Das Querschnittsthema "MEMSY"
Ein Modulares Erweiterbares Multiprozessor SYstem

Prof. F. Hofmann

Lehrstuhl für Betriebssysteme - Informatik IV, Universität Erlangen-Nürnberg

Überblick

Die Erfahrung der Erlanger Informatiker mit Multiprozessoren und Verteilten Systemen umfaßt ein breites Spektrum von Gesichtspunkten, wie Verarbeitungs- und Speichereinheiten, Verbindungsstrukturen, Zerlegung von Anwenderaufgaben in parallel ausführbare Teile, Programmierung praktischer Lösungen der Kommunikations- und Synchronisationsabwicklung, Betriebssysteme für verschiedene Topologien, Messung und Analyse des Ablaufgeschehens, Zusammenhang von Aufgaben bzw. Algorithmenstruktur und erforderlicher Multiprozessorstruktur und Fehlertoleranzaspekte. In diesen umfassenden Rahmen ist die Architektur einer Familie von modular erweiterbaren und konfigurierbaren Multiprozessorsystemen mit breitem Leistungsspektrum realisiert worden. Es sind grundlegende Untersuchungen der "Parallelität" in der Systemstruktur, der Kommunikation, der Berechnungsalgorithmen der Anwenderaufgaben, sowie der Leistungsmessung und -bewertung durchgeführt worden.

1 Motivation

Der starke Entwicklungstrend zu Parallelrechnern in den letzten Jahren hat seine Ursachen u.a. in

- der erkennbaren Annäherung an die technisch mögliche Grenzleistung konventioneller Computer und Vektorrechner
- dem steigenden Rechenleistungs- und Speicherbedarf auf der Anwenderseite
- dem erreichten technologischen Stand sehr leistungsfähiger und preisgünstiger, hochintegrierter Rechnerkomponenten, welche die wirtschaftliche Realisierung parallelverarbeitender Systeme möglich machen
- der Erkenntnis, daß ein weites Spektrum von Anwendungsproblemen nur unzulänglich vektorisierbar ist
- dem extrem besseren Preis/Leistungsverhältnis heutiger Riscarchitekturen in der Floating-Point-Skalarleistung gegenüber konventionellen Vektorprozessoren

Hof93.doc: 27.03.95 17:22

Von den Parallelrechnern sind Systeme der MIMD-Klasse (*multiple instruction - multiple data*) besonders vielseitig einsetzbar, da sie wegen ihrer unabhängigen Befehlswerke gleiche oder verschiedene Datenströme gleich oder unterschiedlich bearbeiten können.

Die Größe solcher Systeme wird im allgemeinen beschränkt durch die Anzahl der Prozessoren, die auf einen Speicher zugreifen, oder durch die Komplexität des Verbindungsnetzwerkes. Diese Beschränkungen können weitgehend aufgehoben werden durch den Einsatz verteilter Speicher (distributed shared memory) und durch die Verwendung von Verbindungsstrukturen mit (größenunabhängiger) konstanter lokaler Komplexität. Mit solchen speichergekoppelten Multiprozessoren können, wie Untersuchungen gezeigt haben, insbesondere Aufgaben mit überwiegend lokalem Datenaustausch mit hoher Effizienz bearbeitet werden. Dazu gehört u.a. die numerische Simulation physikalischer und chemischer Vorgänge, die heute in Forschung und Industrie eine große Rolle spielt.

Bei der Durchsetzbarkeit von Parallelrechnern stellt die Notwendigkeit für die Anwender

- in asynchronen parallelen Abläufen “zu denken” und
- extrem architekturabhängig zu programmieren

das Haupthindernis dar.

Allen Anstrengungen, die Anwender durch geeignete Hilfsmittel bzw. durch Einziehen geeigneter Abstraktionsebenen von diesen Notwendigkeiten zu entbinden, ist bisher nur in Ansätzen gelungen und in der Regel mit erheblichen Leistungsverlusten verbunden. Gegenläufig zu diesen Bemühungen ist nicht zuletzt auch die Tatsache, daß die kommerziellen Anwender - wohl zurecht - zunächst den klassischen “Fortran-Naturwissenschaftler” als potentiellen Kunden sehen, dessen Bedürfnisse in erster Linie zu befriedigen sind.

Für Beiträge zur Lösung der o.g. Fragestellungen sollte ein interdisziplinärer Sonderforschungsbereich besonders prädestiniert sein. Insbesondere dann, wenn durchgängig über alle klassischen Ebenen eines Parallelrechensystems hinweg das notwendige “know how” an einem Ort konzentriert ist. Diesen Zusammenhalt weiter zu fördern, dient das Querschnittsthema MEMSY.

1.1 Zielsetzung

Die Erfahrung der Erlanger Informatiker mit Multiprozessoren und verteilten Systemen an verschiedenen Lehrstühlen erstreckt sich auf ein breites Spektrum von Gesichtspunkten, wie Verbindungsstrukturen, Zerlegung von Anwenderaufgaben in parallel ausführbare Teile, Programmierung praktischer Lösungen der Kommunikations- und Synchronisationsabwicklung, Betriebssysteme für verschiedene Topologien, Messung und Analyse des Ablaufgeschehens, Zusammenhang von Aufgaben- bzw. Algorithmenstruktur und erforderliche Multiprozessorstruktur für den Fall der Fehlertoleranz.

Hof93.doc: 27.03.95 17:22

Die bisher geleisteten Forschungsarbeiten mit den enggekoppelten Multiprozessorsystemen EGPA, SYMPOS, DIRMU, MUPSY und SUPRENUM haben bewiesen, daß grundsätzlich Parallelisierung erfolgreich zur Leistungssteigerung eingesetzt werden kann. Es hat sich gezeigt, daß offene Probleme des E/A- und des Verbindungssystems, der Adaptierbarkeit von Betriebssystemen auf verschiedene Rechnerarchitekturen, der Programmier- und Ablaufumgebung, der Aufgabenverteilung, der Integration von Leistungsmessung und -bewertung, sowie der Einführung von Standardschnittstellen und -komponenten weiter erforscht werden sollten. Dabei kommt der Speicherkopplung als Schnittstelle besondere Bedeutung zu. Die bisherige Forschung ergab aber auch, daß nur die Verbindung von wohlüberlegter Planung mit tatsächlicher Erprobung der Konzepte zu relevanten Aussagen über die Tragfähigkeit von Ideen führen kann.

2 Einführung in den Hardware-Aufbau von MEMSY

Das hier vorgestellte Multiprozessor-Familienkonzept "MEMSY" basiert auf gemeinsamen grundlegenden Architektureigenschaften:

- Interprozessorkommunikation durch Speicherkopplung zwischen benachbarten Prozessoren: Konstante lokale Komplexität bei beliebiger Ausbaubarkeit.
- Die Prozessor-Speicher-Knoten haben denselben Aufbau. Eine Anpassung an die Zielanwendung kann u.a. durch geeignete Spezial-Prozessoren (z.B. für schnelle Gleitkommaverarbeitung) erfolgen.

Es werden Universalprozessoren eines Typs (oder weniger Typen) verwendet. Unterschiedliche Spezialprozessoren können hinzugefügt werden. Jeder Universalprozessor hat Zugriff zu mehreren (i.a. aber nicht zu allen) Speichermoduln. Es gibt Speicher, auf die mehrere Prozessoren zugreifen. Diese verhalten sich logisch wie Multiport-Speicher. Es gibt Speicher mit nur einem Prozessor als Zugreifer (Privatspeicher).

2.1 Topologie

MEMSY hat eine pyramidenförmige Struktur und kann in zwei Ebenen unterteilt werden (A- und B-Ebene). Der schematische Aufbau eines Systems mit 20 Knoten, das als Experimentalsystem aufgebaut wurde, ist in Abb. 2.1 dargestellt. Die Knoten der beiden Ebenen sind jeweils in einem Gitter angeordnet, das zum Torus geschlossen ist. Jedem Knoten der B-Ebene sind genau vier Knoten aus der untersten Ebene, der A-Ebene, zugeordnet. Diese fünf Knoten bilden eine *Elementarpyramide*. Die Kopplung zwischen den Knoten innerhalb einer Ebene erfolgt über gemeinsame Speichermoduln. Dazu ist jedem Knoten ein zusätzlicher Speicher, der Kommunikationsspeicher, zugeordnet. Ein Knoten kann dann auf den Kommunikationsspeicher seiner vier direkten Nachbarn zugreifen (Nearest-Neighbour-Prinzip). Dadurch ergibt sich eine konstante lokale Verbindungskomplexität. Die Verbindung zwischen den Ebenen erfolgt ebenfalls über gemeinsame Speicher, wobei sie nicht, wie innerhalb einer Ebene, symmetrisch ist.

Das heißt, die Knoten der B-Ebene können zwar auf den Kommunikationsspeicher der zugeordneten vier Knoten der A-Ebene zugreifen, nicht aber umgekehrt. Die Knoten der B-Ebene sind über einen Bus mit einem Front-End-Rechner verbunden. Eine ausführliche Beschreibung des Aufbaus und des Verbindungsnetzwerks kann in [Hil92] gefunden werden.

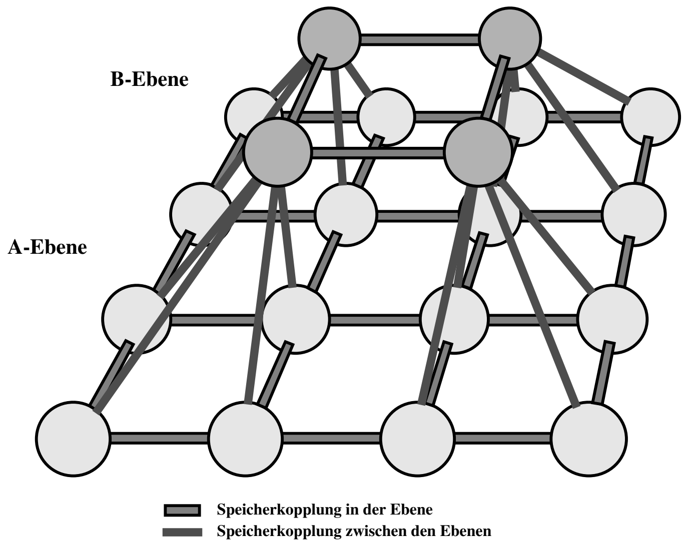

Abb. 2.1 20er MEMSY System (ohne Torusverbindungen)

2.2 Aufbau eines MEMSY-Knotens

Ein Knoten der A- oder B-Ebene setzt sich aus dem Prozessormodul mit integriertem Cache, dem lokalen Hauptspeicher und spezieller Zusatzbaugruppen zusammen, von denen einige im Rahmen des MEMSY-Projekts entwickelt wurden. Diese dienen im wesentlichen zur Ansteuerung der Kommunikationsspeicher und zur Erzeugung von Unterbrechungen bei Nachbarknoten über dedizierte Verbindungen. Darüberhinaus ist pro Knoten ein privater Plattenspeicher und ein FDDI-Anschluß vorhanden. Der prinzipielle Aufbau eines Knotens ist in Abb. 2.2 dargestellt.

Das Konzept von MEMSY sieht vor, daß sich das System soweit möglich aus käuflichen Komponenten zusammensetzen soll. Als Prozessormodul wurde daher eine Multiprozessor-Baugruppe von Motorola gewählt, die vier RISC-Prozessoren und insgesamt acht Cache-Einheiten der 88k-Familie trägt und über einen lokalen Hochgeschwindigkeitsbus und einen VME-Bus verfügt. An den lokalen Bus ist die Ansteuerschaltung für die Kommunikationsspeicher angeschlossen. Die Kommunikationsspeicher werden grundsätzlich wie normaler Hauptspeicher behandelt. Im gekoppelten System entscheidet lediglich die Adressierung auf welchen der verfügbaren Speicher (Hauptspeicher, eigener Kommunikationsspeicher oder Nachbar-Kommunikationsspeicher) zugegriffen wird. Das Interrupt-Subsystem ermöglicht es Unterbrechungen bei anderen Knoten auszulösen. Dazu müssen eigene Knoten-zu-Knoten Verbindungen hergestellt werden. In MEMSY laufen diese Verbindungen parallel zur Speicherkopplung, das heißt ein Knoten hat zu genau dem Nachbarn eine Interrupt-Verbindung, mit dem er einen gemeinsamen Kommunikationsspeicher hat.

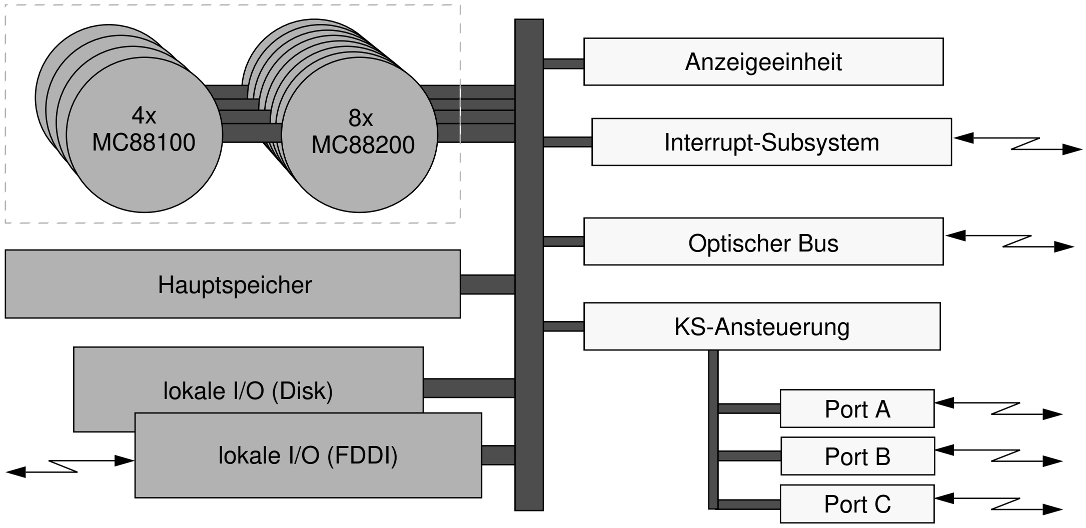

Abb. 2.2 Prinzipieller Aufbau eines MEMSY-Knotens

2.3 Leistungsdaten

Für das Experimentalsystem aus 20 Knoten mit insgesamt 80 Prozessoren ergeben sich folgende Leistungsdaten:

- 2000 MIPS (peak) ca. 1440 MIPS Dauerleistung
- 1000 MFLOPS (peak) ca. 800 MFLOPS Dauerleistung
- 10 MByte physikalischen Cache
- 640 MByte lokaler Hauptspeicher (auf 3840 MByte ausbaubar)
- 80 MByte Kommunikationsspeicher (auf 320 MByte ausbaubar)
- 10 GByte lokale Plattenkapazität (nahezu beliebig ausbaubar)
- 100 MByte/s IO-Bandbreite zu den Festplatten (auf 300 MByte/s ausbaubar)
- 100 MBit/s Bandbreite (FDDI) zum Datentransport von und zu dem System
- Vervielfachung der Bandbreite durch den Einsatz mehrerer FDDI-Ringe
- optisches Experimental-Bussystem (z.Zt. voraussichtlich 5 MBit)
- 400 MByte/s Bandbreite zu den lokalen Hauptspeichern
- 177 MByte/s Bandbreite zu den Kommunikationsspeichern

3 Programmiermodell

Wichtig für den Erfolg oder Mißerfolg eines Multiprozessorsystems sind nicht allein seine technischen Leistungsdaten, sondern auch die Programmierumgebung und das Programmiermodell, das den Anwendern zur Verfügung gestellt wird.

Für das MEMSY-System wurde der Ansatz gewählt, ein neues Programmiermodell zu entwerfen und sich nicht an bestehende Systeme anzulehnen, da die Hardware von MEMSY einige Eigenheiten aufweist, die sich in anderen Systemen nicht oder nur in abgewandelter Form finden lassen (mehrere gemeinsame Speicher mit Zugriff einer Teilmenge der Prozessoren und mehrere unterschiedliche Kommunikationsmedien).

Das im folgenden beschriebene Programmiermodell soll dem Anwender einen einfachen Zugriff auf die Fähigkeiten des MEMSY-Systems bieten und ihn bei der Entwicklung und Effizienzsteigerung seiner Anwendungsprogramme unterstützen.

3.1 Architektur von MEMSY aus der Sicht des Anwenders

Die Sicht des Anwenders auf die Architektur entspricht im wesentlichen der tatsächlichen Architektur, wie sie in Abschnitt 2.1 beschrieben wurde. Unsichtbar für den Anwender sind lediglich einige Besonderheiten der Hardware-Implementierung, wie die teilweise diagonale Kopplung der Knoten und die durch das Verbindungsnetzwerk möglichen Mehrfachzugriffe auf die Kommunikationsspeicher [Hil92].

3.1.1 Speicher

Jeder Prozessor hat einen oder mehrere Kommunikationsspeicher, auf den seine Nachbarprozessoren oder eine Teilmenge davon ebenfalls Zugriff haben. Als Nachbarprozessoren gelten die vier in der Ebene direkt anschließenden Prozessoren, der in der Pyramide übergeordnete Prozessor (für die untere Ebene) und die vier unterhalb eines Prozessors angeordneten Prozessoren (für die obere Ebene).

3.1.2 Knoten

Jeder Knoten des MEMSY-Systems ist wieder ein Mehrfachprozessorsystem, das aus vier Prozessoren aufgebaut ist. Die Anwendung hat die Möglichkeit, mehrere Prozesse auf einem Knoten zu erzeugen, um die volle Parallelität der Knoten nutzen zu können. Die Adressierung der Knoten erfolgt in der Regel durch Angabe der relativen Position, dadurch sind Verbindungen zu den Nachbarknoten uniform für alle Knoten programmierbar. Eine absolute Adressierung ist ebenfalls möglich. Die Position im Feld ist durch das Koordinaten Tripel (x,y,z) definiert, wobei ein Prozessor der oberen Ebene als (0,0,1) definiert ist. Der in der unteren Ebene direkt südwestlich von ihm angeordnete Prozessor definiert den Nullpunkt dieser Ebene (0,0,2).

3.2 Applikationen

Alle Prozesse, die zu einer Anwendung gehören, werden zu einer Applikation zusammengefaßt. Mehrere Applikationen können gleichzeitig im System ablaufen. Wenn sie auf unterschiedlichen Teilmengen der Knoten ablaufen, so beeinflussen sie sich nicht, ansonsten konkurrieren sie um die Betriebsmittel (Rechenzeit, Speicher, etc.).

Jede Applikation hat ihren eigenen Namensraum bei der Benutzung der Mechanismen, die im folgenden erläutert werden. Eine gegenseitige Beeinflussung ist damit ausgeschlossen.

Das System überwacht den Ablauf der zu einer Applikation gehörenden Prozesse. Bei Abbruch eines Prozesses kann die gesamte Applikation terminiert werden und die von ihr belegten Betriebsmittel können freigegeben werden, so daß Fehler in einer Anwendung eine begrenzte Auswirkung auf das Gesamtsystem haben.

3.3 Funktionen

Das MEMSY-Programmiermodell bietet im einzelnen folgende Funktionen an, deren genaue Beschreibung den Rahmen dieses Artikels sprengen würde, und für die deshalb auf [Kar91] verwiesen wird.

3.3.1 Gemeinsamer Speicher

Der Zugriff auf die Kommunikationsspeicher erfolgt durch ein, dem shared-memory Mechanismus von System V ähnliches System. Wie dort werden Abschnitte (Segmente) des gemeinsamen Speichers von einem Prozeß alloziert (*create*) und können dann von anderen Prozessen in ihren Adreßraum eingeblendet werden (*attach*).

Ein Nachteil der MEMSY-Architektur ist, das im allgemeinen die Caches beim Zugriff auf die Kommunikationsspeicher ausgeschaltet werden müssen, da sonst die Konsistenz der Daten zwischen den einzelnen Knoten nicht gesichert wäre. Wenn eine Anwendung sicher ist, daß es bei Zugriff auf ein bestimmtes Segment für eine gewisse Zeit nicht zu Konsistenzproblemen kommen kann (weil z.B. nur lesend oder nur von einem Knoten aus darauf zugegriffen wird), so kann sie für dieses Zeit den Cache für dieses Segment aktivieren.

3.3.2 Messages

Messages sind kurze Nachrichten, die ein Prozeß an Prozesse derselben Applikation auf einem Nachbarknoten senden kann. Die Nachrichten werden beim Empfänger gepuffert. Ein Empfang der Nachrichten kann entweder blockierend oder nicht blockierend erfolgen. Dem Empfänger wird neben dem Inhalt der Nachricht auch die Identität des Absenders mitgeteilt.

3.3.3 Semaphore

Als weitere Koordinierungsmöglichkeit bietet das System globale Semaphore an, welche die üblichen P- und V-Operationen unterstützen. Vor ihrer Benutzung müssen die Semaphore kreiert werden. Es können auf jedem Knoten Semaphore angelegt werden.

3.3.4 Signale

Es ist möglich, einem Prozeß auf einem Nachbarknoten ein Signal zustellen zu lassen, worauf dann dessen normaler Ablauf unterbrochen wird und eine vorher definierte Signalbehandlungsroutine aufgerufen wird. Auf diese Weise kann schnell auf asynchrone oder außergewöhnliche Ereignisse reagiert werden. Es können zwei verschiedene Signale verschickt werden.

3.3.5 Spinlocks

Für die Koordination kurzer kritischer Abschnitte gibt es die Spinlocks. Dies sind gemeinsame Variablen im Kommunikationsspeicher, welche die Zustände „frei" und „belegt" annehmen können.

3.3.6 Transport

Zum Transport größerer Datenmengen ist ein weiterer Nachrichtenmechanismus vorgesehen, den wir zur Unterscheidung von den Messages „Transport" genannt haben. Im Unterschied zu den Messages ist ein Transport auch zu weiter entfernten Knoten möglich. Das Betriebssystem wählt dabei den günstigsten Weg aus (Kommunikationsspeicher, FDDI-Netz oder optisches Bussystem), wobei es auch die aktuelle Lastsituation und Ausfälle von Teilsystemen berücksichtigen kann.

3.3.7 Informationsfunktionen

Zu den Informationsroutinen gehören Routinen, mit denen zwischen absoluten Knotenidentifikation, relativer Knotenidentifikation und Position im Gesamtsystem (x,y,z-Koordinaten) umgerechnet werden kann.

3.3.8 Ein-/Ausgabe

Für jeden Knoten gibt es einen knotenlokalen Plattenbereich und für das Gesamtsystem einen globalen Plattenbereich, auf die mit Varianten der UNIX-Bibliotheksfunktionen für gepufferte Ein-/Ausgabe zugegriffen werden kann.

Jeder Knoten im Experimentalsystem ist mit einer lokalen Platte ausgestattet, so daß die Zugriffe auf die knotenlokalen Datenbereiche schnell und unabhängig von anderen Knoten abgewikkelt werden können.

4 Betriebssystemarchitektur

Es sind verschiedene Überlegungen angestellt worden, welches Betriebssystem für MEMSY eingesetzt werden soll. Grundsätzlich gibt es zwei Möglichkeiten. Zum einen kann man ein komplett neues Betriebssystem entwickeln. Dies bietet den Vorteil, daß das System genau an die vorliegende Hardware angepaßt werden kann. Allerdings besteht ein großer, nicht zu unterschätzender Nachteil darin, daß dies mit sehr viel Zeit- und Arbeitsaufwand verbunden ist. Die zweite Möglichkeit, die sich bietet, besteht darin, auf ein bereits verfügbares Betriebssystem zurückzugreifen. In diesem Fall müßten nur noch die notwendigen Anpassungen an selbstentwikkelte Hardware erfolgen. Im Fall von MEMSY wurde zugunsten der zweiten Alternative entschieden. Als Basis für MEMSOS, dem Betriebssystem von MEMSY, wird das UNIX SYSTEM V/88 Release 3 von Motorola auf jedem Knoten eingesetzt. Diese Unix-Implementierung ist bereits speziell an die knotenlokale Multiprozessorarchitektur der verwendeten Hardware angepaßt.

4.1 Modifikationen und Erweiterungen am eingesetzten Betriebssystem

Vielfältige Änderungen sind am Betriebssystem vorgenommen worden. Einerseits waren Erweiterungen notwendig, um die zusätzliche Hardware in das Betriebssystem zu integrieren. Dies betrifft insbesondere die Ansteuerung der Kommunikationsspeicher und des Interrupt-Subsystems. Obwohl die gängigen Multiprozessorkonzepte bereits integriert waren, konnten sie nicht für den knotenübergreifenden Einsatz verwendet werden, da diesen Konzepten die Annahme zugrunde liegt, daß alle Prozessoren Zugriff auf einen gemeinsamen globalen Speicher haben. Dies trifft in besonderem Maße für die Interprozeßkommunikation zu, die sich mit Hilfe eines globalen gemeinsamen Speichers und simpler Lock-Techniken einfacher durchführen läßt, als in einem System mit verteiltem gemeinsamen Speicher. Innerhalb eines Knotens kann jedoch auf die vorhanden Mechanismen zurückgegriffen werden. Für knotenübergreifende Aktionen sind neue Basismechanismen eingesetzt worden, die auf den bereits erwähnten Erweiterungen aufbauen. Hier sind unter anderem die Kommunikationsspeicherverwaltung, der Unterbrechungsmechanismus und ein einfacher Nachrichtenmechanismus zu nennen. Damit MEMSY den verschiedensten Anforderungen der Anwender gerecht werden kann ist zudem das Applikationskonzept eingeführt worden. Zur Integration der meisten erwähnten Erweiterungen mußten nur wenige Modifikationen am ursprünglichen Kern vorgenommen werden, so daß Standard-Unix Anwendung weiterhin ohne Einschränkungen einsetzbar sind.

Hof93.doc: 27.03.95 17:22

4.1.1 Anwendungsunterstützung - Applikationen

Von Seiten der Anwender werden an ein Hochleistungsrechensystem verschiedene Anforderungen gestellt. Hier sind unter anderem hohe Verfügbarkeit des Systems (Ausfallsicherheit und einfaches Aufsetzen nach einem Knotenausfall), einfache Handhabbarkeit, kurze Aufsetzphasen von Anwendungen, gewohnter oder gar erweiterter Funktionalitätsumfang und die Möglichkeit zu interaktivem Testen von Anwendungen, ohne den Produktionsbetrieb zu stark zu stören, zu nennen.

Damit der Großteil der oben angeführten Anforderungen erfüllt werden kann, ist in MEMSOS das *Applikationskonzept* eingeführt worden. Die Gesamtheit aller Prozesse, die durch eine einzige Anwendung auf dem System erzeugt werden, wird als eine Einheit gesehen. Diese Einheit wird als *Applikation* bezeichnet (vergleiche auch Abschnitt 3.2). Eine Applikation ist somit ein knotenübergreifendes Prozessystem. Um mehrere parallele Applikationen realisieren zu können, muß jede Applikation global eindeutig identifizierbar sein. Dies wird durch eine Applikationsnummer erreicht. Die Prozesse einer Applikation werden *Tasks* genannt. Sie erben die Applikationsnummer und erhalten eine knotenlokal eindeutige Tasknummer. Tasks können somit eindeutig durch ihre Applikations-, Task- und Knotennummer identifiziert werden.

Durch dieses einfache Applikationskonzept ist es möglich eine Applikation, also alle zu einer Anwendung gehörenden Prozesse, eindeutig zu identifizieren und in ihrer Gesamtheit zu überwachen und zu steuern. Dadurch ist es möglich auch mehrere Applikationen zu einer Zeit auf dem System zu erlauben, da diese nun einzeln erfaßt und differenziert behandelt werden können.

5 Zusammenfassung und Ausblick

In Zusammenarbeit zwischen mehreren Lehrstühlen des IMMD und anderen Instituten der Universität Erlangen-Nürnberg wurde mit “MEMSY” ein umfassendes Konzept für ein Multiprozessor-Projekt erarbeitet und ein funktionsfähiges System aufgebaut. Die gewonnenen Ergebnisse werden möglicherweise bei entsprechender Industriebeteiligung in die Entwicklung eines Hochleistungsprototyps einfließen. In diesem umfassenden Rahmen (einschließlich der SFB-Forschungsarbeiten) sollen nicht nur die genannten Multiprozessorkonzepte weiterentwickelt werden, es soll auch eine verbesserte SW- und HW-Architektur entstehen, mit der eine Familie von modular erweiterbaren und konfigurierbaren Multiprozessorsystemen mit breitem Leistungsspektrum realisiert werden kann.

Es sollen grundlegende Untersuchungen der "Parallelität" in der Systemstruktur, der Kommunikation, den Berechnungsalgorithmen der Anwenderaufgaben, sowie der Leistungsmessung und -bewertung durchgeführt werden. In dem gegebenen günstigen Umfeld ist zu erwarten, daß die flexible und effiziente Zusammenarbeit der Gruppen umfassende konzeptionelle und praktische Erkenntnisse über Strukturen und dynamisches Verhalten von Multiprozessorsystemen erbringen wird.

Das MEMSY Experimentalsystem wird bereits jetzt von verschiedenen Anwendungen aus den Bereichen Strömungsmechanik, Theoretische Chemie und Bildverarbeitung genutzt. Es ist zu erwarten, daß sich die Anzahl an Anwendungen noch erhöht und damit auch die Praxistauglichkeit des Systems demonstriert werden kann.

6 Literaturverzeichnis

[Dal92] M. Dal Cin, „Rechnerbaukasten für funktionsorientierte Multiprozessoren mit Fehlertoleranzeigenschaften"; in: *Arbeits- und Ergebnisbericht des SFB 182 Multiprozessor- und Netzwerkkonfigurationen 1990 - 1992*, Seite 114 ff.

[Hil92] U. Hildebrand, *Konzeption, Bewertung und Realisierung einer dynamischen Netzwerkkomponente für speichergekoppelte Multiprozessoren*; Dissertation, Arbeitsberichte des IMMD, Univ. Erlangen-Nürnberg, Band 25, No. 5, 1992

[Hof92] F. Hofmann, „Entwurf und Implementierung eines an Hardwarearchitektur und Aufgabenklassen adaptierbaren Multiprozessorbetriebssystems"; in: *Finanzierungsantrag Multiprozessor- und Netzwerkkonfigurationen 93-95 des SFB182*, Seite 179ff

[Kar91] F. Kardel, W. Stukenbrock, T. Thiel, S. Turowski, *Anleitung zur Benutzung des MEMSY-Programmiermodells für Anwender*; interner Bericht, IMMD 4, Univ. Erlangen-Nürnberg, Oktober 1991

[Rus90] K. Rusnock, P. Raynoha, „Adapting the Unix operating system to run on a tightly coupled multiprocessor system"; *VMEbus Systems*, Oct. 1990, Vol. 6, No. 5, pp. 8-28