The Tofu Interconnect D for Supercomputer Fugaku

20th June 2019
Yuichiro Ajima
Fujitsu Limited
Overview of Fujitsu’s Computing Products

- Fujitsu continues to develop general purpose computing products and new domain specific computing devices.

**General Purpose Computing Products**
- Mainframes
- SPARC Servers
- x86 Servers
- Supercomputers
- PC Clusters

**Domain Specific Computing Devices**
- Deep Learning
- Combinatorial Problems
- Quantum Computing
Supercomputer is a technology driver for Fujitsu’s computing products in technologies including packaging and interconnect.
Fujitsu has developed single-socket node and water-cooled supercomputer with 3D-stacked memory.

Fugaku will integrate memory stacks into a CPU package.
Development of Interconnect Technology

- Tofu Interconnect (Tofu1) for K computer
  - 6D mesh/torus network, virtual torus rank mapping, Tofu Barrier
- Tofu2 added new functions; atomic operations, cache injection
- Tofu Interconnect D (TofuD) for Fugaku
  - Increased resources for high-density node configuration
  - Fault-resilience – dynamic packet slicing
Features of the Tofu interconnect family

- 6D Mesh/Torus Network
- Virtual 3D-Torus Rank-mapping
- Tofu Barrier
- Characteristics of Torus Network
6D Mesh/Torus Network

- Six coordinate axes: X, Y, Z, A, B, C
  - X, Y, Z: the size varies according to the system configuration
  - A, B, C: the size is fixed to $2 \times 3 \times 2$

- Tofu stands for “torus fusion”: $(X, Y, Z) \times (A, B, C)$
Virtual 3D-Torus Rank-mapping

- A rank-mapping option for topology-awareness
- A 3D torus rank can be mapped to a 6D submesh even if there is an offline node
- This fault tolerance contributes to the system availability
Tofu Barrier

- Tofu Barrier offloads Barrier and Allreduce communications
  - Barrier channel (BCH) is an interface
  - Barrier gate (BG) is a communication engine

- Tofu barrier can execute an arbitrary communication algorithm
  - Recursive-doubling algorithm uses \(\log_2(n)\) of BGs in each process
  - Reduce-broadcast algorithm uses a maximum of 5 BGs in each process
Characteristics of Torus Network

- All systems have the same order of bisection bandwidth
  - No significant performance difference in global data exchange
- Torus networks have higher total injection bandwidth
  - Topology-aware communication such as nearest-neighbor data exchange results in higher performance

<table>
<thead>
<tr>
<th>System</th>
<th>Network</th>
<th>Total Injection Bandwidth (PB/s)</th>
<th>Bisection Bandwidth (TB/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blue Gene/Q</td>
<td>Torus (5D)</td>
<td>1.97</td>
<td>49</td>
</tr>
<tr>
<td>K Computer</td>
<td>Mesh/Torus (6D) Virtual Torus (3D)</td>
<td>1.66</td>
<td>36</td>
</tr>
<tr>
<td>Sunway TaihuLight</td>
<td>Tapered Fat-Tree</td>
<td>0.51</td>
<td>70</td>
</tr>
<tr>
<td>Piz Daint</td>
<td>Dragonfly</td>
<td>0.07</td>
<td>36</td>
</tr>
<tr>
<td>Summit</td>
<td>Fat-Tree</td>
<td>0.12</td>
<td>115</td>
</tr>
<tr>
<td>Oakforest-PACS</td>
<td>Fat-Tree</td>
<td>0.10</td>
<td>102</td>
</tr>
</tbody>
</table>
The Design of TofuD

- High Density Node Configuration
- Link Configuration and Injection Bandwidth
- Packaging
- Dynamic Packet Slicing
- Increased Tofu Barrier Resources
High Density Node Configuration

- The processor die size gets smaller from FX100
- The off-chip channels are halved
  - # memory stacks: 8 to 4
  - # high speed serial lanes for Tofu: 40 to 20
- The area of Tofu interconnect shrinks to about 1/3 size
More resources are integrated into the CPU

- # CPU Memory Groups (NUMA nodes): 2 to 4
  - The expected number of processes per node is also doubled

- # Tofu Network Interfaces: 4 to 6
  - Provide more resources and accelerate collective communications
Data transfer rate increased from 25 Gbps to 28 Gbps
Link bandwidth reduced from 12.5 GB/s to 6.8 GB/s
TofuD simultaneously transmits in 6 directions
  - Increased from 4 directions in the case of Tofu1 and Tofu2
Total injection bandwidth per node is 40.8 GB/s
  - Approximately, twice that of Tofu1 or 80% that of Tofu2

<table>
<thead>
<tr>
<th></th>
<th>Tofu1</th>
<th>Tofu2</th>
<th>TofuD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data rate (Gbps)</td>
<td>6.25</td>
<td>25.78125</td>
<td>28.05</td>
</tr>
<tr>
<td>Number of signal lanes per link</td>
<td>8</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>Link bandwidth (GB/s)</td>
<td>5.0</td>
<td>12.5</td>
<td>6.8</td>
</tr>
<tr>
<td>Number of TNIs per node</td>
<td>4</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>Injection bandwidth per node (GB/s)</td>
<td>20</td>
<td>50</td>
<td>40.8</td>
</tr>
</tbody>
</table>
Packaging – CPU Memory Unit (CMU)

- Two CPUs connected with C-axis
  - \(X \times Y \times Z \times A \times B \times C = 1 \times 1 \times 1 \times 1 \times 1 \times 2\)

- Two or three active optical cable (AOC) cages on the board
  - Each cable bundles two lanes of signals from each of the two CPUs
Packaging – Rack Structure

**Rack**
- 8 shelves
- 192 CMUs (384 CPUs)

**Shelf**
- 24 CMUs (48 CPUs)
- $X \times Y \times Z \times A \times B \times C = 1 \times 1 \times 4 \times 2 \times 3 \times 2$

**Top or bottom half of rack**
- 4 shelves
- $X \times Y \times Z \times A \times B \times C = 2 \times 2 \times 4 \times 2 \times 3 \times 2$
Dynamic Packet Slicing – Split Mode

- The physical layer of TofuD is independent for each lane
  - In the ordinary multi-lane transmission, the physical layer has media-independent interface and hides the number of signal lanes
- A packet is sliced and each is injected into a different lane
  - The routing header of the packet is copied to both slices for virtual cut-through packet transfer
- This is normal operation and is called split mode
Dynamic Packet Slicing – Duplicate Mode

- When the error rate is high, the operation mode falls down to duplicate mode that duplicates packets.
- If the error rate returns to low, the link can return to split mode.
  - Each lane is never disconnected independently.
  - The error rates of both lanes are continuously monitored and fed back.
Increased Tofu Barrier Resources

<table>
<thead>
<tr>
<th></th>
<th>Tofu1/2</th>
<th>TofuD</th>
</tr>
</thead>
<tbody>
<tr>
<td>TNI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of BCHs</td>
<td>8</td>
<td>16</td>
</tr>
<tr>
<td>Number of BGs</td>
<td>64</td>
<td>48</td>
</tr>
<tr>
<td>Node</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of TNI w/ Tofu Barrier</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>Number of BCHs</td>
<td>8</td>
<td>96</td>
</tr>
<tr>
<td>Number of BGs</td>
<td>64</td>
<td>288</td>
</tr>
</tbody>
</table>

- The number of Tofu Barrier resources significantly increased
  - All 6 TNIs of TofuD have Tofu barrier
  - Only TNI #0 of Tofu1/2 has Tofu barrier
- This change intended to support intra-node synchronization
Performance Evaluations

- Put Latencies
- Latency Breakdown
- Injection Rates
- Tofu Barrier
Put Latencies

- 8B Put transfer between nodes on the same board
  - The low-latency features were used

<table>
<thead>
<tr>
<th>Communication settings</th>
<th>Latency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tofu1 Descriptor on main memory</td>
<td>1.15 μs</td>
</tr>
<tr>
<td>Direct Descriptor</td>
<td>0.91 μs</td>
</tr>
<tr>
<td>Tofu2 Cache injection OFF</td>
<td>0.87 μs</td>
</tr>
<tr>
<td>Cache injection ON</td>
<td>0.71 μs</td>
</tr>
<tr>
<td>TofuD To/From far CMGs</td>
<td>0.54 μs</td>
</tr>
<tr>
<td>To/From near CMGs</td>
<td>0.49 μs</td>
</tr>
</tbody>
</table>

- Tofu2 reduced the Put latency by 0.20 μs from that of Tofu1
  - The cache injection feature contributed to this reduction
- TofuD reduced the Put latency by 0.22 μs from that of Tofu2
The overhead increase in Tofu2 has been reduced
Injection Rates per Node

- Simultaneous Put transfers to the nearest-neighbor nodes
  - Tofu1 and Tofu2 used 4 TNIs, and TofuD used 6 TNIs

<table>
<thead>
<tr>
<th></th>
<th>Injection rate</th>
<th>Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tofu1 (K)</td>
<td>15.0 GB/s</td>
<td>77 %</td>
</tr>
<tr>
<td>Tofu1 (FX10)</td>
<td>17.6 GB/s</td>
<td>88 %</td>
</tr>
<tr>
<td>Tofu2</td>
<td>45.8 GB/s</td>
<td>92 %</td>
</tr>
<tr>
<td>TofuD</td>
<td>38.1 GB/s</td>
<td>93 %</td>
</tr>
</tbody>
</table>

- The efficiencies of Tofu1 were lower than 90%
  - Because of a bottleneck in the bus that connects CPU and ICC
- The efficiencies of Tofu2 and TofuD exceeded 90%
  - Integration into the processor chip removed the bottleneck
Tofu Barrier – Intra-Node Synchronization

The test program synchronized multiple BCHs in a node

- For 8 and 16 BCHs, some TNIs are shared by multiple BCHs
- Sharing TNI causes the serialization of BCHs/BGs processing

<table>
<thead>
<tr>
<th>Number of BCHs</th>
<th>1</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>48</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of used TNIs</td>
<td>1</td>
<td>4</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>Number of communication stages</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>Max. number of BCHs per TNI</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>Max. number of BGs per TNI</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>9</td>
<td>24</td>
</tr>
</tbody>
</table>

Two delay estimation approaches

- One approach only considers the number of communication stages and the delays of BCH (0.48 μs) and BG (0.13 μs)
- The other approach considers the serialization of BCH and BG processing using the same TNI
Tofu Barrier – Results

- The simple estimate results were too low
- The estimates considering the serialization of BCHs/BGs processing were consistent with the evaluation results
- MPI needs to assign BCHs in the same synchronization group to different TNIs to avoid serialization penalties
Summary

- Supercomputer is a technology driver for Fujitsu’s computing products in technologies including packaging and interconnect.

- TofuD is developed for Fugaku and designed for high-density node configuration and fault-resilience.

- The design of TofuD:
  - Node configuration, injection rate and packaging
  - Dynamic Packet Slicing
  - Increased Tofu Barrier Resources

- The evaluation results of TofuD:
  - Latency was 0.49 μs, which was reduced by 0.22 μs from that of Tofu2
  - Injection rate was 38.1 GB/s and the link efficiency exceeds 90%
  - The evaluation results of Tofu Barrier showed the serialization penalties which a proper MPI implementation can avoid