Entwurf eines parallelen Transportprotokolls

Torsten Braun
Institut für Telematik
Universität Karlsruhe
Postfach 6980, Zirkel 2
W-7500 Karlsruhe
Telefon: 0721 608-3982
Fax: 0721 606097
E-mail: braun@telematik.informatik.uni-karlsruhe.de

Martina Zitterbart*
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598, U.S.A.
Telefon: 001 914 784-6952
Fax: 001 914 784-6219
E-mail: zit@watson.ibm.com

Kurzfassung

Heutige Kommunikationssubsysteme können die Leistungsfähigkeit, wie sie von modernen Anwendungen in Gigabit-Netzwerk-Umgebungen gefordert werden, kaum erbringen. Ein vielversprechender Ansatz, Kommunikationssubsysteme hoher Leistungsfähigkeit zu realisieren, besteht in der Verwendung von Multiprozessorarchitekturen für die Protokollverarbeitung [1, 2]. Basierend auf den Erfahrungen aus der Analyse des Hochgeschwindigkeitsprotokolls Xpress Transfer Protocol (XTP) und den Ergebnissen detaillierter Leistungsuntersuchungen an einer Implementierung auf einer Multiprozessorarchitektur werden Vorschläge hinsichtlich des Entwurfs von parallelen Transportprotokollen entwickelt. Diese Vorschläge führen zum Entwurf eines neuen parallelen Transportprotokolls, welches im wesentlichen durch die Dekomposition in einzelne modulare Protokollautomaten mit einem Minimum an notwendigen Interaktionen untereinander gekennzeichnet ist, um die Parallelverarbeitung effizienter zu unterstützen.

1. Einführung

Hochleistungs-Kommunikationssubsysteme werden einerseits wegen den sich neu entwickelnden Anwendungen, welche hohe Leistungen in Form von Durchsatz und Verzögerungen usw. erfordern, und andererseits durch die Entwicklung von Hochgeschwindigkeitsnetzen in den Gigabit/s-Bereich benötigt. Heutige Kommunikationssubsysteme sind kaum in der Lage, die neuen an sie gestellten Anforderungen zu erfüllen.
Die Gründe für die Leistungsprobleme sind mannigfaltig, sie lassen sich nicht auf ein einzelnes Problem in Kommunikationssubsystemen reduzieren. Viele Faktoren wie Protokollentwurf, Implementierungsarchitektur, Architektur des Arbeitsplatzrechners, Betriebssystem usw. beeinflussen die Leistungsfähigkeit. Lösungen müssen verschiedene Faktoren integrierend berücksichtigen. In der Vergangenheit wurden jedoch verschiedene Forschungsbereiche wie Protokollentwurf und Implementierungsarchitekturen meist isoliert voneinander betrachtet und ohne gegenseitigen Einfluß verfolgt.
Eine der wenigen Ausnahmen ist das XTP-Projekt [3] (XTP: Xpress Transfer Protocol), wo Aspekte des Protokollentwurfs zusammen mit dem Entwurf der Implementierungsplattform behandelt werden. Eine wesentliche Eigenschaft von XTP ist dessen Spezifikation als eine Menge erweiterter endlicher Automaten (extended finite state machines, FSMs) [4]. Dieses Konzept macht XTP besonders attraktiv für parallele Implementierungen sowohl in Software als auch in Hardware (VLSI), wie es ursprünglich im XTP-Projekt geplant war.

---

*1992 von der Universität Karlsruhe (Institut für Telematik) beurlaubt

Basierend auf den Erfahrungen mit diversen Protokollimplementierungen (OSI-Protokolle und XTP) [5] auf Transputernetzen [6] wird ein neuartiger Entwurf eines parallelen Protokolls entwickelt. Das wesentliche Kennzeichen besteht in der Spezifikation modularer Protokollautomaten, um einen hohen Parallelitätsgrad zu ermöglichen. Die Synchronisation zwischen Automaten einer Instanz ist auf ein Minimum reduziert, zur Kommunikation zwischen Automaten in unterschiedlichen Instanzen existieren für jeden Automatentyp spezielle Protokolle. Die Protokollarchitektur eignet sich zur direkten Abbildung auf eine Multiprozessorarchitektur. Eine speziell auf die Anforderungen der Protokollarchitektur zugeschnittene hybride Multiprozessorarchitektur wurde bereits entworfen.
Dieser Beitrag diskutiert in Kapitel 2 nach einem kurzen allgemeinen Abschnitt über XTP dessen Eignung für eine parallele Implementierung. Kapitel 3 beschreibt die mit einer parallelen XTP-Implementierung gesammelten Erfahrungen sowie einige Verbesserungsvorschläge hinsichtlich einer effizienter implementierbaren parallelen Protokollarchitektur. In Kapitel 4 werden die Eigenschaften und Konzepte des neu entworfenen Transportprotokolls erläutert, Kapitel 5 beendet den Beitrag mit einer Zusammenfassung.

2. XTP: Ein Protokoll mit paralleler Struktur

XTP [7] ist ein Hochgeschwindigkeitsprotokoll mit Funktionalitäten der Schichten 3 und 4 gemäß dem OSI-Basisreferenzmodell. Diese Integration der Transport- und Vermittlungsschicht wird auch oft als *transfer layer* bezeichnet. XTP wurde entworfen, um den Anforderungen von zukünftigen Realzeit-, Multi-Media- und verteilten Anwendungen in heterogenen Hochgeschwindigkeitsnetz-Umgebungen zu genügen. XTP unterstützt verschiedene Adressierungsmechanismen wie IP- und OSI-Adressen. Mehrere XTP-Implementierungen auf verschiedenen Implementierungsplattformen wurden bereits realisiert oder entworfen [8, 9, 10, 11, 12, 13]. Die Entwicklung von XTP ist ein Teil des Protocol Engine-Projekts, in dem VLSI-Bausteine zur Leistungssteigerung der Protokollverarbeitung und zu effizienteren Netzwerkankopplungen entwickelt werden [14].
XTP bietet Multicast und unterstützt verbindungslose sowie verbindungsorientierte Dienste. Diverse Protokollmechanismen, die von anderen Hochgeschwindigkeitsprotokollen wie Delta-t [15], NETBLT [16] und VMTP [17] eingeführt wurden, sind in XTP integriert, um effizient in Netzen mit hohem Bandbreiten-Verzögerungs-Produkt arbeiten zu können. Flußkontrolle wird durch Übertragungsratenkontrolle unterstützt, Verbindungen können implizit aufgebaut werden, Fehlerbehandlung und Flußkontrolle sind auswählbar, und die Übertragungswiederholung kann selektiv erfolgen.
Die parallele Protokollverarbeitung wird in XTP durch die Aufteilung in mehrere Automaten unterstützt. Am Institut für Telematik der Universität Karlsruhe wurde XTP auf der Basis der Spezifikation der Version 3.4 auf Transputernetzen implementiert. Die Version 3.4 stellt die aktuellste Version dar, die eine vollständige Spezifikation dieser Automaten beinhaltet. In den folgenden Abschnitten wird die Eignung der Aufteilung der Protokollautomaten im Hinblick auf eine parallele Implementierung diskutiert.

2.1. Das Konzept der parallelen XTP-Automaten

Die XTP Spezifikation Version 3.4 [4] beschreibt elf Automaten mit jeweils unterschiedlichen Aufgaben. Die meisten Automaten werden für jeden Kontext erzeugt, wobei ein Kontext den Zustand einer unidirektionalen Ende-zu-Ende-Verbindung beschreibt. Im wesentlichen sind dies Automaten, welche in Endsystemen angesiedelte Funktionen durchführen. Automaten für Router-Funktionen werden für jede Route erzeugt. Die Automaten kommunizieren über Er-

eignisse und kooperieren über gemeinsamen Daten, z.B. Kontextinformation. Zustandsübergänge werden durch signalisierte Ereignisse (Netzwerk-, Zeitgeber-, interne und externe Ereignisse) ausgelöst, sind aber auch vom Inhalt der aktuellen Kontextinformation abhängig. Die elf Automaten können folgendermaßen klassifiziert werden:

- Verbindungsverwaltung
  Der Kontext-Manager (*context manager state machine*, CMSM) wird für eine neue Verbindung erzeugt und aktiviert die anderen Automaten des generierten Kontexts. Der sichere Verbindungsabbau erfolgt in Zusammenarbeit mit den Automaten WCSM (*write close state machine*) und RCSM (*read close state machine*).
- Austausch von Benutzerdaten
  OSM (*output state machine*) sendet Informationspakete, d.h. Benutzerdaten, an das Netz. Außerdem führt der Automat die Flußkontrolle auf Senderseite durch und entscheidet, ob vom Empfänger Kontrollpakete angefordert werden müssen. RACSM (*rate control state machine*) unterstützt OSM bei der Übertragungsratenkontrolle. Die Informations- und Kontrollpakete auf der Empfängerseite werden von ISM (*input state machine*) überprüft und die Benutzerdaten werden über RSM (*reader state machine*) an den XTP-Benutzer übergeben.
- Austausch von Kontrollinformationen
  CSSM (*control send state machine*) erzeugt und formatiert Kontrollpakete und konkurriert mit OSM um den Zugriff auf das Netzwerk. Die Aktivitäten von CSSM werden durch andere Protokollautomaten ausgelöst, wenn diese das Senden von Kontrollpaketen erfordern. SSM (*sync state machine*) führt den synchronisierenden Handshake durch, der Kontrollaktivitäten zwischen Endsystemen synchronisiert und die Paketumlaufzeit schätzt.
- Vermittlung und Wegewahl
  Der Host-Routen-Manager (*host route manager*, *HRM*) überwacht die aktiven Routen eines Endsystems. ROSM (*router state machine*) führt Wegewahl- und Vermittlungsfunktionen durch und tauscht Routenbezeichner aus.

2.2. Unterstützung paralleler Implementierungen

XTP stellt neben SNR [18] eines der wenigen Protokolle dar, das Parallelität bereits in seiner Spezifikation unterstützt. Standardprotokolle unterstützen Parallelität in der Regel nicht und können deshalb nur mit erheblichem Aufwand auf parallelen Architekturen implementiert werden. Meistens ist eine detaillierte Protokollanalyse erforderlich, um die innewohnende Parallelität zu extrahieren. Im Hinblick auf parallele Implementierungen besitzt XTP folgende ***Vorteile***:

- Parallele Protokollautomaten
  XTP unterstützt eine parallele Implementierung durch die Spezifikation mehrerer parallel arbeitender Protokollautomaten, welche nahezu direkt auf eine Multiprozessorarchitektur abgebildet werden können.
- Integration von Transport- und Vermittlungsschicht
  Diese Kombination reduziert zum einen den durch geschichtete Kommunikationsarchitekturen verursachten Overhead, zum anderen wird Parallelverarbeitung unterstützt, weil damit die fließbandartige Struktur geschichteter Architekturen vermieden werden kann und ein höherer Parallelitätsgrad bei der Bearbeitung einzelner Pakete ermöglicht wird. Außerdem können redundante, d.h. in verschiedenen Schichten mehrfach vorkommende, Protokollfunktionen, reduziert werden.

- Trennung von Sender und Empfänger
  XTP trennt deutlich zwischen Funktionen zum Senden und Empfangen von Paketen. Beide Protokollteile können nahezu unabhängig voneinander arbeiten, was insbesondere bei der Behandlung von Duplex-Verkehr wichtig ist. Der Automat OSM ist für die meisten Funktionen beim Senden von Paketen verantwortlich, die Empfänger-Pipeline wird durch RSM und ISM gebildet. Die Behandlung von Duplex-Verbindungen wird weiterhin durch das Aufspalten einer Verbindung in zwei Simplex-Verbindungen vereinfacht. Beide Endsysteme benutzen unterschiedliche Kontexte für jede Richtung, was eine voneinander unabhängige und parallele Verarbeitung der beiden Datenflußrichtungen erleichtert.
- Unterschiedliche Pakete für Benutzerdaten und Kontrollinformation
  Die Unterscheidung von Protokolldateneinheiten in Informationspakete, die hauptsächlich aus Benutzerdaten bestehen, und in Kontrollpakete, welche Kontrollinformationen, z.B. für Flußkontrolle oder Quittierungszwecke, enthalten, erleichtert die parallele Bearbeitung von Benutzerdaten und Kontrollinformation dadurch, daß Kontrollfunktionen und Funktionen zur Generierung von Benutzerdateneinheiten nicht auf dieselben Datenstrukturen zugreifen müssen. Mit CSSM ist in XTP ein eigener Protokollautomat für die Erzeugung von Kontrollpaketen vorgesehen, der parallel zu den für die Verarbeitung von Benutzerdaten verantwortlichen Automaten arbeiten kann.
- Feste Paketformate
  Feste Paketformate vereinfachen die Parallelverarbeitung von einzelnen Paketfeldern, weil deren Positionen unabhängig voneinander sind und daher verschiedene Funktionen auf relevante Felder ohne aufwendige Analyse der Datenstrukturen zugreifen können. In früheren Versionen von XTP bestanden Kontroll- und Informationspakete aus einem Kopf und einem längeren Anhang, wodurch jedoch die Position des Anhangs in einem Informationspaket von der Länge der Benutzerdaten abhing. In der aktuellen XTP-Version 3.6 [19] besteht der Anhang nur noch aus einer Prüfsumme, so daß der Zugriff auf die anderen Paketfelder erleichtert wird.

Die oben erläuterten Eigenschaften von XTP sind besonders hilfreich für die leistungsfähige parallele Implementierung von XTP auf Multiprozessorarchitekturen, wie z.B. Transputernetzen. Die gesammelten Erfahrungen mit mehreren parallelen Implementierungen, auch von OSI-Protokollen, bestätigen dies ebenso [5]. Sie zeigen jedoch auch, daß die Spezifikation von XTP einige ***Nachteile*** enthält, die effizientere parallele Software-Implementierungen verhindern. In diesem Zusammenhang sollte aber auch erwähnt werden, daß der Entwurf von XTP wesentlich von der Absicht einer späteren Implementierung in VLSI beeinflußt wurde. Im folgenden sind einige wesentliche Punkte aufgeführt, die zum Erzielen effizienter paralleler Implementierungen verbessert werden müssen. Kapitel 3 diskutiert einen Entwurf, welcher Verbesserungen für diese Punkte beinhaltet.

- Phasenabhängige Aktivitäten
  Die meisten Protokollautomaten sind nur während bestimmter Phasen einer Verbindung aktiv, d.h. Verbindungsaufbau, -abbau oder Datentransfer, was zu keiner Verbesserung während einer einzelnen Phase führt. Beispielsweise ist CMSM im wesentlichen nur während des Auf- und Abbaus einer Verbindung aktiv. Nur sechs der elf Automaten (CSSM, SSM, RACSM, OSM, ISM und RSM) sind an der Protokollverarbeitung während der Datentransferphase beteiligt. RACSM ist nur einer kleiner Automat und sollte eher in OSM integriert werden. RSM ist im wesentlichen nur für die Kooperation der Empfangsseite mit dem XTP-Benutzer zuständig.
- Lastausgleich zwischen Protokollautomaten
  Während der Datentransferphase sind hauptsächlich zwei Protokollautomaten (OSM auf der Sendeseite und ISM auf der Empfangsseite) an der Protokollverarbeitung beteiligt,

während der Rest der Automaten nur sehr gering belastet ist. Im wesentlichen wird ein Leistungsgewinn durch die nebenläufige Verarbeitung der Sender- und Empfängeraktivitäten ermöglicht. OSM formatiert zu sendende Informationspakete, führt die Flußkontrolle des Senders durch, ist an der Übertragungsratenkontrolle beteiligt und fordert Kontrollpakete vom Empfänger der Partnerinstanz an. ISM bearbeitet sowohl Kontroll- als auch Informationspakete und löst abhängig von den Ergebnissen der Überprüfung der empfangenen Pakete Aktionen auch in anderen Automaten aus.

- Eng gekoppelte Verarbeitung von Kontroll- und Benutzerdatenfunktionen
  Die Benutzung unterschiedlicher Dateneinheiten für Benutzerdaten und Kontrollinformation spiegelt sich nicht in konsequent getrennten Protokollautomaten für die Bearbeitung der beiden Pakettypen wider. Beispielsweise muß ISM sämtliche Pakettypen bearbeiten, ist aber gleichzeitig an Kontrollaufgaben wie Quittierung, Fluß- und Übertragungsratenkontrolle beteiligt. ISM veranlaßt abhängig von den Ergebnissen der Überprüfung der angekommenen Pakete, daß CSSM entsprechende Kontrollpakete generiert und formatiert.
- Abhängigkeiten zwischen Protokollautomaten
  Der bidirektionale Nachrichtenfluß zwischen einigen Protokollautomaten beeinträchtigt die Parallelität wesentlich, wenn nach dem Versenden von Ereignissen an andere Automaten auf deren Antworten oder Ergebnisse gewartet werden muß. Dadurch wird ein paralleles Arbeiten der Automaten verhindert. Ein wichtiges Beispiel für gemeinsam benutzte Datenbasen sind die Kontextinformationen, welche den Zustand einer Verbindung beschreiben. Zumindest auf Teile dieser zentralen Datenbasis müssen einige Automaten zugreifen können, so daß der Zugriff darauf durch mehrere Automaten seriell erfolgen muß, was schließlich eine Sequenzialisierung der Protokollverarbeitung zur Folge hat.

3. Kriterien zum parallelen Protokollentwurf

3.1. Erfahrungen und Analyse einer parallelen XTP-Implementierung

In [8] wurde eine parallele XTP-Implementierung auf Transputernetzen (s. Abb. 1) vorgestellt, die auf der Spezifikation der parallelen Protokollautomaten der XTP Version 3.4 beruht. Die Struktur der XTP-Automaten wurde dazu auf Transputerprozesse abgebildet [20], d.h. die Protokollautomaten wurden als Prozesse implementiert. Aus Effizienzgründen wurden zum Teil vor der Abbildung auf Prozesse mehrere XTP-Automaten zu einem Produktautomaten zusammengefaßt. Jeder resultierende Prozeß ist in der Lage, auf einem beliebigen Prozessor abzulaufen.
Die Signalisierung der Ereignisse erfolgt durch asynchronen Nachrichtenaustausch zwischen Prozessen. Transputerprozesse kommunizieren jedoch über synchrone Kanäle nach dem Rendezvouskonzept. Asynchrone Kommunikation wird durch sogenannte Warteschlangenprozesse ermöglicht, welche die direkte, synchrone Kommunikation zwischen den beiden beteiligten Prozessen entkoppeln. Die Kommunikationsschnittstelle der Prozesse wurde derart entworfen, daß beliebige Transputernetzkonfigurationen möglich sind, wobei insbesondere die Tatsache, daß Transputer nur vier Kommunikations-Links besitzen, den Entwurf beeinflußte.
Weil das Konzept der Transputer keine gemeinsamen Speicherbereiche vorsieht, verwalten spezielle Speicherverwalterprozesse den Zugriff auf einen logisch globalen Speicher für Kontroll- und Informationspakete. Der globale Speicher wird dabei durch den lokalen Speicher eines ausgezeichneten Transputers realisiert. Aufwendige Datenkopiervorgänge werden damit vermieden. Kontextdaten sind soweit möglich auf die einzelnen Prozesse verteilt und werden, wenn erforderlich, aktualisiert.

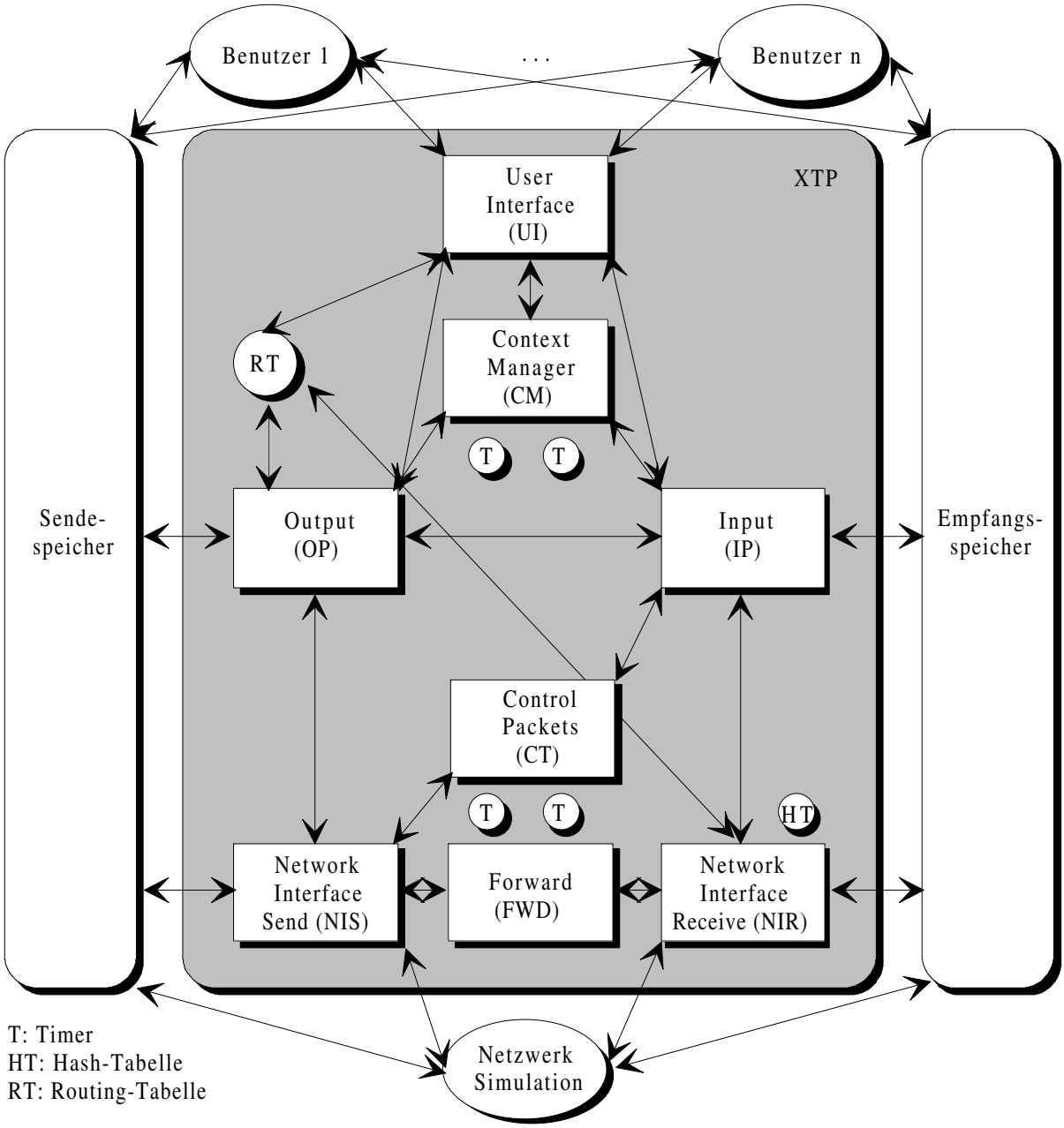

Abb. 1: Parallele XTP-Implementierung auf Transputern

Der Prozeß CM bearbeitet den Automaten CMSM, CT ist für den Austausch von Kontrollpaketen zuständig, FWD führt Wegewahl und Vermittlungsfunktionen aus und UI ist an der Terminierung von Verbindungen beteiligt und übernimmt die Kooperation mit dem XTP-Benutzer. Der Austausch von Benutzerdaten wird durch die Sender-Pipeline bestehend aus OP und NIS bzw. durch die Empfänger-Pipeline mit NIR und IP bearbeitet.
Die parallele Implementierung erreicht auf Transputern des Typs T414 eine Leistung von über 1000 Paketen/s in beiden Übertragungsrichtungen bei Duplex-Verkehr. Mit Code-Optimierungen, effizienterem Compiler und leistungsfähigeren Multiprozessorarchitekturen sollen zukünftig noch höhere Leistungen erzielt werden. Ausführliche Leistungsuntersuchungen [21] an der Implementierung führen zu interessanten Resultaten und Rückschlüssen hinsichtlich paralleler Protokollverarbeitung:

- Leistungsgewinn durch parallele Verarbeitung des Senders und des Empfängers
  Für Duplex-Verkehr wurde durch die parallele Verarbeitung von Funktionen des Senders und des Empfängers ein hoher Leistungsgewinn im Vergleich zur Implementierung auf

einem Prozessor festgestellt. Sender- und Empfängerfunktionen beeinflussen sich gegenseitig kaum.

- Höherer Leistungsgewinn auf Empfängerseite
  Der Leistungsgewinn auf Empfängerseite ist bei der parallelen Implementierung im Vergleich zu einer Konfiguration, bei der alle XTP-Prozesse auf genau einem Prozessor laufen, deutlich höher als der entsprechende Leistungsgewinn auf Senderseite. Dies liegt hauptsächlich daran, daß die Prozesse auf Empfängerseite meist unidirektional kommunizieren, d.h. es entstehen keine Wartezeiten auf eventuelle Rückmeldungen.
- Teilweise parallele Verarbeitung von Kontroll- und Benutzerdaten
  Ein weiteres wichtiges Ergebnis besteht darin, daß die Verarbeitung von Kontrollpaketen - speziell das Generieren und Formatieren von Kontrollpaketen (vgl. Kap. 2.2) - teilweise parallel zur Verarbeitung empfangener oder zu sendender Informationspakete durchgeführt werden kann. Der zusätzliche Aufwand zum Formatieren von Kontrollpaketen kann durch Parallelität um ca. 50 % reduziert werden. Kontrollpakete werden zum Teil auch vom Prozeß NIS bearbeitet, der ebenfalls an der Bearbeitung von Informationspaketen beteiligt ist.
- Kommunikationsoverhead durch Nachrichtenaustausch
  Ein wesentlicher Overhead in dieser parallelen Implementierung wird durch den zusätzlichen Kommunikationsaufwand verursacht. Bis zu 25 % der Prozeßlaufzeiten werden für die Kommunikation aufgewendet. Die Kommunikation ist notwendig, um die einzelnen Prozesse miteinander zu synchronisieren, Parameter und Ergebnisse auszutauschen und Kontextinformation zu aktualisieren.
  Eine Analyse der Kontextinformation zeigt, daß auf die meisten Daten der Kontextinformation nur von einem oder wenigen Prozessen zugegriffen wird, so daß die Anzahl der zu aktualisierenden Daten gering bleiben kann. Trotzdem wird durch die verbleibenden notwendigen Kontextaktualisierungen und die damit verbundene Kommunikation ein nicht zu vernachlässigender Overhead erzeugt.

3.2. Entwurfskonzepte für parallele Protokolle

Aus den Ergebnissen der Analyse der XTP-Protokollautomaten und der Leistungsuntersuchung der entsprechenden Multiprozessor-Implementierung werden die folgenden Verbesserungsvorschläge bzw. Konzepte für den Entwurf eines parallelen Transportprotokolls entwikkelt:

- Auslagern sämtlicher Kontrollfunktionen von der Bearbeitung von Benutzerdaten
  Ein vielversprechender Ansatz zur Erhöhung der Parallelität besteht in der vollständig parallelen Verarbeitung von Kontroll- und Informationspaketen und damit in der Entkopplung von Kontrollfunktionen und Benutzerdaten bezogenen Funktionen. Die Verarbeitung von Benutzerdaten ist oft der kritische Pfad in der Protokollverarbeitung, weshalb versucht werden sollte, so viele Kontrollfunktionen wie möglich von der Behandlung der Benutzerdaten abzuspalten.
  Dies kann durch völlig voneinander unabhängige Protokollautomaten für jeden Pakettyp erfolgen. Auch Funktionen wie Quittierung und Übertragungswiederholung sollten von der regulären Bearbeitung von Benutzerdaten in spezielle Automaten mit separaten Dateneinheiten ausgelagert werden. Um diese Auslagerung zu unterstützen, erscheint es nicht sinnvoll, Felder im Paketkopf von Benutzerdaten zur Auswahl und Steuerung von Kontrollfunktionen, wie z.B. Quittierungen, zu verwenden. Benutzerdatenpakete sollten deshalb nur absolut notwendige Felder zur Beschreibung der Benutzerdaten enthalten.

- Modularer Protokollentwurf
  Um zu vermeiden, daß die Bearbeitung der Kontrollfunktionen einen neuen Engpaß verursacht, können die Kontrollfunktionen zusätzlich unterteilt werden, um weitere Parallelverarbeitung zu ermöglichen. Gerade in Hochgeschwindigkeitsnetzen mit hohem Bandbreiten-Verzögerungs-Produkt gewinnen neuartige und meist aufwendigere Fluß- und Staukontrollverfahren und -strategien zunehmend an Bedeutung. Parallele Implementierbarkeit setzt einen orthogonalen Entwurf von Kontrollfunktionen mit geringen Interaktionen voraus, welcher durch eine deutliche Aufteilung des Protokolls in unterschiedliche Protokollfunktionen erfolgen kann. Beispielsweise kann die Übertragungsratenkontrolle unabhängig von der Quittierung durchgeführt werden.
  Die Dekompostion eines Protokolls in eindeutig voneinander abgegrenzte Einheiten erlaubt einen modularen Protokollentwurf. Mit der erzielten Modularität können einzelne Protokollfunktionen einfacher ausgewählt werden. Mechanismen, Strategien sowie Parameter der Protokollfunktionen sind unabhängig selektierbar. Insbesondere eine flexible und dynamische Konfiguration von Protokollen [22] wird hierdurch wesentlich vereinfacht.
- Benutzung lokaler Ressourcen
  Um einen hohen Parallelitätsgrad zu erreichen, sollten alle parallel ausgeführten Funktionen, wenn möglich, lokale Ressourcen anstatt globaler Ressourcen verwenden, z.B. verschiedene anstatt gemeinsame Kontrollpakete für die einzelnen Kontrollfunktionen. Damit werden Zugriffskonflikte zwischen parallelen Implementierungseinheiten reduziert.
- Reduzieren der Signalisierung von Ereignissen
  Gerade in Hochgeschwindigkeitsnetzen mit hohem Bandbreiten-Verzögerungs-Produkt ist eine global einheitliche Sicht verschiedener Endsysteme auf den Zustand der Verbindung kaum zu erreichen, auch wenn Kontrollinformationen für jedes gesendete oder empfangene Paket ausgetauscht werden. Als Alternative bietet sich der periodische Austausch von Kontrollinformation an. Als Konsequenz daraus erscheint in bestimmten Fällen auch die interne periodische Signalisierung zwischen Protokollautomaten einer Instanz sinnvoll, was schließlich eine Reduzierung des Kommunikationsaufwands zwischen parallelen Automaten erlaubt. Für den zusätzlichen Kommunikationsaufwand bei einer parallelen Protokollimplementierung bildet die Signalisierung von Ereignissen einen wesentlichen Faktor.
- Verteilung von Verbindungszustandsdaten
  Zusammen mit der Dekomposition eines Protokolls in parallel ausführbare Einheiten sollten auch Verbindungzustandsdatenbasen verteilt werden. Global gültige Zustandsdaten sind, soweit möglich, zu vermeiden, weil dadurch entweder die mögliche Parallelität wegen Zugriffskonflikten auf gemeinsame Speicherbereiche beeinträchtigt werden kann oder der Kommunikationsaufwand durch das Aktualisieren verteilt gehaltener Zustandsdaten stark anwachsen kann. Falls eine vollständige Verteilung der Zustandsdaten nicht vermeidbar ist, besteht außerdem die Möglichkeit, in einer Implementierung den Zustandsdatenaustausch zwischen Protokollautomaten mit dem Signalisieren von Ereignissen zu verbinden, sofern die Änderung der Zustandsdaten nur bei bestimmtem Ereignissen relevant ist.

4. Entwurf eines Protokolls für ein paralleles Transportsubsystems

Unter Berücksichtigung der diskutierten Erfahrungen und Ergebnisse wurde ein neues Transportprotokoll entworfen, das sich besonders für Implementierungen auf parallelen Architektu-

ren eignet. Das Protokoll hat nicht nur die Funktionalität konventioneller Transportprotokolle, vielmehr werden auch Funktionen der Sicherungs- und Vermittlungsschicht übernommen, d.h. das Protokoll bietet die Funktionalität eines Transportsubsystems zwischen Medienzugangs- und Transportdienstschnittstelle.
Eine wesentliche Eigenschaft des Protokollentwurfs ist die Aufteilung der Protokollfunktionen auf mehrere Protokollautomaten, wobei für jede Kontrollfunktion sowie für die Benutzerdaten bezogenen Funktionen ein solcher Automat zur Verfügung steht. Ein weiterer Entwurfsaspekt, der hier im folgenden aber nicht näher diskutiert wird, ist die Eignung des Transportsubsystems für zellenbasierte Netze wie DQDB oder ATM, d.h. Breitband-ISDN. Diese Thematik wird in [23] ausführlicher behandelt.

4.1. Aufteilung in modulare Protokollautomaten

Bei dem neuen Transportsubsystem sind für alle Kontrollfunktionen und Benutzerdaten verarbeitenden Funktionen separate Protokollautomaten vorgesehen. Automaten derselben Instanz synchronisieren sich und kooperieren mittels Nachrichtenaustausch. Die Protokollarchitektur besteht aus zwei Typen von Automaten:

- Automaten für Interaktionen mit Nachbarschichten (Schnittstellenautomaten)
- Automaten mit Partnerautomaten in der Partnerinstanz (Protokollautomaten)

4.1.1. Schnittstellenautomaten

Die Interaktionen mit dem Transportdienstbenutzer bzw. mit dem unterliegenden Netzwerk werden durch spezielle Automaten unterstützt. Diese Schnittstellenautomaten sind lediglich zur lokalen Kommunikation innerhalb eines Systems notwendig. Sie haben im Gegensatz zu den im folgenden Abschnitt beschriebenen Protokollautomaten keinen Automaten in der Partnerinstanz, mit dem sie kommunizieren.
Der Automat an der Transportdienstschnittstelle tauscht mit dem Transportdienstbenutzer Transportdienstdateneinheiten aus. Er löst auf vom Transportdienstbenutzer empfangenen Transportdienstdateneinheiten die in den entsprechenden Protokollautomaten erforderlichen Aktionen aus bzw. schickt auf Anforderung der Protokollautomaten Transportdienstdateneinheiten an den Transportdienstbenutzer. Außerdem ist dieser Automat an den Aktionen zu einer ordnungsgemäßen Terminierung einer Verbindung beteiligt.
An der Schnittstelle zwischen Transportsubsystem und unterliegendem Netzwerk kontrolliert ein weiterer Schnittstellenautomat das Senden von Protokolldateneinheiten der Protokollautomaten.

4.1.2. Protokollautomaten

Die Kommunikation zwischen verschiedenen Protokollinstanzen erfolgt durch die direkte Kommunikation zwischen einzelnen Protokollautomaten, d.h. es wird jeweils ein individuelles Protokoll zwischen Automaten in unterschiedlichen Instanzen abgearbeitet. In Abb. 2 sind einige der Protokollautomaten dargestellt. Jede Kommunikationsbeziehung besitzt ein eigenes Protokoll und spezielle Dateneinheiten. Das Multiplexen von Protokolldateneinheiten verschiedener Automaten wird damit vermieden, so daß Synchronisationspunkte zwischen Automaten derselben Instanz minimiert werden.
Die Kontrollfunktionen sind so entworfen, daß sie möglichst unabhängig von den Benutzerdaten bezogenen Funktionen arbeiten, indem der Austausch der Verbindungsdaten vollständig

vom Austausch der Benutzerdaten entkoppelt ist. Quittungen und der Austausch der Verbindungszustandsdaten zwischen Instanzen werden nicht durch gesendete oder empfangene Benutzerdatenpakete sondern periodisch oder durch explizite Anforderungen und Fehlersituationen veranlaßt. Die Protokolle der im folgenden beschriebenen Automaten besitzen, falls erforderlich, ihre eigenen Fehlerbehebungsmechanismen und Zeitgeber. Die meist sehr aufwendige Zeitgeber-Verwaltung kann so auf mehrere Protokollautomaten verteilt werden.

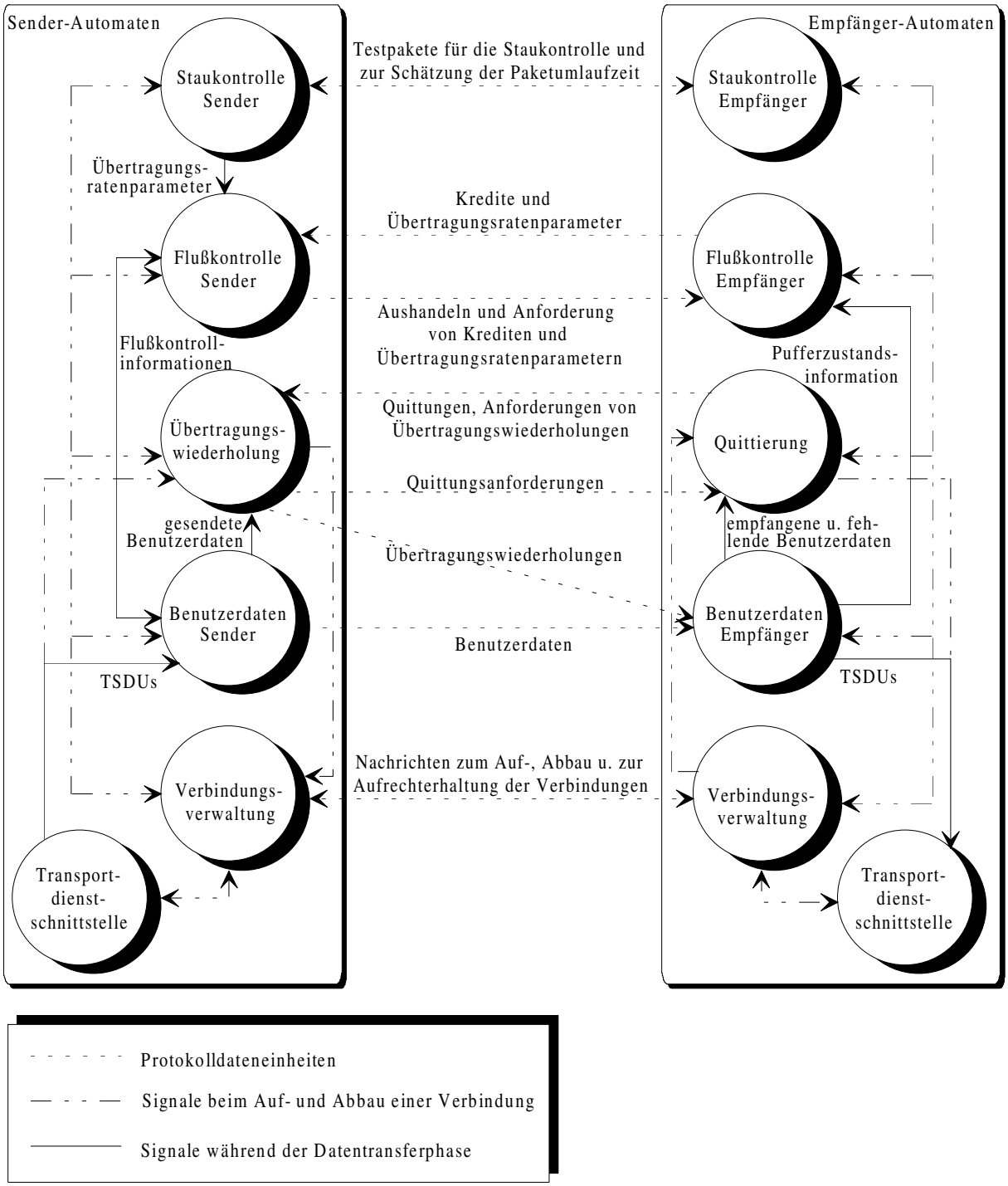

Abb. 2: Parallele Protokollarchitektur

- Verbindungsverwaltung
  Der Automat für die Verbindungsverwaltung verwaltet die gesamte Duplex-Verbindung. Er ist für den Aufbau, die Aufrechterhaltung und den ordnungsgemäßen Abbau der Ver-

bindung verantwortlich. Der Automat benutzt Handshake-basierte oder implizite Verbindungsaufbauverfahren entsprechend dem geforderten Dienst. Außerdem muß er die eindeutige Benutzung von Verbindungsidentifikatoren überwachen.

- Benutzerdatentransfer
  Die beiden Automaten zum Benutzerdatenaustausch senden und empfangen Benutzerdaten. Die wesentlichen Aufgaben sind Segmentieren/Reassemblieren, Adressierung, Formatierung und Analyse von Benutzerdateneinheiten.
- Quittierungen und Übertragungswiederholungen
  Die Quittierungs- und Übertragungswiederholungsfunktionen können ebenfalls in zwei Automaten aufgeteilt werden. Der Übertragungswiederholungs-Automat empfängt Quittungen vom Quittierungs-Automat der Partnerinstanz, an den er aber auch selbst Informationen über gesendete oder quittierte Daten sendet. Übertragungswiederholungen werden in Abhängigkeit von den Quittungen vorgenommen, d.h. die Strategie für Übertragungswiederholungen (z.B. selektive Übertragungswiederholungen oder Go-back-N) ist in diesem Automaten realisiert. Vom Automaten, der in derselben Instanz für das Senden von Benutzerdaten zuständig ist, erhält der Übertragungswiederholungs-Automat dazu in gewissen Abständen Informationen über gesendete, d.h. von der Partnerinstanz zu quittierende Daten. Außerdem initiiert er die Freigabe des von quittierten Daten belegten Speichers. Der Quittierungs-Automat sendet an den Sender der Partnerinstanz negative oder positive Quittungen abhängig von den Ergebnissen der Überprüfung des Benutzerdaten-Empfängers.
- Fluß- und Übertragungsratenkontrolle
  Für die Fluß- und Übertragungsratenkontrolle existieren zwei Automaten, d.h. je einer für jede Übertragungsrichtung. Die Automaten steuern den Übertragungsfluß durch die Auswertung von Signalen der Partnerinstanz wie z.B. Fenster- und Puffergrößen sowie Übertragungsratenparameter. Zur Übertragungsratenkontrolle vereinbaren beide Partnerinstanzen geeignete Übertragungsratenparameter. Der Flußkontroll-Empfänger-Automat kann z.B. Übertragungsratenparameter oder Fenstergrößen entsprechend des augenblicklichen Verkehrs oder des verfügbaren Empfangsspeicherplatzes dem Flußkontroll-Sender-Automaten der Partnerinstanz mitteilen. Dieser steuert dann den Datenfluß, indem er dem für das Senden von Benutzerdateneinheiten verantwortlichen Automaten derselben Instanz entsprechende Parameter mitteilt. Er kann außerdem neue Kredite vom Flußkontroll-Empfänger-Automaten der Partnerinstanz anfordern.
- Staukontrolle
  Die beiden Automaten steuern den Datenfluß unter Berücksichtigung von Feedback-Signalen des Netzes oder der Partnerinstanz, indem anhand der empfangenen Signale Stausituationen erkannt und Übertragungsparameter anderer Automaten beeinflußt werden. Beispiele solcher Feedback-Signale sind Verzögerungen und Umlaufzeiten von speziellen Testpaketen, Zwischenankunftszeiten, Stauanzeigen der Netzwerkzugangseinheit oder der Transitsysteme. Aufgrund der Vielzahl der möglichen Feedback-Signale und der zu erwartenden aufwendigen Berechnung und Abschätzungen von Stausituationen erscheint ein separater Automat für die Staukontrolle gerechtfertigt.
- Jitter-Kontrolle
  Der Jitter-Kontroll-Automat unterstützt die Aufrechterhaltung von Zeitbeziehungen innerhalb eines Datenstroms, wie sie beispielsweise in Videoanwendungen gefordert werden. Dies kann z.B. durch die Übertragung von Zeitstempeln durch den Sender-Automaten erfolgen. Der Empfänger in der Partnerinstanz sorgt dann für die zeitlich korrekte Auslieferung der Benutzerdaten an den Transportdienstbenutzer.

- Wegewahl und Vermittlung
  Der Automat führt in Transitsystemen Funktionen zur Weiterleitung, Wegewahl und Routen-Verwaltung durch.

Die Dekomposition des Protokolls in Protokollautomaten erlaubt neben der effizienteren Parallelverarbeitung durch die erzielte Modularität der Protokollautomaten eine einfachere und flexiblere Konfigurierbarkeit bzw. Anpaßbarkeit an unterschiedliche, sich eventuell dynamisch ändernde Anforderungen durch Anwendungen bzw. Eigenschaften der Netzwerke. In Abhängigkeit davon können bestimmte Strategien, Mechanismen oder Parameter eingestellt werden. Konfigurationsaspekte von Transportprotokollen werden in [22] ausführlicher behandelt.

4.1.3. Datenformate

Die Protokollautomaten besitzen spezielle Adressen, welche Erweiterungen der Instanzenadresse darstellen. Dazu werden einfache Bitfelder (*fsm*-Feld, s. Abb. 3a) benutzt, wobei jedes Bit einen Automatentyp repräsentiert. Dieses Adressierungsschema erlaubt auch die Mehrfachadressierung von Automaten, d.h. ein Automat kann durch Setzen der entsprechenden Bits in der Automatenadresse eine Protokolldateneinheit an mehrere Automaten der Partnerinstanz schicken. Die Mehrzahl der Kommunikationsbeziehungen besteht aber zwischen Automaten gleicher Funktionalität.

Die Protokolldateneinheiten (Abb. 3a) bestehen generell aus einem allgemeinen Paketkopf und einem für jeden Automatentyp spezifischen Teil. Der allgemeine Teil des Paketkopfs umfaßt Adressen, Verbindungsidentifikatoren, Typ-Felder, Längeninformation und ein Feld für die Lebenszeit des Pakets. Der automatenspezifische Teil für die Benutzerdaten (Abb. 3b) enthält einen speziellen Kopf mit zusätzlichen Längeninformationen, Sequenznummern sowie einem variablen Teil für die eigentlichen Daten.

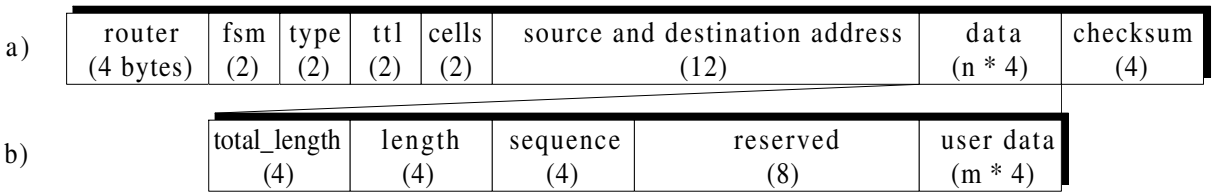

Abb. 3: Datenformate

Unter Benutzung des IP-Adressierungsformats reicht für den gesamten Kopf eines Benutzerdatenpakets, d.h. für den allgemeinen und den speziellen Kopf, eine ATM-Zelle aus. Kontrolldateneinheiten haben in der Regel die Länge von einer oder zwei ATM-Zellen und eignen sich deshalb ebenfalls für die Übertragung in zellenbasierten Netzen. In Netzen mit größeren Übertragungseinheiten, z.B. FDDI, können mehrere Kontrollpakete in einer Übertragungseinheit zusammengefaßt werden.

4.2. Kooperation der Protokollautomaten einer Instanz

Die Kooperation einiger wesentlicher Protokollautomaten bei einer unidirektionalen Datenübertragung ist in Abbildung 2 dargestellt. Die Kommunikation zwischen Automaten einer Instanz ist auf ein Minimum reduziert und so gestaltet, daß sie während der Datenübertragungsphase nur unidirektional abläuft.

Verbindungsaufbau und -abbau

- Der Verbindungsverwaltungs-Automat initialisiert die Automaten bei einer neuen Verbindung und deaktiviert diese wieder beim Abbau.
- Beim Verbindungsabbau muß der Verbindungsverwaltungs-Automat in Zusammenarbeit mit den Übertragungswiederholungs-/Quittierungs-Automaten und dem Automaten an der Transportdienstschnittstelle dafür sorgen, daß alle zuvor übergebenen Daten gesendet und quittiert werden.

Flußkontrolle

- Die Flußkontrolle auf der Senderseite steuert den Benutzerdaten-Sender durch die ausgehandelten Flußkontrollparameter bzw. den vom Staukontroll-Automaten erhaltenen Signale. Der Benutzerdaten-Sender kann gegebenenfalls neue Kredite anfordern.
- Auf Empfängerseite muß der Benutzerdaten-Empfänger dem Flußkontroll-Automaten Informationen über den aktuellen Pufferzustand mitteilen. Letzterer berechnet aus diesen Informationen neue Parameter für den Flußkontroll-Automaten auf der Senderseite der Partnerinstanz.

Übertragungswiederholungen und Quittierungen

- Der Benutzerdaten-Sender übergibt die Kontrolle über einmal gesendete Daten an den Übertragungswiederholungs-Automaten, der dann für eventuell erforderliche Übertragungswiederholungen verantwortlich ist.
- Der Benutzerdaten-Empfänger informiert den Quittierungs-Automaten periodisch über korrekt angekommene bzw. unmittelbar über fehlende Benutzerdaten und gegebenenfalls empfangene Übertragungswiederholungen.

4.3. Implementierung

Die Spezifikation des parallelen Transportsubsystems wurde direkt in eine prototypische Implementierung auf Transputerbasis umgesetzt [24]. Ähnlich wie bei der diskutierten XTP-Implementierung werden die aus den Automaten resultierenden Prozesse auf Prozessoren einer geeigneten Prozessorkonfiguration abgebildet. Die Prozesse werden über virtuelle Kanäle miteinander verbunden, welche auf Softwarekanäle oder Transputer-Links abgebildet werden. Die globalen Speicher für zu sendende oder empfangene Benutzerdaten werden durch den Prozessoren gemeinsam zugängliche Speicher realisiert. Erste Untersuchungen deuten auf wesentlich höhere Leistungsdaten im Vergleich zu Implementierungen entsprechender Protokolle wie XTP hin. Spezialisierte Hardwareeinheiten für besonders zeitkritische Funktionen wie Prüfsummenberechnung, Zeitgeber- und Pufferverwaltung sollen Softwarerealisierungen ersetzen. Die Kombination von Universal- und Spezialprozessoren resultiert in einer hybriden Multiprozessorarchitektur, welche eine hohe Leistungsfähigkeit bei gleichzeitiger Flexibilität garantiert.

5. Zusammenfassung

Dieser Beitrag behandelte den Entwurf eines parallelen Transportsubsystems, der wesentlich durch die Analyse der XTP-Protokollarchitektur sowie aus Erfahrungen und Ergebnissen einer parallelen XTP-Implementierung auf Transputern beeinflußt wurde. Neben der Eignung des Transportsubsystems für zellenbasierte Netze besteht der wesentliche Entwurfsaspekt in der Aufteilung der Protokollinstanz in modulare, weitgehend unabhängige und daher parallel aus-

führbare Protokollautomaten. Der Entwurf sieht für alle Kontrollfunktionen und für die Benutzerdaten verarbeitenden Funktionen separate Protokollautomaten vor. Die Synchronisation zwischen Protokollautomaten einer Instanz wird durch den Einsatz spezieller Protokolle für die Kommunikation zwischen Protokollautomaten verschiedener Instanzen reduziert. Das entworfene parallele Transportsubsystem wird derzeit auf einer hybriden Multiprozessorarchitektur implementiert, wobei Transputer als Universalprozessoren eingesetzt werden und besonders zeitkritische Protokollfunktionen durch spezielle Hardwareeinheiten realisiert werden sollen.

Referenzen

[1] Zitterbart, M.
Funktionsbezogene Parallelität in transportorientierten Komunikationsprotokollen
VDI Verlag, Reihe 10: Informatik/Kommunikationstechnik, Nr. 183

[2] Zitterbart, M.
High Speed Transport Components
IEEE Network Magazine, Vol. 5, No. 1, Januar 1991, S. 54-63

[3] Chesson, G.
The Evolution of XTP
3rd IFIP WG 6.4 Conference on High Speed Networking, Berlin, 18.-22. März 1991, S.15-24

[4] Protocol Engines Inc.
XTP Protocol Definition
Revision 3.4, 17. Juli 1989

[5] Zitterbart, M.
Parallel Protocol Implementations on Transputers - Experiences with OSI TP4, OSI CLNP, and XTP
IEEE Workshop on the Architecture and Implementation of High Performance Communication Subsystems, Tucson, Arizona, U.S.A., 17.-19. Februar 1992

[6] INMOS Ltd.
The Transputer Databook
2nd edition 1989, Redwood Burn Ltd, Trowbridge.

[7] Strayer, W.T.; Dempsey, B.J.; Weaver, A.C.
XTP: The Xpress Transfer Protocol
Addison-Wesley Publishing Company, 1992

[8] Braun, T.; Zitterbart, M.
A Parallel Implementation of XTP on Transputers
16th Annual IEEE Conference on Local Computer Networks, Minneapolis, Minnesota, U.S.A., 14.-17. Oktober 1991, S. 321-329

[9] Diot, C.; Roca, V.
XTP/KRM Implementation on a Transputer Network
16th Annual IEEE Conference on Local Computer Networks, Minneapolis, Minnesota, U.S.A., 14.-17. Oktober 1991, S. 310-320

[10] Heinrichs, B.
XTP Specification and Parallel Implementation
International Workshop on Adavanced Communications and Applications for High Speed Networks, München, 16.-19. März 1992, S. 77-84

[11] Miloucheva, I.; Rebensburg, K.
XTP Service Classes and Routing Strategies for an Integrated Servcices Broadband Environment

International Workshop on Adavanced Communications and Applications for High Speed Networks, München, 16.-19. März 1992, S. 231-240

[12] Mitchell, R. J.; Saulnier, E. T.; Orlovsky, M.C.
A Partitioned Implementation of the Xpress Transfer Protocol
16th Annual IEEE Conference on Local Computer Networks, Minneapolis, Minnesota, U.S.A., 14.-17. Oktober 1991, S. 301-309

[13] Simoncic, R.; Weaver, A. C.; Colvin, M. A.
Experience with the Xpress Transfer Protocol
15th Annual IEEE Conference on Local Computer Networks, Minneapolis, Minnesota, U.S.A., 30. September - 3. Oktober 1990, S. 123-131

[14] Schwaderer, W. D.;
XTP in VLSI: Protocol Decomposition for ASIC Implementation
15th Annual IEEE Conference on Local Computer Networks, Minneapolis, Minnesota, U.S.A., 30. September - 3. Oktober 1990, S. 249-252

[15] Watson, R. W.:
The Delta-t Transport Protocol: Features and Experience
Rudin, H.; Williamson, R. (Hrsg.): Protocols for High Speed Networks, Elsevier Science Publishers B. V. (North Holland), IFIP, 1989, S. 3-18

[16] Clark, D. D.; Lambert, M. L.; Zhang, L.:
NETBLT: A High Throughput Transport Protocol
Proceedings of the ACM SIGCOMM '87, Stowe, Vermont, U.S.A., August 1987, S. 353-359

[17] Cheriton, D. R.; Williamson, C. L.:
VMTP as the Transport Layer for High-Performance Distributed Systems
IEEE Communications Magazine, Vol. 27, No. 6, Juni 1989, S. 37-44

[18] Sabnani, K.; Netravali, A.; Roome, R.
Design and Implementation of a High Speed Transport Protocol
IEEE Transactions on Communications, Vol. 38, No. 11, November 1990, S. 2010-2024

[19] Protocol Engines Inc.
XTP Protocol Definition
Revision 3.6, PEI 92-10, 11. Januar 1992

[20] Schörk, A.
Parallele Realisierung von XTP auf Transputern
Diplomarbeit, Universität Karlsruhe, Oktober 1990

[21] Weber, Th.
Untersuchung und Adaption des XTP-Protokolls auf Transputer-Netzen
Diplomarbeit, Universität Karlsruhe, Juli 1991

[22] Zitterbart, M.; Stiller, B.; Tantawy, A.
A Model for Flexible High-Performance Communication Subsystems
IBM Research Report, RC 17801, Februar 1992, erscheint in: Journal of Selected Areas in Communications, 1993

[23] Braun, T.
A Parallel High Performance Transport System for Metropolitan Area Networks
5th IEEE Workshop on Metropolitan Area Networks, Taormina, Italien, 10.-13. Mai 1992, erscheint in: Computer Communications, 1993

[24] Schmidt, C.
Entwurf eines Transportsystems auf der Basis paralleler Automaten
Diplomarbeit, Universität Karlsruhe, Juli 1992