Scylla: A Memory Controller with Integrated Protocol Engines for Distributed Shared Memory Support

Andreas Nowatzyk, Gunes Aybay,
Michael Browne, Bill Radke, Sanjay Vishin,

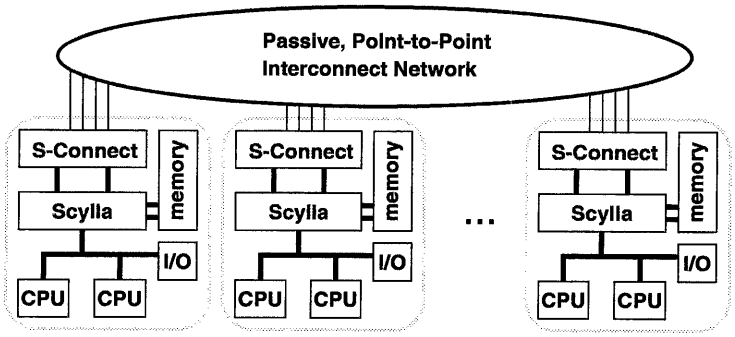
SMCC Technology Development

Sun Microsystems

Outline

- Overview of Distributed Shared Memory and S3.mp
- Where does Scylla fit in the picture?
- Internal Architecture of Scylla
- Advantages of SDRAM based memory subsystem
- Protocol Processors for multiple DSM implementations
- Future Directions

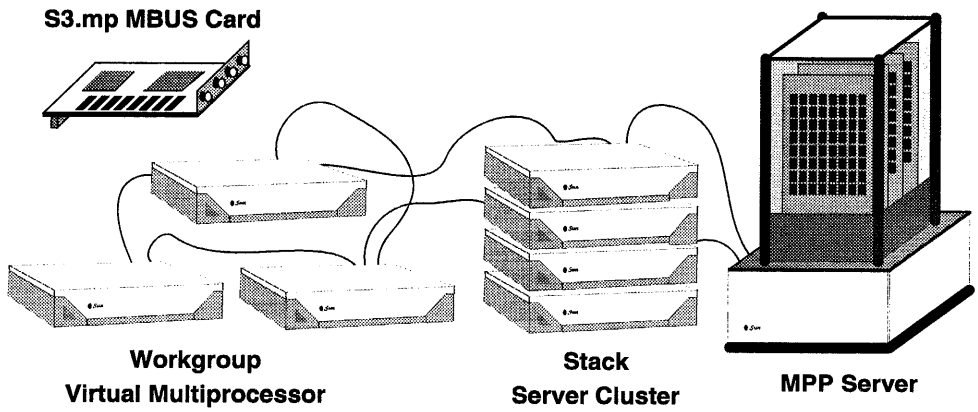
S3.mp: Distributed Shared Memory



- 2 chips per node to support DSM: S-Connect and Scylla
- S-Connect: topology independent & packet-switched
- Scylla: memory control, directory, cache-coherence
- Commodity components for CPU & I/O
- CC-NUMA and S-COMA protocols are being developed

Scylla-HotChips95-3

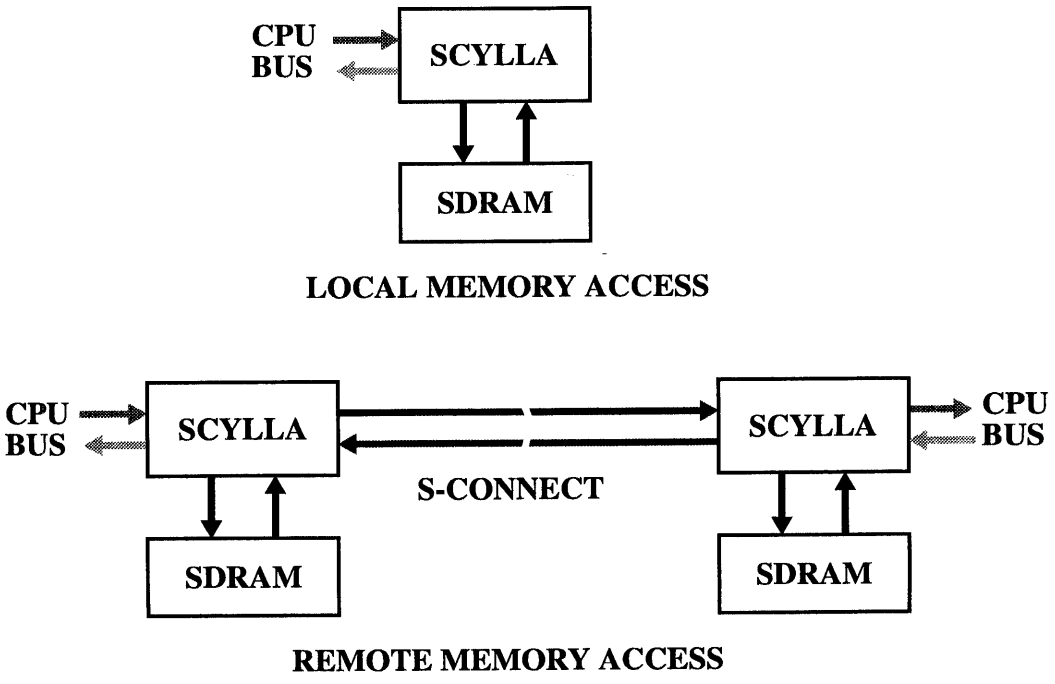
Single Solution, Multiple Configurations



- Most of the DSM system is integrated on Scylla
- Target: 6" x 4" card that can be added to existing systems
- Networks of workstations -> building-wide DSM

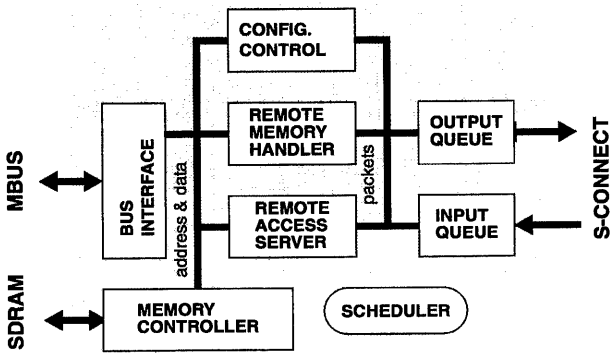
Scylla-HotChips95-4

Scylla Functionality



Scylla-HotChips95-5

Scylla Components

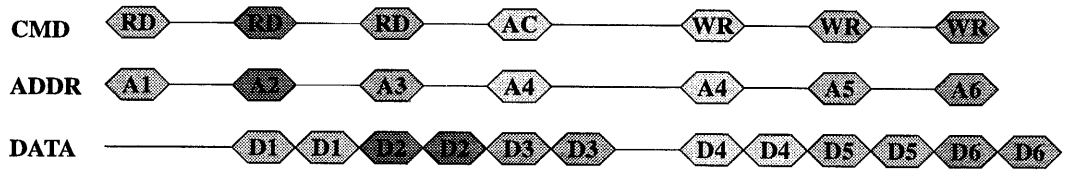


- Bus interface unit: TLB, global address management
- Memory controller: ECC and SDRAM interface
- Protocol engines: cache coherence protocol and block transfers
- I/O queues for interfacing to S-Connect

Scylla-HotChips95-6

Why SDRAM?

- Pipelining is the key for SDRAM utilization
- Can handle multiple outstanding requests



- Schedule transactions like a Tetris game
- At 66MHz > 1Gb/S peak memory bandwidth
- Bandwidth can be close to peak over streams of reads or writes
- 4 banks with 16Mbit chips and 8 banks using 64Mbit chips

Scylla-HotChips95-7

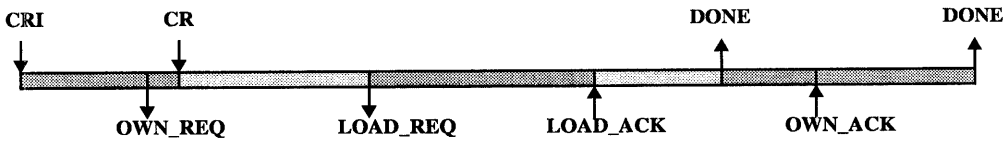
Protocol Processors

- Scylla has two built-in protocol engines: RMH & RAS
- Optimized for running distributed cache coherence protocols
- Protocol instructions are kept in on-chip SRAM
- Protocol can be downloaded from the bus or from the network
- RMH takes care of accessing remote memory on other nodes and maintains a local cache for remote data
- RAS maintains the directory and responds to requests for the portion of the global address space maintained by that node
- Support for fast block transfer in addition to cache coherence
- Mostly CC-NUMA, work in progress for S-COMA support

Scylla-HotChips95-8

Multi-threading in Protocol Engines

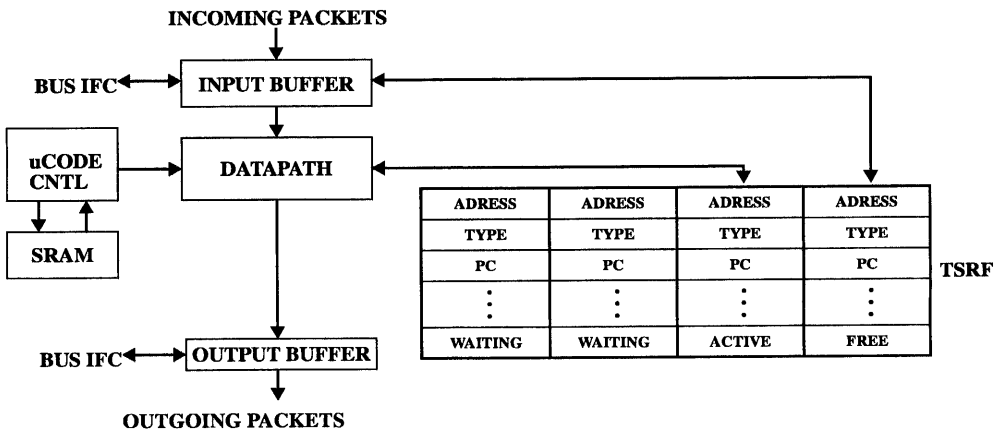
- Scylla's protocol processors are multi-threaded
- Each protocol engine has a Transaction Status Register File
- Every transaction is associated with a TSRF thread. Threads keep running until they execute a RECEIVE instruction. At this point, they are switched to *waiting* state and the protocol engine is free for other threads to run.



- A response that matches a waiting thread changes the state of that thread to *active*. RECEIVE instruction is a multi-way branch. Branch target is evaluated using the opcode of the received response packet.
- Transactions can be completed out of order.

Scylla-HotChips95-9

Internal Structure of The Protocol Processors

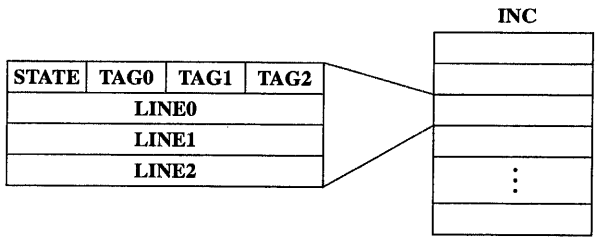


- Current implementation supports 4 threads/protocol processor
- Protocol processors can access memory or the host bus, send and receive packets and generate interrupts.
- Microinstructions can manipulate contents and bit fields of cache lines, TSRF and other global state.

Scylla-HotChips95-10

The Internode Cache

- Up to 32Mb of SDRAM memory controlled by Scylla can be configured as an Internode Cache.



- INC is implemented as a 3-way set-associative cache. Every set consists of 4 contiguous cache lines in memory. The first line is used to store tag information.
- Extra space in INC tags is used to store pointer information for linked list based cache-coherence protocols.
- INC is used as a reverse page translation table in S-COMA operation.

Scylla-HotChips95-11

Scylla Implementation

- Scylla is implemented as a 3.3V 0.5u Gate Array
- Target clock speed is 66MHz.
- Gate count is 190,000, excluding memory and a custom TLB.

Module	Gate Count
Configuration Control	23730
Scheduler	2120
RAS	33691
RMH	33737
Memory Controller	29691
Bus Interface	37315
Input Queue	12597
Output Queue	10470
Miscellaneous	5621

Scylla-HotChips95-12

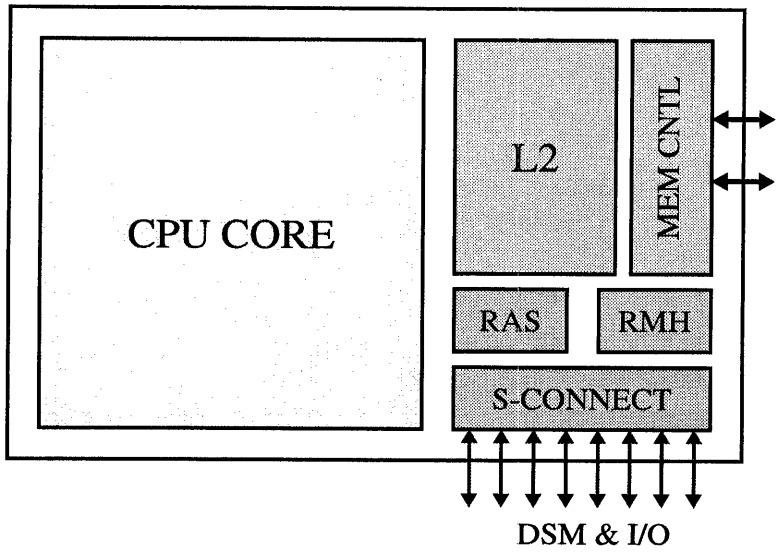
Design Methodology

- Formal verification useful: SMV, Murphi
- High level cache coherence protocol has been verified using Symbolic State Expansion by Fong Pong.
- Clock-accurate C++ model.
- Hybrid system simulation environment: C++ and Verilog
- Extensive random testing and targeted diagnostics.
- Except for the TLB, which is a custom design, Scylla is entirely coded in Verilog and synthesized with Synopsys.
- Layout, routing and scan insertion is done by the ASIC vendor.

Future Directions

- Memory Latency is a performance limiter for real applications.
- Shared bus is a performance limiter for directory based systems
- New generation CPU cores are more superscalar (or VLIW) Speculative execution and prefetching -> need more B/W
- It is possible to build a low cost, low pin count 8/16 bank, high bandwidth memory subsystem using 64Mbit SDRAM chips
- Remote memory accesses and I/O can be handled through high speed serial links, so that most CPU pins can be used for L2/L3 cache and DRAM access.
- Need support from O/S and compiler developers to efficiently support DSM -> prefetching, data allocation

Future CPU?



- Basic building block for wide range of systems
- Minimum possible memory latency for local transactions

Scylla-HotChips95-15

More Information

- WWW: http://playground.sun.com/pub/S3.mp
- Papers: ftp://playground.sun.com/pub/S3.mp

Scylla-HotChips95-16