlaßt sofort das Senden eines entsprechenden Requests. Aufgrund der Ausbreitungsverzögerung können sich noch Requests auf den Bussen befinden, obwohl die zugehörigen Datensegmente bereits gesendet wurden.

Dies kann dazu führen, daß nicht mehr die gesamte Bandbreite der Busse genutzt werden kann. Mit dieser – allerdings geringen – Einbuße an Übertragungskapazität erkauft man sich zwar eine Verbesserung bei der Fairneß des Protokolls, begünstigt aber jetzt diejenige Station, die als erste mit ihrer Übertragung begonnen hat [56, 57] (s. Abschnitt 5.1). Dies führt dazu, daß eine Station, die viel Verkehr erzeugt, während ihrer Übertragungen häufig den Bus dominiert, d. h. sie kann eine deutlich größere Bandbreite nutzen. Bei gleichförmigem Verkehr werden dadurch die Stationen in der Nähe des Headends noch immer begünstigt.

3.5. Der Bandbreiten-Ausgleichs-Mechanismus

Da die Elimination des Standby-Zustands die Probleme mit der Fairneß zwar mildern, aber nicht lösen konnte, wurde der Zugriffsmechanismus erweitert. Dazu wurde unter vielen

Vorschlägen (z. B. [9, 36]) der sog. Bandbreiten-Ausgleichs-Mechanismus (Bandwidth Balancing Mechanism) [11, 12] ausgewählt und zunächst als Option in den Standard aufgenommen [21].

Für den Bandbreiten-Ausgleichs-Mechanismus benötigt jede Station einen zusätzlichen Zähler pro Bus, den sog. bandwidth balancing counter *(BWB_CNTR)*. Für dessen Inhalt wird eine obere Schranke vereinbart, der sog. bandwidth balancing modulus *(BWB_MOD)*. Wenn die Station ein Datensegment überträgt, wird dieser Zähler unabhängig von der Priorität inkrementiert. Erreicht er sein Limit, so wird er auf null zurückgesetzt, und die Request- bzw. Countdown-Zähler sämtlicher Prioritäten werden um eins erhöht.

Jede Station verzichtet also auf einen Teil $B/(1 + BWB\_MOD)$ der Bandbreite B, die sie für ihre Datensegmente hätte nutzen können. Dafür wird die Gesamtbandbreite jetzt unter den sendenden Stationen unabhängig von der Reihenfolge aufgeteilt, in der sie ihre Übertragungen begonnen haben [12]. Der Anteil ist proportional dem für die jeweilige Station gewählten BWB_MOD [44]. Im Standard wird vorgeschlagen, generell $BWB\_MOD = 8$ zu wählen, so daß die Gesamtbandbreite gleichmäßig aufgeteilt wird. N gleichzeitig sendende Stationen können dann eine Gesamtbandbreite von $B_{ges} = C \times (N \cdot BWB\_MOD)/(N \cdot BWB\_MOD + 1) = C \times (8N)/(8N + 1)$ nutzen, wobei C die Kanalkapazität bezeichnet [12].

Wenn eine weitere Station mit einer Übertragung beginnt, bekommt sie nicht sofort ihren Anteil. Es dauert einige Zeit (die sog. transiente Phase), bis die Gesamtbandbreite neu verteilt ist (s. Abschnitt 5.1). Die Länge dieser Phase wächst sehr rasch mit dem Wert vom BWB_MOD und dem Abstand der sendenden Stationen. Sie kann bei $BWB\_MOD = 8$ und einem Abstand von 100 Zellen mehrere tausend Zellen betragen. Eine Übertragung kann also zu kurz sein, um die Bandbreite gerecht aufzuteilen. Der Bandbreiten-Ausgleichs-Mechanismus verbessert also die Fairneß bei hoher Last, kann sie aber nicht immer garantieren [52].

Ein weiteres für den Standard relevantes Problem ist, daß der Bandbreiten-Ausgleichs-Mechanismus nicht mit mehreren Übertragungsprioritäten arbeitet [42]. Normalerweise sollte der Verkehr mit hoher Priorität durch den mit niedriger Priorität kaum (im Idealfall gar nicht) beeinflußt werden. Man kann aber Konfigurationen angeben, bei denen dieser Einfluß bereits sehr deutlich ist, selbst wenn nur zwei Prioritäten benutzt werden. Es kann z. B. zu Situationen kommen, in denen die Bandbreite von Stationen ausgeglichen wird, die verschiedene Prioritäten benutzen. Falls mehrere Stationen mit hoher Übertragungspriorität senden, ist der Anteil an Bandbreite, den jede einzelne für sich nutzen kann, sehr stark von der Lage derjenigen Stationen abhängig, die nur niedrige Prioritäten benutzen [51].

Obwohl das REQ3-Bit durch ein zusätzliches reserviertes Bit ersetzt wurde, das für einige Vorschläge zur Beseitigung dieses Problems nötig wäre [13, 14], wurde der Standard nicht nochmals erweitert. Stattdessen darf jetzt für den asynchronen Zugriff nur noch die niedrigste Übertragungspriorität verwendet werden [22].

Die wichtigsten Punkte beim asynchronen Zugriff im aktuellen Standard IEEE 802.6 sind:

- Der Zugriff erfolgt nach dem Prinzip des verteilten Wartens.
- Für jedes Datensegment wird ein Request abgesetzt (kein Standby-Zustand).
- Es wird der Bandbreiten-Ausgleichs-Mechanismus (Default: $BWB\_MOD = 8$)[9] zur Verbesserung der Fairneß benutzt.
- Es darf nur die niedrigste Übertragungspriorität verwendet werden.

4. Modellierung und Analyse des DQDB-Protokolls

Durch die verteilte Zugriffsorganisation ist das DQDB-Protokoll einer exakten analytischen Untersuchung schwer zugänglich.[10] Während des Entwicklungs- und Standardisierungsprozesses wurden neben Simulationsstudien einige analytische Untersuchungen vorgestellt.

Wir werden in diesem Unterabschnitt die von zwei der Autoren dieses Beitrages in [49] vorgestellte approximative Analysemethode genauer in Betracht ziehen. Ziel dieser Untersuchung war es, einfache geschlossene Formeln für die Leistungsbeurteilung von DQDB-Systemen unter stationären Lastbedingungen zu entwickeln. Im Anschluß werden wir kurz auf zwei detailliertere Analysen eingehen, die im wesentlichen auf derselben Modellbildung basieren [4, 32].

Das DQDB-System habe N aktive Stationen, die isochrone und asynchrone Dienste in Anspruch nehmen. Obwohl die beschriebene Analyse für DQDB-Systeme mit mehreren Prioritäten angewendet werden kann, wird hier der Übersichtlichkeit halber nur die einfache Version mit einer Prioritätsebene im Detail betrachtet. Analyse und Resultate für mehrere Prioritätsklassen finden sich in [46].

Wie bei der Protokollbeschreibung im vorausgegangenen Abschnitt bereits erwähnt, ist der Datenstrom auf Bus A mit dem Requeststrom auf Bus B gekoppelt, analog der Datenstrom auf Bus B mit dem Requeststrom auf Bus A. Da sich Daten- und Requeststrom auf Bus A wie auch Bus B aber nicht gegenseitig beeinflussen, kann man die beiden Datenübertragungsrichtungen völlig voneinander trennen. Infolgedessen wird nachfolgend nur der Datenstrom auf Bus A zusammen mit dem Requeststrom auf Bus B betrachtet. Die Analyse für die andere Übertragungsrichtung kann analog erfolgen.

Ankommender Datenverkehr in den Stationen wird mit einem Poisson-Prozeß beschrieben. Die Verkehrsintensität von Station i nach Station j sei λ_{ij} ($\lambda_{ij} = 0$). Der Gesamtverkehr Λ_i von Station i, der auf Bus A übertragen werden soll, und der Gesamtverkehr Λ auf Bus A können wie folgt angegeben werden:

$$\Lambda_i = \sum_{j=i+1}^{N-1} \lambda_{ij} \quad \text{und} \quad \Lambda = \sum_{i=1}^{N-1} \Lambda_i. \tag{1}$$

Ein Anteil p_I der Bandbreite des Übertragungsmediums wird nun für isochronen Verkehr reserviert. Die Restkapazität von $(1 - p_I)$ kann für asynchrone Dienste gemäß dem DQDB-Zugriffsmechanismus benutzt werden. Wird mit τ die Übertragungsdauer einer Zelle bezeichnet, so sind die von Station i belegte Buskapazität ρ_i und die gesamte Busauslastung ρ wie folgt:

[9] Er kann mit $BWB\_MOD = 0$ deaktiviert werden.

[10] Man beachte im folgenden den fundamentalen Unterschied zwischen einer exakten Analyse von DQDB und der exakten Analyse eines – approximativen – Modells eines DQDB-Systems!

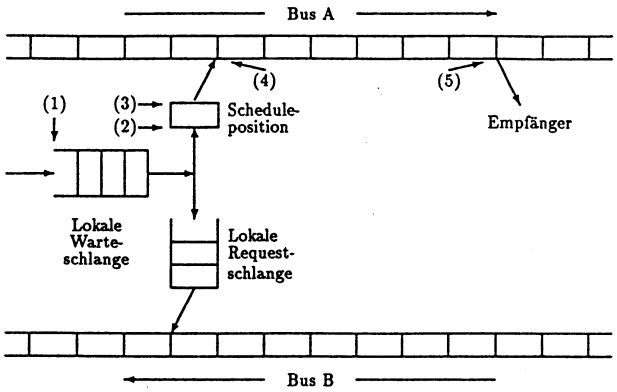

Abb. 9. Modell einer Station (nur Nutzdatentransfer auf Bus A)

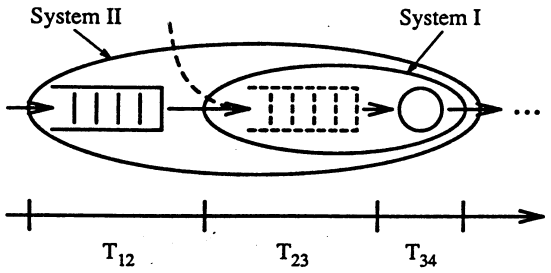

Abb. 10. Dekomposition der Transferzeit

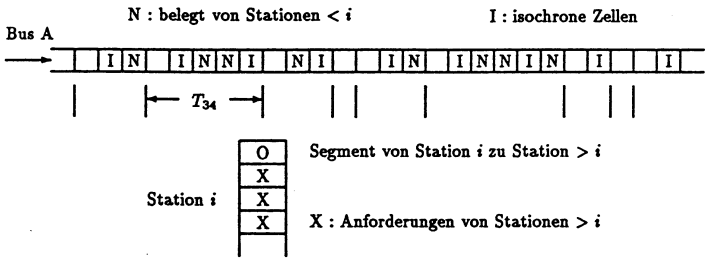

Abb. 11. Virtuelle Übertragungszeit bei Station i

$$\rho_i = \Lambda_i \cdot \tau \text{ und } \rho = \sum_{i=1}^{N-1} \rho_i. \tag{2}$$

Betrachtet werde nun ein Datensegment, das vom Sender i zum Empfänger j ($j > i$) übertragen werden soll. Die Übertragung des Datensegmentes erfolgt auf Bus A, die des dazugehörigen Request-Signals auf Bus B. Wie in Abb. 9 illustriert, werden die zur Bestimmung der Zugriffs- und Transferzeit des Segmentes signifikanten Zeitpunkte wie folgt markiert:

(1): Ankunftszeitpunkt; Datensegment wird in Station i (z. B. von höheren Protokollinstanzen) erzeugt.

(2): Schedule-Zeitpunkt; Datensegment verläßt die lokale Warteschlange und rückt in die Scheduleposition vor. Zu diesem Zeitpunkt wird für das Segment ein Request zur Reservierung einer Zelle erzeugt und in der Requestschlange abgespeichert. Nachdem alle älteren Requests abgesetzt sind, darf dieser in der nächsten verfügbaren Zelle auf Bus B übertragen werden. Das Datensegment ist nun bereit für die Übertragung, muß aber noch gemäß der globalen Abfertigungsdisziplin in der verteilten – und damit nur virtuell vorhandenen – Warteschlange warten.

(3): Übertragungsbeginn; Beginn der virtuellen Übertragungszeit des Segmentes. Das Datensegment erreicht die erste Position der (fiktiven) globalen Warteschlange, muß jedoch noch auf eine freie vorbeifließende Zelle auf Bus A warten.

(4): Übertragungsende.

(5): Empfangszeitpunkt; die für den Transfer benutzte Datenzelle ist vollständig bei Station j angekommen.

Diese Modellbetrachtung legt eine Dekomposition der Transferzeit des Datensegmentes nahe. Dazu werden folgende Zufallsvariablen (ZV) eingeführt (vgl. Abb. 10):

T_{12}: Lokale Wartezeit; ZV für die Wartezeit in der lokalen Warteschlange der Station i. Werden mehrere Prioritäten betrachtet, so hat jede Prioritätsebene eine separate Warteschlange.

T_{23}: Schedule-Wartezeit; ZV für die Wartezeit in der Schedule-Position der Station i. Diese Wartezeit hängt vom momentanen Zustand der globalen Warteschlange ab, gemäß der verteilt realisierten Warteschlangenorganisation.

T_{34}: Virtuelle Übertragungszeit auf Bus A.

T_{45}: Übertragungsverzögerung von Station i nach Station j.

Aus dieser Betrachtung ergeben sich die Buszugriffszeit T_{14} und die Transferzeit T_{15} als Additionen der beschriebenen Zufallsvariablen. Um die Verteilungsfunktion der Zugriffszeit T_{14} zu berechnen, werden zunächst mittels approximativer Modellbetrachtungen die Verteilungsfunktionen der Zeitintervalle T_{34}, T_{23} und T_{12} bestimmt.

Wenn ein Segment in Station i die erste Position in der globalen Warteschlange erreicht hat, besteht – aus der Perspektive von Station i – der vorbeifließende Zellstrom auf Bus A aus folgenden Typen von Zellen (vgl. Abb. 11):

1. Zellen, die für isochrone Dienste vorreserviert sind.
2. Zellen, die von den vorausgegangenen Stationen $1, 2, \ldots i-1$ belegt worden sind.
3. Freie Zellen, die von Station i belegt werden können, falls diese dazu berechtigt ist.

Demgemäß ist die virtuelle Übertragungszeit T_{34} eines Segments in Station i die Zeitspanne zwischen zwei Zellen vom Typ 3. Ferner wird vereinfachend angenommen, daß bei den Zellen vom Typ 1 die zeitliche Aufteilung gleichverteilt ist und sie mit einer Wahrscheinlichkeit von p_I anzutreffen sind. Für die Spezialfälle $p_I = 0\,\%$ und $p_I = 50\,\%$ [11] kann T_{34} mit einer geometrischen Verteilung wie folgt approximativ angegeben werden:

$$Pr\{T_{34} = k \cdot \frac{1}{1-p_I} \, slots\} = q_i^{k-1}(1-q_i),$$
$$\text{für } k = 1, 2, \ldots \text{ mit } q_i = \sum_{j=1}^{i-1} \frac{\rho_j}{1-p_I}. \tag{3}$$

Damit ergibt sich die zugehörige Laplace-Stieltjes-Transformierte (LST)

$$\Phi_{34}(s) = \frac{(1-q_i)\cdot z}{1-q_i \cdot z}, \text{ wobei } z = e^{-s\tau/(1-p_I)} \tag{4}$$

[11] Und allgemein für den Fall $p_I = 1 - 1/n$ mit natürlicher Zahl n.

Aus verkehrstheoretischer Sicht kann T_{34} als die Bedienzeit aller Segmente betrachtet werden, die von den Stationen $i, i+1, \ldots, N$ zur Übertragung auf Bus A generiert werden, d.h. deren Reservierungen von Station i vermerkt werden. Diese Segmente bilden somit den Ankunftsprozeß des globalen Wartesystems aus der Sicht der Station i. Dieses System wird hier als System I bezeichnet (vgl. Abb. 10). Die Wartezeit im System I ist identisch mit der Schedule-Wartezeit T_{23}. In [49] wird für das System I in erster Näherung das Warteschlangenmodell M/G/1 verwendet. Daraus ergibt sich folgende Laplace-Stieltjes-Transformierte für T_{23} (vgl. [47]):

$$\Phi_{23}(s) = \frac{s \cdot (1 - \Gamma_i \cdot ET_{34})}{s - \Gamma_i(1 - \Phi_{34}(s))},$$

$$\text{wobei } \Gamma_i = \sum_{j=i}^{N} \Lambda_j. \tag{5}$$

Eine interessante Eigenschaft des Systems I ist, daß mit steigendem Stationsindex die Verkehrsintensität abnimmt, während die mittlere Bedienzeit zunimmt. Werden mehrere Prioritäten herangezogen, kann die Analyse des Systems I anhand des Warteschlangenmodells M/G/1 mit unterbrechender Priorität (vgl. [46]) durchgeführt werden. Aus Gl. (4) und Gl. (5) kann die Laplace-Stieltjes-Transformierte des Zeitintervalls T_{24} hergeleitet werden:

$$\Phi_{24}(s) = \Phi_{23}(s) \cdot \Phi_{34}(s). \tag{6}$$

Das Intervall T_{24} bildet die virtuelle Bedienzeit von Datensegmenten, die im lokalen Speicher der Station i eintreffen und auf Bus A übertragen werden. Dieses lokale Wartesystem wird als System II bezeichnet (vgl. Abb. 10). Die Bedienzeit des Systems II ist T_{24}, der Ankunftsprozeß ist identisch mit dem Ankunftsprozeß der betrachteten Station, und die resultierende Wartezeit ist die lokale Wartezeit T_{12}. Die Analyse von System II wird wieder mit einem M/G/1-System approximativ durchgeführt. Die Laplace-Stieltjes-Transformierte der lokalen Wartezeit lautet demgemäß:

$$\Phi_{12}(s) = \frac{s \cdot (1 - \Lambda_i \cdot ET_{24})}{s - \Lambda_i(1 - \Phi_{24}(s))}. \tag{7}$$

Man erhält schließlich für die Zugriffszeit einer Segmentes:

$$\Phi_{14}(s) = \Phi_{12}(s) \cdot \Phi_{23}(s) \cdot \Phi_{34}(s). \tag{8}$$

Die gesamte Transferzeit T_{15} eines Datensegmentes kann dann aus der Zugriffszeit und der Übertragungsverzögerung zwischen Station i und Station j ermittelt werden.

Die Autoren von [4] legen nun ein leicht modifiziertes Modell zugrunde und schlagen eine Dekomposition der Busy- und Requestströme vor, um die Komplexität der Analyse des Modells im Vergleich zu einer exakten Lösung – wie in [32], siehe weiter unten – zu reduzieren. Dabei wird die virtuelle Bedienzeit des lokalen Wartesystems, also die Durchlaufzeit des inneren Systems I mit Hilfe eines M/G/1-Systems, bei dem jeweils der erste Job während einer Busy-Phase eine spezielle Bedienzeit[12] erfährt, näherungsweise bestimmt. Die Zugriffsverzögerung eines Knotens wird dann wieder über das Lösen eines herkömmlichen M/G/1-Systems mit der genannten Bedienzeit ermittelt.

Der Vorteil dieser Analyse liegt vor allem darin, daß sie die Explosion des Zustandsraumes, die mit dem in [32] vorgestellten Verfahren verbunden ist, vermeidet. Dies geht schon in einem der einfachsten Fälle mit dem Lösen einer sechsdimensionalen Markow-Kette einher, die – bei zwei aktiven Stationen und einer Entfernung von 6 Zellen zwischen den Stationen – bereits über 90000 erreichbare Zustände enthält. Nach Aussage der Autoren ist das Verfahren damit auch nur bei sehr kompakten Systemen mit sehr wenigen Stationen anwendbar.

Es soll nicht verschwiegen werden, daß die geschilderten Analysen zunächst nur gemäß einem Poisson-Strom ankommende einzelne Segmente betrachten, was natürlich nur bedingt den in einem solchen System zu erwartenden Datenstrom annähern kann. Dennoch lassen sich wesentliche Charakteristika im Leistungsverhalten durchaus hinreichend genau feststellen.

Eine direkte Erweiterung z.B. auf Poisson-verteilte Ankünfte mit beliebig verteilten Paketgrößen scheint jedenfalls problematisch zu sein (s. dazu die Ergebnisse in [29, 30]). Mit Hilfe zeitdiskreter Analyseverfahren ist allerdings eine Verallgemeinerung der oben ausführlich dargestellten Analyse auf beliebig verteilte Zwischenankunftszeiten möglich (s. [45]). Diese Modifikation läßt dann aber keine Untersuchung mehrerer Prioritäten zu.

5. Zugriffsverhalten

Wir wollen in diesem Abschnitt näher auf das Zugriffsverhalten des DQDB-Protokolls eingehen. Da in diesem Zusammenhang immer wieder von Unfairneß die Rede ist, sollen am Anfang einige Betrachtungen für den saturierten Fall stehen (5.1). Hier wird die grundlegende Problematik beim verteilten Warten aufgezeigt.

Im Anschluß (5.2 und 5.3) werden wir einige Ergebnisse für das statistische Gleichgewicht präsentieren, die auf der folgenden MAN-Konfiguration basieren:

- Länge des Systems: 100 km.
- 49 Stationen, in konstanten Abständen an das System angeschlossen.
- Nettoübertragungskapazität der Busse: 136 Mbps.
- Länge der Zellen: 53 Byte (1 Byte ACF, 4 Byte Adresse, 48 Byte Nutzdaten).
- Eine isochrone Grundlast von 50 %.

In den zu Validierungszwecken markierten Simulationsergebnissen werden Konfidenzintervalle mit 95 %-Quantil dargestellt. Wenn nicht anders vermerkt, beziehen sich die angegebenen Zugriffsverzögerungen auf den Nutzdatenverkehr auf Bus A.

5.1. Fairneß-Analyse für den saturierten Fall

Im saturierten Fall, in dem jede Station immer (mindestens) ein Datensegment zu übertragen hat (z.B. während langer File-Transfer-Anwendungen), betrachtet man die Aufteilung der Gesamtbandbreite unter den einzelnen Stationen. Die Effekte, die zu unfairem Verhalten führen, lassen sich hier sehr deutlich beobachten und einfach erklären, denn im Gegensatz zur Analyse im statistischen Gleichgewicht braucht man keine zufallsabhängigen Größen zu betrachten. Um unfaires Verhalten zu quantisieren, betrachten wir den Quotienten aus den Bandbreiten der beiden Stationen, die den größten bzw. den kleinsten Anteil an Bandbreite erhal-

[12] „M/G/1 system with an exceptional first service time" (s. [47]).

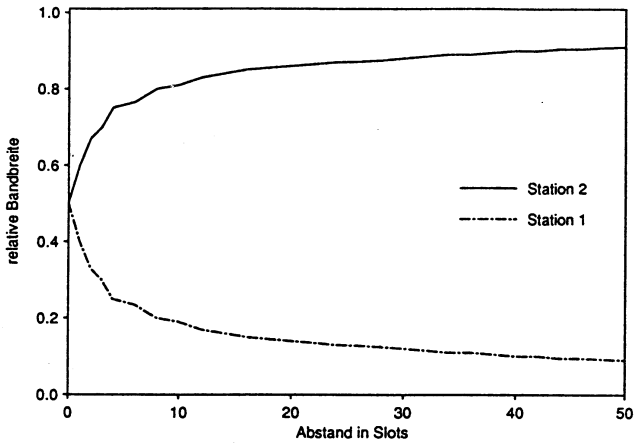

Abb. 12. Bandbreitenaufteilung, falls die untere Station (2) zuerst zu senden beginnt

ten: $B^* = B_{max}/B_{min}$. Das Protokoll ist „fair", falls gilt: $B^* = 1$; je größer B^* ist, desto „unfairer" verhält es sich.

Um Unfairneß zu bestimmen, betrachten wir den einfachsten Fall, nämlich daß nur zwei Stationen auf dem betrachteten Bus A senden können. Der Abstand zwischen den beiden betrage d Zellen. Ferner sollen alle Datensegmente die gleiche Priorität besitzen, und auf dem Bus sollen sich anfangs nur leere Zellen befinden. Wenn eine Station nun eine Übertragung beginnt, so benutzt sie alle freien Zellen, da ihr „Countdown"-Zähler null ist.

5.1.1. Protokollversion mit „Standby"-Zustand

Nehmen wir an, die obere Station sende bereits seit mindestens d Zeiteinheiten (= Übertragungszeit für eine Zelle) und befinde sich im saturierten Zustand. Die andere Station sieht dann nur noch belegte Zellen. Wenn sie jetzt ebenfalls mit einer Übertragung beginnt, so geht sie zunächst in den Standby-Zustand über. Einen Request sendet sie erst eine Zeiteinheit später, da sie dann eine bereits benutzte Zelle erreicht. Dieser Request benötigt d Zeiteinheiten, bis er die obere Station erreicht. Dort beeinflußt er zunächst nur den „Request"-Zähler, so daß eine weitere Zeiteinheit vergeht, bevor eine einzelne freie Zelle durchgelassen wird. Diese Zelle braucht wieder d Zeiteinheiten, um die untere Station zu erreichen. Nachdem diese nun ihr erstes Datensegment senden kann, wiederholt sich der Vorgang für alle weiteren.

Die untere Station kann somit von $2d + 2$ Zellen eine einzige benutzen. Es gilt also [53]:

$$B_{min} = \frac{1}{2d+2}, \quad B_{max} = \frac{2d+1}{2d+2} \quad \Rightarrow B^* = 2d+1 \tag{9}$$

Auch wenn die untere Station zuerst mit einer Übertragung beginnt, stellt sich d Zeiteinheiten, nachdem auch die obere zu senden angefangen hat, die obige Bandbreitenaufteilung ein. Denn die untere Station erzeugt zunächst keine Requests auf dem anderen Bus, da für ihre Datensegmente bereits im „Standby"-Zustand eine freie Zelle ankommt. Daher kann die obere Station bei Beginn ihrer Übertragung beliebig auf den Bus A zugreifen. Erst nachdem die erste bereits benutzte Zelle bei der unteren angekommen ist, wird ein Request gesendet. Damit hat man wieder die oben beschriebene Situation vor sich.

Man kann diese Version also nur für $d = 0$ als fair bezeichnen. MANs haben aber im allgemeinen logische Buslängen von $d > 100$. Bei der Protokollversion mit „Standby"-Zustand gilt also [53]:

- Es werden unabhängig von der Reihenfolge, in der Stationen ihre Übertragungen beginnen, die Stationen in der Nähe des Headend des betrachteten Busses (obere Stationen) bevorzugt.
- Die Bandbreitenaufteilung ist vom Abstand der gleichzeitig aktiven Stationen abhängig. Für zwei aktive Stationen gilt: $B^* = 2d + 1$.

5.1.2. Protokollversion ohne „Standby"-Zustand

Wenn die obere Station bereits d Zeiteinheiten vor der unteren zu senden begonnen hat, verhält sich diese Protokollversion ähnlich wie die oben beschriebene. Der einzige Unterschied besteht darin, daß die untere Station jetzt einen Request absetzt, sobald sie Datensegmente übertragen will. Es dauert daher „nur" noch $2d + 1$ Zeiteinheiten, bis sie eine freie Zelle erhält. Es gilt also:

$$B_{min} = \frac{1}{2d+1}, \quad B_{max} = \frac{2d}{2d+1} \quad \Rightarrow B^* = 2d \tag{10}$$

Völlig anders verhält sich diese Protokollversion, wenn die untere Station zuerst zu senden beginnt [56, 57]: Nach d Zeiteinheiten sieht die obere alle Request-Bits auf Bus B gesetzt. Solange diese nicht sendet, ist ihr Request-Zähler abwechselnd eins (ein Request fließt auf Bus B vorbei) und null (die nächste freie Zelle auf Bus A fließt vorbei). Nehmen wir an, daß der Request-Zähler null ist, wenn die obere ihre Übertragung beginnt. Sie kann also die nächste freie Zelle sofort benutzen. Zum Schreiben ihres Datensegments benötigt sie jedoch eine Zeiteinheit, so daß danach der Request-Zähler eins enthält. Dieser Wert wird sofort in den Countdown-Zähler geladen, und der Request-Zähler wird zurückgesetzt, da die Station saturiert ist. Sie muß daher eine Zeiteinheit warten, bis sie mit dem Senden auf Bus A beginnen darf ($CD = 0$). Zudem benötigt sie eine weitere Zeiteinheit, um ihr Datensegment vollständig auf den Bus zu schreiben. Da pro Zeiteinheit ein Request eintrifft, ist der Inhalt des Request-Zählers nach einer Übertragung um eins größer als bei der vorhergehenden. Die obere Station muß also zwischen dem Schreiben von Datensegmenten von Mal zu Mal länger warten (d. h. ihre relative Bandbreite sinkt), bis ihre belegten Zellen die untere Station erreichen, wodurch diese wiederum daran gehindert wird, die Request-Bits aller Zellen auf Bus B zu setzen. Simulationen haben gezeigt, daß sich die Bandbreiten der beiden Stationen wie in Abb. 12 einpendeln [57].

Um eine Abschätzung für die Bandbreite der oberen Station zu bekommen, betrachten wir zunächst, wieviele Zellen sie in den ersten $d + 1$ Zeiteinheiten benutzen kann:

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2} = d+1 \Rightarrow n = \frac{\sqrt{8d+9}-1}{2} \tag{11}$$

Zu Beginn ihrer Übertragung kann die obere Station mehr Bandbreite benutzen als später, so daß Gl. (11) ihre mittlere Bandbreite überschätzt. Jetzt ist die erste Lücke im Requeststrom der unteren Station entstanden, die aber die obere erst in d Zeiteinheiten erreicht. Sie kann in den nächsten $d + 1$ Zeiteinheiten nur relativ wenige Zellen benutzen, da sie immer länger warten muß, bis sie eine freie Zelle benut-

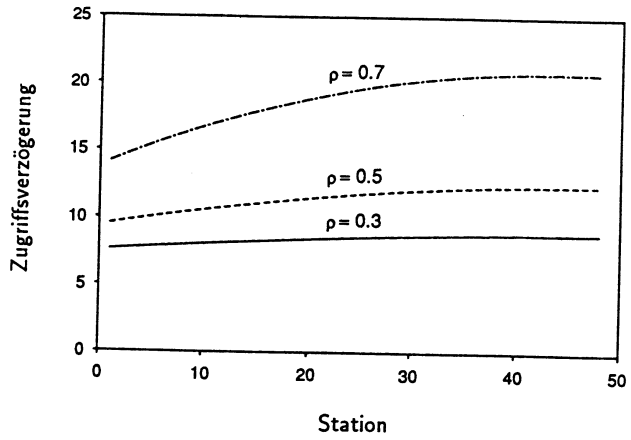

Abb. 13. Abhängigkeit der Zugriffszeiten von der Position am Bus

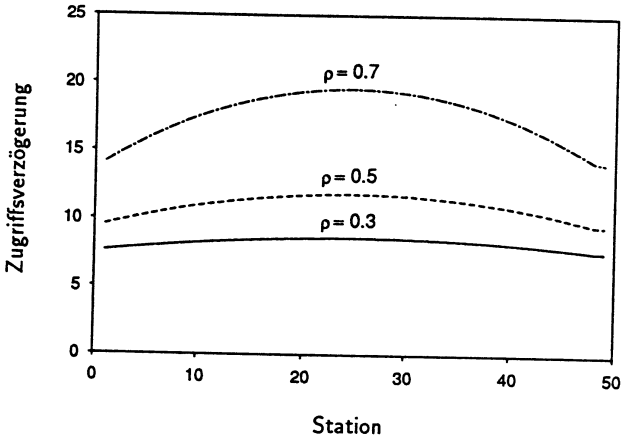

Abb. 14. Zugriffszeiten für beide Busse

zen darf. Wir nehmen vereinfachend an, daß sie in dieser Zeitspanne etwa so viele Zellen nutzen kann, wie sie am Anfang über ihre mittlere Bandbreite hinaus belegt hat. Damit erhalten wir die folgende Abschätzung für $B_{\min}$ [7]:

$$B_{\min} = \frac{\sqrt{8d+9}-1}{2(2d+2)} \qquad (12)$$

Da in einem MAN die Distanzen zwischen gleichzeitig aktiven Stationen d im allgemeinen recht groß sind, kann man hier die gröbere Abschätzung $B_{\min} \approx 1/\sqrt{2d}$ [57] gut verwenden. Man erhält damit $B^* \approx \sqrt{2d}$.

Wenn beide Stationen gleichzeitig mit ihrer Übertragung beginnen, gleichen sich die oben beschriebenen Effekte aus. Sie teilen sich nach einiger Zeit die Bandbreite gleichmäßig und greifen in den meisten Fällen sogar auf jede zweite Zelle zu [57].

Zusammenfassend kann man über die Protokollversion mit zwei aktiven Stationen folgendes sagen:

- Die Unfairneß ist weniger gravierend als beim ursprünglichen Protokoll, aber noch immer recht deutlich.
- Die Aufteilung der Bandbreite ist abhängig von der Reihenfolge, in der Stationen zu senden beginnen.
- Falls diejenige Station, die später zu senden beginnt, weiter oben als die bereits aktive sitzt, erhält sie mehr Bandbreite ($\approx 1/\sqrt{2d}$) als eine, die weiter unten liegt ($1/(2d+1)$).

5.1.3. Bandbreiten-Ausgleichs-Mechanismus

Beim Bandbreiten-Ausgleichs-Mechanismus (Bandwidth-Balancing Mechanism, BWB) benutzt eine einzelne, sendende Station $S1$ nicht die gesamte Kanalkapazität C, sondern nur $\gamma_1 = C \times M/(M+1)$. Wenn eine weitere Station $S2$ eine Übertragung beginnt, kann sie $\gamma_2 = (1-\gamma_1) \times M/(M+1)$ verwenden, um Datensegmente und Requests zu senden. Da $S1$ die Requests von $S2$ berücksichtigen muß bzw. die von $S2$ bereits genutzten Zellen nicht verwenden kann, sinkt die Bandbreite von $S1$ auf $\gamma_1 = (1-\gamma_2) \times M/(M+1)$. Dadurch bleibt jetzt mehr Bandbreite für $S2$ übrig, so daß diese $S1$ weiter bremsen kann. Nach einer gewissen Zeit (transiente Phase) pendeln sich die Bandbreiten so ein, daß jede der beiden je $\gamma_i = C \times M/(2M+1)$ erhält. Wenn N Stationen gleichzeitig aktiv sind, erhält nach der Einschwingphase jede Station $\gamma_i = C \times M/(NM+1)$. Für den Bandbreiten-Ausgleichs-Mechanismus gilt also:

- Alle gleichzeitig aktiven N Stationen erhalten die gleiche Bandbreite $\gamma_i = C \times M/(NM+1)$, unabhängig von ihrer Lage und der Reihenfolge, in der sie ihre Übertragungen begonnen haben.
- Der Bandbreiten-Ausgleichs-Mechanismus benötigt eine gewisse Zeit, um faire Bedingungen herzustellen (s. Abschnitt 3.5).
- Es kann nicht mehr die gesamte Kanalkapazität genutzt werden, sondern $C \times NM/(NM+1)$ von N gleichzeitig aktiven Stationen.

5.2. Zugriffsverzögerung von DQDB ohne Bandbreiten-Ausgleich

Im folgenden legen wir symmetrische Verkehrsannahmen zugrunde, d. h. die Verkehrsintensitäten λ_{ij} aller Stationen (vgl. Abschnitt 4) sind identisch, $\lambda_{ij} = \lambda$. Selbst unter dieser Annahme zeigt sich, daß die mittlere Zugriffszeit stark von der Position der Station am Bus abhängig ist (s. Abb. 13). Das Maximum für eine relativ hohe Last von 80 % (bezogen auf die nicht für isochrone Dienste benutzte Bandbreite) wird bei Station 40 erreicht, wo die Kombination aus Ankünften von Requests auf Bus B und freien Zellen auf Bus A die schlechtesten Ergebnisse liefert. Wenn man die Mittelwerte für beide Transferrichtungen geeignet aufsummiert, ergibt sich eine symmetrische Kurve, die ihr Maximum bei Station 25 annimmt (vgl. Abb. 14), d. h. in einem DQDB-System haben – unter obigen Vorgaben – die Stationen am oberen bzw. unteren Ende des Systems eindeutige Vorteile gegenüber den in der Mitte lokalisierten Stationen. Dieses Verhalten kann sich allerdings auch umkehren, wenn man nicht mehr Poisson-Ströme, sondern Ströme mit deutlich höherer Varianz zugrunde legt (s. [41]).

Solange man sich allerdings nicht im Hochlastbereich bewegt, sondern das System nur bis etwa 60 % belastet, zeigen sich zwischen den Zugriffszeiten der Stationen nur marginale Unterschiede (Abb. 13 und 15). Zudem wächst die mittlere Zugriffszeit bis $\rho = 0{,}6$ kaum mehr als linear – mit schwacher Steigung (Abb. 15). Das System zeigt also ein ausgesprochen gutartiges Verhalten im Niedrig- und Mittellastbereich.

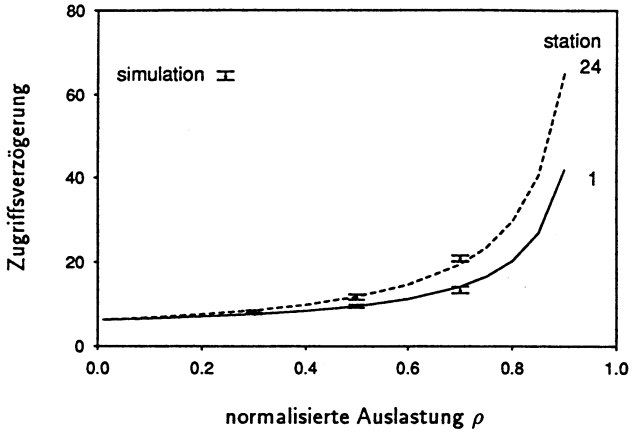

Abb. 15. Abhängigkeit der Zugriffszeiten von der nonisochronen Last

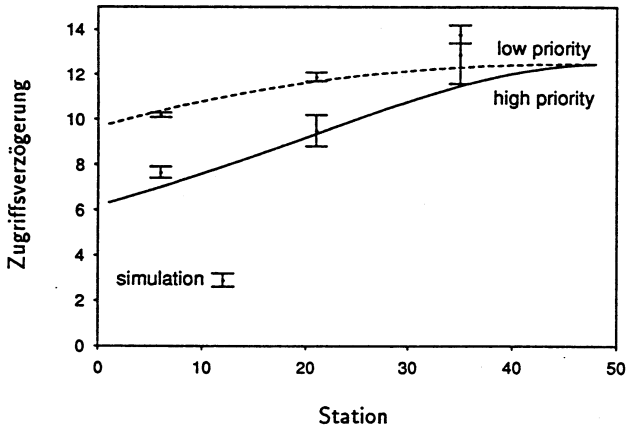

Abb. 16. Erweiterung auf zwei Prioritäten

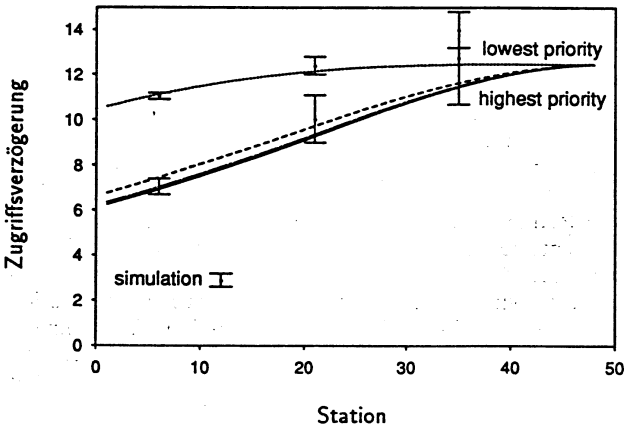

Abb. 17. Vier Prioritätsklassen

5.3. Zugriffsverhalten bei Verwendung mehrerer Prioritäten

Im aktuellen Standard 802.6-DQDB sind explizit keine Zugriffsmechanismen für mehrere Prioritätsklassen enthalten, sie waren allerdings in früheren Drafts vorgesehen. Da aber weiterhin im Access Control Field drei Request-Bits vorhanden sind und die Stationen weiterhin über drei Zustandsautomaten verfügen, eine Station aber derzeit nur die Request-Bits der niedrigsten Prioritätsklasse nutzen darf, kann über eine zukünftige Unterstützung mehrerer Prioritäten bzw. Verkehrsklassen spekuliert werden (siehe z. B. [15]). Wir werden daher im folgenden auch einige Resultate für mehrere Prioritätsklassen präsentieren, die allerdings auf den ursprünglichen Entwürfen basieren (bis Draft 12 [21], in dem der Bandwidth-Balancing Mechanism bereits als Option vorhanden war).

In Abb. 16 werden zwei Prioritäten betrachtet. Von der gesamten nichtisochronen Last von 50 % – bezogen auf die nicht für isochrone Dienste benutzte Restbandbreite – entfallen 5 % auf die hohe, die restlichen 95 % auf die niedrige Priorität. Man erkennt auch hier wieder, daß die Zugriffsverzögerung stark stationsabhängig ist. Insofern wird das unfaire Verhalten von DQDB widergespiegelt. Allerdings unterscheiden sich die Verzögerungen für die beiden Klassen an den unteren Stationen kaum, während sie bei den oberen Knoten signifikante Differenzen aufweisen, d. h. eine weit unten gelegene Station hat keine Vorteile, wenn sie eilige Segmente mit höherer Priorität versieht.

Abbildung 17 zeigt eine ähnliche Situation mit vier Prioritätsklassen, wobei die Aufteilung der Klassen in Teilen von 0,01, 0,04, 0,15 und 0,80 (von hoher nach niedriger Priorität) erfolgte. Die Konfidenzintervalle beziehen sich auf die Klassen 0 und 2 (niedrigste und zweithöchste Priorität). Im Prinzip zeigt sich das gleiche Bild wie in Abb. 16: Die Gradienten der Prioritäten 1 bis 3 sind deutlich größer als die der niedrigsten Klasse. Andererseits unterscheiden sich Werte für die Klassen 1 bis 3 nicht deutlich, so daß ein Verzicht auf eine oder zwei Prioritäten zumindest in der ursprünglichen Form wohl kaum nachteilig wäre.

6. Zukünftige Entwicklungen

6.1. Erweiterungen zum Bandbreiten-Ausgleichs-Mechanismus

Beim Bandbreiten-Ausgleichs-Mechanismus, wie er im Standard definiert ist, kann man nur allen Stationen die gleiche Bandbreite zuteilen. Manchmal wäre aber eine flexiblere Aufteilung wünschenswert. Zudem würden manche Anwender und Netzbetreiber gerne weiterhin mehrere Übertragungsprioritäten benutzen.

6.1.1. Flexible Bandbreitenaufteilung

Da beim aktuellen Standard nur Übertragungen mit einer Priorität erlaubt sind, brauchen die Netzbetreiber eine andere Methode, um die Bandbreite für die inviduellen Benutzer kontrollieren zu können. Man kann dies z. B. auf einfache Weise dadurch erreichen, daß man den Stationen verschiedene BWB_MOD zuordnet, denn die Bandbreite γ_i der Station i ist proportional ihrem BWB_MOD_i und beträgt [44]:

$$\gamma_i = \frac{BWB\_MOD_i}{1 + \sum_{j=1}^{N} BWB\_MOD_j} \tag{13}$$

wobei N die Anzahl der aktiven Stationen ist.

Eine Möglichkeit, jeder Station eine maximale Bandbreite zuzuteilen, ist in [1] beschrieben: Zunächst berechnet man, welchen Anteil α_i der noch verfügbaren Bandbreite die Station i benutzen darf, also $\alpha_i = B_{MAX_i}/B_i$, wobei B_i die Bandbreite darstellt, die sie ohne Kontrolle nutzen könnte. Dann wählt man zwei ganze Zahlen β_i und θ_i so, daß $\alpha_i \approx \beta_i/(\beta_i + \theta_i)$ gilt. Der Netzbetreiber weist nun jeder Station ihr β_i als BWB_MOD_i zu und setzt ein zusätzlich benötigtes Register auf θ_i. Jedesmal, wenn die Station i β_i Zellen benutzt hat (d.h. $BWB\_CNT_i = 0$), werden der Request- bzw. Countdown-Zähler um θ_i statt um eins inkrementiert. Dadurch ist gewährleistet, daß keine Station mehr als die ihr zugeteilte (und von ihrem Betreiber bezahlte) Bandbreite nutzen kann. In [1] sind Beispiele für verschiedene Fairneß-Definitionen angegeben.

6.1.2. Bandbreiten-Ausgleich mit mehreren Prioritäten

Es gibt auch einige Vorschläge, den Bandbreiten-Ausgleichs-Mechanismus so zu erweitern, daß er mit mehreren Prioritäten arbeiten kann [13, 14, 35, 43]. Das Problem beim Bandbreiten-Ausgleich mit mehreren Übertragungsprioritäten besteht darin, daß jeder Knoten zwar durch die Request-Bits Informationen über die von den unteren Stationen benötigten Zellen pro Priorität besitzt, aber nicht feststellen kann, wieviel Bandbreite bereits von den oberen Stationen pro Priorität genutzt wurde. Es gibt zwei Möglichkeiten, ein Gleichgewicht an Information zu schaffen:

- Reduktion der Anzahl der Request-Bits auf eins. Dadurch kennt jede Station nur noch die Aufteilung ihres eigenen Verkehrs auf die verschiedenen Prioritäten (lokale Information).
- Einführung einer Kennung, die es erlaubt festzustellen, welche Priorität das Datensegment einer belegten Zelle hat. Dies kann z. B. mit den beiden reservierten Bits im Zugriffskontrollfeld realisiert werden (globale Information).

Bei den Methoden mit lokaler Information wird die oben gezeigte Eigenschaft genutzt, daß die Bandbreite proportional BWB_MOD aufgeteilt wird. Man weist nun allen Stationen (verschiedene) BWB_MOD_p für die Prioritäten $p = 0, 1, 2, \ldots$ zu. Wenn man den BWB_MOD um so größer wählt, je höher die assoziierte Priorität ist, erzielt eine Station mit hoher Priorität eine größere Bandbreite als eine mit niedriger Priorität. Die Bandbreiten gleichzeitig mit gleicher Priorität sendender Stationen werden weiterhin entsprechend dem Bandbreiten-Ausgleichs-Mechanismus gleichmäßig unter diesen aufgeteilt [13]. Diese Methode hat den Nachteil, daß man sehr große BWB_MOD für hohe und recht kleine für niedrige Prioritäten wählen muß, um die hohen Prioritäten effektiv nutzen zu können. Dies führt zu langen transienten Phasen bei hohen und zu geringer Bandbreitenausnutzung bei niedrigen Prioritäten.

Falls die Stationen über globale Informationen verfügen, wird allen Stationen der gleiche BWB_MOD zugewiesen. Eine Station, die Datensegmente der Priorität p senden will, wartet, bis eine Zelle vorbeikommt, die kein Datensegment mit gleicher oder höherer Priorität enthält. Erst dann darf die Station BWB_MOD Datensegmente der Priorität p übertragen. Dadurch werden zunächst die Bandbreiten der mit der höchsten Priorität sendenden Stationen ausgeglichen, so als ob es keinen anderen Verkehr gäbe. Die Stationen mit der nächsten Priorität können also nur die infolge des Bandbreiten-Ausgleichs-Mechanismus ungenutzte Restbandbreite für ihre Übertragungen verwenden. Dadurch erreicht man hohe Effektivität für hohe Prioritäten. Auch hier erhalten alle Stationen, die mit gleicher Priorität senden, dieselbe Bandbreite [14]. Die Implementierung dieser Methode ist jedoch aufwendiger. Eine detaillierte Beschreibung der Methoden und ihrer Implementierung sowie ein Vergleich mit Hilfe von Simulationsergebnissen findet sich in [15].

6.2. *„Slot Reuse" und „Erasure Nodes"*

In allen zellbasierten Bussystemen kann ein Leistungsgewinn erzielt werden, wenn bereits empfangene Zellen möglichst bald wieder freigegeben werden (d.h. das Busy-Bit wird zurückgesetzt), statt sie belegt bis zum Busende laufen zu lassen. Dadurch wird es möglich, eine Zelle mehrfach zu benutzen, wodurch wiederum die verfügbare Bandbreite wächst.

6.2.1. Mögliche Funktionsweisen

Es gibt im Prinzip zwei Möglichkeiten zur Freigabe von bereits benutzten Zellen:

- Jede Station darf die an sie adressierten Zellen sofort freigeben. Dies wird Destination Release (Freigabe durch Empfänger) genannt und erfordert, daß alle Stationen entgegen der eigentlichen Logik des Protokolls aktiv an das Medium gekoppelt sind. Eine Ankoppelung über ein ODER-Glied ist deswegen nicht mehr möglich.
- Nur einige spezielle Stationen dürfen bereits benutzte und empfangene Zellen wieder freigeben. Solche Stationen werden Erasure Nodes genannt und können in normale Stationen integriert sein.

Ob eine Zelle bereits empfangen wurde, kann durch die Zieladresse oder mittels einer speziellen Kennung (bei DQDB das PSR-Bit) entschieden werden. Bei der Freigabe durch den Empfänger reicht im allgemeinen[13] ein Vergleich der Zieladresse mit der eigenen Adresse aus, um zu entscheiden, ob die Zelle empfangen und danach freigegeben werden kann.

Ein Erasure Node müßte also die Zieladresse mit denen aller Stationen vergleichen, die auf dem Bus oberhalb dieses speziellen Knotens liegen. Dies kann bei großen Netzen mit sehr viel Aufwand verbunden sein. Man bevorzugt daher eine Kennung im Zugriffskontrollfeld, um Zellen als bereits empfangen zu markieren. Damit nicht jede belegte Zelle in einer Station solange verzögert werden muß, bis die Zieladresse ausgewertet ist, wird diese Kennung in die direkt auf sie folgende Zelle eingetragen. Dies hat natürlich zur Folge, daß Erasure Nodes jede Zelle bis zum Eintreffen dieser Kennung verzögern müssen.

Ein Problem bei der Freigabe von Zellen ist die Behandlung von Requests auf dem entgegengesetzt gerichteten Bus. Wenn man nur belegte Zellen, aber keine Request-Bits freigibt, treten in der Folge viele Requests auf, die bereits durch Wiederverwendung von Zellen veraltet sind. Die globale

[13] Falls ein Datensegment an eine Gruppe von Stationen adressiert ist, darf nur die letzte Station die Zelle freigeben bzw. als empfangen markieren.

Warteschlange kann also nicht mehr korrekt an allen Stationen mit ihren Zählern gebildet werden. Dies kann zur Verschlechterung des Fairneß-Verhaltens führen, wodurch die Stationen im unteren Bereich des Busses bevorteilt werden. Werden auf der anderen Seite zuviele Requests rückgängig gemacht (z.B. indem man für jede freigegebene Zelle das nächste Request löscht), kann es vorkommen, daß einige Stationen im unteren Bereich nicht genügend freie Zellen erhalten. Es gibt bereits einige Vorschläge für die Behandlung der Requests [37, 38, 58].

6.2.2. Optimale Positionierung

Man kann mit relativ – im Vergleich zur Anzahl der Stationen – wenigen geschickt positionierten Erasure Nodes Bandbreitengewinne erzielen, die mit denen bei der Freigabe durch den Empfänger vergleichbar sind. Ein Algorithmus zur Berechnung der optimalen Positionen von Erasure Nodes und des erzielten Gewinns wird in [10] vorgestellt. Man kann auch zeigen, daß mit einer bestimmten Anzahl von *optimal* plazierten Erasure Nodes der gleiche Gewinn wie bei der Freigabe durch den Empfänger erzielt werden kann. Diese Anzahl hängt stark von der Verkehrsmatrix eines betrachteten Netzes ab [38].

6.3. Multirequest-Verfahren – DQDB-ähnliche Protokolle

Eine der Ursachen für das unfaire Verhalten von DQDB ist, daß eine Station immer nur eine Zelle in der eigenen Schedule-Position haben darf. Es wurden daher einige Modifikationen vorgestellt, die genau an diesem Punkt ansetzen und von denen im folgenden zwei Varianten kurz beschrieben werden.

Kamal stellt in [26] mit dem sog. DQDB/SR eine Version vor, bei der das Schedulen nicht mehr auf Basis von Zellen, sondern von ganzen Paketen erfolgt, d.h. wenn ein Paket in die verteilte Schlange eingereiht wird, werden die entsprechenden Zähler nicht nur um eins erhöht, sondern um die Anzahl von Zellen, die das Paket lang ist. Das Absetzen von Requests erfolgt allerdings nach wie vor zellweise.

Eine weitere Ergänzung ist das sog. Continuation-Bit-Verfahren, das z.B. in [32] beschrieben wird und ebenfalls in DQDB/SR Verwendung findet. Dabei werden die Zellen eines Paketes über Marken sowie eine Start- und eine Ende-Zelle als zusammengehörig gekennzeichnet und so die vollständige Adreßinformation in der ersten (und der letzten) Zelle einer Nachricht konzentriert, mithin wird der ansonsten erforderliche Overhead auf ein Minimum reduziert. Zudem ist dieses Verfahren (DQDB/SR – Slot Reuse) noch mit einem Destination-Release-Mechanismus ausgestattet, der durch einen zusätzlichen Zähler möglich wird, was hier aber nicht näher beschrieben werden soll.

Noch einen Schritt weiter gehen Müller et al. in [31] mit DQMA (Distributed Queue Multiple Access), das Requests für ganze Pakete zuläßt. Dazu wird das ACF modifiziert, indem man u.a. das Requestfeld auf 8 bit erweitert und eine Station bis zu 255 Zellen auf einmal reservieren läßt, wenn sie Zugriff auf ein leeres Requestfeld (= 0) hat. Durch zyklisches Ändern eines zusätzlichen 2-bit-Prioritätsfeldes im ACF durch das Headend sind weiterhin Prioritäten möglich. Dadurch werden natürlich Kontrollverfahren notwendig, um ein Überlaufen der Requestschlangen zu vermeiden. Man kann dies allerdings recht einfach z.B. über einen Fenstermechanismus realisieren.

Zusätzlich können über das Einfügen von Verzögerungselementen im Requestpfad die Zellen eines Paketes nicht nur – wie beim Continuation-Bit-Verfahren – mit vermindertem Overhead, sondern sogar in aufeinanderfolgenden Zellen übertragen werden. Dies vereinfacht die Reassemblierung der Pakete im Empfänger zwar erheblich, bringt aber natürlich unter Umständen deutliche Verzögerungen für einzelne Zellen mit sich.

6.4. SMDS

Gerade in den USA scheint die Notwendigkeit zur Verknüpfung der enorm vielen LAN-Inseln immer dringlicher zu werden. Man hat sich daher dazu entschlossen, bis zu einer breiten Verfügbarkeit von B-ISDN einen landesweit zugreifbaren Datendienst einzurichten: SMDS – Switched Multimegabit Data Service (vgl. [3, 5]).

SMDS ist, um einem häufigen Mißverständis vorzubeugen, nicht identisch mit IEEE 802.6; es werden z.B. weder isochrone noch verbindungsorientierte Dienste unterstützt. Vielmehr wird lediglich ein verbindungsloser Datagrammdienst angeboten, der allerdings gegenüber IEEE 802.6 um verschiedene Punkte erweitert wurde. So verwendet man logische Adressen von 60 + 4 bit Länge – die physikalischen Adressen im Zell-Header sind analog DQDB 4 Byte lang –, die sowohl Gruppenadressierung als auch Mehrpunktdienste ermöglichen: Bis zu 16 SMDS-Adressen können mit einem SNI[14] identifiziert werden. Zudem sind z.B. auch Funktionalitäten in den Bereichen Netzmanagement und Netzsicherheit in den Spezifikationen enthalten.

Trotz der genannten und weiterer Differenzen sind SMDS und IEEE 802.6 auf der Ebene des verbindungslosen MAC-Dienstes kompatibel, so daß DQDB-MANs durchaus über öffentliche SMDS-Systeme gekoppelt werden können.

Bisher sind auf Level 1 bereits zwei Zugriffsraten definiert, für die auch schon Produkte, z.B. Router, angekündigt oder verfügbar sind:

- DS 3 für 44,736 Mbps (T3-Leitung)
- DS 1 für 1,544 Mbps (T1-Leitung).

Es ist abzusehen, daß in Kürze auch Spezifikationen für andere Transferraten – man denke insbesondere an SONET (synchronous optical network) [27] – festgelegt werden, was eine breite Akzeptanz von SMDS sicherstellen dürfte.

6.5. GFC – MAN als Zubringer für B-ISDN

Neben den eher pragmatischen Ansätzen, die sich in Entwicklungen wie SMDS niederschlagen, wird in den Gremien des CCITT intensiv an der Anbindung von Endgeräten an das zukünftige B-ISDN gearbeitet. Diese Anbindung entspricht der Funktionalität des B-NT2-Blocks in Abb. 18. Da dieser Block in der Verantwortung des Kunden liegt, spricht man auch von einem CPN (Customers Premises Network) oder SPN (Subscribers Premises Network). Das GFC-Protokoll (Generic Flow Control) soll nun den Zugriffsmechanis-

[14] Subscriber network interface.

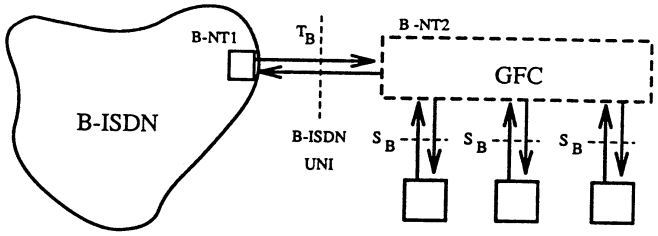

Abb. 18. Anbindung an B-ISDN über GFC

mus implementieren, der innerhalb dieses Blocks eine faire Aufteilung der Bandbreite unter den angeschlossenen Endgeräten sicherstellt.

Ausgehend von dem Fairneßkriterium, daß während einer Überlastphase alle Stationen die gleiche prozentuale Verringerung ihrer vereinbarten Spitzenbandbreite erfahren sollen, wurden verschiedenste Vorschläge eingereicht, die auf den gängigen MAN-Protokollen aufbauen und diverse Topologien unterstützen (z. B. Dual Bus, Stern, Ring, konzentriertes System, . . .).

Es sei angemerkt, daß einer der beiden derzeit noch diskutierten Entwürfe Konzepte von DQDB enthält (Schweizer PTT und Australian Telecom), während der andere, ein gemeinsamer Vorschlag von NTT und British Telecom, unter die Epigonen von Orwell einzuordnen ist.

Die Autoren danken Frau I. Fromm, Siemens AG München, und Herrn W. Schödl, IND, Universität Stuttgart, sowie den Gutachtern für die überaus konstruktive Kritik.

Literatur

1. Banerjee, S., Mukherjee, B.: Internal report, University of California, Davis 1990
2. Budrikis, Z.L., Hullett, J.L., Newman, R.M., Economou, D., Fozdar, F.M., Jeffery, R.D.: QPSX: A Queued Packet and Synchronous Circuit Exchange. Proc. ICCC '86, Munich 1986, pp. 288–293
3. Clapp, G.H.: LAN Interconnection Across SMDS. IEEE Network Magazine, September 1991, pp. 25–32
4. Conti, M., Giordano, S., Gregori, E., Lenzini, L.: DQDB Modeling: Problem complexity reduction and solution via Markov Chain. Int. Conf. on the Performance of Distributed Systems and Integrated Communication Networks, Kyoto 1991, pp. 47–62
5. Cox, T.A., Piscitello, D.M., Tesink, K.: SNMP Agent Support for SMDS. IEEE Network Magazine, September 1991
6. Decina, M.: Open Issues regarding the universal application of ATM for multiplexing and switching in the B-ISDN. Proc. ICC'91, Denver 1991
7. Dittmann, R.: Leistungsbewertung von DQDB Zugriffsprotokollen in Kommunikationssystemen. Diplomarbeit Universität Würzburg 1990
8. Falconer, R.M., Adams, J.L.: Orwell: A Protocol for an Integrated Service Local Network, in: Advances in Local Area Networks, pp. 465–479 (K. Kümmerle et al., eds.). IEEE Press 1987
9. Filipiak, J.: Access Protection for Fairness in a Distributed Queue Dual Bus Metropolitan Area Network. Proc. ICC'89, Boston 1989, pp. 635–639
10. Garett, M.W., Li, S.Q.: A study of slot reuse in dual bus multiple access networks. Proc. IEEE INFOCOM '90, San Francisco 1990, pp. 617–629
11. Hahne, E.L., Choudhury, A.K., Maxemchuk, N.F.: Improving DQDB Fairness. Contribution No. 802.6–89/52 to the IEEE 802.6 Working Group, 1989
12. Hahne, E.L., Choudhury, A.K., Maxemchuk, N.F.: Improving the Fairness of Distributed-Queue-Dual-Bus Networks. Proc. IEEE INFOCOM '90, San Francisco 1990, pp. 175–184
13. Hahne, E.L., Maxemchuk, N.F.: Bandwidth Balancing with Local Priority Information. Contribution No. 802.6–90/06 to the IEEE 802.6 Working Group, 1990
14. Hahne, E.L., Maxemchuk, N.F.: Bandwidth Balancing with Global Priority Information. Contribution No. 802.6–90/07 to the IEEE 802.6 Working Group, 1990
15. Hahne, E.L., Maxemchuk, N.F.: Fair Access of Multi-Priority Traffic to Distributed-Queue Dual-Bus Networks. Proc. INFOCOM '91, Miami 1991, pp. 889–900
16. Huber, M.N., Sauer, K., Schödl, W.: QPSX and FDDI-II Performance Study of High Speed LAN's. Proc. EFOC/LAN 88, Amsterdam 1988
17. IEEE: IEEE Standard: Distributed Queue Dual Bus (DQDB) Metropolitan Area Network (MAN), P802.6, 1990 [also IEEE Working Group, Proposed IEEE Standard 802.6 – Distributed Queue Dual Bus (DQDB) – Metropolitan Area Network (MAN), Draft versions: June 1988 – December 1990]
18. IEEE 802.6 Working Group: DQDB Metropolitan Area Network, Draft D0 of Proposed Standard, 1988
19. dito, Draft D6, 1988
20. IEEE 802.6 Working Group: DQDB Subnetwork of a MAN, Draft D7 of Proposed Standard, 1989
21. dito, Draft D12, 1990
22. dito, Draft D15, 1990
23. dito, Final Draft, 1990
24. IEEE: IEEE Standard 802.6-1990 – Distributed Queue Dual Bus (DQDB) Subnetwork of a Metropolitan Area Network (MAN), 1990
25. IEEE: Special Issue on Broadband Networks, 1989
26. Kamal, A.E.: Efficient Multi-Segment Message Transmission with Slot Reuse on DQDB. Proc. INFOCOM '91, Miami 1991, pp. 869–878
27. Klein, M.J.: Synchrone Digitale Hierarchie – Prinzipien und Anwendungen. Philips Innovation – Technische Mitteilungen Telekommunikation *2*, 4–10 (1991)
28. Le, M., Neitzel, L.: FDDI-II Working Paper, February 1987
29. de Moraes, L.F.M.: Simple Approximation for Frame Delays in DQDB Networks and Comparison with a Slotted Bus without Reservations. Proc. 8th EFOC/LAN '90, Munich 1990
30. de Moraes, L.F.M.: Frame Delay Analysis of the DQDB Protocol, in: High-Capacity Local and Metropolitan Area Networks – Architecture and Performance Issues pp. 299–310 (G. Pujolle, ed.). NATO ASI Series F, Vol. 72. Berlin: Springer 1991
31. Müller, H.R., Nassehi, M.M., Wong, J.W., Zurfluh, E., Bux, W., Zafiropulo, P.: DQMA and CRMA: New Access Schemes for Gbit/s LANs and MANs. Proc. IEEE INFOCOM '90, San Francisco 1990
32. Mukherjee, B., Banerjee, S.: Alternative Strategies for Improving the Fairness in and an Analytical Model of DQDB Networks. Research Report, Division of Computer Science, Department of Electrical Engineering and Computer Science, University of California, Davis, CSE-90-32, 1990
33. Newman, R.M., Hullett, J.L.: Distributed Queueing: A Fast and Efficient Packet Access Protocol for QPSX. Proc. ICCC '86, Munich 1986, pp. 294–299
34. Newman, R.M., Budrikis, Z.L., Hullett, J.L.: The QPSX MAN. IEEE Comm. Mag. *26*, 20–28, April 1988
35. Phung, V., Breault, R.: Enhancement to the Bandwidth Balancing Mechanism (version 2). Contribution No. 802.6–90/25 to the IEEE 802.6 Working Group, 1990
36. Potter, P., Zukerman, M.: Cyclic Request Control for Provision of Guaranteed Bandwith within the DQDB Framework. Proc. ISS '90, Stockholm 1990, Paper No. A4.1
37. Potter, P., Zukerman, M.: Analysis of a DQDB Subnetwork with Eraser Nodes. Proc. 13th ITC, Copenhagen 1991, pp. 941–946
38. Rodrigues, M.A.: Erasure node: performance improvements for the IEEE 802.6 MAN. Proc. IEEE INFOCOM '90, San Francisco 1990, pp. 636–643

39. Ross, F.E.: An Overview of FDDI: The Fiber Distributed Data Interface. IEEE J. Select. Areas Comm. *7*, 1043–1051 (1989)
40. Sauer, K., Schödl, W.: Performance Aspects of the DQDB Protocol. Proc. ITC Specialist Seminar, Adelaide 1989; Comput. Networks ISDN Syst. *20*, 253–260 (1990)
41. Schödl, W., Tangemann, M.: Performance Comparison of HSLANs for Correlated and Uncorrelated Source Models. Proc. 13th ITC, Copenhagen 1991, pp. 953–958
42. Spratt, M., Gifford, S.: A Problem in the Multi-Priority Implementation of the Bandwidth Balancing Mechanism. Contribution No. 802.6–89/06 to the IEEE 802.6 Working Group, 1989
43. Spratt, M.: Implementing the Priorities in the Bandwidth Balancing Mechanism. Contribution No. 802.6–90/05 to the IEEE 802.6 Working Group, 1990
44. Spratt, M.: Bandwidth Allocation in IEEE 802.6 Using non Unity Ratio Bandwidth Balancing. Proc. ICC '91, Denver 1991, pp. 729–735
45. Stock, T., Tran-Gia, P.: A Discrete-Time Analysis of the DQDB Access Protocol. Proc. ITS '90, Rio de Janeiro 1990, pp. 121–126
46. Stock, T.: Influences of Multiple Priorities on DQDB Protocol Performance. Proc. 13th ITC, Copenhagen 1991, pp. 947–952
47. Takagi, H.: Queueing Analysis, Vol. 1. Amsterdam: North-Holland 1991
48. Tanenbaum, A.S.: Computer-Netzwerke. Attenkirchen: Wolfram 1990
49. Tran-Gia, P., Stock, T.: Approximate Performance Analysis of the DQDB Access Protocol. Proc. ITC Specialist Seminar, Adelaide 1989; Comput. Networks ISDN Syst. *20*, 231–240 (1990)
50. Tran-Gia, P., Stock, T.: Modelling of the DQDB Access Protocol and Closed-Form Approximation, in: High-Capacity Local and Metropolitan Area Networks – Architecture and Performance Issues, pp. 299–310 (G. Pujolle, ed.). NATO ASI Series F, Vol. 72. Berlin: Springer 1991
51. van As, H.R.: Performance Evaluation of Bandwidth Balancing in the DQDB MAC Protocol. Proc. 8th EFOC/LAN '90, Munich 1990, pp. 31–39
52. van As, H.R.: Major Performance Characteristics of the DQDB MAC Protocol. Proc. ITS '90, Rio de Janeiro 1990, pp. 113–120
53. Wong, J.W.: Throughput of DQDB Networks under Heavy Load. Proc. EFOC/LAN 89, Amsterdam 1989, pp. 146–151
54. Zukerman, M.: Queueing Performance of QPSX. Proc. 12th Int. Teletraffic Congress, Torino 1988, paper 2.2B.6
55. Zukerman, M.: Approximations for Performance Evaluation of the Packet Access Queue in QPSX: the IEEE 802.6 Evolving MAN Standard. Aust. Telecomm. Res. *22* (2), 53–62 (1988)
56. Zukerman, M., Potter, P.: The DQDB Protocol and its Performance under Overload Traffic Conditions. Proc. ITC Specialist Seminar, Adelaide 1989; Comput. Networks ISDN Syst. *20*, 261–270 (1990)
57. Zukerman, M., Potter, P.: The Effect of Eliminating the Standby State on DQDB Performance under Overload. Int. J. Digit. Analog Cabled Syst. *2*, 179–186 (1989)
58. Zukerman, M., Potter, P.: A Protocol for Eraser Node Implementation within the DQDB Framework. Proc. GLOBECOM '90

Eingegangen 27.3.1992; in überarbeiteter Form 25.1.1993

Dipl.-Math. Rainer Dittmann
Dr. rer. nat. Thomas Stock
Prof. Dr.-Ing. Phuoc Tran-Gia
Lehrstuhl für verteilte Systeme (Informatik III)
Universität Würzburg
Am Hubland, D-97074 Würzburg
E-mail:
{dittmann,stock,trangia}@informatik.uni-wuerzburg.dbp.de.