

Scalable Coherent Interface (SCI)

- ANSI/IEEE Std 1596-1992
- Result of an effort in defining a superbus in 1980s
- Defines:
 - Distributed solution
 - Hardware and protocols providing shared-memory view of buses
 - Transactions to read, write, and lock memory locations
 - Hardware protocol to keep processor caches coherent

Goals

- High performance
 - High throughput, low latency, low CPU overhead
- Scalability
 - Performance, physical distance of node, coherence mechanism, technology, addressing capability
- Coherent memory system
- Interface characteristics
 - Enable compatibility among products of multiple vendors
 - SCI should serve as an *open distributed bus*

Concepts

- Point-to-point links
 - Concurrent data transfers
- Sophisticated signaling technology
 - Unidirectional links without any back-propagating signals
 - Multiple link technologies
- Nodes
 - Whole SMP-nodes, processors with cache, memory nodes, IO nodes
 - Up to 64K
- Topology independence
 - Complex networks can be build
 - Standard anticipates simple topologies, ring, rings of rings, multidimensional tori

Concepts contd.

- Fixed addressing scheme
 - 64 bit addressing scheme
 - Most significant 16 address bits are node ID
 - Remaining 48 bits are used for addressing within a node
 - 32bit nodes require address conversion to 64 bit SCI space
- Hardware-based global address space
 - Distribution is transparent to processors
 - Memory access is mediated to target memory by SCI hardware
 - Integration with the memory architecture of the node
- Bus-like services
 - Transaction to read, write, and lock memory locations
 - Prioritization of transaction for fair protocols
 - Message passing and broadcast functionality

Concepts contd.

- Split transactions
 - All transaction are split into request and response phases
 - Send as packets with transaction ID, addresses, commands, status and data
 - 64 outstanding transactions for high utilization
- Optional cache coherence
 - Cache coherence based on distributed-directory approach
 - Doubly linked sharing list
 - Multiple reader single writer
 - Invalidation-based
 - No memory consistency model
- Reliability in hardware
 - 16-bit CRC based error detection
 - No in-order delivery

Concepts contd.

- Layered specification
 - Physical layer
 - Logical layer
 - Cache coherence layer (optional)
- C-code
 - Major portions are specified via C-code
 - Allowed validation of standard by simulations

Physical layer

- Three initial models
 - Parallel link for short distances (meters) at 1GB/s
 - Serial link for intermediate distances (tens of meters) at 1Gb/s
 - Optical serial link over long distances (kilometers) at 1Gb/s
- Today
 - Systems use specific physical layer implementations
 - Incompatibility with other vendor's systems
 - 500 MB/s
 - Few meters 16-bit parallel links, 125 MHz, CMOS technology
 - 1 GB/s
 - Closely coupled SMP, GaAs link controller
 - 1 GB/s
 - 100 meters, parallel fiber-ribbon cables, BiCMOS technology

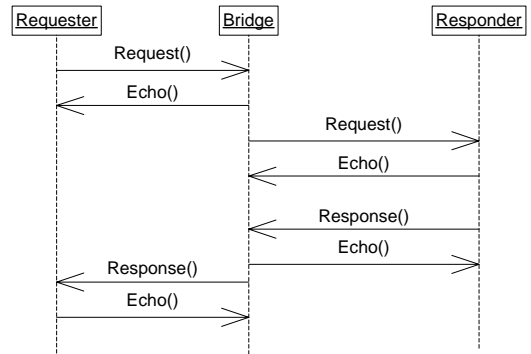
Logical Layer

- Specifies
 - Transaction types and protocols
 - Packet types and format
 - Packet encodings
 - Standard node interface structure
 - Bandwidth and queue allocation protocols
 - Error processing
 - Addressing and initialization issues

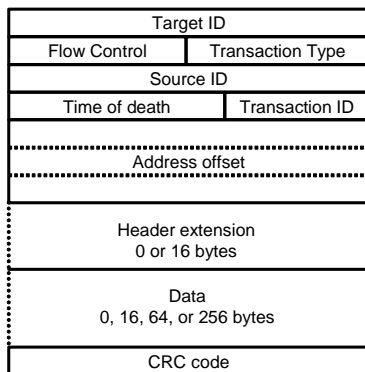
Transactions

- Split transactions
 - Request and response subactions
 - Each subaction consists of
 - Send packet
 - Echo (acknowledgement) packet by receiver confirming acceptance by receiver for further processing
 - Timeout/retransmission of packet by sender
 - No in-order delivery
 - Up to 64 outstanding transactions

Bridges for Complex Topologies



Packet Format



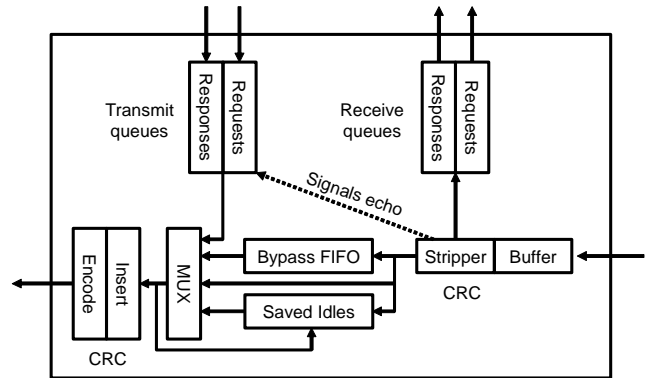
Transaction Types

- Read/Write transactions
 - Coherent and non-coherent
 - Read response delivers data
 - Write response signals success or failure
- Lock transactions
 - Indivisible read-modify-write operation (also compare&swap, fetch&add)
 - Data is carried in both directions (new and old value)
- Move transactions
 - Same as non-coherent write without response subaction
- Event transactions
 - No response, no flow-control
 - Intended for distributing a time stamp

Packet Encoding

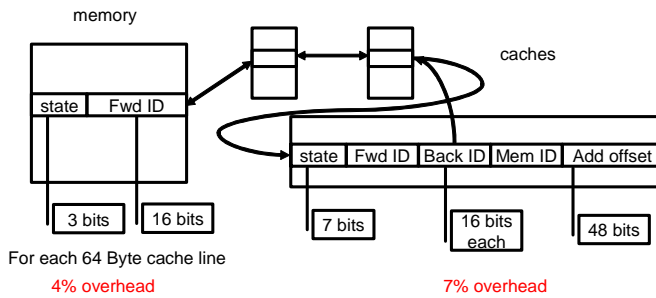
- Packets are transmitted in individual symbols (2 bytes)
 - Plus *clock* signal determining boundary of symbols
 - Plus *flag* signal determining end of packet
- Idle symbols
 - Inserted between packets
 - Allow SCI nodes to permanently synchronize to incoming data
 - Transfer allocation and other control information
 - Idle symbols are created whenever a node takes a packet of the incoming link, and are replaced when a node has to send a packet

SCI Link Control



Cache Coherence Layer

- Implementation of coherence protocols is optional
- Multiple reader, single writer with write invalidation
- Central concept: [distributed sharing lists](#)



Memory States

- Home
 - No cached copy exists
- Fresh
 - Read-only copies may exist and copy in memory is valid
- Gone
 - A writable (exclusive or dirty) copy exists. No valid copy exists on the local node.

Cache States

- 29 stable states and many pending states
- Stable states can be thought of having two parts
 - Position in sharing list
 - ONLY: single node list
 - HEAD, TAIL, MID
 - Actual state
 - Dirty: modified or writable
 - Clean: writable but same as in memory
 - Copy: readable
 - ...

Basic Operations

- List construction
 - adds new sharer to head
- Rollout
 - removes node from list
- Purging (invalidation)
 - head node may invalidate other copies

Reasons for Write Back and Replacement

- Node in sharing might need to delete itself because
 - It must become head of the list to perform write.
 - It must replace data for capacity or conflict reasons.
- Priority scheme to avoid deadlock
 - If two adjacent nodes try to roll out at the same time, they may set themselves to pending state simultaneously.
 - By convention, the node closer to the tail of the list has priority and is rolled out first.
- Negative acknowledgements
 - SCI includes NACKS
 - NACKS do not simply lead to a retry in traditional sense.
 - They indicate inappropriate requests.
 - Requester has to generate a different type of request.

Serialization of Operations to a Given Location

- Home node serializes operations.
 - SCI
 - Home node always accepts new requests.
 - New requests are inserted in pending list.
 - Origin
 - Busy state while protocol is executed.
 - New requests are discarded by sending NACKs
 - Requester just retries after some delay.
- SCI does not specify memory consistency.

Protocol Extensions

- Pairwise sharing
 - For situation where two nodes share a cache block and they ping-pong ownership, both writing repeatedly.
- Scalability of invalidation
 - Latency tends to be larger than in memory protocols.
 - If lists get long, the serialized implementation get very expensive.
 - Several extensions have been proposed, but are not included in the standard.

Applications of SCI

- Memory interconnect for cache-coherent NUMA
- System area network for clusters
- IO subsystem interconnect
- Large-scale data acquisition system

IBM NUMA-Q

- Up to 64 nodes
- Quad nodes
 - Four Intel Pentium III, 700 MHz
 - Split transaction bus
 - Up to 8 GB memory
 - 2 PCI interfaces
- IQ-Link
 - SCI link, 1 GB/s
 - 32 – 256 MB remote cache
- Shared IO system

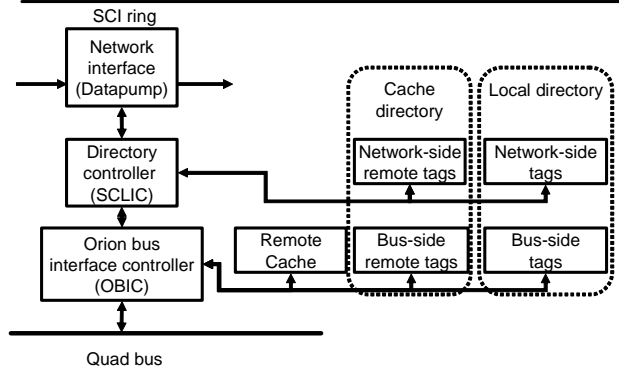
Architecture



Enhancement to SCI Protocol

- Pairwise sharing
- Optimized invalidation scheme
- Less memory overhead by shorter node addresses
- Protocol extension to force update of home location.

NUMA IQ-Link Board



IQ-Link Design

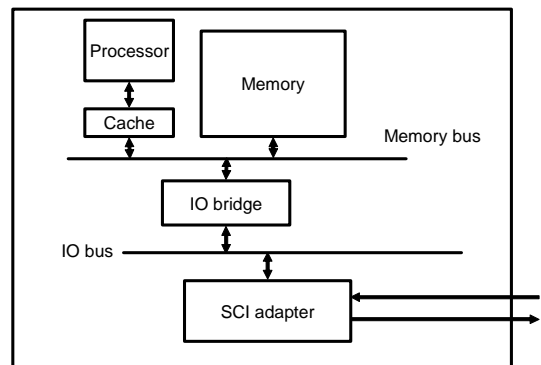
- Plugs into the memory bus
- Orion bus interface controller (OBIC)
 - Interface to quad's memory bus
 - Acts as pseudo memory controller that snoops and translates accesses to remote data.
 - Acts as pseudo processor that puts incoming transaction from the network onto the bus.
- Data pump
 - Gallium arsenide chip for link and packet-level protocol.
- SCI link interface controller (SCLIC)
 - SCI coherence protocol using multiple protocol engines.
- Directory for local data, split among OBIC and SCLIC.
- Remote cache

Interaction with Memory Bus

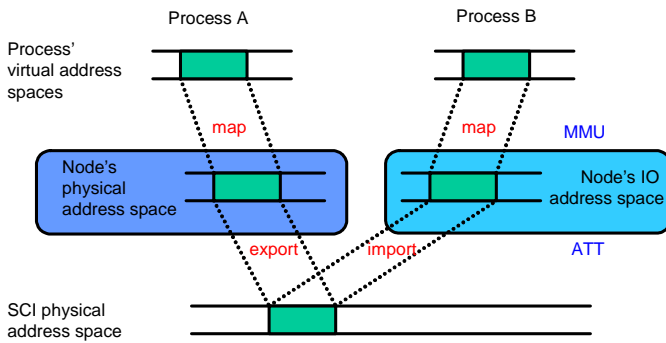
- Read request by processor of quad
 - OBIC snoops request on bus
 - It looks up remote cache and local directory whether request can be satisfied locally.
 - If so, main memory or one of the other caches satisfies the request.
 - Otherwise, OBIC detects that the request must be propagated off node.
- Write request by processor of quad
 - OBIC checks whether local or remote address.
 - SCLIC performs coherence protocol and signals completion on bus.
- If an updated value from another node arrives
 - OBIC has to update memory.

SCI as High-Performance Network for Clusters

SCI-Adapter attached to IO-Bus



Address Mapping in SCI Clusters



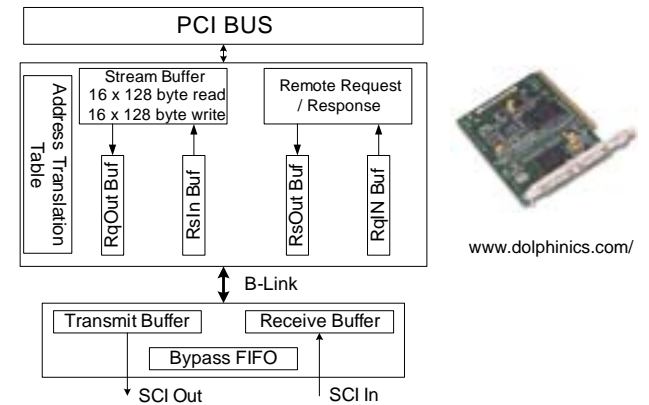
Implementation of Load/Stores

- To local shared memory
 - Load / Stores are executed in the normal way via memory bus.
- To remote shared memory
 - Load / Stores are mapped to PCI addresses
 - At requestor: PCI addresses are mapped to SCI addresses by SCI card
 - At responder: SCI addresses are mapped to physical memory addresses.
- Mapped virtual pages have to be pinned in memory.
- Accesses to shared segments are transparent to remote node and are executed without OS intervention.

Cache Coherence

- Since SCI-card is attached to IO-bus it cannot observe operations on the memory bus.
 - This precludes caching and coherence maintenance for memory regions mapped to SCI address space.
 - This implies that remote addresses are uncacheable.
- Accesses to local shared addresses can be cached.
 - Request from other nodes appear on memory bus and will be handled by local cache coherency protocol.
- Each access to a remote address has to go through the network.

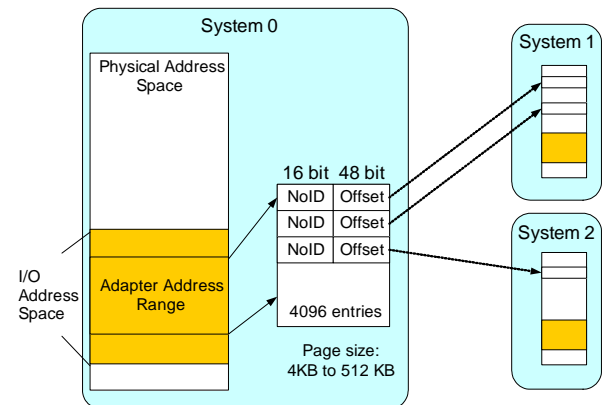
Dolphin's PCI-SCI Bridge



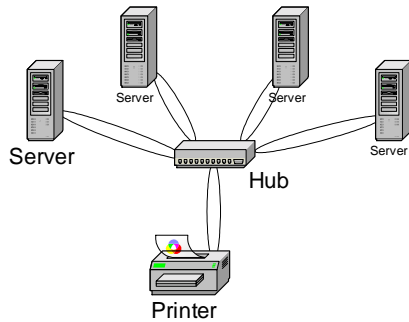
Streams

- Write stream buffers
 - Gather subsequent writes
 - Explicit flushing or if buffer is filled
- Read stream buffers
 - Prefetching of data
 - One read on PCI can result in up to 32 64-byte SCI read ops
 - When a stream read buffer was emptied, a new read request is automatically initiated.

Address Translation 64-bit PCI Card



Interconnection topologies



Wulfkit3

- Three dimensional topology
- Eliminates need for switches



Caching of Remote Addresses in SMiLE SCI-VM

- Put remote addresses in processor caches.
- Usage of write through modus.
 - Word granularity
- Impact on memory coherency
 - Weak memory coherency
 - Special operations to make memory coherent
 - Flush write buffers in processors and on SCI card
 - Invalidate read copies in caches and on SCI card

Page Management in SMiLE SCI-VM

- Goals:
 - Same virtual address space assigned to global SCI address space in all processes.
 - Distribution of virtual pages should be adaptable to the needs of the application (Coarse granularity).
 - Physical pages have to be pinned in memory. But virtual pages should be pageable.
- SCI-VM
 - Provides allocation routine
 - It has to be called by all processors with the same parameters.
 - It allocates virtual memory.
 - The distribution can be influenced via a special parameter.
 - Virtual pages can be paged out but the mapping information has to be adapted in all SCI cards. Exported physical pages are acting as a memory cache.

SCI for Scalable IO

- IO subsystem on Cray T3E
- Two SCI rings (600 MB/s each)
- Clients can send in both directions.
- Hot swap of faulty clients.



Large-Scale Data Acquisition

- Data acquisition system for the Large Hadron Collider currently under development at CERN (see also Datagrid project).
- 2000 data sources and 1000 processors
- Multiple GB/s required bandwidth
- SCI-based prototype demonstrated feasibility.

Summary

- SCI is IEEE standard
- Different implementation are incompatible.
- Standard defines global address space with cache coherency.
- ccNUMA systems implement coherency protocol.
- IO cards for clusters do not support coherency.
- Required software for page management and weak consistency has been developed in the SMiLE project.