# Vorwort

Die vorliegende Arbeit nutzt die Methoden der parallelen Programmierung und realisiert auf Kernel-Ebene Algorithmen und Datenstrukturen, die das Zusammenwirken der vielen Teile eines echtzeitfähigen Kommunikationsnetzwerkes transparent machen.
Das Netzwerk wird als Echtzeit-Multitaskingsystem modelliert, das als „Fundamente“ die typischen Kernfunktionen eines Echtzeitbetriebssystems und die Multitasking-Fähigkeit der Intel-Prozessorarchitektur (Pentium) nutzt.
Die Arbeit zeigt, wie die typischen Layer eines tokenbus-basierten Netzwerks (Phys Layer 1, MAC Layer 2a, LLC Layer 2b und Transport Layer 4 mit UDP) als Tasks abgebildet und in das Multitaskingsystem integriert werden können. Die Tasks führen die Protokolle ihrer Layer als parallele Programme aus, die sich als Kernelerweiterungen präsentieren und dafür sorgen, dass der rege vertikale und horizontale Botschaftenaustausch innerhalb der Rechner und zwischen den Rechnern verklemmungsfrei und sicher abläuft.
In Konsequenz führt die Nutzung dieser Methode zu einer Implementierung, die die vielen Rechner im Kommunikationsnetz wie einen einzigen Parallelrechner mit Echtzeitverhalten aussehen lässt.

**Die Arbeit besteht aus fünf Kapiteln**.

Das 1. Kapitel behandelt den MAC Layer 2a ( Medium Access Control).

Als Mediumzugangskontrolle wird das priorisierte tokenpassing-Verfahren implementiert und verifiziert. Die wesentlichen Strukturmerkmale des MAC-Layers sind die Receive Machine (RxM), die Transmit Machine (TxM) und als Herzstück die Access Control Machine (ACM). Die Receive Machine korrespondiert mit dem Physical Layer und bekommt von dort sogenannte MAC Protocol Data Units (PDUs). Sie überprüft die Frame Check Sequence (FCS) der individuellen PDUs, stellt Fehlermeldungen des UART fest und gibt die Ergebnisse weiter an den LLC Layer und die Access Control Machine. Die Transmit Machine bekommt Daten Frames von der Access Control Machine, berechnet von jedem die Frame Check Sequence, hängt sie an die Daten Frames an und schickt sie als MAC PDUs an den Physical Layer.
Die Access Control Machine stellt sich als eine endliche Zustandsmaschine dar (finite state machine) und bestimmt, wann ein Frame in das Netz geschickt wird. Zu diesem Zweck kooperiert sie mit den ACMs aller anderen Rechner.

Das 2. Kapitel behandelt den LLC Layer 2b (Logical Link Control).

Sie ist die logische Schnittstelle zwischen dem MAC Layer und dem Transport Layer. Ihre Aufgabe ist es, für eine effiziente und zuverlässige Kommunikation zwischen zwei Rechnern zu sorgen. Das dort implementierte Sicherungsprotokoll realisiert einen sogenannten bestätigten verbindungsunabhängigen Dienst, der für tokenbus-basierte Kommunikationsnetze typisch ist. Es zeichnet sich durch seine Robustheit aus, das mit allen Attacken auf den Bitstrom im Netz fertig wird. Durch die Verteilung von Exemplaren dieses Protokolls auf „maßgeschneiderte“ LLC-Tasks, werden sichere Datenübertragungen parallel zwischen den Anwender-Tasks beliebig vieler Rechnern möglich.

Das 3. Kapitel behandelt das User Datagram Protocol (UDP) im Transport Layer 4.

In tokenbus-basierten Kommunikationsnetzen wird bevorzugt UDP eingesetzt. Die typischen Merkmale für dieses Protokoll sind folgende: Es garantiert, dass Botschaften eines Anwenderprogramms abgeschickt werden, aber es garantiert

nicht, dass diese Botschaften beim Anwenderprogramm des Empfängers auch ankommen. Bei UDP werden als ultimative Ziele nicht Tasks „genannt“, sondern abstrakte Zielpunkte, sogenannte protocol ports. Auch ordnet das Protokoll nicht durcheinandergekommene Botschaften. Hier Ordnung zu schaffen, liegt optional beim Anwender. Um dies zu ermöglichen, sind die erforderlichen Datenstrukturen und Parallel-Algorithmen als Kernelerweiterungen implementiert.

Das 4. Kapitel behandelt die verbindungsorientierte LLC-Kommunikation.

Es analysiert die verbindungslose Kommunikation und zeigt, dass sich der Nutzungsgrad des LANs mit der Anzahl der Rechner dramatisch verringert. In einem solchen Netzwerk sind alle Rechner über „ihre“ LLC-Tasks für immer miteinander in Kontakt. Falls keine Anwender-Daten bereit stehen, tauschen die Rechner ununterbrochen nutzlose Quittungs-PDUs aus. Ein Ausweg aus diesem Dilemma ist ein Netzwerk, das verbindungsorientierte Kommunikation unterstützt. In einem solchen System sind die Rechner lediglich während des Datenaustausch miteinander verbunden. Der Nutzungrad des LANs ist jetzt nicht mehr von der Anzahl der Rechner abhängig und kann den Maximalwert erreichen. Die Funktionen zum Aufbau und Abbau von LLC-Verbindungen werden als Kernel-Erweiterungen entwickelt und stehen zur Unterstützung des User Datagram Protocols zusammen mit dem checksum-Algorithmus als Service Access Points (SAPs) zur Verfügung. Von einem erfolgreichen Verbindungsaufbau/-abbau muss zusätzlich gefordert werden, dass sich die beteiligten Anwender-Tasks synchronisiert verbinden oder trennen. Eine Analyse der Rendezvous-Bedingungen, gefolgt von Lösungen zur Verhinderung von deadlocks, schließt das Kapitel ab.

Das 5. Kapitel enthält eine Einführung in verteilte Systeme.

Die Existenz leistungsstarker Mikroprozessoren auf der einen Seite und lokaler Netzwerke auf der anderen Seite, führt zu der Idee, beide Technologien zu vereinigen und ein Rechner-System zu konstruieren, das aus einer Vielzahl von Prozessoren besteht, die über ein Kommunikationsnetzwerk miteinander verbunden sind. Diese Idee ist in den vorangegangenen Kapiteln bereits konsequent umgesetzt. Ziel eines solchen Systems ist es, dem Benutzer die Illusion zu vermitteln, dass er es mit einem einzigen Rechner zu tun hat und nicht mit einer Ansammlung von mehreren unterschiedlichen Rechnern. Dabei spielt die Task-Kommunikation eine wesentliche Rolle. Ein solches System verhält sich wie ein virtuelles Einprozessorsystem mit dem Makel, dass es dort keinen gemeinsamen Speicher und keine gemeinsamen Betriebsmittel gibt. So müssen ausgesuchte Rechner ihre lokale Einheiten wie Speicher und IO als virtuelle globale Datenstrukturen für den gemeinsamen Zugriff zur Verfügung stellen. Dies macht sie zu kritischen Regionen, die nur unter gegenseitigem Ausschluss betreten werden dürfen. Die Simulation der dafür notwendigen P/V-Operationen in einem zentralen Algorithmus und die Strategie zur Vermeidung von deadlocks bei vielfachem Betriebsmittelbedarf sind die Kernaspekte dieses Kapitels. Damit sich die Kommunikation nicht ausschließlich auf explizite Datenübertragungen beschränkt, greift es die Idee von Nelson und Birrell auf und schließt mit dem Mechanismus des entfernten Prozeduraufrufs (RPC) einschließlich den Input-/Output-RPCs ab.

Klaus-Dieter Thies, Ph.D.