Coupling I/O channels for the IBM eServer z900: Reengineering required

by T. A. Gregg
R. K. Errickson

The IBM eServer z900 introduces new Parallel Sysplex® coupling channels that satisfy evolving requirements in a way that minimizes product and development costs. Their design also provides backward compatibility with earlier S/390® models, spans all three coupling channel design points, and anticipates future end-of-life technology issues. The original intersystem channel (ISC) design was improved, and new features added, but the core chips were retained. This paper describes the efforts that led to the improved design.

Introduction

From 1994, when the Parallel Sysplex* was first introduced, until the introduction of the z900, a few significant improvements were made to its coupling channels, the communication mechanism for the Parallel Sysplex. The first sections of this paper contain an overview of the Parallel Sysplex and a description of the prior design of the coupling channels. The next sections describe how new requirements led to a rethinking of the coupling channel design. The remainder of the paper describes how these new requirements were fulfilled for the z900 by using new techniques.

When the Parallel Sysplex was first introduced [1–3], it had the standalone structure shown in **Figure 1**. The primary goal of the Parallel Sysplex is to provide a highly scalable computing platform while displaying a very high degree of fault tolerance. Multiple servers running one or more instances of an operating system (OS) are typically connected to two coupling facility (CF) images. The connections, designated as intersystem channels (ISCs), are described in more detail later. Each of these links has a sender channel (S) at the OS side in the server connected to a receiver channel (R) at the CF side. Primary messages are sent from an OS to a CF, and these messages read or modify shared data structures in the main memory of the CF. When the CF determines that other OS images have interest in the data structure that was changed or accessed, it sends alerts to these OS images. These alerts are sent by secondary messages from the CF to one or more of the OS images, but the alerts do not interrupt the OS image. Instead, they are registered in special areas of system memory to be queried by the OS at some later time. With no interruptions to the OS images for the alert function, the Parallel Sysplex can scale to larger configurations more easily.

Buffer sets

Figure 2 depicts the structure of a buffer set. The sender at the OS originates a primary message with a command

0018-8646/02/$5.00 © 2002 IBM

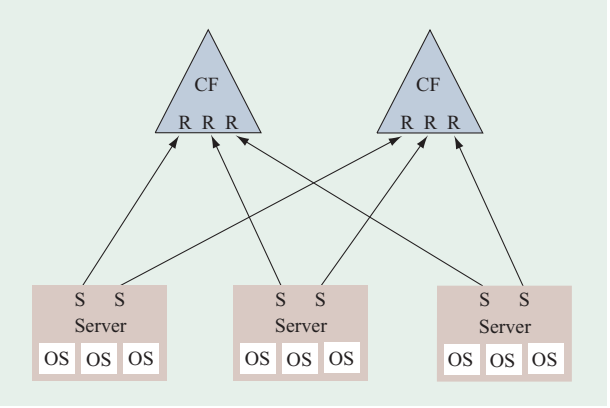

Figure 1

Standalone coupling facilities of the Parallel Sysplex.

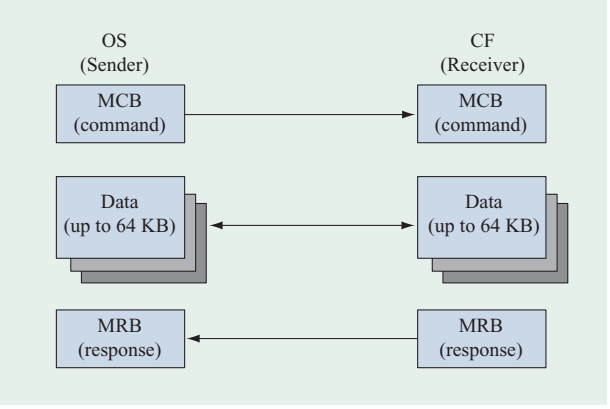

Figure 2

Buffer set for exchanging messages.

designated as a message command block (MCB) and places it in an MCB buffer in main memory. If there is any data to be moved, the OS either places the data to be sent into data buffers in main memory (the write case) or allocates buffers in main memory for the receipt of data (the read case). The OS also allocates a buffer in main memory for the receipt of the response called the message response block (MRB). The OS then initiates the message exchange by executing a processor instruction called *send message*. The instruction specifies, in essence, the buffer set that contains the buffers for the MCB, data, and MRB. Meanwhile, the receiver at the CF has prepared to receive the MCB by allocating a buffer in its main memory. When the CF receives an MCB, it examines its contents to determine whether there is any data to be moved. For a write operation, the CF prepares data buffers in its main memory (a data structure) to receive the data. For a read operation, the CF sends data buffers in its main memory back to the OS. After data transfer (if any) is complete, the CF sends the MRB from a buffer in its main memory back to the OS. The CF executes special instructions to find any pending messages (*locate channel buffer*), move data (*move channel buffer data multiple*), and send the MRB (*signal channel buffer*).

Similarly, buffer sets are used by the CF to send secondary messages to the OSs. These secondary message buffer sets have only MCB and MRB buffers; no data areas are required.

Until the development of the z900, each ISC, ICB, and IC connection had two buffer sets for the OSs to send primary messages to the CF and two more buffer sets for the CF to send secondary messages to the OSs. As we later describe, one of the biggest changes for the z900 was a considerable increase in the number of buffer sets.

Intersystem channel 1 (ISC-1)

The first coupling channel was the intersystem channel (ISC-1), which was a bit-serial, optical interface capable of operating at ten kilometers or more. **Figure 3** is a diagram of the ISC-1 and ISC-2 structure (discussed below). The same channel controller and link adapter application-specific integrated circuit (ASIC), or chip, part numbers were used on all Parallel-Sysplex-capable servers, including the bipolar product lines. Unique host adapters were required in order to accommodate each of the various product lines because each product line had differences in its connections to main memory.

The channel controller chip included an 801 microprocessor and interface logic connecting the host adapter with the link adapter. The 801 microprocessor was the first IBM RISC design [4, 5] and was the foundation of the later PowerPC* architectures. To minimize the unique code tools required, the 801 architecture was chosen, since it was also used in the I/O processor (IOP) in the largest IBM bipolar servers. In these servers, the IOP is used primarily for offloading I/O operations from the normal central processors. In the IBM CMOS servers, the 801-based IOP was replaced by a normal central processor; from a machine-organizational viewpoint, the IOP looks like any other central processor in the server, but it does not execute code on behalf of any OS or CF image. The function of the IOP is still to offload I/O and other system functions from the normal central processors.

In ISC-1, the 801 microprocessor is a unique implementation that has robust error detection while meeting performance requirements. The Licensed Internal Code is cached in outboard SRAM chips with main memory containing a complete image of the code. The channel controller takes commands from the host adapter, queues them, and sends them to the link adapter. It

receives events from the link adapter and queues them on their way to the host adapter. The channel controller chip also manages the data transfer between the host adapter and the link adapter.

The link adapter contains the buffer set arrays (embedded RAM) for the two primary and two secondary buffer sets. It buffers data to and from the optical link on its way from and to the host adapter. It also controls the optical transceivers, while detecting errors and managing low-level link activities such as word synchronism acquisition.

Multiple host adapter chips were developed, two for the two bipolar product lines and one for the CMOS product line. Every attempt was made to keep the 801 microprocessor code for all three product lines the same. However, the host adapter and code for the CMOS product line was not performance-optimized, and the second CMOS host adapter chip with its supporting code (described below) had an improved design.

Intersystem channel 2 (ISC-2; HyperLink)

In ISC-1, as the message processing progressed, either a central processor or an IOP was interrupted multiple times, depending on the quantity of data that was to be transferred. In the bipolar product line, the IOP interruptions were efficient, and the performance was acceptable. On the other hand, the CMOS IOP implementation was not as efficient, and performance suffered. By moving most of these message-passing processing steps out of the central processors and IOPs and into hardware state machines, performance was greatly improved while utilization of the central processor and IOP was reduced. These new hardware state machines were implemented in a new host adapter chip developed for the IBM single-product line, the CMOS product line. This new host adapter required significant code changes in the central processors, IOP, and the channel controller 801 microprocessor. To control the new host adapter, the central processors and IOPs used a new set of hardware commands, which comprise command set 2. Command set 1 drives the ISC-1. HyperLink is another term used for ISC-2.

Integrated cluster bus 2 (ICB-2)

The next improvement in coupling links, ICB-2 [6], was introduced in 1998. The ICB-2 designation is derived from the fact that ICBs use command set 2, the same as the one used for ISC-2. ICB-2 was developed after command set 2 was adopted, and it never used command set 1. ICB-2 uses very fast, limited-distance (10 meters) links to improve the performance of small clusters of servers.

ICB links are built directly on the internal system area link called the self-timed interface (STI). This link is dual simplex, one byte wide, in copper cable, and runs at 333 MB/s at a distance of 10 meters. By adding a relatively small volume of very complex logic to the system hub chips, ICB channels improve performance while reducing product cost and improving reliability, availability, and serviceability.

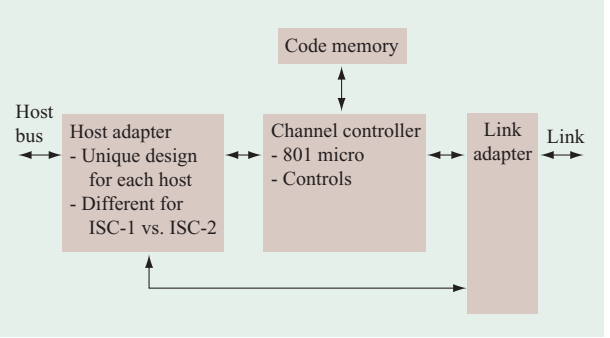

Figure 3

Structure of ISC-1 and ISC-2.

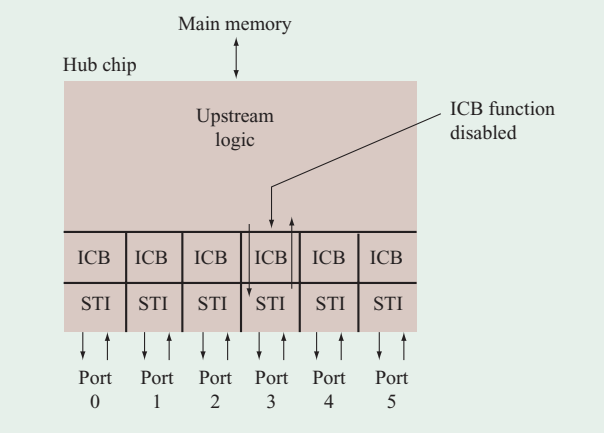

Figure 4

ICB structure.

Figure 4 shows the structure of the hub chips. Such chips connect input/output adapters to main memory and provide six STI links. Behind each link is an ICB channel. When enabled, this channel receives hardware commands (command set 2) from the central processors and IOPs to logically perform the same message passing as ISC and IC links (see next section). The ICB link protocol is based on STI and uses its low-level flow control. This simplifies the ICB hardware implementation to the point where it is practical to use hardware state machines, and, in this case, avoids the added chip area and development expense of an imbedded microprocessor. Further, because the link protocol is based on STI (logical and physical), it is limited

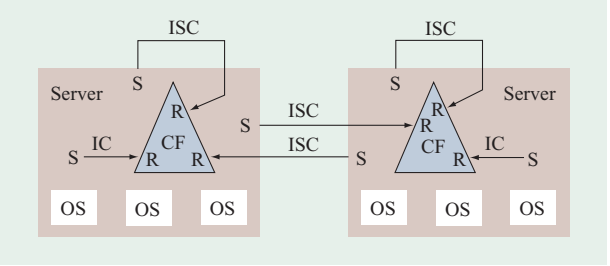

Figure 5

Integrated coupling facilities.

with respect to distance. Additional flow-control handshakes between the command and optional data transfer eliminate large link buffers, but because distance is limited, these handshakes add very little latency to message exchanges.

Internal channel (IC)

It quickly became apparent that while a standalone CF provides a certain level of error isolation between the CF and the server, it is expensive. The CF could be used only as a CF and could not run OS images. By dedicating some of the processors in a server and using logical partitioning (LPAR), CF image(s) are added to a server, as shown in **Figure 5**. However, external ISC links are required to connect an integrated CF to the OS images. These links are the interconnected pairs of ISC links shown in the upper portion of Figure 5. The same sender (S) and receiver (R) concepts shown in Figure 1 still apply in Figure 5. In particular, note that two links (one in each direction) are required between the two servers in Figure 5.

Relief from the external looped-back links came in 1999 with the introduction of internal coupling (IC) links, also shown in Figure 5. These logical links were implemented in Licensed Internal Code (or simply "code") and required no ISC channel hardware. From a software and CF view, IC links look like ISC links with senders and receivers. IC links were also limited to two buffer sets for primary and two buffer sets for secondary messages. To further make the IC links look like real ISC links to the software (OS) and code (CF), they consume the highly constrained channel path ID (CHPID) space, currently limited to 256 paths.

IC links proved to have very good performance because the code path lengths are no longer than those required by ISC links, and the data movement is performed by very fast memory-to-memory transfers.

ISCs, ICBs, and ICs are three different implementations of coupling channels, and each is suited to a particular environment. It should be understood that all three may be present in any given Parallel Sysplex configuration.

New Parallel Sysplex requirements

In the early stages of the development of the z900, we examined the Parallel Sysplex strategy and considered how the ISC-2, ICB-2, and IC designs had to evolve in order to meet the demands imposed by this strategy. We designated the new channels as ISC-3, ICB-3, and IC-3, with the "3" designation representing the third generation of coupling links and a new set of commands, command set 3.

Link bandwidth

The first and most obvious requirement was to increase link bandwidth. The OS images use the option of synchronous message passing when the total round-trip time to process a message is less than the time it takes the OS to switch to and from a different task. With synchronous message passing, the processor (process or thread) waits for the message to complete before proceeding. As processors become faster, the links must also become faster to allow efficient synchronous message passing.

For ISC channels, we followed the Fibre Channel lead and increased the link speed of 1.062 to 2.125 Gb/s. With the 8B/10B transmission code, the bandwidth increased to more than 200 MB/s. The resulting coupling efficiency remained close to 80% [7, 8].

For ICB links, we had to consider the "normal" STI exploiters [9] (input/output adapters). To satisfy both ICB and input/output adapters, the STI link speed was increased from 333 to 1000 MB/s [10, 11]. With the overhead of the STI packets and flow control, this resulted in an increase in bandwidth to about 650 MB/s. Because STI is a byte-wide interface, the signaling rate on each conductor went from 333 MB/s on the previous STI to 1 GB/s, close to the limit for noncoded (such as the 8B/10B code) information over 10 meters of copper cable.

Channel latency

The hardware path length of ISC and ICB channels had to be improved. The operating frequency of ISC-1 and ISC-2 has remained the same for all previous servers. Any new ISC design would have to increase the operating frequency. With ICB, the improvements in the hub chip operating frequency automatically reduced the path length.

Buffer sets

Our experience with ISC-2, ICB-2, and IC-2 taught us that some workloads required more than one link between CF images and OS images to improve throughput. Also, redundant links were added to improve availability. We were not constrained by the raw bandwidth of the links, but rather by the limited number of buffer sets. As

described above, the number of buffer sets determines the number of messages that can be handled concurrently. With only two buffer sets in each direction, and considering the time it takes to process a message, the link utilization is well below 10%. Improving the link bandwidth further decreases link utilization because even less time is used to transmit the messages.

With a link utilization as low as 10% or less, more buffer sets can be supported on each link without increasing link utilization to the point at which link queueing effects become noticeable. The number of buffer sets was increased from two to seven, thus reducing the number of links required by a factor of up to 3½.

Peer mode

As Parallel Sysplex implementations move to internal CFs and the CFs become more and more distributed among multiple servers, many more ISC links are required, and often multiple links are needed to connect the same two servers. Redundant links are recommended for high availability, doubling the number of links required. As indicated in connection with Figure 5, two links are required to interconnect the two servers. **Figure 6** shows how four servers, each containing multiple OS images and an internal CF, must be interconnected. This figure depicts the concept of peer links. Each end of a peer link has the capabilities (in the form of buffer sets) to be both a sender (owned by multiple OS images) and a receiver (owned by a CF) at the same time. In the example of Figure 6, peer mode further decreases the number of ISCs and/or ICBs by a factor of 2.

CF duplexing connectivity

In Figure 1, the CF and OS images are in different physical machines. If a CF fails, its structures can be rebuilt from information in the multiple OS images. With internal CF images distributed in multiple servers, it becomes increasingly difficult for the OS images to avoid putting CF data structures in the same server that is running an OS that is required to rebuild the structures, thus eliminating single points of failure. In order to remove this complexity from the OS, CF duplexing was introduced. CF duplexing allows pairs of CFs to automatically synchronize two copies of CF data structures, one in each CF. Each CF of a duplexed pair is placed in a different physical server.

The two CF images of a duplexed pair exchange messages to keep themselves synchronized, and a link is required to exchange these messages. Before duplexing was introduced, the coupling links were defined to connect one or more OS images to a single CF image; interconnecting two CF images required a new connection. To avoid creating a unique coupling channel to allow duplexed CF images to directly communicate with each other, the definition of existing links was changed to allow a single CF to share a sender with the OS images. Secondary messages between the two CF images could be sent in both directions with a pair of coupling links. As shown in Figure 5, the CF in one server sends secondary messages using its receiver channel to a CF in the other server using its sender channel. A second link is required to send messages in the opposite direction. The introduction of peer links, with the capability of being simultaneously both a sender and receiver, reduces the additional number of links required, depending on the Parallel Sysplex configuration.

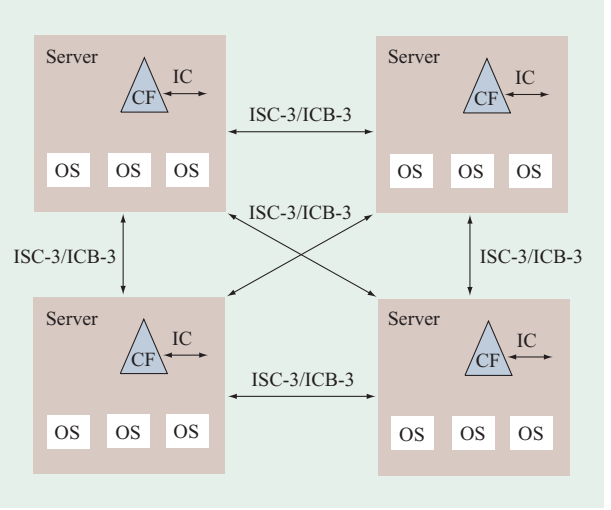

Figure 6

Integrated coupling facilities with ISC-3/ICB-3/IC-3 (peer).

Command set 3

Besides Parallel Sysplex requirements of adding more buffer sets with peer mode, the z900 has new system requirements that could not be met by the existing ISC-2 and ICB-2 command set.

Larger main-memory addressing and more LPAR images

Main-store physical addressing supported by ISC-3 and ICB-3 was increased from 36 bits to 48 bits. This increase should protect the design for a few more generations of processors. The number of LPAR images supported also affects the command structure. Earlier command structures could support only 15 images.

Elimination of ISC/ICB access to expanded memory

Before the z900, OS and CF images were constrained to two gigabytes of main memory because of 31-bit memory addressing. When the CF needed more memory, it had

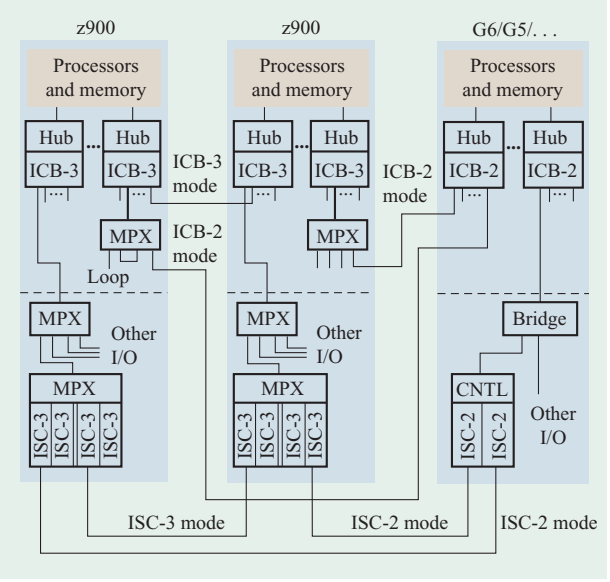

Figure 7

External channel connections.

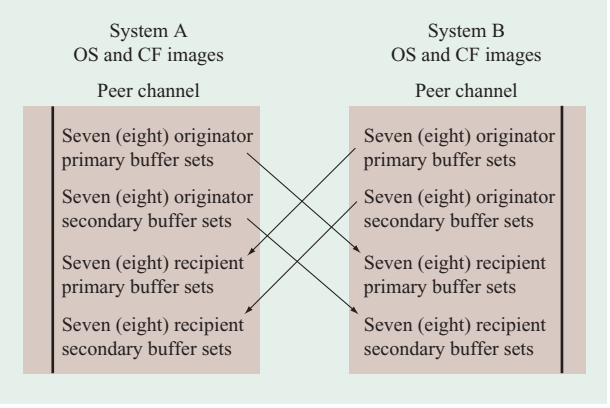

Figure 8

ISC-3, ICB-3, and IC-3 peer-mode buffer sets.

to use expanded memory, thus providing more physical memory access through a separated address space. With the z900 64-bit architecture [12], the CF no longer has to use expanded memory. Zone relocation gives each image its own view of a contiguous physical memory address space, or zone, starting at address zero. The computation used to determine the physical memory address of the server adds a base address to the LPAR physical address. Eliminating CF access to expanded memory worked well with moving the LPAR zone-relocation function from code to special hardware in the hub chip because zone relocation for expanded memory requires a different algorithm than main memory. Since ISCs and ICBs were the only I/O with access to expanded memory, eliminating this function simplified the hub chip design.

z900 coupling channels

Figure 7 is an overview of the way z900s are interconnected and the way they are connected to the previous generations of S/390 (G6, G5, etc.) servers. In the figure, the two z900 servers are interconnected with an ISC-3 link and an ICB-3 link operating in peer mode. Connections to the servers of previous generations were achieved by ISC-3 operating in ISC-2 compatibility mode and by ICB-3 operating in ICB-2 compatibility mode through multiplexor chips (MPXs) that convert the STI links from 1 GB/s to 333 MB/s.

Common aspects of ISC-3, ICB-3, and IC-3

The part of the channel design that is common across the three link types is the number of buffer sets. The additional buffer sets and peer nature of ISC-3, ICB-3, and IC-3 are depicted in **Figure 8**. The number of each of the four types of buffer sets (originator primary, recipient primary, originator secondary, and recipient secondary) was increased from two to seven. ICB-3 actually contains eight buffer sets of each type to help implement compatibility mode, as described below.

ICB-3 implementation

With the given requirements, the ICB-3 design direction was relatively clear: Add more buffer sets, provide peer mode, and push the STI link speed to a gigabyte per second. But there was an additional requirement—to provide attachment to the previous generation of ICBs (ICB-2). Recall that ICB-2 operates at 333 MB/s in sender/receiver mode, with two buffer sets in each direction.

To provide connections to ICB-2, we investigated several approaches. One was to add a set of ICB-2 ports on the hub chip; this approach was quickly abandoned because it required too many chip pins dedicated to this function. Another approach was to create a mode in which the 1GB/s STI ports could also operate at 333 MB/s. This second approach was also abandoned because it would add considerable complexity in the line drivers and receivers. For example, clock extraction at multiple signaling speeds requires special circuits. An even more difficult problem was related to the operating voltages. The operating voltages of silicon circuitry continue to decrease as device geometries shrink. As a consequence, the signaling voltages are also becoming smaller. In fact, the newer receiver circuits cannot tolerate the voltage swings of the older driver circuits. In our case, these

concerns make it impractical to have line driver and receiver circuits that can operate in both 333MB/s and 1GB/s modes.

The approach we chose was to add a converter/multiplexor chip that connects ICB-3 ports to ICB-2 ports. This design point fits into the rest of the I/O plan, since a converter is also required for normal I/O. For normal I/O, the converter chip is a speed-matching buffer and a multiplexor. It connects a new 1GB/s STI link to four old 333MB/s STI links. The protocols on both the old and new STI links are same; therefore, the complexity of the converter chip is minimized. The converter simply routes outbound STI packets to the appropriate STI port and adds source port information to incoming packets. The converter chip contains no state information as to packet ordering or content.

The same converter chip can also create four ICB-2 links from one ICB-3 link. Since the converter has no capability to implement any of the ICB protocols, all of the state information for all four ICB-2 links is retained in the ICB-3 hardware. When operating in ICB-3 peer mode, all 32 buffer sets are used by the single link. When ICB-3 is operating in compatibility mode, the buffer sets are distributed among the four ICB-2 links at the far end of the converter chip. **Figure 9** shows an example of how the ICB-3 buffer sets are distributed. Since each ICB-2 requires only four buffer sets (two originator primary and two recipient secondary for sender channels, or two originator secondary and two recipient primary for receiver channels), half of the ICB-3 buffer sets are not used when it is in compatibility mode. In the figure, systems B and E have a sender channel and systems C and D have a receiver channel. With all of the state information in the ICB-3 hardware, the converter chip need only provide the speed-matching and multiplexing functions.

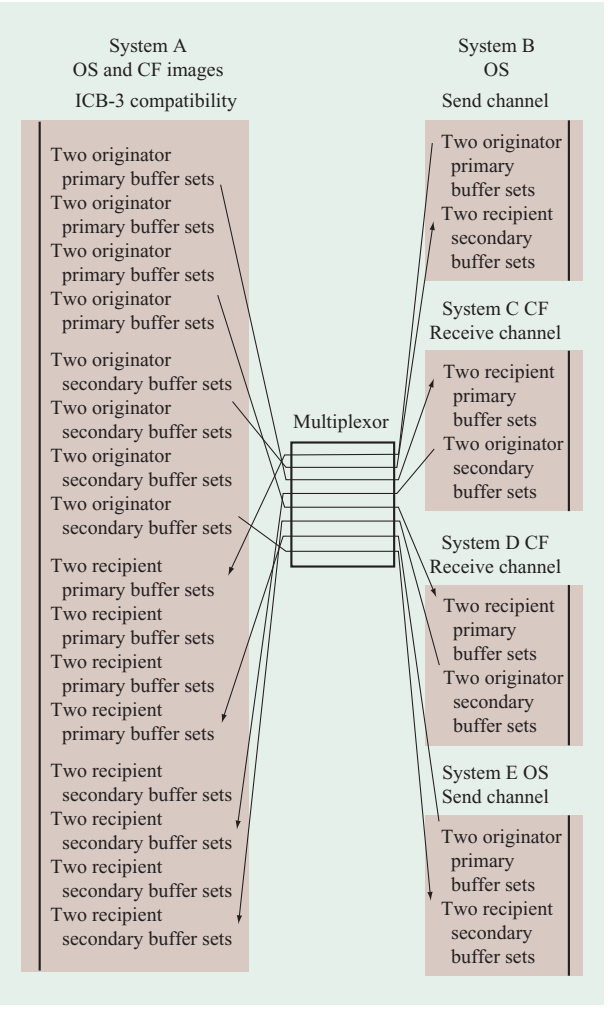

Figure 9

ICB-3 compatibility-mode buffer set allocation.

ISC-3 implementation

All of the ISC-2 components were nearing their "end-of-life" phase. Production of the ASIC and SRAM technologies was soon to be discontinued. The gigabit link module (GLM), which includes the optical transceivers, was becoming expensive, and there was the threat that its production would soon be discontinued. GLM uses open fiber control (OFC), an obsolete method of providing laser safety (described later). We also had many new requirements, and using ISC-2 on the z900 was not an option.

First, we needed to sweep all of the multiple chips in ISC-2 into a single state-of-the-art ASIC. This in itself reduces product cost while improving reliability. Using a single ASIC also reduces the message-passing latency by eliminating the communication between the multiple chips in ISC-2.

Using denser chip technology also allowed us to package more ISC-3s into the same space required by ISC-2. The new package [13] has four ISC-3s in a single card slot. However, with four channels on a card, the user would need to purchase increments of four, which would be costly. Hence, we divided the card into a mother card containing a converter/multiplexor chip and two daughter cards, each containing two ISC-3s.

After carefully looking at ISC-2 design, we found that we could reuse much of the ISC link protocol engine and optical transceiver interface logic from the link adapter, but the general bus structure and data transfer engines to main memory were inadequate. ISC-2 had internal bottlenecks that could not keep up with the 2.125Gb/s optical transceivers. The main-memory attachment had to be changed from the existing "internal bus" to a direct STI attachment. Implementing command set 3 caused

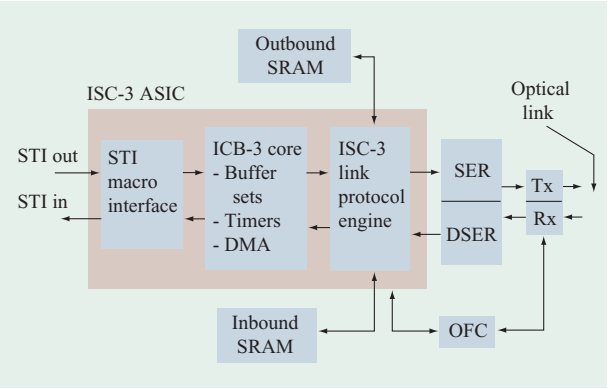

Figure 10

ISC-3 hardware structure.

many more changes that the underlying ISC-2 structure could not handle without considerable redesign. With 28 buffer sets, the entire design direction of the rest of the link adapter chip had become unworkable.

We decided to discard most of the ISC-2 design and start with the ICB-3 core design, as shown in **Figure 10**. This design already implements command set 3, so we needed to add the ISC link protocol engine and optical transceiver interface derived primarily from ISC-2. We also had to add many more physical buffers to handle an increase in the number of buffer sets. All of the existing chip interfaces in ISC-2 would have to be eliminated.

Data buffering

Increasing the number of buffer sets from four to 28 requires more link buffers to receive inbound messages. Message information is transmitted over ISC links in frames, and each frame consists of a link header, a variable-size payload up to 4096 bytes, and checking fields. The credit-based ISC link flow control requires inbound link buffers of sufficient size to receive frames for all of the advertised buffer sets. ISC-2 has sufficient inbound frame buffers for two primary and two secondary buffer sets. Each command and response has 256 bytes, and each data area has 4096 bytes. With only 4096 bytes of data area buffering for each primary buffer set, messages transferring more than 4096 bytes require an intermediate link acknowledgment protocol. Each time 4096 bytes are received and sent to main memory, ISC-2 generates a link acknowledgment allowing the other end of the link to send the next 4096 bytes. Therefore, at a minimum, the inbound buffering must be increased to handle 28 buffer sets.

Further increases in the size of the inbound link buffers were made to improve performance by eliminating the link acknowledgments. The largest data transfer allowed in a single message is 64 KB. Therefore, in ISC-3 peer mode, increasing the inbound buffering for data areas from 4096 KB to 64 KB eliminated the link acknowledgments. Consider, for example, a ten-kilometer link. At an optical link propagation delay of 5 μs/km, the round-trip delay at 10 km is 100 μs. At 2.125 Gb/s (212.5 MB/s), the 64KB data transfer takes about 300 μs. With only one link turnaround (no link acknowledgments required), the link time for the message is about 400 μs. Compare this to the ISC-2 case, in which 15 link acknowledgments are required, adding another 1500 μs to the link time, and the slower transfer rate adds 300 μs more (the inbound buffering is implemented by an off-chip SRAM).

ISC-2 did not have outbound link buffers large enough to hold the last transmitted frame for each buffer set. There are situations at the receiver where a request for retransmission is received after the CF has changed the contents of the data structure. In these cases, the receiver cannot retransmit the data, and it simply ignores the request, letting the sender time out. Once the sender times out, the OS is informed, and it redrives the message. Because this time-out processing requires OS overhead and takes time, sufficient outbound buffering was added so that all outbound frames could be retransmitted. This buffering is the same size as the inbound buffering and is implemented in another off-chip SRAM.

ISC-3 compatibility mode and optical transceivers

ISC-3 compatibility mode has two parts. First, it must operate as a sender or receiver with two originator and two recipient buffer sets. It must also exchange link acknowledgments during long data transfers at 4096-byte boundaries. As it turns out, with the exception of the required link acknowledgments, compatibility mode is a subset of ISC-3 peer mode. The additional hardware complexity to support compatibility mode is limited primarily to recognition and generation of link acknowledgments.

Speed compatibility proved to be much more difficult than just accommodating the speed difference, because ISC-1/2 uses the GLM with OFC, and the OFC protocol has to be implemented when connected to ISC-1/2. OFC addresses lasers whose optical power output is not controlled well enough to keep it below safe levels. With these lasers, when the optical fiber connection is opened, personnel could be exposed to harmful power levels. When a receiver detects no light, it assumes that the fiber connection has been opened at some point, and it turns off its laser to prevent high levels of exposure. To ensure that the transceivers at both ends of the link are "playing by the rules" and to power up the lasers, an elaborate handshake is required. Only the very early users of Gb/s laser transceivers needed to use OFC, because the safety standards changed. The newer lasers have a much tighter

control of their optical power output, and an open fiber is no longer a hazard. Introducing OFC for the higher 2.125Gb speeds is undesirable and is not required, but OFC is still required when operating with ISC-1/2.

We wanted to use industry-standard optical transceivers that do not provide the OFC function, so we ended up developing the equivalent of the GLM with three discrete parts: the optical transceiver, the serializer/deserializer, and the OFC controller. We added a special circuit that disables the OFC when operating at 2.125 Gb/s. This circuit detects when the system disables OFC, and it shuts down the laser driver long enough to engage the OFC circuit at the other end of the link (if one exists). Using the circuit, we could remain compliant with the OFC safety standard, since any bug in the code controlling OFC would be detected by the OFC at the other end of the link. We finally decided to use a single ISC-3 card that could be programmed to operate at both 1.0625 Gb/s with OFC enabled and 2.125 Gb/s with OFC disabled.

Microprocessor elimination in ISC-3

Earlier ISC designs incorporated an 801 microprocessor uniquely designed for ISC, and our biggest problem was figuring out a new implementation. The flexibility of the microprocessor was required in ISC-2 to keep the design synchronized with an evolving link protocol. By 1998, the link protocol was stable enough to consider putting all of the mainline function into hardware state machines. Also, our experience with ICB-2 taught us that a hardware-intensive design was not too complex to attempt. The ISC-2 801 microprocessor design was unique and included good error checking, and the instruction set was the same as that used in the IOP in the bipolar processors. Using a common architecture allowed tools to be shared with other code-development teams. As the ISC found its way into the CMOS machines and the bipolar product lines were discontinued, maintenance of the tools (compilers, simulators, libraries, release processes) became problematic. It became difficult to get other development sites to maintain compilers, since the instruction set was nearing obsolescence. It was clear that any microprocessor implementing the 801 architecture was not an option. It was also becoming a problem to retain the skills required to maintain the 801-based code.

We next considered using a newer microprocessor such as a PowerPC. However, neither the single-chip Power microprocessors nor the embedded cores provided the robust level of error checking that is a hallmark of zSeries* servers. To achieve this level of checking, we would either have to duplicate and cross-check the microprocessors, or develop our own. But even though a new microprocessor architecture would allow us to use the newer, better-supported tools, in the long run, the tools would again become a problem. Unlike the Intel x86 architecture, which is backward compatible, the Power architecture evolves over time, and the tools used to support earlier processors become obsolete and lose support. We knew that whatever Power processor we chose, the tools would eventually become a problem.

We finally considered eliminating the ISC microprocessor altogether and shifting the ISC-2 microprocessor functions to hardware state machines or the IOP. Our experience with ICB-2 and ICB-3 taught us that the level of hardware complexity required in ISC-3 was attainable, and we knew that eliminating the ISC microprocessor had many advantages. First, the product cost is much lower. Second, the development of the microprocessor interface to the rest of the hardware is about the same as moving the mainline functions from code to hardware state machines. Third, the development cost is reduced (unique tools not required). Fourth, the performance is better, since we are thus forced to put all of the performance-critical functions into hardware state machines, and there is no microprocessor code path length. Finally, the tools' end-of-life problems are avoided.

Eliminating the ISC microprocessor required more than simply putting all of the mainline functions into hardware state machines. We also had to dramatically reduce the dependence on a microprocessor for initialization and exception handling. We still had a processor that could control ISC-3, but it was the IOP. As shown in Figure 7, IOPs are the same as normal processors and are physically and logically much further from the ISC than a local microprocessor would be, so the interaction with them must be minimized. Figure 7 shows that the connection between an IOP and an ISC-3 includes a hub chip and two levels of multiplexors. Also, the IOP is shared by many different ISCs and other system functions, so their utilization to support ISC-3 must also be minimized. This led us to develop many new hardware state machines and structures to more efficiently handle the initialization and exceptions.

Link frame handling in hardware

In ISC-1 and ISC-2, limited frame handling was performed by hardware state machines. Improvements in ISC-2 further reduced the main processor and IOP path length, but the link acknowledgment reception and generation for long data transfers is still handled by the ISC-2 microprocessor.

In ISC-3, all mainline frame handling is moved to a new hardware state machine. This hardware provides state information for each area of each buffer set. Within a single buffer set, the hardware coordinates the states of all buffer areas to recognize all possible error sequences. When multiple frames with intervening link acknowledgments are required for long data transfer, the hardware detects the inbound link acknowledgment frame

and automatically begins transmission of the next data area. Similarly, when received data has been properly received, the hardware automatically generates an outbound link acknowledgment instructing the other end of the link to proceed.

To completely define the operation of the hardware state machines, detailed state tables describe the actions during both valid and error sequences of frames. For example, if a frame is received when it is not expected (in the wrong sequence), the hardware interrupts the IOP and gives it precise information about the error.

Priority scheduling

With only two originator and two recipient buffer sets in ISC-2 and ICB-2, queueing frames and packets is a relatively simple process; a first-come-first-served algorithm is sufficient. In ISC-3 and ICB-3, the number of message buffer sets is increased from four to 28. With this much larger number of message buffer sets, the first-come-first-served algorithm becomes much more "unfair," causing some message buffer sets to be starved for service.

We developed a hardware priority scheduling mechanism to recognize the priorities of different kinds of messages presented to the scheduling mechanism, to process the different kinds of messages presented with fairness within a priority, and to guarantee the forward progress of all message buffer sets. Rather than simply creating a very large combinatorial logic tree, a very small sequential state machine was used in order to take advantage of the nature of the traffic to calculate priorities in parallel with frame transmission. Because frame transmission requires multiple cycles, the state machine has multiple cycles to perform the calculations required to determine the next user to transmit on the link or store into memory.

Disable inbound link function

Serial optical communication links experience many different kinds of errors and failure behaviors. At one extreme are the occasional single-bit errors. These errors cause a single bit in the bit stream to be flipped or complemented and are caused by random noise. When these errors occur during an idle sequence, no recovery action is required. When they occur within an information frame, various recovery actions are taken depending on what part of the frame is corrupted. At the other extreme are link failures, which lead to a total loss of communication. These can be caused by anything from a critical component failure to losing power at one end of the link, or to a physical disconnection of the optical fiber transmission medium. Very often, especially when an optical connector somewhere in the operating link is physically disconnected, the link failure condition is preceded by an increasing bit-error rate. In some cases, a link failure condition is detected followed by a re-establishment of the link followed by a link failure condition multiple times, as the optical connector is being slowly disconnected.

In ISC-2, the local processor is interrupted as each link error and failure event is recognized by the inbound decoder state machines. Even when the link error rates are very high, as when an optical connector is being disconnected, the local processor can keep up with the error events, since it is relatively idle during these high-error-rate periods.

With no microprocessor in ISC-3, hardware is provided that interrupts an IOP only when specific error conditions are detected. This hardware also responds on the outbound link with a special continuous sequence signaling the type of failure. The link events for an inbound link are atomically filtered by detecting link failures or the receipt of a continuous sequence and subsequently disabling the inbound link by ignoring all events until re-enabled by the IOP.

The detection of the link failure or continuous sequence event is reported to an IOP with information describing the nature of the report. ISC-3 handles the receipt of special continuous sequences on the inbound link with the proper special continuous sequence response on the outbound link. In both cases, ISC-3 is inhibited from causing subsequent interruptions to the IOP before the IOP performs the appropriate recovery actions. Also, the IOP can arm ISC-3 to cause an interrupt when the link starts to receive a valid sequence.

Link error statistics

In ISC-2, link errors are monitored using interrupts to the local microprocessor, and keeping this approach in ISC-3 would over-utilize the IOP. To address this problem, we developed hardware state machines that do not interrupt the IOP each time a bit error is detected and automatically gather link error statistics with minimal processor (IOP) involvement.

In ISC-2, the first indication of a bit error is a code violation (CV). If the bit error is in a frame, either it may be recognized immediately (CRC error in the information field) or a message timeout condition may be detected. In either case, error-recovery mechanisms are invoked, and the damaged frame (or operation) is retried. Bit errors in either the idle sequence or a continuous sequence (another type of idle sequence) do not damage frames, but they still must be detected and tracked to determine the overall link error performance (bit-error rate). In many lightly utilized links such as ISC, most of the bit errors occur in idle sequences. Previously in ISC-2, these harmless bit errors caused interrupts of the local microprocessor, and this microprocessor collected the bit-error statistics.

In ISC-3, there are two types of bit-error events. The first type is the isolated bit error caused by noise events. The second type is a burst of bit errors, which may indicate that the receiver has lost bit/byte/word synchronism. If synchronism is reestablished within 100 ms, a temporary loss of sync event is recognized. If the loss of sync condition persists for more than 100 ms, a link failure condition is recognized, and the IOP is immediately interrupted to handle the error. In this failure case, there is sufficient time to change the physical connection, and the IOP must bring up the link from scratch, verifying what server is at the other end.

In ISC-3, both of these kinds of errors are counted in a register and start a timer. All subsequent errors increment the counter. The counter does not wrap back to zero when it reaches its maximum value, but stays at the maximum value (saturates). When the timer expires, a single interrupt to the IOP is generated. The IOP can then read the error counter, reset the counter, and re-enable the interrupt. The IOP can program the interrupt time-out.

Time-stamp coordination

Event tracing and logging are important in the debugging of designs, and one of the most useful pieces of information in each entry is a time stamp. Time stamps obviously point out time delays in the system, but they are also very valuable in coordinating multiple trace entries generated by multiple processes in the system.

In ISC-3, trace entries are generated that include a local time stamp; however, the time stamps provided in these trace entries have only a loose relationship to the IOP time-of-day (TOD) register. We needed a way to synchronize this local time stamp with the TOD. In particular, our method for synchronizing the time stamps has a master TOD register in the IOP driven by an oscillator, and same oscillator drives the ISC-3 time register through a timing pulse over the STI link. But this timing pulse has any arbitrary phase with respect to the TOD register. Timing facilities in ISC-3 measure this phase relationship, and the IOP code can access this phase information to coordinate the time-stamp information.

Inbound event buffering

Because the IOP is so far from ISC-3, we needed to develop a very efficient way for the IOP to gather exception information. The frequency of these events is sometimes too high for a remotely located processor such as the IOP to process each one in real time, so a small memory element in the form of a first-in-first-out (FIFO) buffer was added in ISC-3 to queue these events.

To make FIFO access efficient, ISC-3 provides the maximum information to the IOP each time it reads the FIFO buffer. ISC-3 presents different information depending on the state of the FIFO (its fullness), and the state of ISC-3 itself. ISC-3 writes discrete events into the FIFO at a location determined by a write pointer, and the IOP reads the FIFO from a location determined by a read pointer. Reading FIFO entries conditionally returns event and status information and conditionally increments the FIFO read pointer. A fullness indication of the FIFO is returned in the read information as the value of the FIFO read pointer and write pointer. Also, ISC-3 substitutes status when the FIFO is completely empty, and the event description when the FIFO has one or more valid entries.

The ISC-3 FIFO has a mode in which the IOP can read multiple entries of the FIFO using a single command. Once again, the format of the returned data is different from the variable information returned by a single FIFO access.

The ISC-3 FIFO also reduces the number of interrupts presented to the IOP by sharing information on the fullness of the FIFO, as observed by the IOP and known to the FIFO.

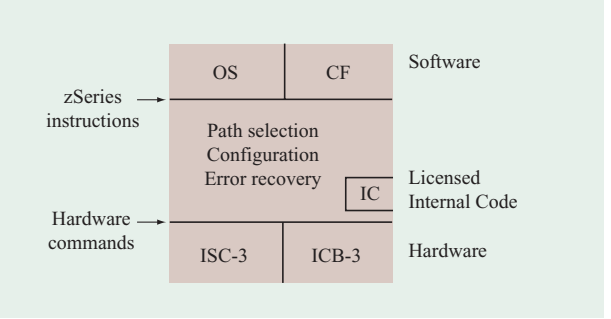

Figure 11

Code functions.

Link initialization

After the functions described above have been transferred to the hardware, the remaining code support for the coupling channels is left to adapt the z/Architecture* programming interface to the hardware control interface, as shown in **Figure 11**. The most interesting function of these is the link initialization part of the configuration function. The primary remaining requirement for code support is in the initialization of the connection. The design of the third-generation links, particularly the introduction of the peer mode, placed some interesting requirements upon the supporting code.

ICB initialization

The initialization of an ICB link consists of determining the identity of the attachment at the remote end of the link. The protocol for the initialization of the ICB link

Table 1 Parallel Sysplex channel comparison.

	ISC-1	*ISC-2*	*ISC-3*	*ICB-2*	*ICB-3*	*IC-2*	*IC-3*
Type	Sender /receiver	Sender /receiver	Peer	Sender /receiver	Peer	Sender /receiver	Peer
Hardware commands	Command set 1	Command set 2	Command set 3	Command set 2	Command set 3	NA	NA
Buffer sets	2+2	2+2	7+7+7+7	2+2	7+7+7+7	2+2	7+7+7+7
Speed	531/1062 Gb	531/1062 Gb	1062/2125 Gb	250 MB	650 MB	700 MB	850 MB
Latency (lock)	100 μs	45 μs	28 μs	18 μs	15 μs	18 μs	15 μs
Distance	10 km @ 1 G	10 km @ 1 G	10 km @ 2 G	10 m	10 m	NA	NA
Code path	SAP/PU intensive	Command set 2	Command set 3	Command set 2	Command set 3	Processor moves data	Processor moves data
Flow	Link ACK on 4K	Link ACK on 4K	Link ACK on 64K	STI flow control	STI flow control	NA	NA
Microprocessor	Yes	Yes	No	No	No	No	No

was fairly well established in the design for the ICB-2 link. However, this protocol was based upon a design in which the code at either end of the link supported different functions—one end was a sender and the other was a receiver. We used this asymmetry to simplify the design by using the sender to drive the protocol. Peer mode has changed this. The link is now symmetric, so both ends drive the protocol, and the link is fully operational only after both ends have driven the sequence to completion.

The first step in the sequence is simply to send a signal, called a ready signal, which indicates that this end of the link is operative. When an ICB-3 channel receives this signal, it knows that the other end of the link is ready to step through the initialization protocol. Generally, only one side of the link detects this signal, since the first side that becomes operative sends it to an inoperative partner, which cannot detect it.

After it has received the ready signal, the ICB-3 support code responds with a signal called a trigger signal. This signal indicates that the ICB is ready to exchange identification information with the remote partner.

When the ICB receives the trigger signal, it sends an MCB that starts a message containing the node identifier for the ICB channel. This comprises the type and model of the server, the serial number of the server, and the channel identifier of the ICB that is initiating the message.

The proper response to any MCB is to send an MRB in order to complete the message. This response contains the node identifier for the channel that is sending the MRB.

This is the end of the sequence. At this point, one end of the link has determined that it can initiate a message to the other end. However, only one end of the link has driven through the sequence, and both ends must do this to make the link ready for communication. Therefore, after it receives the MRB, the ICB sends a trigger signal. This allows the other end of the link to drive an MCB with a node descriptor, so that the sequence can be completed from both ends. (So that this does not continue forever, the trigger signal is ignored if the ICB has already completed the initialization sequence.)

ISC initialization

In addition to identifying the partner at the remote end of the link, ISC link initialization also exchanges some capability information (e.g., the number of buffer sets available for communication) between the ends of the link. In the past, the sender side has driven the messages that exchange that information. This obviously had to be changed for peer mode. When the link is in peer mode, ISC link initialization begins with the same sequences that have been used in ISC-1 and ISC-2. After the node identifiers have been exchanged with an "operational transceiver request/operational transceiver response" frame sequence, the ISC with the higher serial number assumes the role of the dominant ISC and drives the remainder of the link-initialization sequence.

Extensibility

ISC-3 is also designed for longevity. We felt that the number of buffer sets was sufficient to handle the Parallel Sysplex requirements for many years to come. Enough space was allocated in the commands to handle the expected increases in main-memory size and the number of logical partitions. Thus, when the components reach their end-of-life phase, a straightforward technology remap is all that should be required. This expected remap(s) should have only the slightest changes to the code, primarily to handle the new part numbers and expected improvement in packaging density.

In the future, the ISC-3 link speed may also be increased. The ISC-3 design allows remaps to increase the speed of the ASIC to accommodate faster optical transceivers. We believe that we may have to use the next optical transceiver standard (5, 10 Gb/s) to meet future Parallel Sysplex performance requirements.

We also added a few more features we expect to be useful in the future, and these are described below.

Larger command buffers (MCBs)

Many Parallel Sysplex exploiters are running out of space in the 256-byte MCB. ISC-3 (and ICB-3) provide space for a larger MCB. As with ISC-2 data frames, transmission of the new MCB can be suspended on 256-byte boundaries. The start bit protocol used in ISC-2 data frames and the logic required to control it have been added to MCB and MRB transmission.

Interrupts for new features

We added the hardware capability in both ISC-3 and ICB-3 to generate hardware interrupts to the IOP at key points in the message-passing sequences, and we are envisioning several applications. First, the present CF code polls for work. We believe that an interrupt-driven CF implementation could improve performance. Second, interrupts could enable multiple CF images to share receiver facilities. Third, we may want to share the coupling links with new traffic. We are currently evaluating these applications.

Concluding remarks

Table 1 contains a summary of all of the Parallel Sysplex coupling links. For each type of link, several key attributes are shown, all of which have been described in this paper. The latency numbers for lock commands indicate how improvements in the code and hardware path lengths and link speeds yield better performance. As we evolved from ISC-1 to ISC-2 and ICB-2, many important improvements were made to both the functionality and the performance. The development of ISC-3, ICB-3, and IC-3 proved to be more revolutionary. As developers, we try to reuse as much of the previous design as possible, but in this case we had to make a much bigger break with the past. We believe that our resulting coupling-channel design is optimized for both the current and at least several more zSeries generations.

Acknowledgments

The engineering efforts on the design of the z900 coupling channels took several years and required the contributions of many workers in the IBM Poughkeepsie, Boeblingen, Austin, and Endicott laboratories. Special acknowledgment goes to Kulwant Pandey for her expert chip design work and to Ambrose Verdibello, Pat Sugrue, and Steve Burrow for their design and code work. We also thank Albert Ing for his comments regarding this paper, and Dave Elko not only for his comments but also for his extensive suggestions for improving the paper.

References

1. J. M. Nick, B. B. Moore, J.-Y. Chung, and N. S. Bowen, "S/390 Cluster Technology: Parallel Sysplex," *IBM Syst. J.* **36,** No. 2, 172–201 (1997).
2. J. M. Nick, J. Chung, and N. S. Bowen, "Overview of IBM System/390 Parallel Sysplex—a Commercial Parallel Processing System," *Proceedings of the IEEE Symposium on Parallel and Distributed Processing*, Los Alamitos, CA, 1996, pp. 488–495.
3. IBM Corporation, *MVS/ESA Programming: Sysplex Services Guide*, Order No. GC28-1495-02, 1995; available through IBM branch offices. Chapter 6 describes coupling-facility cache structures, Chapter 7 describes list structures, and Chapter 8 describes lock structures.
4. John Cocke and V. Markstein, "The Evolution of RISC Technology at IBM," *IBM J. Res. & Dev.* **34,** 4–11 (January 1990).
5. G. Radin, "The 801 Minicomputer," *SIGARCH Computer Architecture News* **10,** 39–47 (March 1982).
6. T. A. Gregg, K. M. Pandey, and R. K. Errickson, "The Integrated Cluster Bus for the IBM S/390 Parallel Sysplex," *IBM J. Res. & Dev.* **43,** No. 5/6, 795–806 (September/November 1999).
7. C. L. Rao, G. M. King, and B. A. Weiler, "Integrated Cluster Bus Performance for the IBM S/390 Parallel Sysplex," *IBM J. Res. & Dev.* **43,** No. 5/6, 855–862 (September/November 1999).
8. C. L. Rao and C. Taaffe-Hedglin, "Parallel Sysplex Performance," *CMG Trans.* **87,** 3–7 (Winter 1995).
9. T. A. Gregg, "S/390 CMOS Server I/O: The Continuing Evolution," *IBM J. Res. & Dev.* **41,** No. 4/5, 449–462 (July/September 1997).
10. J. M. Hoke, P. W. Bond, R. R. Livolsi, T. C. Lo, F. S. Pidala, and G. Steinbrueck, "Self-Timed Interface of the Input/Output Subsystem of the IBM eServer z900," *IBM J. Res. & Dev.* **46,** No. 4/5, 447–460 (2002, this issue).
11. J. M. Hoke, P. W. Bond, T. Lo, F. S. Pidala, and G. Steinbrueck, "Self-Timed Interface for S/390 I/O Subsystem Interconnection," *IBM J. Res. & Dev.* **43,** No. 5/6, 829–846 (September/November 1999).
12. K. E. Plambeck, W. Eckert, R. R. Rogers, and C. F. Webb, "Development and Attributes of z/Architecture," *IBM J. Res. & Dev.* **46,** No. 4/5, 367–379 (2002, this issue).
13. D. J. Stigliani, Jr., T. E. Bubb, D. F. Casper, J. H. Chin, S. G. Glassen, J. M. Hoke, V. A. Minassian, J. H. Quick, and C. H. Whitehead, "IBM eServer z900 I/O Subsystem," *IBM J. Res. & Dev.* **46,** No. 4/5, 421–445 (2002, this issue).

Received December 4, 2001; accepted for publication March 5, 2002

Thomas A. Gregg *IBM Server Group, 2455 South Road, Poughkeepsie, New York 12601 (tomgregg@us.ibm.com).* Mr. Gregg is a Distinguished Engineer and a member of the IBM Academy of Technology. He received an Sc.B. degree in engineering from Brown University in 1972 and continued under a university fellowship, receiving an Sc.M. degree in electrical engineering in 1974. He joined IBM at Poughkeepsie, New York, in 1973. Mr. Gregg has held various technical positions in the area of I/O subsystem design. He holds numerous patents utilized in IBM ESCON and Parallel Sysplex channel products, and has received many IBM Invention Achievement Awards. For these products, he has also received five IBM Outstanding Innovation Awards and two IBM Corporate Awards.

Richard K. Errickson *IBM Server Group, 2455 South Road, Poughkeepsie, New York 12601 (rerricks@us.ibm.com).* Mr. Errickson received a B.S. degree in computer engineering from Lehigh University, joining IBM in 1985. He is a Senior Engineer, working on the development of microcode for the I/O subsystem for IBM eServer processors.