OMNI-PATH FABRIC TOPOLOGIES AND ROUTING

Renae Weber, Software Architect

Intel Corporation

March 30, 2017
LEGAL DISCLAIMERS

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL’S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

A "Mission Critical Application" is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL’S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS’ FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined". Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The cost reduction scenarios described in this document are intended to enable you to get a better understanding of how the purchase of a given Intel product, combined with a number of situation-specific variables, might affect your future cost and savings. Circumstances will vary and there may be unaccounted-for costs related to the use and deployment of a given product. Nothing in this document should be interpreted as either a promise of or contract for a given level of costs.

Results have been estimated or simulated using internal Intel analysis or architecture simulation or modeling, and provided to you for informational purposes. Any differences in your system hardware, software or configuration may affect your actual performance.

Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families: Go to: Learn About Intel® Processor Numbers

All products, computer systems, dates and figures specified are preliminary based on current expectations, and are subject to change without notice.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or go to: http://www.intel.com/design/literature.htm

The High-Performance Linpack (HPL) benchmark is used in the Intel® FastFabrics toolset included in the Intel® Fabric Suite. The HPL product includes software developed at the University of Tennessee, Knoxville, Innovative Computing Libraries.

Intel, Intel Xeon, Intel Xeon Phi™ are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries.
AGENDA

- Omni-Path Routing Algorithms & Constructs
- Omni-Path Fabric Topologies
- Omni-Path Routing Algorithm Extensions
Fabric Topologies and Routing

- Routing Algorithms supported by Intel® Omni-Path FM
  - Shortest Path
  - Device Group Routing
  - Up/down Fat Tree
  - DOR Mesh/Torus
  - Enhanced Hypercube
The following Routing Algorithms are supported:

- **shortestpath** - pick shortest path and balance lids on ISLs
- **dgshortestpath** - A variation of shortestpath that uses the RoutingOrder parameter to control the order in which switch egress ports are assigned to LIDs being routed through the fabric. This can provide a better balance of traffic through fabrics with multiple types of end nodes.
- **fattree** - A variation of shortestpath with better balancing and improved SM performance on fat tree-like fabrics.
- **dor** - Dimension ordered routing for use with mesh and torus topologies cabled in port-to-dimension order.
- **hypercube** - A variation of dg shortest path for enhanced hypercube and mesh. (Supported via 3rd parties)

```
<RoutingAlgorithm>shortestpath</RoutingAlgorithm>
```
Fabric Topologies and Routing

- Balance between the following
  - Performance
  - Avoidance of deadlocks due to credit loops
  - Resiliency to fabric disruptions
    - Minimal disruption
  - Fabric bring up time
OMNI-PATH DEVICE GROUP ROUTING
Patterns in physical fabric layout → patterns in traffic routing → congestion

- Routing all Storage traffic through single switch
  - Single point of failure
  - Compromises QoS
    - VL resources per device type
    - All storage using single ISL
Device Group Routing

- Define device groups used to specify routing order (up to 8 groups)
  - Example: compute nodes, storage nodes
- Balance traffic across these groups
- Provides better static load balancing
  - Leverages QoS
Configuring Device Group Routing

```xml
<DGShortestPathTopology>
    <!-- RoutingOrder lists the device groups in the order they should be handled. Each device group must have been declared in the DeviceGroups section. -->
    <RoutingOrder>
        <DeviceGroup>Compute</DeviceGroup>
        <DeviceGroup>Storage</DeviceGroup>
    </RoutingOrder>
</DGShortestPathTopology>
```
● Sample device group constructs

```
<DeviceGroup>
   <Name>SampleDeviceGroup</Name>
   <SystemImageGUID>0x123567812345678</SystemImageGUID>
   <NodeGUID>0x123567812345678</NodeGUID>
   <PortGUID>0x123567812345678</PortGUID>
   <NodeDesc>Some Name</NodeDesc>
   <IncludeGroup>AllEndNodes</IncludeGroup>
</DeviceGroup>
```

● Wildcard on Node Description

```
<DeviceGroup>
   <Name>Rack1DG</Name>
   <NodeDesc>*Rack1*</NodeDesc>
</DeviceGroup>
```
ANATOMY OF A CREDIT LOOP
FAT TREE CREDIT LOOP

Tier 0
- SW0
- SW1
- SW2
- SW3

Tier 1

Tier 2
**Credit loop**
- Due to use of path from SW0 to SW2 using intermediate SW1

**Spine First Routing**
- If both up and down routes exist, always route up
  - Deadlock avoidance
- Director Class Switch
  - System Image GUID is used to determine up link
  - If two equal cost paths exist, select next hop with the same system image GUID
**FAT TREE ROUTING**

- **Omni-Path Fat Tree Routing Algorithm**
  - Provides up/down routing for deadlock avoidance
    - Extends spine first routing
    - Identifies direction of links in fat tree topologies
      - If both up & down routes exist, chooses up links
  - Discovery walks fat tree to determine
    - Uplink ports
    - Downlink ports
    - Trunk groups
  - Uses topology information calculated during discovery
    - Balances routes across fabric and between each switch in sub-cluster
Well-formed fat tree, all hosts at lowest tier.

<FatTreeTopology>

<!-- The number of tiers in the fat tree -->
<TierCount>3</TierCount>

<!-- Indicates whether or not the HFIs are on the same tier in the fat tree. -->
<FIsOnSameTier>1</FIsOnSameTier>

<!-- The following must be setup if the HFIs are not on the same tier. It identifies the root/core switches in the fat tree. -->
/CoreSwitches>CoreDeviceGroup</CoreSwitches>

<!-- Nodes may be specified for exclusion from initial round-robin to give better route balancing of remaining nodes. -->
<RouteLast>hfi_device_group</RouteLast>

</FatTreeTopology>
Asymmetric fat tree, hosts at core or on different tiers

```xml
<FatTreeTopology>
  <!-- The number of tiers in the fat tree -->
  <TierCount>3</TierCount>
  <!-- Indicates whether or not the HFIs are on the same tier in the fat tree. -->
  <FIsOnSameTier>0</FIsOnSameTier>
  <!-- The following must be setup if the HFIs are not on the same tier. -->
  <!-- It identifies the root/core switches in the fat tree. -->
  <CoreSwitches>CoreDeviceGroup</CoreSwitches>
  <!-- Nodes may be specified for exclusion from initial round-robin -->
  <!-- to give better route balancing of remaining nodes. -->
  <!-- This may be useful in asymmetric fat trees or to initially balance across compute nodes in the tree. -->
  <!-- <RouteLast>hfi_device_group</RouteLast> -->
</FatTreeTopology>
```
Specify core switch group:

- Wildcard on Node Description
  
  \[
  \text{<DeviceGroup>}
  \text{<Name>CoreSwitchGroup</Name>}
  \text{<NodeDesc>*Core*</NodeDesc>}
  \text{<DeviceGroup>}
  \]

- Explicit NodeGUID
  
  \[
  \text{<DeviceGroup>}
  \text{<Name>CoreSwitchGroup</Name>}
  \text{<NodeGUID>0x123567812345678</NodeGUID>}
  \text{<DeviceGroup>}
  \]

  \[
  \text{<NodeGUID>0x123567887654321</NodeGUID>}
  \]

**Dimension Ordered Routing with Dateline**

- Routing order is by dimension
- Dateline
  - Used for torus deadlock freedom
  - Credit loop free
  - Increment SC/VL when crossing dateline in given dimension
  - Requires 2 VLs

**Balanced routing**

- Balanced by locality (switch to switch) and across fabric
DOR ROUTING DISRUPTION HANDLING

**Disruption Handling**
- Additional set of VLs used for disruption handling
  - Escape VLs
    - Jump to new set on illegal turn to lower dimension
    - Stays on new set for remainder of path
  - Accomplished via SC2SC mapping
    - On an illegal turn: \( SC' \rightarrow SC_{i+2} \)
    - *No SL or path record changes required*
DOR ROUTING HANDLING MULTIPLE FAILURES

**Disruption Handling**

- Routing close to failure causes multiple bad turns
  - Would require another set of VLs
  - Credit loop avoidance
Fault Region

- Avoids multiple bad turns
  - Credit loop avoidance
- Switch within a fault region
  - Route to them
  - Route legal DOR paths through them
  - Don’t route through on broken path
  - Avoid multiple illegal turns
OMNI-PATH FAST FABRIC TOOLS

- opareport --o validatevlcreditloops

  Getting All Node Records...
  Done Getting All Node Records
  Done Getting All Link Records
  Done Getting All Cable Info Records
  Done Getting All SM Info Records
  Done Getting vFabric Records
  Getting All FDB Tables...
  Done Getting All FDB Tables
  Getting All Port VL Tables...
  Done Getting All Port VL Tables
  Validate Credit Loop Routes
  Done Building All Routes
  Fabric summary: 459 devices, 408 HFIs, 51 switches,
  14128 connections, 186864 routing decisions,
  166056 analyzed routes, 0 incomplete routes
  Done Building Graphical Layout of All Routes
  Routes are deadlock free (No credit loops detected)
  Done Deallocating All Vertices
**Enhanced Hypercube**

- Hypercube and mesh routing
- Configure port order and cost to control dimension order routing
- Balanced by locality (switch to switch) and across fabric

```
<HypercubeTopology>
  <EnhancedRoutingCtrl>
    <Switches>SwitchDeviceGroup1</Switches>
    <PortData>
      <pPort>25</pPort>
      <vPort>1</vPort>
      <Cost>10</Cost>
    </PortData>
    <PortData>
      <pPort>28</pPort>
      <vPort>2</vPort>
      <Cost>11</Cost>
    </PortData>
  </EnhancedRoutingCtrl>
</HypercubeTopology>
```
Adaptive Routing

- Supported by all Omni-Path Routing Algorithms
- Medium grained
  - Fabric Manager provides alternate paths to switch FW
  - Switch FW makes changes in event of congestion
  - Uses alternate routes to quickly respond to failures
  - Redundant ISL failover at switch
Routing Module Extensions

- Used to add new routing algorithms to the Omni-Path FM
- Function pointers, wrappers, and structures that store behaviors for a specific topology
- Decoupling of discovery processing, routing, and QoS logic for specific topologies from main line code

`topop->routingModule->funcs.<function>`
13th ANNUAL WORKSHOP 2017

THANK YOU

Renae Weber
Intel Corporation