Eigenschaften einer modernen Ein-/Ausgabe-Architektur

Characteristics of a Modern In/Out Architecture

Helge Lehmann, IBM Entwicklung GmbH, Böblingen, Wilhelm G. Spruth

Zusammenfassung Das Leistungsverhalten moderner kommerzieller Großrechner wird in ganz besonderem Maße durch die eingesetzte Ein-/Ausgabe (E/A)-Architektur mitbestimmt. Auf diesem Gebiet haben die S/390- und zSeries-Rechner eine Führungsposition mit inne, die auf einer ganzen Reihe von Einrichtungen beruht, die in anderen Rechnerarchitekturen (zum Teil noch) nicht verfügbar sind.

Hierzu gehören neben Anderem ein getrenntes E/A-Kanal-Subsystem, das parallel zur CPU arbeitet, die Möglichkeit, über unterschiedliche Pfade dynamisch auf den gleichen Plattenspeicher zuzugreifen, oder eine Prioritätssteuerung, die über einen Work Load Manager eine kontinuierliche Anpassung der E/A-Ressourcen an sich dynamisch ändernde Belastungsprofile ermöglicht.

Der vorliegende Beitrag enthält eine Übersicht über moderne E/A-Einrichtungen am Beispiel der S/390- und zSeries-Rechner und ihre Zuordnung in dem kommenden InfiniBand-Standard. ►►► **Summary** The performance characteristics of modern high end commercial systems are to a significant extent determined by their In/Out (I/O) architecture. Here zSeries and S/390 computers feature a large number of leading edge characteristics not (yet) available on other systems.

Among these are the Channel Subsystem, which executes I/O related code concurrently with the CPU, the capability to access a disk alternatively over several parallel paths, or priority controls which permit a work load manager to reassign I/O resources dynamically in response to changing work load situations.

The following paper provides an overview of modern I/O characteristics which are implemented on S/390 and zSeries systems and their relationship to the forthcoming InfiniBand standard.

KEYWORDS Channel Subsystem, I/O Architecture, Priority Queuing, S/390, System Area Network, zSeries

1 Einführung

Das Leistungsverhalten moderner Großrechner wird im Wesentlichen durch drei Faktoren bestimmt: CPU-Geschwindigkeit, Hauptspeichergröße und E/A-Leistung.

Für eine Reihe von Anwendungen ist (fast) nur die CPU-Leistung entscheidend. Darunter fallen beispielsweise die Wettervorhersage und Klimamodelle. Viele Anwendungen verlangen einen ausreichend großen Hauptspeicher, um Leistungseinbußen, z. B. durch eine exponentiell ansteigende Seitenwechselrate, zu vermeiden. Die Anzahl der gleichzeitig aktiven Prozesse wird ebenfalls durch die Größe des Hauptspeichers vorgegeben.

Für die allermeisten Transaktionsverarbeitungs- und Datenbank-Anwendungen ist die Güte des E/A-Subsystems der leistungsbestimmende Faktor. Eine deutliche Mehrheit der heute im Einsatz befindlichen Großrechner bearbeitet Aufgaben dieser Art. Es wird allgemein anerkannt, dass die Rechner der S/390- und zSeries-Architektur an dieser Stelle über eine führende Technologie verfügen. Weniger bekannt sind jedoch die Elemente, die eine moderne E/A-Technologie ausmachen.

Dies ist ein Themenkreis, der in modernen Lehrbüchern häufig vernachlässigt wird. Beispielsweise widmet das Standardwerk über Computer-Architektur von Hennesy und Patterson von 1100 Seiten lediglich etwa 50 Seiten der Ein- und Ausgabe [5]. Dies steht im Gegensatz zur Struktur von Großrechnern, in denen die Kosten für die E/A typischerweise um den Faktor 10 höher liegen als für den Prozessorkomplex [10].

Die vorliegende Veröffentlichung erläutert moderne E/A-Einrichtungen am Beispiel der S/390- und zSeries-Architektur.

2 S/390- und zSeries-Architektur

Die S/390- und heutige zSeries-Architektur ist historisch-technologisch in Jahrzehnten gewachsen. Die 1964 von Amdahl, Blaauw und Brooks [2] eingeführte S/360-Architektur stellte seinerzeit einen Meilenstein in der Entwicklung des Computers dar [14; 15]. Zum damaligen Zeitpunkt waren die Unterschiede in den Rechnerarchitekturen der einzelnen Hersteller sehr viel größer, als dies heute der Fall ist. Viele Eigenschaften, die heute als selbstverständlich gelten, entstanden erst mit der Einführung der S/360-Architektur.

Diese historischen Wurzeln führen zu der weit verbreiteten Meinung, dass die S/390- bzw. zSeries-Hard- und Software-Technologie veraltet sei und über kurz oder lang aussterben würde. In Wirklichkeit wurde die Architektur (und Technologie) kontinuierlich weiter entwickelt und verbessert, wobei gleichzeitig eine sehr gute Rückwärtskompatibilität für existierende Anwendungen erreicht wurde.

Die ausgezeichnete Marktposition der S/390- und zSeries-Rechner im Marktsegment der großen kommerziellen Server ist vor allem auf Hardware- und Software-Technologieeigenschaften zurückzuführen, über die andere Rechner (zum Teil noch) nicht verfügen. Beispiele hierfür sind die Hardwaretechnologie [4], die E/A-Architektur, die Virtualisierung mit Hilfe der *Virtual Machine* (z/VM), die *Logical Partition* unter Benutzung von *Processor Resource/System Manager* (PR/SM) [13], der z/OS *Goal Mode Work Load Manager* [16], das Clustern, die *Sysplex*-Technologie [8; 9] und die Skalierung mit Hilfe der *Coupling Facility* [12].

IBM bezeichnet seine Hardware als zSeries (früher S/390) und das am meisten eingesetzte Betriebssystem als z/OS (früher OS/390). Die zSeries-Systeme mit z/OS weisen gegenüber S/390-Systemen mit OS/390 eine zusätzliche 64-Bit-Unterstützung auf. Die wichtigste zSeries-Implementierung wird als z900 bezeichnet; eine kleinere z800 wird seit 2002 ausgeliefert. Im Rahmen dieses Aufsatzes verwenden wir die Bezeichnungen zSeries und z/OS, schließen damit aber ausdrücklich S/390 und OS/390 mit ein.

In dem vorliegenden Beitrag werden die technologisch herausragenden E/A-Eigenschaften des zSeries-Systems beschrieben. Hierzu zählen folgende Hardware-Einrichtungen:

- ein fortschrittliches *Cache-Coherence Network*, das allen CPUs eines symmetrischen Multiprozessors (SMP) einen schnellen Zugriff über den jeweils lokalen L1-Cache auf einen gemeinsam genutzten L2-Cache erlaubt, und über diesen mittels eines *Main Storage Controllers* auf den Hauptspeicher,
- eine hohe Bandbreite zum L2-Cache und zum Hauptspeicher,
- eine große Anzahl von leistungsfähigen I/O-Hubs, deren Ports zum 1st order SAN (*System Area Network*) oder zu E/A-Adaptern führen,
- eine große Anzahl von verschiedenen E/A-Adaptern, die ihrerseits den Zugang zu verschiedenen Kommunikationsnetzwerken wie LANs, WANs, oder zu 2nd order SANs (*Storage Area Networks*) wie ESCON, FICON oder Fibre Channel Fabrics ermöglichen,
- Einrichtungen zur Fehlererkennung und -korrektur auf allen Hardware- und Software-Ebenen, die damit einen Beitrag zu der erreichten Ausfallsicherheit von 99,999 % einer zSeries-Installation leisten.

Weitere Eigenschaften der zSeries-Architektur sind:

- das Offloading von E/A-Operationen weg von der (oder den) CPU(s) hin zu einem (oder mehreren) *System Assist Prozessoren* (SAP), die im Wesentlichen das zSeries *Channel Subsystem* beherbergen,
- neuartige Einrichtungen wie z. B. die vollautomatische Behandlung von Belegtzuständen an beliebiger Stelle der E/A-Topologie oder das *Channel Subsystem Priority Queuing*,
- die Weiterentwicklung von *Start Subchannel/Channel Command Word* basierenden E/A-Protokollen hin zu *Request Queue/Control Block* basierenden Eigenschaften (siehe z. B. QDIO in Absatz 4.2).
- weitgehende Einrichtungen zur Verbesserung der E/A-Konfigurationflexibilität, die sowohl statische als auch umfangreiche dynamische Änderungen ermöglichen.

3 E/A-Topologie

In Bild 1 ist die Struktur des *z900 Central Electronic Complex* und sein Anschluss an externe Netze wie Kommunikationsnetzwerke und *Storage Area Networks* dargestellt. Ein gemeinsam genutzter L2-Cache verbindet bis zu 20 Prozessoren (16 CPUs, 3 SAPs, 1 Reserveprozessor), 4 *Memory Bus Adapter* (MBA) Chips für den Anschluss von E/A-Adapterkarten und mehreren Hauptspeicherkarten. Die MBA-Chips stellen die zSeries-Implementierung eines generischen I/O-Hubs dar. Sie erfüllen eine ähnliche Aufgabe wie die *Southbridge* in einem PC, allerdings mit deutlich höherer Funktionalität. Der gemeinsam genutzte L2-Cache, die Prozessoren und die MBA-Chips befinden sich auf einem einzigen *Multi Chip Module* (MCM).

Alle Verbindungen sind Punkt zu Punkt und aufgrund der hohen Packungsdichte des MCMs sehr kurz, sodass die Busse mit bis zu 450 MHz betrieben werden können. Dies führt zu sehr guten Latenz- und Bandbreitewerten. So besitzt beispielsweise jede CPU und jeder SAP eine Bandbreite zum L2-

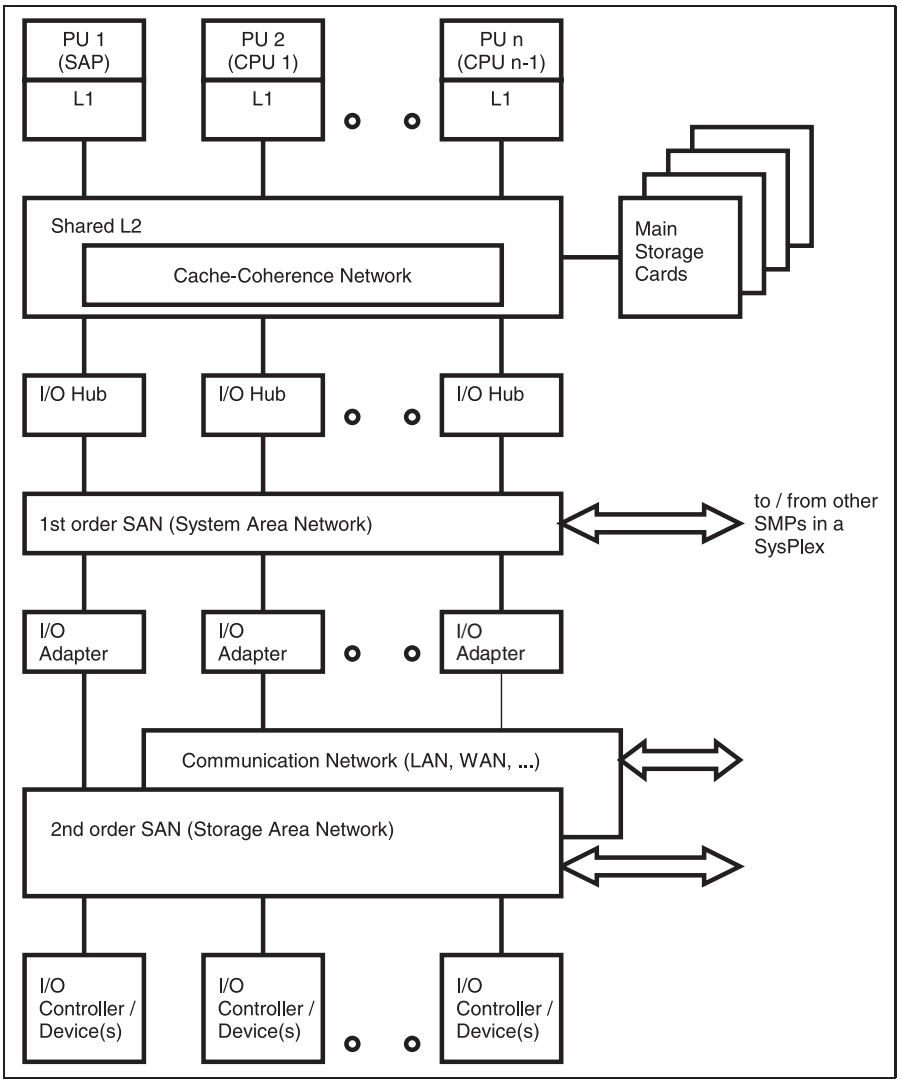


Bild 1 zSeries SMP E/A-Konfiguration.

Cache von 14,5 GByte/s mit einer Zugriffszeit von 20 ns. Der L2-Cache stellt für alle Prozessoren eine gesamte Bandbreite von 290 GByte/s zur Verfügung. Die Bandbreite des Hauptspeichers beträgt 29 GByte/s. Jeder MBA-Chip schließt 6 full-duplex byte-serielle Busse an, die mit einer 2 ns Double Data Rate Clock betrieben werden. Dies ergibt für jeden Bus eine E/A-Bandbreite von 1 GByte/s in jeder Richtung. Nach Abzug des Protokoll-Overheads beträgt die effektive Datenbandbreite in jeder Richtung 750 MByte/s. Die Bandbreite für alle I/O Hub Chips beträgt somit theoretisch 36 Gbyte/s, hiervon sind 29 GByte/s in der Praxis realisierbar. Die Busse verwenden das *Self-Timed Interface* (STI) Protokoll [7], welches eine ähnliche Rolle wie der *Peripheral Component Interconnect* (PCI) Bus in einem PC übernimmt, allerdings mit wesentlich höherer Datenrate und Funktionalität. Da der STI-Bus differenzielle Treiber und Receiver benutzt, kann die Kabellänge bis zu 15 m betragen.

Das *System Area Network* (Bild 2) besteht aus Switches, die ebenfalls mit STI-Bussen verbunden sind. Jeder Switch hat 4 STI full-duplex Bus-Ausgänge. Jeder dieser Ausgänge geht zu einer I/O-Karte, z. B. einer *Common I/O Card* oder einer *Inter System Channel* (ISC) Card. Während die STI-Busse zwischen MBAs und Switches mit 1 GB/sec betrieben werden, beträgt die Datenrate der Busse zwischen den Switches und den angeschlossenen Karten 500 MB/sec oder 333 MB/sec. Von den maximal 96 Ausgängen sind 84 nutzbar für I/O Cards (z. B. FICON, Gigabit Ethernet und andere). Diese sind in E/A-Rahmen (*I/O cages*) untergebracht. Bis zu 3 solcher Rahmen mit jeweils bis zu 28 E/A-Adapterkarten können an das System angeschlossen werden.

Solch ein System kann nun auf verschiedene Art und Weise in einen als *Parallel Sysplex* be-

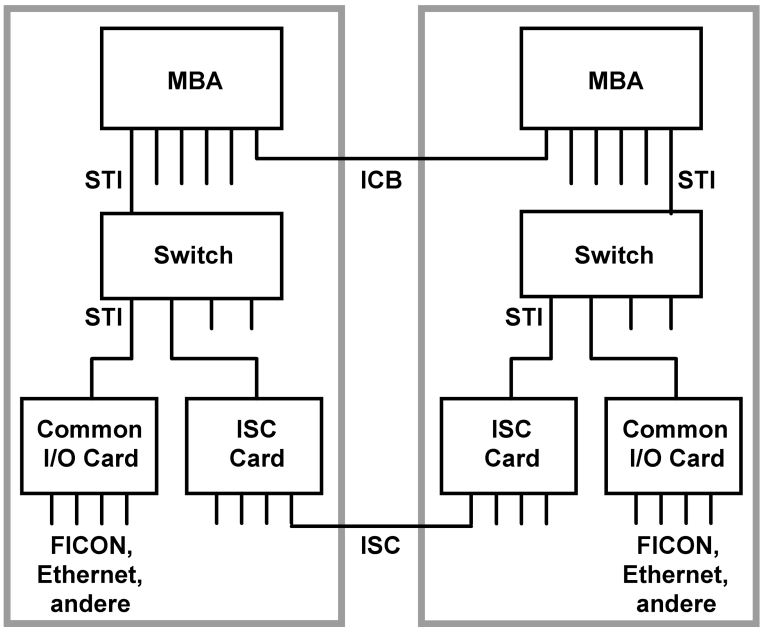


Bild 2 System Area Network.

zeichneten Rechnerverbund (Cluster) integriert werden [12]. Zum einen können mit Hilfe des *Integrated Cluster Buses* (ICB), der das Clustering-Protokoll auf der Basis eines STI-Kabels implementiert, mehrere z900-Systeme von MBA zu MBA direkt miteinander verbunden werden. Die maximale Entfernung beträgt hierbei 10 m. Zum anderen werden für räumlich verteilte Cluster (*Geographically Dispersed Parallel Sysplex* [3]) bis zu 20 km lange optische Verbindungen benutzt, wofür eine ISC-Karte benötigt wird.

4 S/390-z900 E/A-Architektur

4.1 Traditionelle Architektur

Die traditionelle E/A-Architektur wird beim S/390 durch Kanäle (*Channels*), Unterkanäle (*Subchannels*) und Kanalprogramme (*Channel Programs*) charakterisiert.

Der Kanal ist ein leistungsfähiger E/A-Adapter, der speziell in Hinblick auf die Anforderungen eines zSeries E/A-Subsystems hinsichtlich Leistungsfähigkeit, aber insbesondere auch hinsichtlich RAS (*Reliability, Availability, Serviceability*) Anforderungen entwickelt wurde. Die wichtigsten E/A-Adapter der z900- und z800-Systeme basieren auf einer *Common I/O Card*, die in Abschnitt 6 näher beschrieben wird. Ein Unterkanal ist ein logisches Abbild eines E/A-Gerätes, das über eine Steuereinheit an den Kanal angeschlossen ist.

Sowohl diese Architektur als auch die *Small Computer System Interface* (SCSI) Architektur gehen auf den 1964 entwickelten *Selectorkanal* der S/360-Architektur zurück. Der SCSI-Bus und das serielle SCSI-Interface haben eine ähnliche Funktion wie der S/390-Kanal, wenn auch mit geringerer Funktionalität.

Die Verbindung zwischen Kanal und Steuereinheit – ursprünglich basierend auf einem elektrischen Kabel mit Busstruktur (*parallel channel*) – hat sich zu einem optischen Netzwerk (*FICON Channel*) weiterentwickelt (Bild 3). Es erlaubt bei Datenübertragungsraten bis zu 2 Gbit/s Entfernungen bis zu 100 km zwischen Rechnersystem und Steuereinheit. FICON ist ein Mitglied der *Fibre Channel* Protokollfamilie. Es unterscheidet sich von anderen Mitgliedern der Familie dadurch, dass es in der obersten Schicht FC-4 ein FICON-Protokoll an Stelle eines Audio-, Video-, 802.2- oder seriellen SCSI-Protokolls definiert, das die E/A-Architektur der zSeries unterstützt. FICON ist die Voraussetzung für zahlreiche Einrichtungen, über die z. B. SCSI in der parallelen Ausführung nicht verfügt. Neben dem zSeries-spezifischen FICON-Protokoll unterstützen die neuesten zSeries-Systeme aber auch das SCSI-Protokoll über *Fibre Channel* (SCSI FCP).

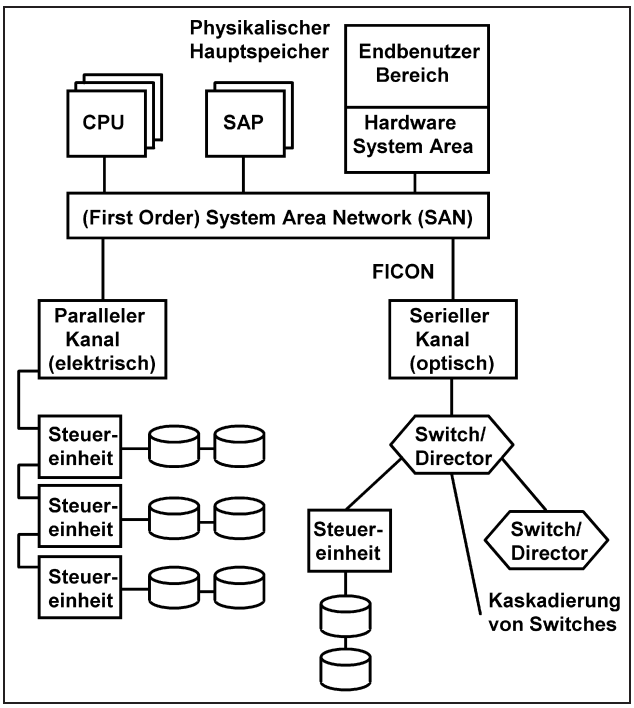


Bild 3 Parallele/serielle Kanäle bei S/390 und zSeries.

In einem *Fibre Channel Netzwerk*, basierend auf dem FICON- oder SCSI-Protokoll, können über *Fibre Channel Switches* und *Directors* eine Vielzahl von Servern und Steuereinheiten bzw. Endgeräten in einem *Storage Area Network* miteinander verknüpft werden. Als Director bezeichnen wir einen Switch mit erweiterten Funktionen, z. B. Redundanz- oder Recovery-Einrichtungen. Storage Area Netzwerke im zSeries-Umfeld verwenden in der Regel Directors an Stelle von einfachen Switches.

Eine beliebig komplexe E/A-Operation wird durch ein Kanalprogramm definiert. Dieses besteht aus einzelnen Befehlen, die als *Channel Command Words* (CCW) bezeichnet werden. Es stehen Mechanismen wie z. B. Bedingungsabfragen und bedingte Sprünge zur Verfügung. Jede E/A-Operation besteht aus den folgenden Schritten:

- Die Ausführung des Kanalprogramms wird durch einen speziellen E/A-Maschinenbefehl der CPU, den *Start Subchannel* (SSCH) Befehl, angestoßen.
- Die Ausführung des Kanalprogramms erfolgt in Kooperation zwischen dem Kanalsubsystem, das auf dem SAP läuft, dem Kanal, und einer Steuereinheit.
- Die Beendigung eines Kanalprogramms wird durch eine E/A-Unterbrechung zusammen mit entsprechender Statuspräsentation an die CPU gemeldet.
- Eine besondere Eigenschaft der zSeries-Architektur besteht darin, dass der Zugriff auf einen Kanal wie auf die daran angeschlossenen Steuereinheiten und Endgeräte gleichzeitig von mehreren Betriebssystemen er-

folgen kann, die parallel auf der oder den zSeries-CPU(s) ausgeführt werden. Kanalsubsystem und Kanal stellen dabei sicher, dass sich diese E/A-Operationen nicht gegenseitig stören.

4.2 Queued Direct I/O

Während die eigentliche Ausführung eines Kanalprogramms durch einen SAP sowie den Kanal erfolgt und damit parallel zur Bearbeitung von zSeries-Programmen durch die CPU abgewickelt wird, belastet sowohl das Starten eines Kanalprogramms als auch dessen Beendigung die CPU (letztere wird durch eine Unterbrechung signalisiert). Hier bringen die *Queued Direct I/O* (QDIO) Protokolle der zSeries, die von der *Common I/O Card* unterstützt werden, eine deutliche Verbesserung der Leistungsfähigkeit eines Systems, da sie sowohl die CPU entlasten, als auch die zum Starten bzw. für die Beendigung einer E/A-Operation benötigte Zeit reduzieren. Damit werden sowohl die Zugriffszeiten als auch der Durchsatz, gemessen in E/A-Operationen pro Sekunde, verbessert.

Die QDIO-Protokolle der zSeries basieren auf (theoretisch) endlos laufenden Kanalprogrammen (Bild 4). Nachdem eine E/A-Operation mit einem SSCH-Maschinenbefehl gestartet wurde, werden nachfolgende E/A-Operationen vom CPU-Programm (in der Regel auf Betriebssystemebene) in eine Output oder Request Queue gepackt, die der Kanal (implementiert auf Basis der Common I/O Card) ausliest und bearbeitet. Solange diese Queue nicht leer wird, braucht der Kanal hierfür keinen weiteren Anstoß. Lediglich wenn das CPU-Programm einen Request in eine leere Queue ablegt, muss der Prozessor des Kanals aufgeweckt werden, um die Bearbeitung der Queue weiterzuführen. Umgekehrt legt der Kanal eingehende Nachrichten oder Fertigmeldungen in eine Input oder Completion Queue, die vom CPU-Programm sequenziell ausgelesen wird. Auch hier ist ein Unterbrechungssignal nur dann erforderlich, wenn eine solche Nachricht bzw. Fertigmeldung in eine leere Queue abgelegt wird.

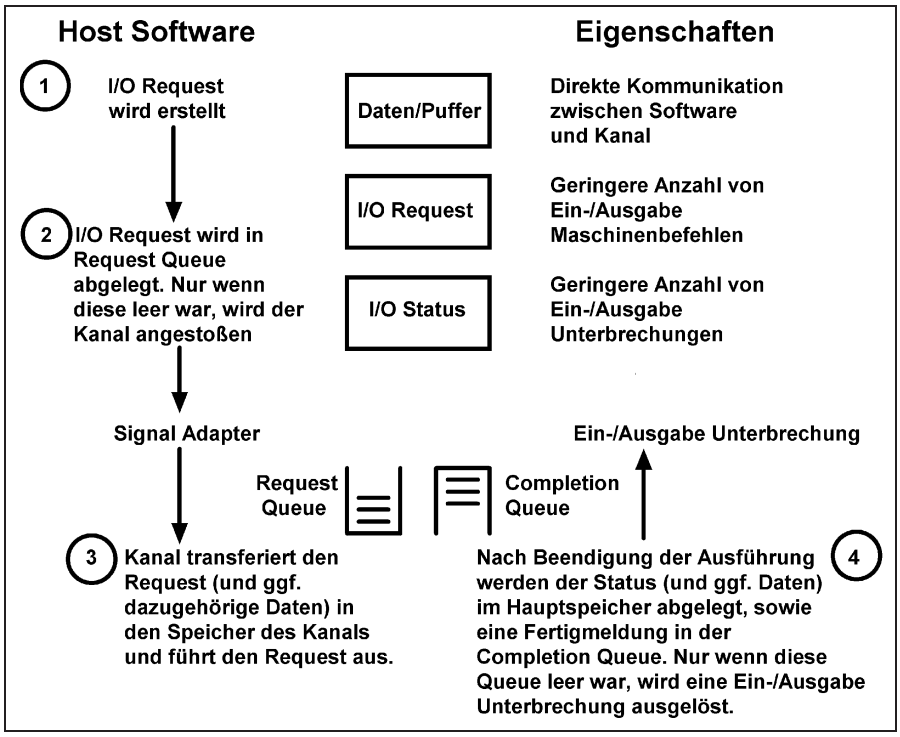


Bild 4 QDIO Channel Schnittstelle (z. B. Fibre Channel).

4.3 Andere Entwicklungen

Das zSeries *Fibre Channel Protocol* (FCP) *for SCSI* ermöglicht den Zugriff auf Fibre Channel SCSI-Plattenspeicher. Einzelheiten sind in [1] beschrieben.

Manche Rechnerarchitekturen setzen ebenfalls auf Queuing-Mechanismen basierende E/A-Protokolle ein. Allerdings bieten diese in der Regel nicht die Schutzmechanismen, die die RAS-Anforderungen der zSeries zwingend vorschreiben. Sie erlauben auch keine unabhängige und geschützte Kommunikation mit verschiedenen Betriebsystemen, die auf derselben Hardware-Plattform laufen. Beides wird bei den QDIO-Protokollen der zSeries garantiert. Insbesondere kommt hier die Firewall-Funktion, die in der STI-PCI-Bridge implementiert ist, zum Tragen.

Eine wichtige Neuentwicklung auf dem Gebiet der E/A-Protokolle stellt der Standard *InfiniBand* (IB) dar, eine gemeinschaftliche Entwicklung der Firmen Compaq, Dell, Hewlett-Packard, IBM, Intel, Microsoft und Sun Microsystems. Dieser IB-Standard ist als eine leistungsfähige E/A-Architektur für zukünftige Rechnersysteme geplant. Insbesondere sind für Standard-E/A-Operationen keine Interaktion mit dem I/O-Supervisor des Betriebssystems mehr erforderlich.

InfiniBand benutzt ein ein- oder mehradriges optisches oder elektrisches Kommunikationsnetz, das in der optischen Ausführung maximale Entfernungen und Datenraten (pro Ader) in der gleichen Größenordnung wie FICON zulässt. Das *InfiniBand*-Protokoll verbindet queuing-basierende E/A-Mechanismen, definiert – basierend auf einem Firewall-Konzept – einen geschützten Datentransfer zwischen dem Hauptspeicher des Rechners und der E/A-Welt, und unterstützt mehrere unabhängige, parallel laufende Betriebssysteme auf einem Rechnersystem. Es greift damit einige der wichtigsten Eigenschaften der heutigen zSeries E/A-Systeme auf.

Die zSeries-Architektur einschließlich der E/A-Architektur ist in einem offiziellen Architektur-Handbuch beschrieben [11].

5 Channel Subsystem

5.1 Definition

Der größte Unterschied zwischen der heutigen z/Architektur und anderen Rechner-Architekturen besteht im E/A-Subsystem. Sein zentraler Bestandteil ist das *Channel Subsystem* (CSS), das ursprünglich 1977 eingeführt und seitdem kontinuierlich verbessert wurde. Während vorher eine starke Verflechtung von Betriebssystemkomponenten und der I/O-Welt existierte, wurde mit der Einführung des CSS der Versuch unternommen, das E/A-Subsystem konsequent abzukapseln, von den Betriebssystemkomponenten zu entkoppeln und es mit einer definierten Schnittstelle zu versehen. In heutiger Terminologie könnte man hier von einem objektorientierten Ansatz sprechen, der den Rechnerarchitekten mit der Definition des CSS bereits vor 25 Jahren gelungen ist. Unter CSS verstehen wir dabei die Summe aller physikalischen und logischen Ressourcen und Einrichtungen, die notwendig sind, um den Informationsaustausch zwischen dem Hauptspeicher und den angeschlossenen E/A-Geräten zu veranlassen und zu kontrollieren. Bei Systemen der zSeries gehören zu den physikalischen Ressourcen:

- dedizierte Prozessoren (dies sind die erwähnten SAPs),
- ein gesonderter, dem Betriebssystem und den Anwendungen nicht zugänglicher Speicherbereich (dies ist die *Hardware System Area*, HSA),
- die I/O-Hubs und
- die E/A-Adapter.

Die logischen Ressourcen bestehen im Wesentlichen aus

- dem Microcode, der bei der zSeries-Architektur *als Licenced Internal Code* (LIC) bezeichnet wird [13] und der zum Teil auf den CPUs, vor allem aber auf den SAPs und in den Prozessoren der E/A-Adapter ausgeführt wird, sowie
- den zur Durchführung und Kontrolle der E/A-Operationen notwendigen Datenstrukturen. Diese befinden sich zum größten Teil in der HSA, aber auch im lokalen Speicher der E/A-Adapter.

Das CSS verwaltet bis zu 256 Kanäle und bis zu 65 536 angeschlossene E/A-Geräte für bis zu 15 Betriebssysteme, die gleichzeitig auf einem zSeries-Rechner laufen können. Damit werden diese Betriebssysteme von immer wiederkehrenden Arbeiten entlastet (*offloading*) und können überlappend zu der laufenden E/A-Operation andere Maschinenbefehle ausführen. Die Architektur unterstützt bis zu 8 parallel zueinander verlaufende Datenpfade zu einem E/A-Gerät (*multipathing*). Ein solcher Datenpfad kann im Allgemeinen Kanal, Switch, Steuereinheit und E/A-Gerät als Zwischenstationen beinhalten.

Das CSS entscheidet, welcher Pfad genommen wird (*path selection*). Insbesondere werden nur funktionsfähige Pfade ausgewählt und möglichst solche, die frei, d. h. nicht gerade mit anderen Aufgaben betraut sind. Tritt trotzdem der Fall ein, dass der ausgewählte Kanal, der Switch, die Steuereinheit oder das Gerät den anstehenden Auftrag nicht sofort erledigen können (*busy condition*), so übernimmt das CSS die Aufgabe, diese E/A-Operation erneut zu starten (*busy handling*). Da das CSS sehr kurze Kommunikationspfade zu den Kanälen besitzt, ist diese zentrale Verwaltung der E/A-Ressourcen sehr effektiv.

5.2 Behandlung von Belegt-Zuständen

In der oben gezeigten E/A-Topologie mit CSS, Kanalpfaden, Switches, Steuereinheiten und E/A-Geräten kann es beim Starten einer E/A-Operation an allen Stellen zu meist temporären Belegt-Zuständen kommen:

- Die erste Hürde ist das Finden eines freien Kanalpfades aus einer Auswahl von bis zu 8 Pfaden zu dem angesprochenen E/A-Gerät. Sollten alle hierfür vorgesehenen Pfade belegt sein, wird der Auftrag im CSS gehalten und nach kurzer Zeit erneut versucht zu starten.
- Die zweite Hürde stellen die Switches auf dem Weg zur Steuereinheit dar, die ein „Belegt“ auf dem für diese Verbindung vorgesehenen abgehenden Pfaden entdecken können. Auch in diesem Fall wird der Auftrag im CSS gehalten und nach kurzer Zeit erneut versucht zu starten.
- Die dritte Hürde stellt die Steuereinheit dar, die u. U. wegen Überlastung diesen Auftrag zu diesem Zeitpunkt nicht annehmen kann. Auch in diesem Fall wird der Auftrag im CSS gehalten und sofort wieder versucht zu starten, sobald die Steuereinheit ein „no longer busy“ signalisiert.
- Die vierte Hürde stellt das Gerät selbst dar, das zufälligerweise zu diesem Zeitpunkt eine E/A-Operation im Auftrag eines anderen Systems ausführen könnte. Auch in diesem Fall wird der Auftrag im CSS gehalten und sofort wieder versucht zu starten, sobald das Gerät ein „no longer busy“ signalisiert.

Alle diese Hürden sollten bei einer sorgfältig ausgelegten E/A-Topologie eher selten auftreten. Trotzdem ist mit dem hier aufgezeigten Mechanismus ein Konzept vorhanden, das vollautomatisch und ohne Einbuße an CPU-Leistung alle potenziellen Belegt-Zustände behandelt, ohne dass sich das Betriebssystem oder die Anwendungen darum kümmern müssen.

5.3 Asynchronität der E/A-Maschinenbefehle

Eine wesentliche Fähigkeit des CSS ist die Entlastung des Betriebssystems durch überlappende Ausführung von E/A-Maschinenbefehlen mit anderen, nicht auf die Ein-/Ausgabe bezogenen Maschinenbefehlen. Hierzu werden die E/A-Maschinenbefehle asynchron ausgeführt. Das bedeutet, dass nach einer kurzen formalen Prüfung des E/A-Auftrages der Maschinenbefehl aus Sicht des Betriebssystems abgeschlossen und zu Ende ist (der *Condition Code* wird gesetzt), der Auftrag inhaltlich aber erst danach bearbeitet wird. Die Effektivität dieses Verfahrens wird noch dadurch unterstützt, dass diese asynchrone Abarbeitung des E/A-Auftrages auf dem SAP durchgeführt wird.

5.4 Channel Subsystem Priority Queuing

Neben dieser soeben beschriebenen Entlastung der Betriebssysteme durch das asynchrone Ausführen von E/A-Operationen auf einem eigenen Prozessor leistet das CSS weitere eigene Beiträge zur Steigerung des Durchsatzes im E/A-Bereich. Einer dieser Beiträge läuft unter dem Stichwort *Priority Queuing*.

In einem hoch ausgelasteten System mit vielen Prozessoren laufen in der Regel viele verschiedene Anwendungen mit stark unterschiedlichen Anforderungen an das Antwortzeitverhalten der von ihnen angestoßenen E/A-Operationen. Ein extremer Fall sind Online-Datenbank-Abfragen mit nur wenigen E/A-Operationen und einer *sub-second response time*-Anforderung. Ein anderer extremer Fall sind Batch-Programme, die im Hintergrund laufen.

Mit dem *Channel Subsystem Priority Queuing* (CSSPQ) wird in Verbindung mit dem z/OS *Work Load Manager* und dem *Dynamic Channel Path Management* jeder E/A-Operation eine individuelle Priorität (von 0 bis 15) mitgegeben. Dieser Prioritätswert wird vom CSS beim Starten von E/A-Operationen beachtet, wobei die Operationen mit einem höheren Prioritätswert Vorrang haben. Der Algorithmus ist so ausgelegt, dass auch E/A-Operationen mit niedrigen Prioritäten nicht unendlich lange warten müssen. Die im Einzelfall in einem System verwendeten Prioritätswerte werden durch den System-Programmierer festgelegt. Die Verbesserung des Antwortzeitverhaltens von E/A-Operationen mit hoher Priorität ist deutlich messbar, und kann z. B. für die Erfüllung von *Service Level Agreements* wichtig sein.

6 Datensicherheit und Zuverlässigkeit

Die meisten Marktforschungsunternehmen attestieren z900 und z/OS eine sehr gute Zuverlässigkeit und Verfügbarkeit im Vergleich zu anderen Rechnern. Eine Verfügbarkeit von 99,999 % wird an dieser Stelle häufig genannt. Dies bedeutet nicht mehr als 5 Minuten Ausfallzeiten während eines Jahres für geplante und ungeplante Unterbrechungen des Rechnerbetriebs. Für viele Unternehmen sind derartig gute Verfügbarkeitswerte wichtig.

Eine große Anzahl unterschiedlicher Hardware- und Softwareeigenschaften trägt zu der guten Verfügbarkeit bei. Auch das E/A-Subsystem leistet hierzu einen Beitrag. Zwei besonders wichtige Maßnahmen bestehen darin, zu verhindern, dass falsche oder fehlerhafte Daten vom/zum Hauptspeicher transferiert werden, und im Falle eines Adressierungsfehlers *Direct Memory Access* (DMA) Daten an die falsche Adresse im Hauptspeicher geschrieben werden.

6.1 E/A in PCs und Workstations

Die Welt der E/A-Adapter wird heute von PCI-Adapterkarten dominiert, unabhängig von der Art der Ein-/Ausgabe (wie z. B. Speicherzugriff, lokale oder nichtlokale Kommunikation) und des verwendeten Übertragungsprotokolls.

Bei PCs und Workstations wird die Operation der PCI-Adapter vom Betriebssystem oder einer Applikation initiiert, die auf dem zentralen Prozessor des Rechners ausgeführt wird. Dazu greift das Programm über *Memory Mapped I/O* (MMIO) direkt auf Register und Datenbereiche in dem Adapter zu, unter Verwendung von Adressen im Adressbereich des PCI-Busses. Die zu übertragenden Daten werden dann meist vom PCI-Adapter per DMA vom Hauptspeicher des Rechners zu einem Puffer im PCI-Adapter bzw. umgekehrt übertragen.

Um an einem größeren Server mehrere PCI-Busse zu betreiben und zu diesem Zwecke auch etwas größere Entfernungen zwischen dem oder den Prozessoren und den PCI-Bussen überbrücken zu können, implementieren viele Server eine proprietäre Verbindung zwischen Prozessor und PCI-Bussen. Häufig wird auch hier die Operation des PCI-Adapters direkt durch MMIO vom Prozessor aus gesteuert (auch als *load/store I/O mode* bezeichnet). Dieser Zugriff wird lediglich über die proprietäre Verbindung durchgeschleust, transparent sowohl für das ausführende Programm als auch für den PCI-Adapter. Entsprechendes gilt auch für den DMA-getriebenen Datentransfer.

Der PCI-Bus in seiner ursprünglichen Form hat nur sehr limitierte Schutzmechanismen, um die Integrität der übertragenen Speicheradressen und Daten zu garantieren. Dieses Problem wurde erst mit späteren Weiterentwicklungen, wie dem PCI-X-Bus und dem PCI-Express verbessert, an dessen Spezifikation derzeit gearbeitet wird.

Weiterhin sitzen auf den meisten PCI-Adaptern selbst Prozessoren und andere Komponenten unterschiedlichster Ausprägung, die wenig oder gar keine Prüfmechanismen zur Erkennung interner Fehler besitzen. Da in vielen Fällen der PCI-Adapter selbst einen DMA-Transfer in den Hauptspeicher des Systems durchführt, kann er dabei fehlerhafte Daten transferieren, oder aber im Falle eines Adressierungsfehlers Daten an die falsche Adresse im Hauptspeicher schreiben.

6.2 zSeries Ein- und Ausgabe

Die Kanäle, die eine zentrale Rolle im zSeries E/A-System spielen, wurden ursprünglich aus diskreten Bauelementen unter Berücksichtigung der besonderen Anforderungen der IBM-Großsysteme hinsichtlich Datensicherheit, Zuverlässigkeit und Verfügbarkeit entwickelt. Heute werden für die Entwicklung der Kanäle vielfach die gleichen Standardkomponenten verwendet, die auch in PCs eingesetzt werden, da diese aufgrund der großen Stückzahlen ein sehr gutes Preis-Leistungsverhältnis bieten. So spielen PCI-Adapter auch in zSeries-Rechnern eine wesentliche Rolle.

Dabei wird jedoch der zSeries *Central Electronic Complex* Bereich wesentlich stärker von dem PCI-Bereich mit seinen Standardkomponenten abgeschirmt, als dies auf den meisten anderen Rechnersystemen üblich ist. Insbesondere führt der PCI-Adapter auf der zSeries keinen direkten DMA-Transfer in den zSeries-Hauptspeicher durch. Weiterhin werden spezielle Maßnahmen zur Erkennung und Korrektur von Fehlern ergriffen.

Eine wichtige Komponente des E/A-Subsystem der zSeries-Systeme ist die *Common I/O Card*. Diese Karte bildet die Basis sowohl für den FICON- und Fibre Channel-Kanal, als auch für Ethernet, ATM und andere schnelle Kommunikationsprotokolle.

Die Common I/O Card stellt ein PCI- bzw. PCI-X-Interface zur Verfügung. An dieses kann entweder eine Standard PCI-Karte angeschlossen werden, oder aber Komponenten mit integriertem PCI-Interface. Bild 5 zeigt die Common I/O Card in ihrer Verwendung als FICON- bzw. Fibre Channel-Karte.

Die Common I/O Card besteht aus einem PowerPC Prozessor, Memory Controller, lokalem Speicher für Programmcode und Daten, sowie einer STI-PCI-Bridge. Dazu kommt eine protokollspezifische Interface-Komponente, in diesem Falle ein Interface Controller für Fibre Channel.

Sowohl der PowerPC als auch der Memory Controller sind doppelt vorhanden, jeweils als sog. Master und Checker. Alle Operationen der Master-Komponenten werden parallel von den Checkern durchgeführt. Eine Funktionseinheit in der STI-PCI-Bridge vergleicht die Ergebnisse. Bei Nicht-Übereinstimmung wird die Operation abgebrochen. Damit werden insbesondere Adressierungsfehler vermieden, die sonst zu einem Zugriff auf falsche Bereiche im zSeries-Hauptspeicher führen würden.

Der Transfer von Daten zwischen dem Hauptspeicher der zSeries und dem E/A-Medium (wie z. B. dem Fibre Channel) erfolgt jeweils mit Zwischenpufferung im Speicher der Common I/O Card. Der Transfer zwischen zSeries-Hauptspeicher und Puffer in der Common I/O Card wird per DMA von der STI-PCI-Bridge durchgeführt, während der PCI-Adapter den DMA-Transfer zwischen Puffer in der Common I/O Card und dem externen Übertragungsmedium (jeweils in beide Richtungen) durchführt. Damit ist sichergestellt, dass niemals der PCI-Adapter, sondern nur die STI-PCI-Bridge auf den Hauptspeicher der zSeries sowohl lesend als auch schreibend zugreifen kann. Durch interne Schutz- und Prüfmechanismen in der STI-PCI-Bridge werden sowohl Datenübertragungs- als auch Adressierungsfehler ausgeschlossen.

Die STI-PCI-Bridge spielt somit eine gewisse Firewall-Rolle. Der gesamte systemseitige Bereich, also vom zSeries-Prozessor bis zur STI-PCI-Bridge, ist durch eine Vielzahl von Mechanismen gegen Fehler jeglicher Art geschützt. Wenn Daten über die STI-PCI-Bridge diesen Bereich verlassen, um extern gespeichert zu werden, wird ein *Cyclic Redundancy Check* (CRC) generiert, der mit den Daten zu der externen Steuereinheit geschickt wird, sodass die Integrität der Daten verifiziert werden kann. Umgekehrt enthalten alle Daten, die von einer Steuereinheit empfangen werden, einen solchen CRC-Wert, der von der STI-PCI-Bridge überprüft wird. Damit ist sichergestellt, dass Übertragungsfehler an einer beliebigen Stelle auf dem Weg von der STI-PCI-Bridge zur externen Steuereinheit oder zurück erkannt werden.

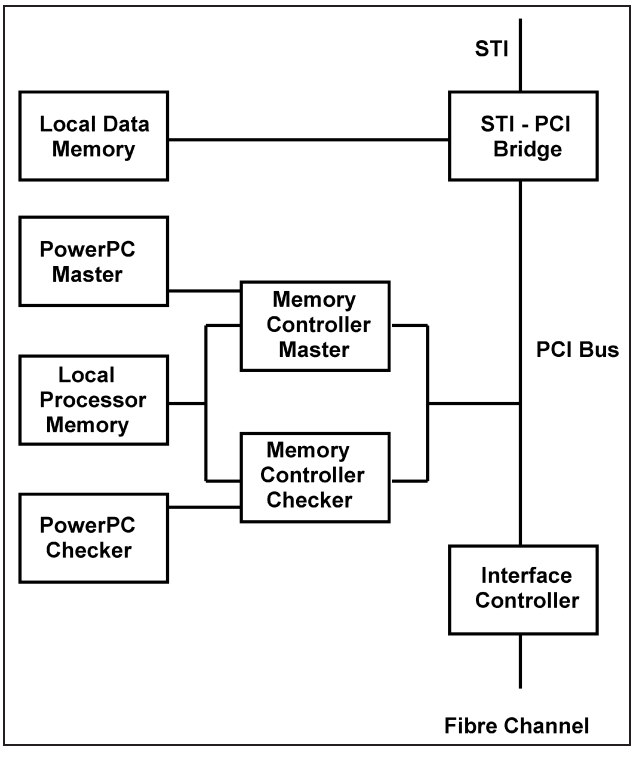


Bild 5 zSeries Fibre Channel, basierend auf der Common I/O Card.

Der CRC-Mechanismus garantiert, dass die gelesenen Daten korrekt und konsistent sind. Er garantiert jedoch nicht, dass es auch die richtigen Daten sind. Es könnte ja sein, dass durch einen Adressierungsfehler in der Steuereinheit oder der Elektronik der Platteneinheit selbst Daten von der falschen Stelle auf der Platte gelesen werden. Wenn dabei ein komplettes Datenpaket gelesen wird, hat dieses einen gültigen CRC, sodass dieser Fehler unerkannt bleiben würde. Deshalb wird auf der Platte mit jedem Datenpaket und seinem CRC auch die Adresse dieses Datenpakets auf der Platte abgelegt; diese wird dann beim Lesen der Daten wieder überprüft, sodass auch dieser Adressierungsfehler abgedeckt wird.

7 Partitionierung

Die Flexibilität der E/A-Architektur eines zSeries-Servers wird mit Hilfe eines Virtualisierungskonzeptes deutlich erhöht.

PR/SM *(Processor Resource/System Manager)* ist ein weiterer Microcode-Befehl. Sie ist ein Bestandteil der z/Architektur und in allen zSeries-Rechnern implementiert. Hiermit wird die Partitionierung eines physikalischen Rechners in mehrere logische Rechner ermöglicht, die *Logical Partition* (LPAR) genannt werden (Bild 6). Jeder logische Rechner hat sein eigenes Betriebssystem, seinen eigenen unabhängigen realen Hauptspeicherbereich und seine eigenen Kanäle sowie E/A-Geräte. Der gleichzeitige Einsatz von z/OS und Linux auf dem gleichen Rechner ist eine von vielen Einsatzmöglichkeiten. PR/SM koordiniert die einzelnen Betriebssysteme auf den verschiedenen CPUs eines symmetrischen Multiprozessors. Jedes Betriebssystem kann nur die Ressourcen benutzen, die für die logische Partition, in der es läuft, definiert sind. Eine gemeinsame Nutzung von Krypto-Koprozessoren und E/A-Adaptern und Geräten durch mehrere LPARs ist möglich.

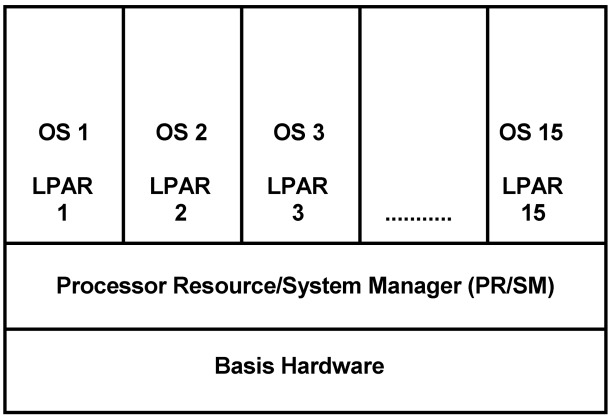


Bild 6 PR/SM und LPAR.

LPAR Logical Partition
OS Operating System

Das z/VM-Betriebssystem bietet ebenfalls die Möglichkeit des Betriebes von Gastbetriebssystemen. Historisch sind PR/SM und der z/VM-Kernel aus der gleichen Codebasis entstanden. S/390-Rechner anderer Hersteller verfügen über mit PR/SM vergleichbare Einrichtungen. Beispiele sind Hitachis MLPF oder Amdahls Multiple Domain Facility. IBM hat damit begonnen, auch andere ihrer Rechner-Plattformen mit einer PR/SM-ähnlichen Funktionalität auszurüsten. Für die Intel-Architektur bietet die Firma VMware ein entsprechendes Softwareprodukt (mit nicht ganz vergleichbaren Fähigkeiten) an.

PR/SM LPARs haben entsprechend einer Zertifizierung die gleichen Sicherheitseigenschaften wie räumlich getrennte Rechner (E4-Zertifizierung).

Die einfachste Stufe der Konfigurations-Flexibilität stellen die rekonfigurierbaren Kanalpfade dar. Definiert man einen Kanalpfad als rekonfigurierbar, so kann er, ohne andere E/A-Aktivitäten zu stören oder zu unterbrechen, von einer logischen Partition abgegeben und einer anderen logischen Partition zugeordnet werden. Noch größere Flexibilität erreicht man durch die Eigenschaft, dass Kanäle von verschiedenen Partitionen gemeinsam genutzt werden können. Dies bedeutet, dass ein einziger physikalischer Kanal virtualisiert wird, sodass jede Partition, die auf diesen Kanal zugreifen darf, meint, einen eigenständigen Kanalpfad zu dem angegebenen Switch oder der entsprechenden Steuereinheit zu besitzen. Somit laufen bis zu 15 logische Kanalverbindungen über eine einzige physikalische Strecke. Die entsprechende zSeries-Einrichtung wird als *Multiple Image Facility* (MIF) bezeichnet.

Man kann eine LPAR so definieren, dass sie eine oder mehrere CPUs (oder aber auch nur Bruchteile einer im *Time Slice*-Verfahren genutzten CPU), einen Hauptspeicher, Expanded Storage (optional) und Kanäle enthält. Es ist außerdem möglich, eine LPAR als *Coupling Facility* zu definieren [12].

LPARs besitzen die folgenden Eigenschaften:

- Es können maximal 15 LPARs definiert werden.
- Der reale Hauptspeicher für jede LPAR ist isoliert, ebenso der Expanded Storage.
- Über die dynamische Speicher-Rekonfiguration kann eine LPAR realen Hauptspeicherplatz abgeben oder erhalten.
- Alle Kanäle können als rekonfigurierbar definiert werden. Kanäle werden den LPARs zugewiesen. Es ist möglich, rekonfigurierbare Kanäle zwischen den LPARs zu bewegen. Die Kanäle können zwischen den LPARs durch Kommandos bewegt werden, ohne das Betriebssystem zu unterbrechen. Die entspre-

chende Einrichtung wird als *Dynamic Channel Path Management* (DCM) bezeichnet. Hierzu können Tasks benutzt werden, die in der z/OS Management Console verfügbar sind.

- MIF erlaubt es, Kanäle von zwei oder mehr LPARs zur selben Zeit zu nutzen.
- LPARs können per Definition so viele CPUs besitzen, wie installiert sind. Es ist möglich, CPUs den LPARs zuzuordnen oder diese gemeinsam zu benutzen. CPUs, die man ausschließlich einem LPAR zuordnet, sind für andere aktive LPARs nicht verfügbar. Die Ressourcen von gemeinsam benutzten CPUs werden aktiven LPARs zugewiesen, wie sie benötigt werden. Man kann CPU-Ressourcen begrenzen, wenn es erforderlich ist.

Der in der z/Architektur implementierte *Intelligent Resource Director* (IRD) ist eine Erweiterung des PR/SM- und LPAR-Konzeptes. IRD übernimmt die Optimierung der Prozessor- und Kanal-Ressourcen über die verschiedenen LPARs. Die drei wichtigsten Funktionen des IRD sind das *LPAR CPU-Management,* das DCM und das CSSPQ (siehe Absatz 5.4). Sie werden vor allem von dem Work Load Manager des z/OS-Betriebssystems genutzt.

Marken

Folgendes sind Marken der International Business Machines Corporation in den USA und/oder anderen Staaten: IBM, ESCON, FICON, Geographically Dispersed Parallel Sysplex, OS/390, Parallel Sysplex, PowerPC, PR/SM, S/390, z/Architektur, z/OS, zSeries, z/VM

Danksagung

Der vorliegende Beitrag entstand aus einem Gemeinschaftsprojekt der beiden Verfasser mit mehreren Mitarbeitern der IBM Laboratorien in Böblingen. Besonders hervorzuheben sind die Beiträge von Herrn Gerhard Banzhaf, Herrn Jürgen Märgner sowie Herrn Thomas Schlipf. Die genannten Herren waren zusammen mit Herrn Helge Lehmann maßgeblich an der Entwicklung des z900 E/A-Systems beteiligt.

Literatur

[1] I. Adlung, G. Banzhaf, W. Eckert, G. Kuch, S. Mueller, and C. Raisch: FCP for the IBM eServer zSeries systems. Access to distributed storage. IBM Journal of Research and Development, Vol. 46, No. 4/5, Jul/Sep 2002.

[2] G. Amdahl, G. Blaauw, F. Brooks: *Architecture of the IBM System/360.* IBM J. Res. Devel. 8 (2), 87–101 (1964).

[3] T. A. Gregg, R. K. Erickson: Coupling I/O channels for the IBM eServer z900. IBM Journal of Research and Development, Vol. 46, No. 4/5, Jul/Sep 2002.

[4] H. Harrer et al.: First- and second-level packaging for the IBM eServer z900. IBM Journal of Research and Development, Vol. 46, No. 4/5, Jul/Sep 2002.

[5] J. Hennessy, D. Patterson: *Computer Architecture: A Quantitative Approach.* 3rd Edition, Morgan-Kaufman, 2002.

[6] P. Herrmann, U. Kebschull, W. G. Spruth: *Einführung in z/OS und OS/390.* Oldenbourg, 2002. ISBN 3-486-27214-4.

[7] J. M. Hoke et al.: Self-timed interface of the input/output subsystem of the IBM eServer z900. IBM Journal of Research and Development, Vol. 46, No. 4/5, Jul/Sep 2002.

[8] Sonderheft des IBM Systems Journal zum Thema Sysplex, Vol. 36, No.2, 1997.

[9] Sonderheft des IBM Journal of Research and Development zum Thema Sysplex, Vol. 36, No.4, 1992.

[10] G. F. Pfister: In Search of Clusters, Seite 469–470. Prentice Hall 1998, ISBN: 0138997098.

[11] z/Architecture Principles of Operation. IBM Form No. SA22–7832.

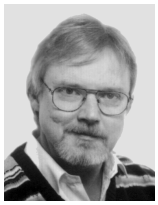
[12] E. Rahm, W. G. Spruth: Sysplex Cluster-Technologien für Hochleistungsdatenbanken. Datenbank Spektrum, Heft 3, 2002, S. 16.

[13] W. G. Spruth: The Design of a Microprocessor. Springer, 1988. ISBN 3–540–51395–7.

[14] T.A. Wise: I.B.M.'s $5,000,000,000 Gamble. Fortune Sept. 1966, p. 118. Eine Kopie ist unter http://jedi.informatik.uni-leipzig.de zu finden.

[15] T.A. Wise: The Rocky Road to the Market Place. Fortune Oct. 1966, p. 138. Eine Kopie ist unter http://jedi.informatik.uni-leipzig.de zu finden.

[16] http://www-1.ibm.com/servers/eserver/zseries/zos/wlm/documents/velocity/velocity.html. Eine Kopie ist unter http://jedi.informatik.uni-leipzig.de zu finden.



1

2

1 Dr. Helge Lehmann ist Chefberater in der Hardware-Entwicklung der IBM Deutschland Entwicklung GmbH und arbeitet an zukünftigen Strategien im I/O-Bereich der zSeries-Systeme. Er promovierte in Köln über *Numerische Lösungen Partieller Differenzialgleichungen* (Mehrgittermethoden) und ist seit 1985 bei IBM tätig. Dort hat er seinen Schwerpunkt im Bereich *Channel Subsystem Design* und *I/O Configuration Definition* für S/370-, S/390- und z900-Systeme.
Adresse: IBM Deutschland Entwicklung GmbH, Schoenaicherstraße 220, D-71032 Böblingen, E-Mail: hhlehmann@de.ibm.com

2 Hon.-Prof. Dr.-Ing. Wilhelm G. Spruth studierte Elektrotechnik an der RWTH Aachen. Er ist Honorarprofessor am Institut für Informatik der Universität Leipzig, sowie am Institut für Informatik der Universität Tübingen. Davor war er langjähriger Entwicklungsleiter in den IBM Laboratorien Böblingen. In dieser Funktion war er unter anderem als Projektleiter für die Entwicklung des ersten S/370- Mikroprozessors in CMOS-Technologie verantwortlich. Herr Spruth ist Verfasser mehrerer Fachbücher und Inhaber zahlreicher Patente. Er ist Mitglied in der ACM, GI und der Computer Society des IEEE.
Adresse: Eberhard-Karls-Universität Tübingen, Wilhelm-Schickard-Institut für Informatik, Arbeitsbereich Technische Informatik, Sand 13, Raum 207, D-72076 Tübingen, E-Mail: spruth@informatik.uni-tuebingen.de