Das modular erweiterbare Multiprozessor System MEMSY

Prof. Dr. F. Hofmann (IMMD IV)

Prof. Dr. M. Dal Cin, Prof. Dr. G. Fritsch, W. Henning, Dr. H. Hessenauer, Dr. U. Hildebrand
Dr. C. U. Linster, W. Stukenbrock, T. Thiel, S. Turowski

Überblick

- ❏ Architekturmerkmale und Designziele
- ❏ Realisierung
  - Programmiermodell
  - Betriebssystem
  - Hardware
- ❏ Leistungsmerkmale und Skalierbarkeit
- ❏ Stand der Arbeiten und Ausblick

Architekturprinzipien

Anwendersicht:		HW-Realisierung:
❑ Nachrichten (send / receive)	(SUPRENUM)	Blocktransfer (Bus / Netz)
❑ global gemeinsame Daten	(KSR-1)	Seitenaustausch (Bus / Netz)
❑ lokal gemeinsame Daten	**MEMSY**	Speicherkopplung
• Nearest neighbour Struktur		Koppelelemente und
2-Ebenen Hierarchie		Kommunikationsspeicher
• Ferntransporte:		
Transfer KS -> Proz.-> KS		Speicherkopplung
Nachrichten		Netz (FDDI)
• Koordinierung:		
NN-Nachbarschaft		Interruptkopplung
global		optischer Sync.-Bus

MEMSY-Architektur

- ❑ Entsprechung von Programmiermodell und HW-Architektur
  - ⇨ geringe Verluste durch Kommunikation
  - ⇨ hohe Effizienz der Verarbeitung

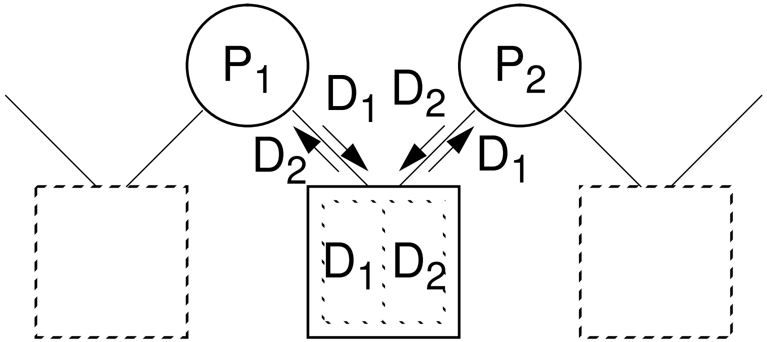
- ❑ Konstante lokale Komplexität ermöglicht beliebige Erweiterbarkeit
  - ⇨ Skalierbarkeit
  - ⇨ hohe Leistung durch massive Parallelität

- ❑ MEMSY erfüllt diese Voraussetzungen bei Anwendungen mit vorwiegend NN-Datenabhängigkeiten und hohem parallelisierbarem Rechenaufwand, typisch für viele Aufgaben der numerischen Simulation:
  - Diskretisierte partielle Differentialgleichungen (Gittermethoden)
    z.B. Aerodynamik, Plasmaphysik, Meteorologie
  - Vielteilchenmodelle (Molekulardynamik, Monte Carlo)
    z.B. Makromolekulare Chemie, Oberflächenphysik

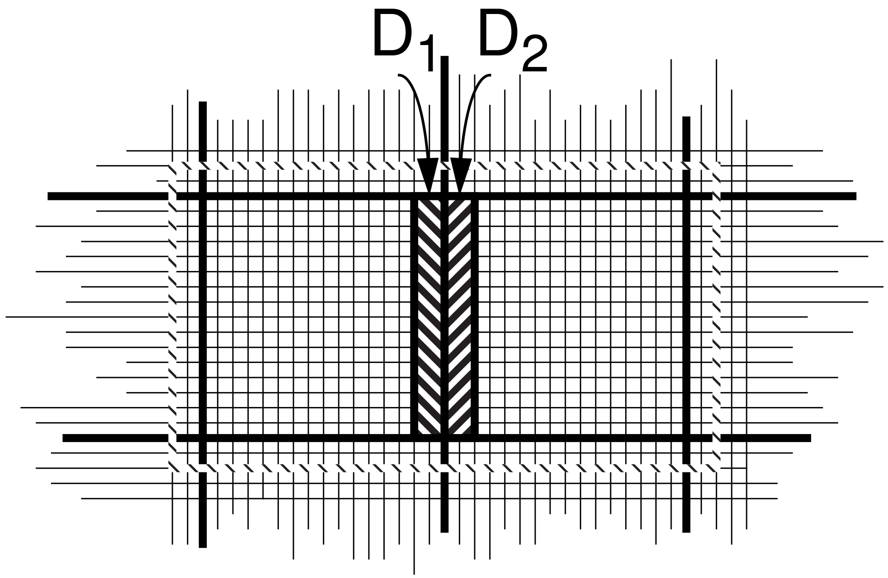
ARCHITEKTUR

Lokale Kommunikation in Distributed-Shared-Memory-Multiprozessoren (z. B. MEMSY)

Austausch von Daten (D_1 , D_2) zwischen benachbarten Prozessoren (P_1 , P_2) über gemeinsame Variable im (lokal) gemeinsamen Speicher (z.B. Randwerte von benachbarten Teilräumen numerischer Gitter)



Kommunikationsspeicher



Programmiermodell

- ❑ Programmiersprachen C und C++
- ❑ explizite Verwaltung des Kommunikationsspeichers durch die Anwendungen
- ❑ Adressierung der Knoten relativ über Himmelsrichtungen oder absolut
- ❑ Angebot an verschiedenen Kommunikations- und Koordinierungsmechanismen
  - Shared Memory
  - Messages
  - Signale
  - Spinlocks
  - Semaphore
  - Transport
- ❑ Weitere Mechanismen
- ❑ Simulator

Programmiermodell

- Shared Memory
  - dynamisches Erzeugen (create, attach) und Vernichten (detach, destroy) von Segmenten
  - Zugriff über Zeiger (C!)
  - expliziter Cache-Einsatz möglich
- Messages (auch zuverlässige)
  - kurze (2 Worte) Nachrichten (send, receive)
  - blockierend oder nicht blockierend
  - Absender kann festgestellt werden
  - Einsatz vorwiegend für Koordinierungszwecke
- Signale
  - Unterbrechungen zur Behandlung von Ausnahmesituationen

Programmiermodell

❑ Spinlocks
- Koordinierungsvariable im Kommunikationsspeicher
- implementiert durch spezielle Prozessorinstruktion (XMEM)
- timeout - Mechanismus
- “busy-wait” -> nur für kurze kritische Abschnitte geeignet

❑ Semaphore
- globale Koordinierungsvariable
- implementiert durch das Betriebssystem
- kein “busy-wait”

❑ Transport
- effizienter Transport von größeren Datenmengen
- implementiert durch das Betriebssystem
- dynamische Auswahl der günstigsten Alternative

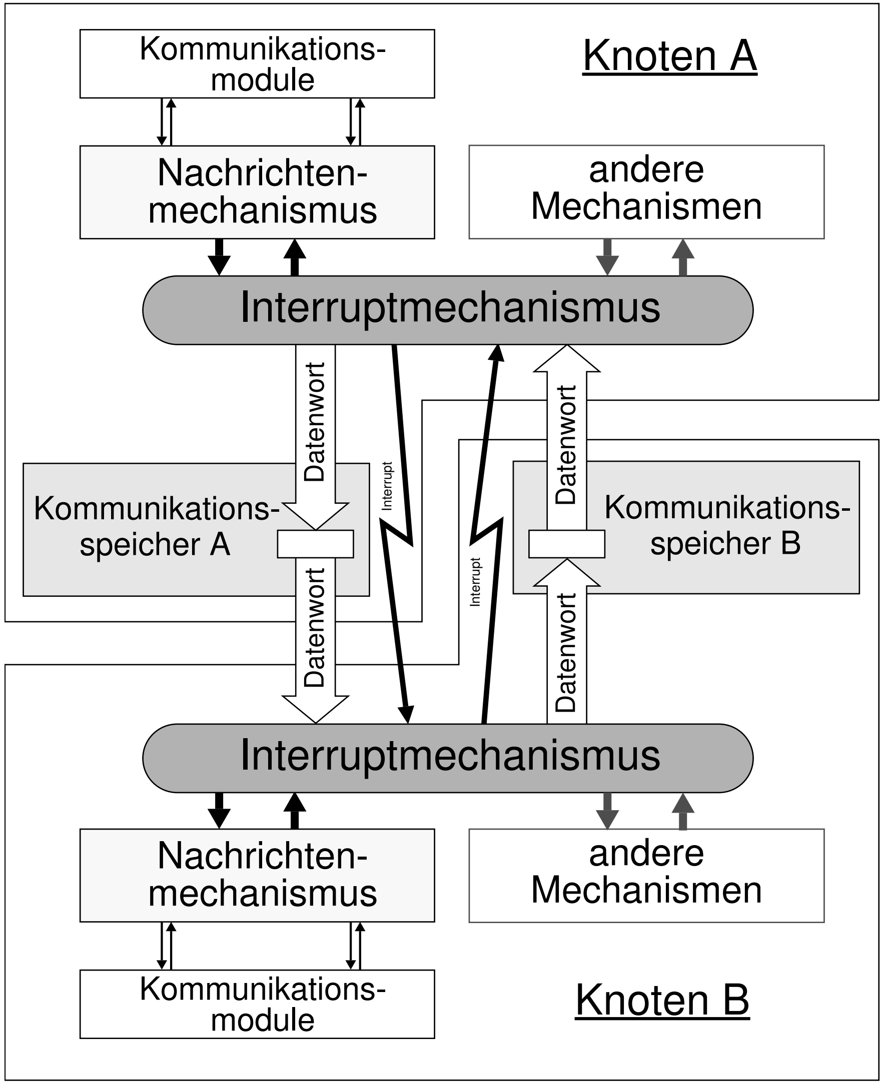
Programmiermodell

- ❑ Informationsmechanismen
  - Prozessoridentifikation und Position im Feld
  - Systemzustand
- ❑ I/O
  - normale UNIX-I/O (fopen, fread, fwrite, ...)
  - knotenlokale Dateisysteme
  - globale Dateisysteme
- ❑ fork (knotenlokal)
- ❑ Simulator
  - Testen der Anwendungen auf einer Sequent S2000/700 mit 16 Prozessoren

Betriebssystem Architektur

- ❑ Standard UNIX System V3.2 von MOTOROLA (multiprozessorfähig)
- ❑ Erlanger Erweiterungen:
  - Verwaltung des Kommunikationsspeichers
  - Interrupt- und Nachrichtenmechanismus
  - Unterstützung der Funktionen des Programmiermodells
  - Anwendungsunterstützung (mehrere Anwendungen gleichzeitig!)
  - Fehlererkennungs- und Fehlerbehebungsmaßnahmen
  - erweitertes Scheduling

Betriebssystem Architektur



- ❏ Interruptmechanismus
  - selbständiger Verbindungsaufbau
  - Verbindungsüberwachung
  - Pufferung (zur Entkopplung)
  - Zustellung der Datenworte
  - Statusmeldungen
  - statistische Information
- ❏ Nachrichtenmechanismus
  - Verbindungsüberwachung
  - Pufferung und Verteilung von empfangenen Nachrichten
  - Wahrung der Ressource-Konsistenz
  - Statusmeldungen

Betriebssystem Architektur

❑ Anwendungsunterstützung durch Applikationskonzept

- Alle Prozesse einer Anwendung werden zu einer Applikation zusammengefaßt
- Prozesse einer Applikation werden durch Knoten-ID, Applikationsnummer und Jobnummer identifiziert
- Applikationen können knotenübergreifend sein
- Ein Applikations-Daemon pro Knoten überwacht und regelt das Erzeugen und Zerstören von Applikationen
- Mehrere voneinander unabhängige Applikationen sind auf dem Gesamtsystem gleichzeitig möglich

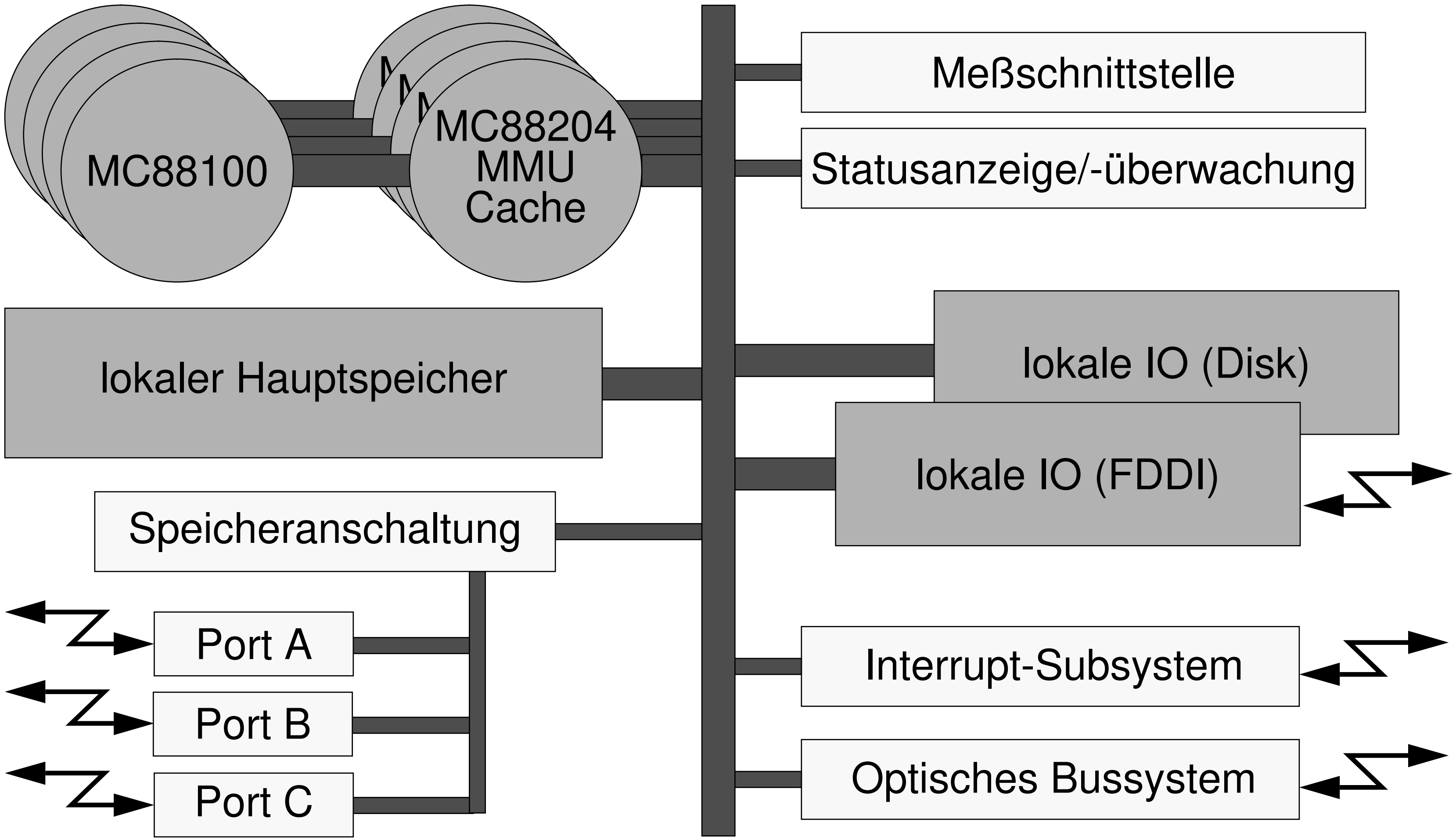
Betriebssystem Architektur

❑ Erweiterung des SYSTEM V Schedulingmechanismus durch eine zusätzliche *Applikationswarteschlange*

- Applikationsprozesse werden ausschließlich in der Applikations-warteschlange eingetragen
- Alle sonstigen UNIX-Prozesse werden in der ”Standard”-Systemwarte-schlange eingetragen
- Prozesse mit Systempriorität werden vor allen anderen Prozessen bedient
- Lauffähige Prozesse der Applikation mit der höchsten Applikationspriorität werden bevorzugt

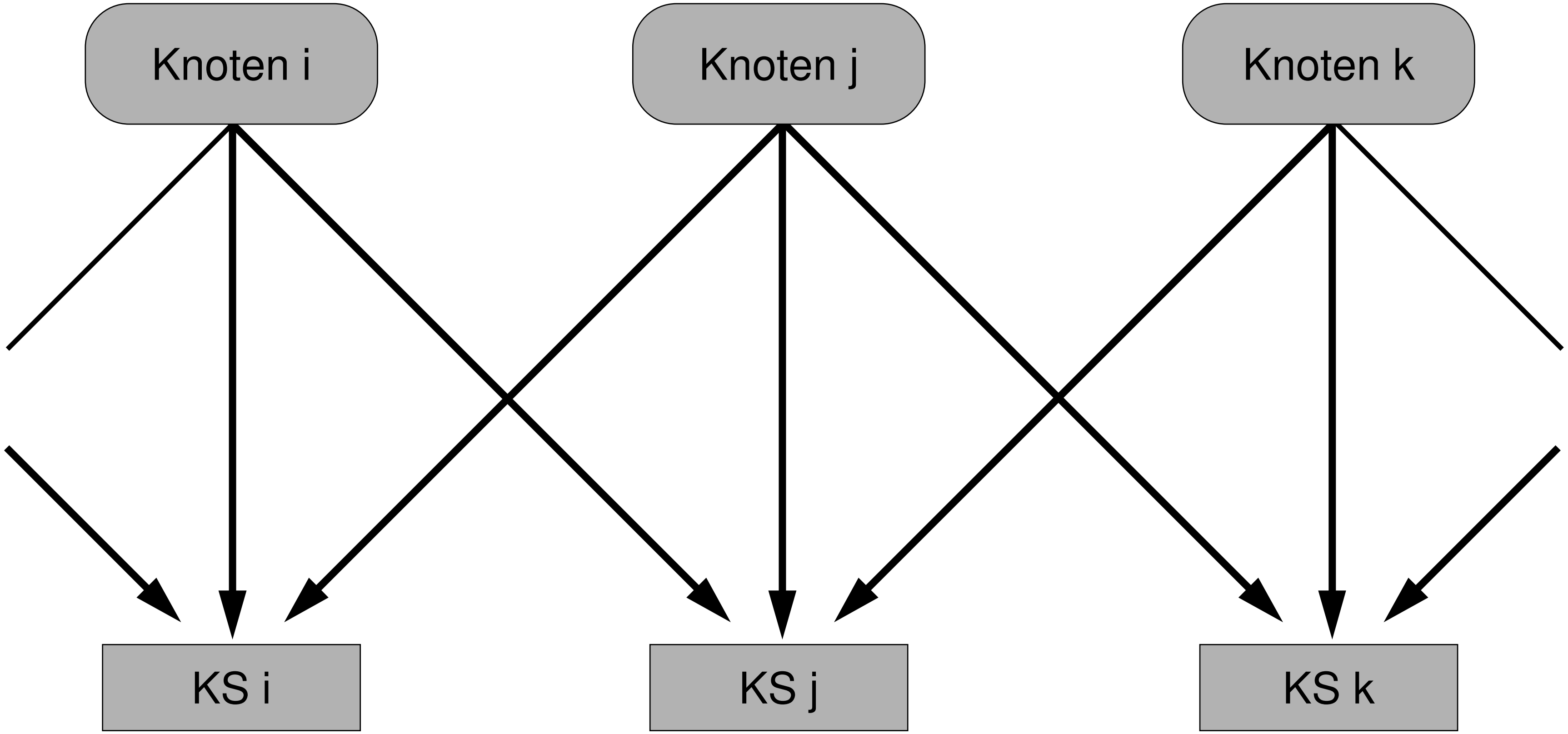
MEMSY - Hardware Architektur

❑ Aufbau eines MEMSY-Knotens



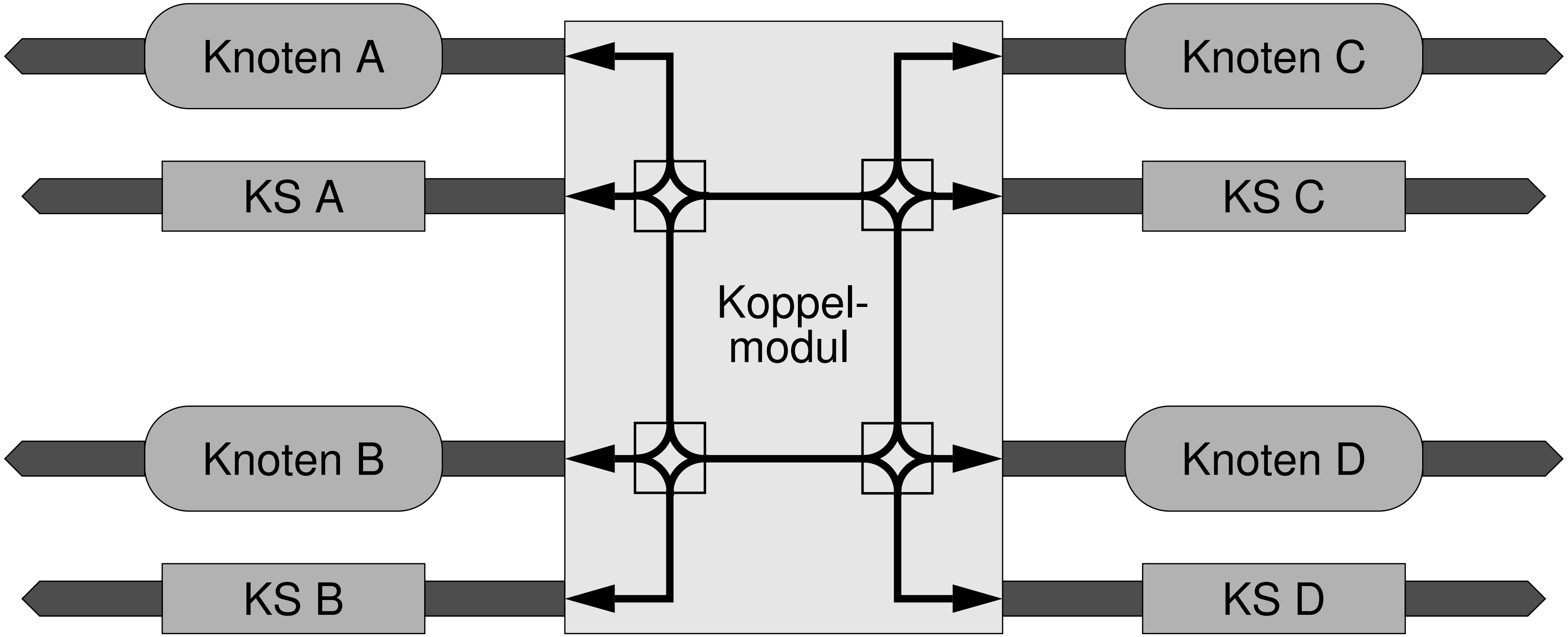
MEMSY - Hardware Architektur

❏ Prinzip der Speicherkopplung (eindimensionaler Ausschnitt)

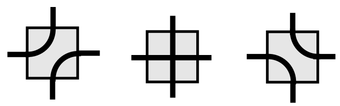


MEMSY - Hardware Architektur

❑ Speicherkopplung innerhalb der Ebene

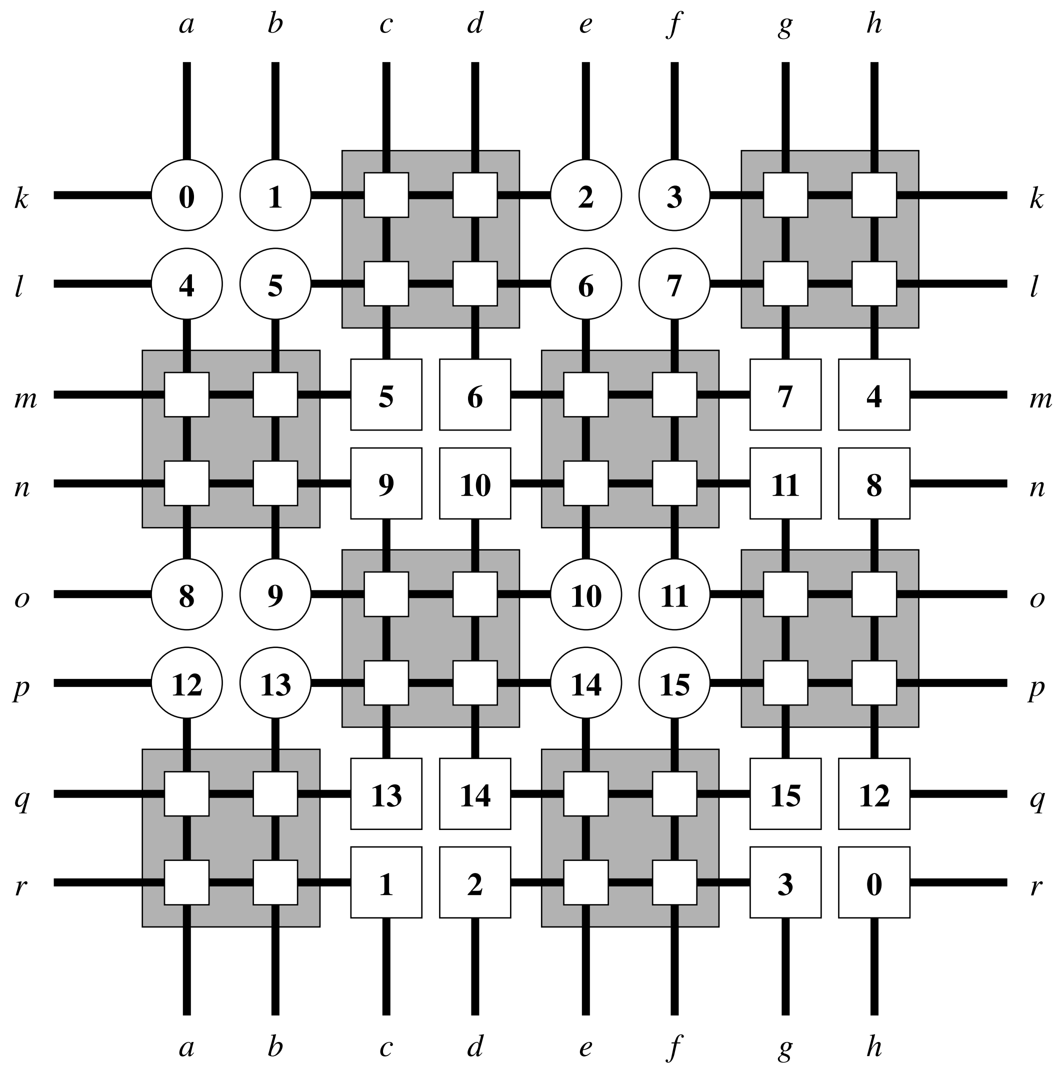


Mögliche Schaltzustände



MEMSY -Hardware Architektur

❑ Einsatz der Koppelelemente in einer Ebene mit 16 Prozessoren



MEMSY - Leistungsmerkmale

- Prozessor MC88100/MC88204
  - 25 MIPS (peak) - ca. 18 MIPS max. erreichbar (handoptimiert)
  - 12,5 MFLOPS (peak) - ca. 10 MFLOPS max. erreichbar (handoptimiert)
  - je 64 kByte Cache für Daten und Code (Konsistenz-Protokoll in HW)
- Leistungsdaten je Knoten
  - 4 Prozessoren MC88100
  - 2 Cache/Memory Management Units pro Prozessor
  - 32 MByte lokaler Hauptspeicher (max. 192 MByte)
  - 4 MByte Kommunikationsspeicher (max. 16 MByte)
  - 512 MByte Festplatte (IO-Bandbreite 5 MByte/s - max. 15 MByte/s)
  - FDDI-Interface mit 100 MBit/s Bandbreite
  - optisches Experimental-Bussystem (z.Zt. voraussichtlich 5 MBit/s)
- Experimentiersystem
  - 16 + 4 Knoten

Skalierbarkeit der MEMSY-Architektur

❑ Beispielhafte Anforderungen von Gitteralgorithmen:

- 1000×1000 Gitterpunkte mit je 10 Variablen
  ⇨ 160 MByte Speicherbedarf für Variable
- 300 Zeitschritte mit je 100 Iterationen
- 20 arithmetische Operationen pro Variable und Iteration
  ⇨ 10^{13} Floatingpoint-Operationen erforderlich

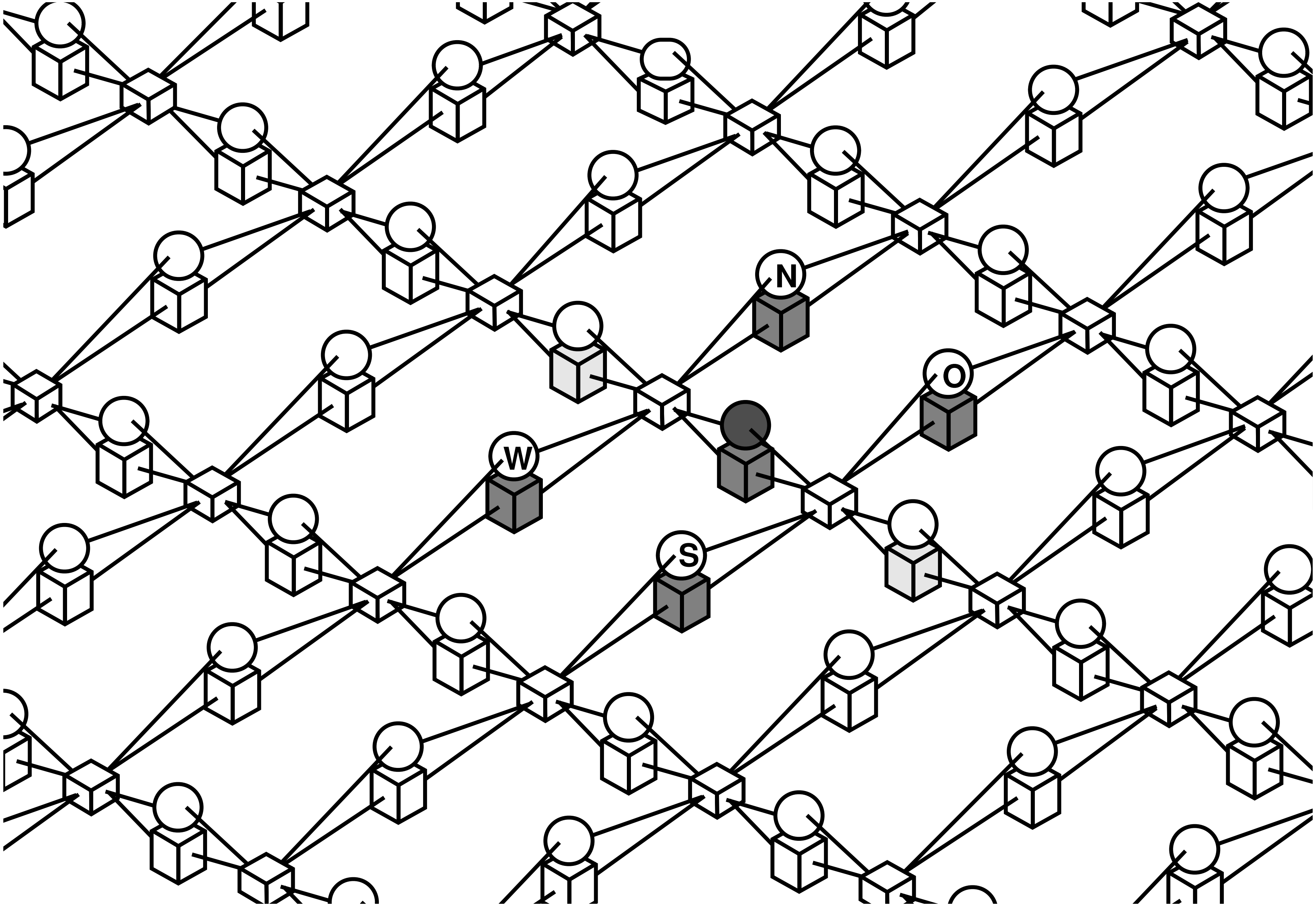
❑ Parallelisierung auf 16×16 Feld (Datenpartitionierung) ergibt je Knoten:

- 4×10^{10} arithmetische Operationen
  1.5×10^{8} Zugriffe zu Kommunikationsspeichern
  ⇨ Kommunikationsverluste gering

❑ Projektion der Leistungsdaten der Implementierung auf ein 16x16 Arbeitsfeld:

- 10 GFLOPS erreichbare Rechenleistung und 9 GByte Arbeitsspeicher
- erforderliche Netto-Rechenzeit: ca 1000 s

MEMSY - Skalierbarkeit



Stand der Arbeiten (Oktober 1992)

- ❑ 4er System lauffähig mit Testanwendungen
- ❑ 20er System im Aufbau - geplant lauffähig Ende 1992

Geplante Untersuchungen und Arbeiten

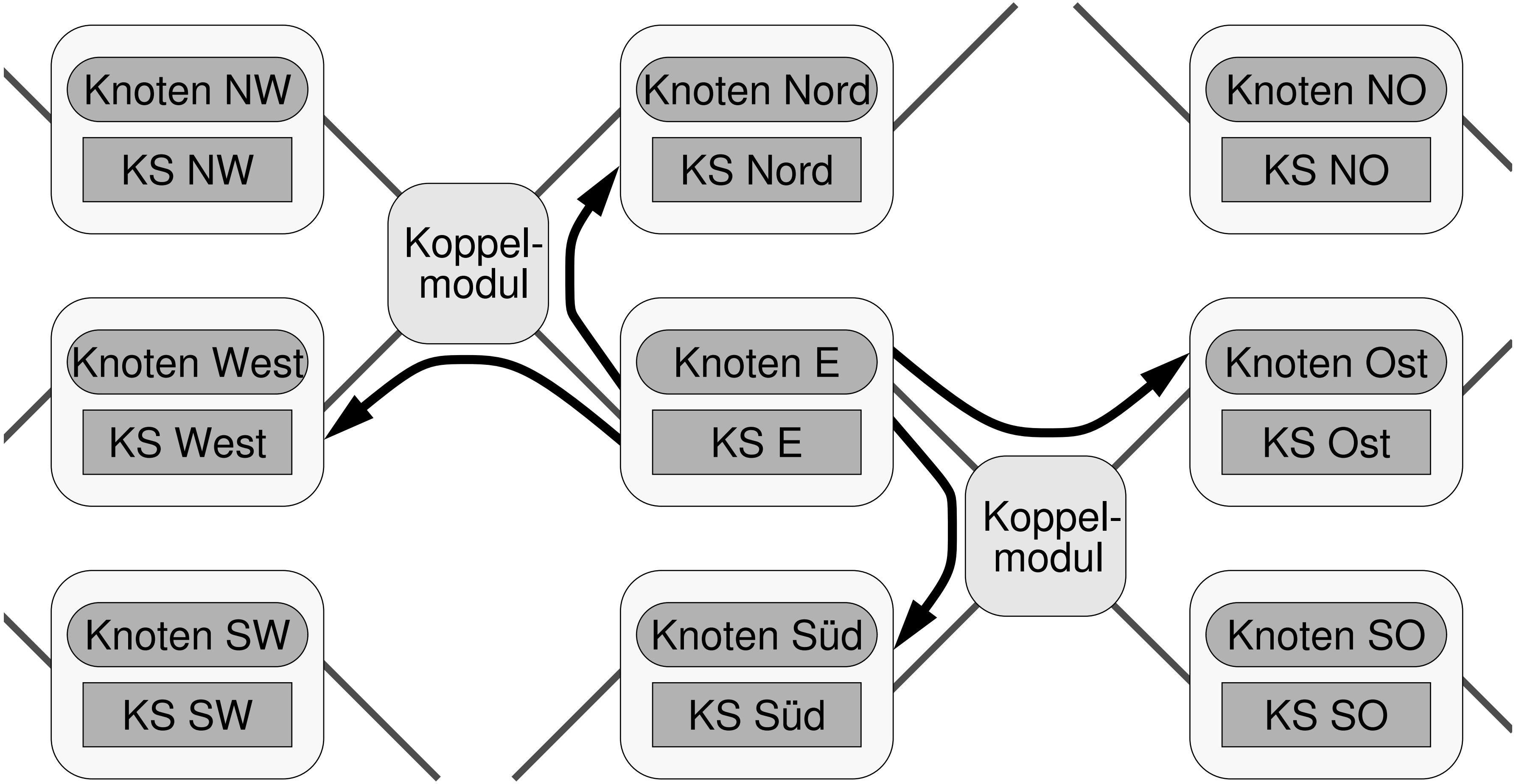
- Bewertung und Verbesserung der HW-Architektur
  Verbindungssystem, Synchronisier-HW
- Validierung des Architekturkonzepts
  Anwendungsklassen bezogene Leistungsmessung,
  Extrapolation auf große Systeme
- Verfeinerung und Ausweitung bestehender Mechanismen zur Kontrolle und Verteilung von Anwendungen
- Machine Control Layer
- Entwicklung von Fehlertoleranzmaßnahmen in Hard- und Software
  Fehlererkennung, Fehlerbehandlung
- Paralleles Raytracing in Realzeit mit hoher Auflösung und Farbtiefe

Querschnittsthema MEMSY des SFB 182

Design und Aufbau	IMMD III (Prof. Dr. M. Dal Cin) IMMD IV (Prof. Dr. F. Hofmann)
Zusätzliche HW-Komponenten	IMMD III (Prof. Dr. M. Dal Cin)
Betriebssystem, Laufzeitsystem, Simulator, Debug-Hardware	IMMD IV (Prof. Dr. F. Hofmann)
Messungen	IMMD VII (Prof. Dr. U. Herzog)
Optisches Bussystem	Lst. für Angewandte Optik (Dr. habil. J. Schwider)
Bisherige Anwendungen	IMMD III (Prof. Dr. M. Dal Cin) IMMD IV (Prof. Dr. F. Hofmann) Lst. für Strömungsmechanik (Prof. Dr. F. Durst) Lst. für Theoretische Chemie (Prof. Dr. P. Otto)

MEMSY - Hardware Architektur

❑ Realisierung der Speicherkopplung in der Ebene (Draufsicht)



MEMSY - Architektur

❏ Auswirkungen auf die Effizienz beim Einsatz von Koppelmoduln