Cray XC System Node Level Diagnosability

Jeffrey J. Schutkoske
Platform Services Group (PSG)
Cray, Inc.
St. Paul, MN, USA
jjs@cray.com

***Abstract*— Cray XC System node level diagnosability is not just about diagnostics. Diagnostics are just one aspect of the tool chain that includes BIOS, user commands, power and thermal data, and event logs. There are component level tests that are used to checkout the individual components, but quite often issues do not appear until full scale is reached. From experience over the last few years, we have seen that no single tool or diagnostic can be used to identify problems, but rather multiple tools and multiple sources of data must be analyzed to provide proper identification, isolation, and notification of hardware and software problems. This paper provides detailed examples using the existing tool chain to diagnose node faults within the Cray XC system.**

Keywords-Cray XC, diagnostic, diagnosability, computer architecture

I. INTRODUCTION

Cray XC System node level diagnosability is not just about diagnostics. Diagnostics are just one aspect of the tool chain that includes BIOS, user commands, power and thermal data, and event logs. There are component level tests that are used to checkout the individual components, but quite often issues do not appear until full scale is reached. From experience over the last few years, we have seen that no single tool or diagnostic can be used to identify problems, but rather multiple tools and multiple sources of data must be analyzed to provide proper identification, isolation, and notification of hardware and software problems.

The diagnosability of a Cray XC system centers on the compute node. There are various configurations for a compute node including a dual socket Intel processor (Intel SandyBridge, IvyBridge or Haswell), single socket Intel processor with an Nvidia GPU (Kepler or Atlas), and single socket Intel processor with an Intel Xeon PHI co-processor Knights Corner (KNC). The compute node is connected to the Aries High Speed Network using a PCIe link.

There are a number of tools used to diagnose a faulty node. Initially BIOS is used to verify that the node has powered on, initialized, and all ports are trained successfully. Any errors or warnings are reported to the Hardware Supervisory System (HSS).

The HSS system management platform supports a variety of hardware platforms with varying management and control capabilities. For this reason, HSS system management provides abstracted interfaces for important hardware access operations and allow the development of customized drivers, where necessary, which interface directly with the hardware. There are a number of basic infrastructure capabilities that enable higher-level functions within the HSS system management environment.

HSS provides a highly scalable management infrastructure allowing a system from a single blade to thousands of blades to be controlled and monitored from a single system management node. It features a relatively light footprint in terms of hardware and software, and performs monitoring and management out-of-band, so impact on user applications is minimal, with no significant jitter introduced on the compute node. It also uses standard Linux images, with some additional drivers where needed, and a standard Gigabit Ethernet network.

The Cray Linux Environment (CLE) is booted after the node is successfully initialized by BIOS. CLE utilizes the Resiliency Communication Agent (RCA) Interface to communicate between HSS and nodes. Cray has implemented a Hardware Error Log channel using the RCA interface to send hardware errors to HSS.

System on-line diagnostics are utilized to validate the various components individually or concurrently in the system under CLE. The on-line diagnostic tests can be executed from the Cray login node in interactive mode or scripted in batch mode.

HSS monitors a number of power and thermal sensors within the Cray XC system at the blade and cabinet level. These sensors and devices are managed via the out-of-band paths on the blade by HSS.

HSS utilizes a number of Intel Reliability, Availability, and Service (RAS) features including Platform Environment Control Interface (PECI) and In-Target Probe (ITP). It also utilizes the Nvidia SMBus Post-Box Interface (SMBPBI).

The remainder of this paper describes the diagnosability tool chain with detailed examples of hardware failures and how to diagnosis them in the areas as follows:

- Node Initialization Errors
- Node Hardware Errors
- Node Performance
- Node / Aries Performance
- Node / GPU Performance
- Node / Co-Processor Performance
- HSS Out-Of-Band Diagnosis
- HSS Out-Of-Band Debug

This paper reviews system diagnostics in Section II and how to launch the on-line diagnostic tools in Section III. Sections IV and V describe diagnosing initialization and hardware errors. Performance related diagnosis is discussed in Sections VI – X. Out-of-Band (OOB) diagnosis and debug are handled in Sections XI and XII.

II. System Diagnostics

System Diagnostics are used to validate the various components individually or concurrently in the system. The system diagnostics focus testing in a number of areas as follows:

- **Boot**: Boot level diagnostics perform tests at boot time or under special diagnostic kernel images.
- **Confidence**: Confidence level diagnostics perform tests that validate the functionality of the component.
- **Stress**: Stress level diagnostics perform tests that are designed to maximize stress on the hardware. These tests are deliberately intense and thorough to determine the stability of a given system or component.
- **Performance**: Performance level diagnostics perform tests that can measure and calculate the performance of a given component. These tests are also configured to have an expected performance level for the given component to compare actual results against.
- **Workload**: Workload level diagnostics are tests that either simulate a generic application workload, benchmark or are actual applications used to verify that they system is ready to execute user applications.
- **Reporting**: Error reporting of warnings, faults, and errors, as well as, thermal, power, and status data from all devices into a central location on the SMW allows for better system analysis of problems.

III. On-line Diagnostic Execution

The node and network on-line diagnostics are installed with CLE and are located in /opt/cray/diag/default. The on-line diagnostics can be executed from the CLE login node via interactive mode or batch mode. The compute nodes are accessed by submitting jobs through the batch management system (typically PBS or TORQUE Resource Manager). They generally have access to a high-performance parallel file system (typically Lustre) and are dedicated resources for the duration of the batch reservation. The examples in this paper were executed interactively using Application Level Placement Scheduler (ALPS).

To execute an Aries Fast Memory Access (FMA) diagnostic test on NIDs 28 and 29 with 2 ranks per node as follows:

```
aprun -n 2 -N 1 -L 28,29 ./xtfma_ata –R 2
```

In this example the same copy of the diagnostic is placed on each node. If multiple ranks per node are selected, a thread is started for each rank on the node. The rank id is set sequentially by ALPS, starting with the first node. The diagnostic allocates a buffer for the diagnostic and a buffer for each additional node and rank under test. For performance considerations, the diagnostic statically allocates the required buffer space to support the number of cores & ranks selected. Available memory per node limits the number of Network ASICs that can be tested at any given time.

The Cray GPU on-line diagnostics are installed in /opt/cray/cray-nvidia/default. The Cray KNC on-line diagnostics are installed in /opt/cray/cray-intel/default.

IV. Node Initialization Errors

Cray provides BIOS to initialize the Intel processor and memory. The Cray BIOS also initializes, trains and reports on various hardware components as follows:

- QPI bus
- Memory DIMM
- Aries PCIe bus
- Co-Processor PCIe bus
- PCIe device bus

The Cray BIOS reports any DIMM failures during memory training including any MCA errors that were detected. Memory size and speed are verified and reported. It also reports all QPI and PCIe link width, speed, and status.

The Cray BIOS logs each device that is detected, bus (B), device (D), and function (F), as well as, the PCIe width and speed.

```
Aries (B0:D2:F0) completed all PCIe Gen3 Phases successfully,
LNKSTS2=0x1f

   PciBus: Aries NIC detected @ B1|D0|F0, RevId=0x10
   PciBus: Aries NIC detected @ B1|D0|F1, RevId=0x10
   PciBus: Aries NIC detected @ B1|D0|F2, RevId=0x10
   PciBus: Aries NIC detected @ B1|D0|F3, RevId=0x10

   PciBus: Aries NIC [B1|D0|F0] initialized, Width: x16, CurSpd: 8.0
Gbps
   PciBus: Aries NIC [B1|D0|F1] initialized, Width: x16, CurSpd: 8.0
Gbps
   PciBus: Aries NIC [B1|D0|F2] initialized, Width: x16, CurSpd: 8.0
Gbps
   PciBus: Aries NIC [B1|D0|F3] initialized, Width: x16, CurSpd: 8.0
Gbps

   Wait4Boot: Boot Protocol version 1
```

Figure 1. BIOS Successful Output

BIOS detected errors or failures are logged to the BIOS logs. All errors are logged to the BIOS log on the Blade Controller (BC). Controller log forwarding with the Lightweight Log Manager (LLM) can be configured to forward the BIOS logs to the SMW. The Blade and Cabinet Controllers are diskless Linux systems, so log forwarding is enabled by default.

```
WARNING: Timed out waiting for BC to complete Aries PCIe Comparator Offset setup
   WARNING: Aries (B3:D0:F0) did not complete all PCIe Gen3 Phases, LNKSTS2=0x%x
   WARNING: Aries (B3:D0:F0) not running at Gen3, PCIe Phase 1 did not complete, LNKSTS=0x%x, LNKSTS2=0x%x
   WARNING: Aries (B3:D0:F0) link training failed, LNKSTS=0x%x
   WARNING: Aries (B3:D0:F0) did not train to Gen3, LNKSTS=0x%x
```

Figure 2. BIOS Failure to Initialize Aries

Any errors are logged and reported to the SMW command, *xtbounce*, as part of the system boot sequence.

```
***** node_up *****
09:12:12 - Beginning to wait for response(s)
09:15:14 - Received 50 of 54 responses
ERROR: c0-0c0s15n0 - 370 - SXM (GPU) PCIe link speed mismatch
ERROR: c0-0c0s15n1 - 370 - SXM (GPU) PCIe link speed mismatch
ERROR: c0-0c0s15n2 - 370 - SXM (GPU) PCIe link speed mismatch
ERROR: c0-0c0s15n3 - 370 - SXM (GPU) PCIe link speed mismatch
```

Figure 3. BIOS Failure to Train PCIe Link

The node is prevented from booting if any hardware failure is detected during BIOS execution. The Cray BIOS logs are copied to the Cray SMW on any BIOS error or failure for further analysis.

This ensures that at the time of system boot all devices are properly discovered, initialized, and trained successfully.

V. Node Hardware Errors

The CLE kernel captures node hardware errors. It reads MCA errors (correctable and uncorrectable) and logs them in the node console log, which is saved on System Management Workstation (SMW). It also reports them via the Hardware Error Log Channel connected from the node to the Blade Controller (BC), which forwards them to the SMW Hardware Error Logger Daemon (*xthwerrlogd*). The SMW command, *xthwerrlog*, displays all hardware errors logged via the Hardware Error Channel including MCA errors. The SMW command, *xtmcadecode*, decodes and provides detailed explanation for Intel MCA errors.

The output from *xthwerrlog* shows the correctable memory errors and displays the node, count, bank, type, and DIMM location.

TABLE I. xthwerrlog DIMM Output

xthwerrlog Correctable DIMM Data				
Node	**Count**	**Bank**	**Type**	**DIMM**
c1-0c0s1n0	1	8	CORRECTABLE	J10
c1-0c0s1n1	16	9	CORRECTABLE	J7
c1-0c0s7n2	50	9	CORRECTABLE	J11
c1-0c0s7n2	1	10	CORRECTABLE	J12
c1-0c1s1n1	24	9	CORRECTABLE	J11

Advanced Error Reporting (AER) is enabled in the CLE kernel by default. With AER enabled the Aries NIC logs PCIe errors to the root complex on the Node side. This is also true for both the Nvidia Kepler/Atlas GPUs and Intel Xeon PHI Knights Corner co-processor PCIe errors. The CNL kernel reads these PCIe errors and logs them in the node console log, which is saved on System Management Workstation (SMW). It also reports them via the Hardware Error Log Channel connected from the node to the Blade Controller (BC), which forwards them to the SMW Hardware Error Logger Daemon (*xthwerrlogd*). The PCIe errors are viewable using the SMW command, *xtpcimon*.

The output from *xtpcimon* shows the Aries PCIe errors and displays the time, node, description, and count is available.

```
c0-0c0s7a0n1   CorrectableMemErr   0:0:0 AER Correctable: Non-fatal error (mask bit: 1)

c0-0c1s6a0n2   CorrectableMemErr   Link CRC error (cnt: 3)
```

Figure 4. PCIe Monitor Error Output

The KNC kernel reads the KNC MCA Detected Unrecoverable Errors (DUE) and Corrected Errors (CE) and logs them in the KNC console log, which is saved on System Management Workstation (SMW). Compute blades with Intel Xeon processors and Intel KNC coprocessors have 2 console streams, one for the Xeon/host and one for the KNC coprocessor. The Xeon host processor processes the MCA error from the KNC coprocessor and sends it to HSS via the same Hardware Error Log Channel that is used between the Xeon host processor and HSS. The SMW command, *xthwerrlog*, displays all hardware errors logged via the Hardware Error Channel including MCA errors. The SMW MCA decode command, *xtmcadecode*, is used to decode the MCA Status Register as follows:

```
xtmcadecode -t xeon-phi -m 0 2 0x9490114e0100010d

Bank 2: Tag: TAG Directory:
MCA Status              = 0x9490114e0100010d:
MCACOD = 0x010d, MSCOD = 0x0100
Other Info = 0x0e
Corrected Error Count                        = 16453
Threshold-based Error Status       = 0, No Tracking

Common Status Info:
VALID       = 1 = Valid Error Detected
OVER        = 0 = No overflow
UC          = 0 = Error Corrected by HW
EN          = 1 = Reporting enabled for this error in MCA
MISCV       = 0 =
ADDRV       = 1 = Error address in MCi_ADDR
PCC         = 0 =
S           = 0 =
AR          = 1 = Immediate recovery action required for UCR


MEM Error: Errors in cache hierarchy.
```

```
Error Code:  CACHEL1_ERR_ERR
F            = Correction Report Filtering = 0 = Normal
TT           = TransactionType     = 3 = Invalid
LL           = Cache Level                        = 1 = L1
RRRR         = MemoryTransType  = 0 = Generic Error

Decode Complete
```

Figure 5. KNC MCA Decode Output

VI. Node Performance Diagnosis

The Cray XC system provides 3 different types of compute node configurations in the dual socket Processor Daughter Card including Intel SandyBridge, Intel IvyBridge, or Intel Haswell processors. There are a number of diagnostic tests available for the compute nodes that validate the nodes performance.

A. *Xeon Memory Performance*

The memory test, *xtmemtester*, targets all of the processor memory. *xtmemtester* is an effective user space memory test for stress-testing the memory subsystem. It is very effective at finding intermittent and non-deterministic faults. The maximum amount of memory that *xtmemtester* can test is less than the total amount of memory installed in the system; the kernel, libraries, and other system usage limits the memory available for testing.

It is extremely rare to have *xtmemtester* actually provide failure information. If the hardware indicates a SBE has occurred, it is corrected and the diagnostic will not see the SBE as it is by definition corrected. If the diagnostic forces a DBE, the diagnostic is terminated with an EC_NODE_FAILED status by the kernel.

B. *Xeon Non-Uniform Memory Access (NUMA) Performance*

The Non-Uniform Memory Access (NUMA) test, *xtnumatest*, is a set of tests that exercise and test the NUMA capability of a Symmetric Multi-Processors (SMP) node. *xtnumatest* is a group of tests that verifies that each processor core within an SMP is able to allocate and access memory that is local to the socket and across the Intel QuickPath Interconnect (QPI) to memory on the remote socket. The remote memory in the context of this test does not include memory connected to other SMPs that may be accessible via the Aries network. *xtnumatest* also provides a suite of tests that stress the node by exercising all cores simultaneously to stress the memory paths from each CPU to local and remote memory. Finally it also provides a performance test to validate the QPI performance.

```
Running test08: ***** QPI Bandwidth Test *****
Expected socket 0 to socket 1 is:
             (22990 MB/s - 25410 MB/s)
Expected socket 1 to socket 0 is:
             (22990 MB/s - 25410 MB/s)
```

Figure 6. QPI Bandwidth Test Successful Output

C. *Xeon Processor Performance*

The Intel processor stress test, *xtcpudgemm*, provides a computationally intensive processor tests to validate the Intel SandyBridge or IvyBridge processor. This test outputs the performance and the power for the processor during each pass of the diagnostic test. This test uses the standard CBLAS DGEMM. It also validates the results of the DGEMM matrix multiply.

Three arrays are summed to maximum memory usage:

```
A x B = C

Array A:[k,m] B:[n,k] and C:[n,m]
```

Figure 7. DGEMM Maximum Memory Usage

An extra matrix is used to store the initial C matrix data.

```
So {k x m} + {n x k} + {n x m} + {n x m} < max mem

Where default values are: m = 8192, n = 8192, k = 8192
```

Figure 8. DGEMM Matrix Multiply

When the *xtcpudgemm* diagnostic detects unexpected results, the diagnostic prints the actual value and the expected value. The DGEMM performance and p-state must always be greater than the value defined in the INI file. A non-zero output is returned upon failure.

```
c0-0c0s7n0 nid00028  Iteration,  GFlops, Node Power(W)
c0-0c0s7n1 nid00029  Iteration,  GFlops, Node Power(W)
c0-0c0s7n0 nid00028  0,            24.1141,          89

c0-0c0s7n0 nid00028 Failed:
c0-0c0s7n0 nid00028  0,            24.1141,          89
             Processor actual: 515.110009109461,
             Processor expected: 514.110009109461
c0-0c0s7n0 nid00028 Failed,1 failure(s)

c0-0c0s7n1 nid00029  0,            24.1123,          93
c0-0c0s7n1 nid00029 Failed:
c0-0c0s7n1 nid00029  0,            24.1123,          93
             Processor actual: 515.110009109461,
             Processor expected: 514.110009109461
c0-0c0s7n1 nid00029 Failed,1 failure(s)

Application 136517 exit codes: 2
```

Figure 9. Node DGEMM Diagnostic Miscompare Output

The *xtcpudgemm* diagnostic has been enhanced for greater control over testing each Intel processor core. In the past the diagnostic tested the dual socket node as an entity. It has also been enhanced to utilize the Intel AVX2 instruction set to increase the processor utilization.

The output format has been enhanced to support the additional test output as follows:

- **Time**: A timestamp of when this output was posted to the screen, at run's end
- **C-Name**: The C-name of the node the test was run on
- **System name**: Name of the system the test was run on
- **Iteration**: States which iteration of the run the test data was from.
- **GFlops minimum**: First the core GFlops is calculated by taking the runtime of the test on a core. Next using the smallest core GFlop value, multiply it by the number of cores per processor to calculate the GFlops minimum value.
- **GFlops maximum**: This is just like the GFlops minimum value except the largest individual core value is used and multiplied by the number of cores per processor.
- **GFlops average**: This is the average value reported by all the cores and then multiplied by the number of cores.
- **Socket**: This is the socket # the test was run on.
- **Brand**: This is the make and model of the processor.
- **CPU Count**: The number of sockets on the node (normally 2).
- **Cores/CPU**: How many cores per CPU.
- **Clock Speed (MHz)**: What the processor states is the maximum factory speed.
- **Effective Clock (MHz):** This is calculated from the GFlops and provides what *xtcpudgemm* determines is the processor clock.
- **Bin:** The delta from nominal clock in MHz.
- **Effective Bin:** This is calculated Bin from *xtcpudgemm*.

When the *xtcpudgemm* diagnostic detects a performance issue, the diagnostic prints the actual value and the expected value. The bin value is the delta from the normal clock in MHz. The diagnostic expects the bin value to be greater than or equal to -450 for Intel Haswell processors. As indicated in the table below the processor on c0-0c2s9n2 is failing to meet the expected performance.

TABLE II. XTCPUDGEMM PERFORMANCE OUTPUT

xtcpudgemm Performance Output Sample					
Node	**GFlops Min**	**GFlops Max**	**GFlops Avg**	**Bin**	**Eff Bin**
c0-0c2s8n0	503.813	506.223	505.066	-332.98	-77.2488
c0-0c2s9n0	501.591	504.382	503.217	-341.659	-87.0555
c0-0c2s9n1	495.502	498.846	497.501	-365.447	-113.934
c0-0c2s8n1	489.865	493.652	492.368	-387.466	-138.814
c0-0c2s8n2	477.858	481.309	480.039	-434.366	-191.809
c0-0c2s8n3	477.336	479.782	478.613	-436.406	-194.114
c0-0c2s9n3	473.895	477.102	475.889	-449.848	-209.302
c0-0c2s9n2	**472.54**	**474.801**	**473.787**	**-455.14**	**-215.283**

When the *xtcpudgemm* diagnostic detects a performance issue, the diagnostic prints the actual value and the expected value.

```
c0-0c2s9n2, nid00166, Fail:
        Bin actual: -455.14,
        Bin expected greater than or equal to: -450
c0-0c2s9n2, nid00166, Fail
c0-0c2s9n2, nid00166, 1 failure(s)
```

Figure 10. Node Processor Performance Failure Output

These diagnostic enhancements have allowed the diagnostic to isolate down to the failing core within the Intel processor.

VII. ARIES HIGH SPEED NETWORK PERFORMANCE DIAGNOSIS

The Cray XC system High Speed Network (HSN) connects each processing node to a single Aries network interface (NIC). Since there are four NICs on each Aries ASIC, there are four processor nodes connected to each Aries. Each Aries has 4 PCIe links and 10 optical ports. There are 96 Aries ASICs in a full sized optical group, composed of 16 per chassis times 6 chassis.

The Dragonfly topology is a hierarchical network consisting of two layers of a flattened butterfly topology. The first layer is a two dimensional flattened butterfly that connects all of the Aries ASICs with an optical (local) group. The optical group refers to a pair of cabinets or six chassis.

The first dimension within the optical group is the "green" dimension that connects the 16 Aries ASICs within the chassis. The second dimension within the optical group is the "black" dimension that connects the six chassis within the two cabinet optical group. The optical ports are the "blue" dimension that connects the optical groups within the Cray XC system. The five network ranks correspond to traversing green, black, blue, green, and black link in order.

A. Aries Stress Test

There are a number of on-line diagnostics that validate the Aries Block Transfer Engine (BTE) and Fast Memory Access (FMA) transfer types. The Aries FMA and BTE concurrent test, *xtfbc*, is designed to test the dedicated FMA and BTE logic blocks concurrently, while stressing the shared hardware like the Processor Interface (PI), Network Interface (NICs), Netlink (NL), network tiles and high speed links. The FMA and BTE threads exchange data between like thread types as well as FMA to BTE and BTE to FMA thread data exchanges. All threads have the ability to synchronize globally to keep all threads/ranks operating together as a single system test.

This test also supports partitioning and nearest neighbor mode.

The Aries on-line diagnostic tests can be executed periodically as batch jobs or interactively from the login nodes to validate system functionality between customer application job runs. The on-line diagnostic tests execute in user space using the uGNI library, the Generic Hardware Abstraction Layer driver (GHAL) and the Generic Network Interface driver (GNI) to access the Aries Network ASIC. These diagnostics do not use Message Passing Interface (MPI) or Distributed Memory Application (DMAPP) API.

When *xtfbc* diagnostic test fails, the failing NIC and rank indicate a failure and saves a log file with additional failure information.

```
10:31:22  c0-0c0s0n2: [naddr=0x2] thread 4 failed.

=== c0-0c0s0n2 ===
[diag-sft, rank:10] fma_ata_put_test:
              Data Miscompare at Offset 20.
[diag-sft, rank:10] Saved error file
              /var/log/fbc_ata_rank10_log.1626
[diag-sft, rank:10] thread_main: FMA All to All PUT failed,
              status: FMA_PUT_DATA_MISCOMPARE
```

Figure 11. Aries Data Miscompare Failure Output

If multiple ranks per node are selected, a thread is started for each rank on the node and each thread is mapped to a core. The diagnostic allocates a buffer for the diagnostic and a buffer for each additional node and rank under test. For performance considerations, the diagnostic statically allocates the required buffer space to support the number of cores and ranks selected. The available memory per node limits the number of Aries Network ASICs that can be tested at any given time.

It is not recommended to attempt to run all cores within the Cray XC system, unless the Cray XC system is very small (less than 4 cabinets). The FMA and BTE diagnostic can saturate the bandwidth from the node processor to the Aries Network ASIC at around 3 to 4 cores depending on the core processor speed, memory size and memory bandwidth.

B. Aries Network Performance Test

The Aries All-To-All performance test, *xta2a*, is used to measure the performance on all-to-all communication for sets of nodes corresponding to the physical structure of an XC30 system: blades, chassis, groups, and the whole system. The test is designed to run on as many nodes as are available, reporting variation in performance over sets of nodes of a given size. For example, to run 512 instances of a blade level test on 2048 nodes and report variation between them.

The *xta2a* diagnostic generates a high network load. In particular it stresses the PCIe interfaces that connect each node to Aries. Poor performance on this test correlates well to high rates of PCIe errors (logged on the SMW). The *pcitest.sh* wrapper script exercises this use of *xta2a*. The test executes on all nodes for a period of ten minutes. This is sufficient to detect nodes with rates of PCIe errors that impact application performance. The *pcitest.sh* script reports start and end times for the test. These times are used with the SMW command, *xtpe*, to select errors reported in the same interval.

Running the *xta2a* test in pcitest mode for 5 minutes on 20 nodes.

```
nprocs=160 sets=5 ppn=8 maxlongs=512 dmapp_mode=2 bte=65536
Bytes     Min    Mean    Max     Dev  Scaled
  4096    6258    6340   6499     97    1.5%
  4096    6259    6344   6506    100     1.6%
  4096    6259    6344   6507    100     1.6%
  4096    6258    6344   6508    100     1.6%
  4096    6259    6344   6507    100     1.6%
nprocs=160 sets=5 ppn=8 maxlongs=512 dmapp_mode=2 bte=65536
Bytes     Min    Mean    Max     Dev  Scaled
  4096    6250    6339   6506    101     1.6%
  4096    6252    6343   6508    102     1.6%
  4096    6252    6344   6509    102     1.6%
  4096    6252    6344   6509    102     1.6%
  4096    6252    6344   6509    102     1.6%
02:59:24 PM 2014-04-13 The xta2a test Completed.
```

Figure 12. xta2a Diagnostic Test Output

The *xta2a* test uses MPI_Alltoall with the DMAPP optimizations enabled. As such it is representative of a real application. The *xta2a* test does not require dedicated access to the whole system. It can be run on a subset of nodes allocated by the batch system. The impact on performance of other applications is low if all nodes in an electrical group are allocated to a test. The *pcitest.sh* script can be adapted to create a batch script suitable for this purpose.

```
nprocs=1536 sets=32 ppn=12 maxlongs=512 dmapp_mode=2 bte=1024
Bytes     Min    Mean    Max     Dev  Scaled
  4096    5457    5603   5668     56    1.0%
```

Figure 13. xta2a Diagnostic Successful Output

Note the low variation between minimum and maximum bandwidth. This is a good result.

The second example output is typical of a system in which a small number of nodes are showing performance problems.

```
nprocs=42008 sets=1318 ppn=8 maxlongs=512 dmapp_mode=2
bte=1024
Bytes     Min    Mean    Max     Dev  Scaled
  4096    3626    4887   4891     74    1.5%
  4096    3944    4916   4918     74    1.5%
  4096    4068    4916   4918     74    1.5%
```

```
 4096    3617     4915     4919       84      1.7%
Bandwidth low for set  994 nodes 4056 4059:     3617GB/s
Bandwidth low for set 1051 nodes 4288 4291:     4012GB/s
```

Figure 14. xta2a Diagnostic Performance Failure Output

The test reports sets of nodes (blades in this case) that show poor performance.

Details of PCIe errors that cause the variation in performance can be found by running *xtpe* on the SMW.

```
Input file is        : pcimon-20130809.1
Elapsed sample time  : 77.07 minutes

'Bad link' threshold : 4.0000 (CRCTLP|DLLP Errors)/minute

CORRECCTABLE ERRORS:
==========================
ERRORS   ERROR STRINGS
------   ---------------------------------------------------
626      'Info              0:3:0 AER Correctable: Replay timeout (mask bit:
0)'

LINK CRC|TLP|DLLP ERRORS:
==========================
ERROR                    NODE ID              b:d:f
          ERRORS         PER MIN        FIX?
Link CRC error             c2-0c0s1a0n2          -
          837           10.8607        Fix this link
Bad TLP/DLLP               c0-1c2s7n2           0:3:0
          2469/0        32.0372        Fix this link
Bad TLP/DLLP               c1-0c1s10n2          0:3:0
          3800/0        49.3080        Fix this link
Bad TLP/DLLP               c2-1c2s8n2           0:3:0
          586/0         7.6038         Fix this link
Bad TLP/DLLP               c3-0c2s8n2           0:3:0
          9656/0        125.2941       Fix this link
Bad TLP/DLLP               c3-1c0s3n2           0:3:0
          463/0         6.0078         Fix this link
Bad TLP/DLLP               c4-1c1s14n2          0:3:0
          2913/0        37.7984        Fix this link
Bad TLP/DLLP               c4-1c2s4n2           0:3:0
          11422/0       148.2093       Fix this link
Bad TLP/DLLP               c5-0c0s9n2           0:3:0
          1573/0        20.4109        Fix this link
Bad TLP/DLLP               c5-1c1s3n2           0:3:0
          573/0         7.4351         Fix this link
Bad TLP/DLLP               c6-0c0s1n2           0:3:0
          2049/0        26.5874        Fix this link
Bad TLP/DLLP               c6-0c2s14n2          0:3:0
          3539/0        45.9213        Fix this link
Bad TLP/DLLP               c7-0c0s14n2          0:3:0
          435/0         5.6445         Fix this link
Bad TLP/DLLP               c7-0c0s1n2           0:3:0
          340/0         4.4118         Fix this link
```

Figure 15. PCIe Error Output

VIII. NVIDIA GPU PROCESSOR PEROFRMANCE DIAGNOSIS

There are a number of diagnostic tests available for the compute node with Nvidia GPUs that validate the performance of the GPU. The Cray GPU DGEMM test, *xkdgemm*, is a standard double precision floating point matrix multiply application. It utilizes CUDA to execute the matrix multiply on the Nvidia GPU and uses the standard CUDA BLAS library. The Cray GPU Memory test, *xkmemtest*, is used to test the GPU memory for hardware errors and soft errors using CUDA. The Cray GPU PCIe bandwidth test, *xkbandwidth*, is used to measure the memory copy bandwidth of the GPU. It can measure device-to-device copy bandwidth, host to device copy bandwidth for pageable and pinned memory, and device to host copy bandwidth for pageable and pinned memory.

The Cray GPU stress test, *xkstress*, performs stress and performance test across nodes. This test includes STREAMS, GEMM, and PCIe tests. It is an MPI application that compares performance results across the blades, cabinets, and system.

When the *xkdgemm* diagnostic detects unexpected results, the diagnostic prints the actual value and the expected value. The DGEMM performance and p-state must always be greater than the value defined in the INI file. A non-zero output is returned upon failure.

```
c1-0c2s0n2 nid00288 Iteration,
          Gflops,
          Temp (C),
          AvgPower (W),
          P-state,
          AvgClockrate (Mhz)

c1-0c2s0n2 nid00288 0,
          1217.59,
          27,
          180.74,
          0,
          732

c1-0c2s0n2 nid00288 Failed:
    GPU actual: 502.630097504761,
    GPU expected: 502.630099412109
```

Figure 16. GPU DGEMM Data Miscompare Failure Output

The *xkdgemm* diagnostic is also good at finding GPU performance issues especially when the MPI mode is enabled. This allows the diagnostic to use MPI to compare the results across each node. When the *xkdgemm* diagnostic detects unexpected performance results, the diagnostic prints the actual performance results and the expected performance.

```
xkdgemm -m 12100 -n 4096 -k 14100 -i 1 -l 10 -g –v
c0-0c2s13n2 nid00182 MPI mode enabled, root node

Iteration, GFlops, Temp, Power, P-state, SMClockrate

c0-0c1s5n1 nid00085
     Failed: c0-0c1s5n1 nid00085
     8, 1295.29, 74, 194.934, 0, 745
     GFlops actual: 1295.29,
     GFlops expected greater than: 1300
```

Figure 17. GPU DGEMM Performance Failure Output

IX. Intel Co-Processor Performance Diagnosability

There are a number of standalone Intel Xeon PHI co-processor diagnostic tests. The Cray PHI DGEMM test, *xtphidgemm*, is a standard double precision floating point matrix multiply application. The test also outputs the KNC PHI temperature, memory temperature, and power usages on each pass.

When the *xtphidgemm* diagnostic detects unexpected results, the diagnostic prints the actual value and the expected value. A non-zero output is returned upon failure.

```
c0-0c0s13n1 nid00053  Iteration,
            GFlops,
            Die Temp(C),
            Memory Temp (C),
            Avg Power (W),
            Avg Clock(MHz)

c0-0c0s13n1 nid00053       0,
            696.908,
            68,
            53,
            196.252,
            1052.63

c1-0c2s13n2 nid00053 Failed:
            Co-processor actual: 502.630097504761,
            Co-processor expected: 502.630099412109
```

Figure 18. KNC DGEMM Data Miscompare Failure Output

The Cray PHI PCIe bandwidth test, *xtphibandwidth*, measures the PCIe bandwidth and memory bandwidth of the Intel MIC KNC. Measurements include device-to-device (DTOD) copy bandwidth, host to device (HTOD) copy bandwidth, and device to host (DTOH) copy bandwidth. Random floating-point data of a default size of 128 MB is verified after each copy.

When the *xtphibandwidth* diagnostic detects PCIe bandwidth performance that is less than the target as defined in the INI file, the diagnostic prints out the actual value and the expected value. The bandwidth performance must be greater than the value defined in the INI file in order to pass. The diagnostic also has a non-zero output on failure.

```
c0-0c0s13n1 nid00053  Iteration, Size(MB),
            HTOD(MB), DTOD(MB), DTOH(MB)

c0-0c0s13n1 nid00053       0,     128,  6213.79,   127506,  6491.41

c0-0c0s13n1 nid00053 Failed:
c0-0c0s13n1 nid00053       0,     128,
            6213.79,   127506,  6491.41
            Actual DTOD bandwidth: 127506,
            Expected DTOD bandwidth: 130000

c0-0c0s13n1 nid00053       1,     128,   6214.4,   127060,  6489.24
c0-0c0s13n1 nid00053 Failed:
c0-0c0s13n1 nid00053       1,     128,
            6214.4,   127060,  6489.24
            Actual DTOD bandwidth: 127060,
            Expected DTOD bandwidth: 130000

c0-0c0s13n1 nid00053 xtphibandwidth Failed, 2 failures
```

Figure 19. GPU PCIe Bandwidth Failure Output

X. HSS Out-Of-Band Diagnosis

The Cray XC liquid cooled cabinet provides a significant number of temperature, velocity, humidity, and water temperature and pressure, and power sensors to maintain optimal levels of power and thermal control within the Cray XC system. Cray HSS software polls these interfaces on a consistent frequency to monitor the data in the cabinet.

The HSS software installed on the SMW, blade controllers, and cabinet controllers monitors the data from the various sensors in the Cray XC cabinet.

To validate the HSS hardware and software, the SMW HSS diagnostic utility, *xtcheckhss*, is used. The SMW command validates the Cray XC system HSS infrastructure by checking each blade and each cabinet. It can be used to get a quick validation that the HSS infrastructure is functioning normally and can also be used to trouble-shoot a blade or a given cabinet. On a blade it validates the basic blade functionality. It can also validate HSS voltages, Aries voltages and current, temperatures, PDC sensors, as well as, the Intel processor and DIMM temperatures and voltages. For a cabinet, it validates the basic cabinet functionality. It can also validate the HSS voltages and rectifiers in the cabinet. It is also used to validate the HSS infrastructure for a compute cabinet, blower cabinet, pre-conditioner cabinet and all air sensors in the compute or pre-conditioner cabinet. Additionally it verifies all of the HSS firmware versions are properly flashed and verifies the HSS Inventory Serial EEPROM (SEEP).

The HSS diagnostic utility can be used to detect low voltage failures as shown below.

```
xtcheckhss --volts --blade=c0-0c0s7

Component   Module
            Sensor
            HMIN      SMIN      DATA                UNIT
            SMAX      HMAX

c0-0c0s7n2  qpdc0_n0_s0_mem_vrm           vdd_vdr01_s0_c_i
            1200      1350      1339      v*1000
            1650      1800
```

Figure 20. HSS Low Voltage Output

XI. HSS Out-Of-Band (OOB) Debug

The HSS system management platform performs monitoring and management out-of-band to the compute node. HSS also implements interfaces to various vendors Out-Of-Band interfaces to enhance OOB debug of the Cray System. These interfaces are as follows:

- Intel Platform Environment Control Interface (PECI)
- Nvidia SMBus Post-Box Interface (SMBPBI)
- Intel In-Target Probe (ITP)

A. Intel Platform Environment Control Interface (PECI) Debug

Intel processors provide a single wire signal for their Platform Environment Control Interface (PECI). Intel uses the Intelligent Platform Management Interface (IPMI) Specification [1] as the protocol to connect to the Intel Management Engine (ME). This interface is used to monitor thermal, power and electrical conditions of the Intel processor. This interface is primarily used to collect power, thermal, and status information by the HSS utility System Environment Data Collections (SEDC), which monitors the system health and records the environmental data coming from the system's hardware components. It is also provides the interface for Cray Power Management software stack. And finally the data can be viewed real-time using the HSS diagnostic utility, *xtcheckhss*.

B. Nvidia SMBus Post-Box Interface (SMBPBI) Debug

Nvidia provides OOB debug capabilities. Cray utilizes the OOB to monitor the Nvidia SXM GPU accelerators via an i2c bus using the Nvidia SMBus Post-Box Interface (SMBPBI). This interface is used to monitor aggregate Double Bit Errors (DBE), aggregate Single Bit Errors (SBE), thermal, power, and electrical conditions of the Nvidia SXM GPU accelerator.

The Nvidia driver supports "retiring" (or “mapping out”) framebuffer memory pages when it contains a memory cell that is degrading in quality. Retiring a degraded page and preventing it from being used can increase the longevity of an otherwise good board. This feature is enabled by default.

The SMW diagnostic command, *xtaccecc*, is used to display the aggregate DBE and SBE errors, as well as, the retired DBE and SBE error counts of the GPU via the OOB debug path as shown below.

```
./xtaccecc c0-0c0s6
=============================================
ACCEL location Serial Number
             Agg DBE Agg SBE DPR DBE DPR SBE
=============================================
c0-0c0s6n0    0332612070036
            3     2     3     1
c0-0c0s6n1    0332612060220
            0     0     0     0
c0-0c0s6n2    0332612060254
            0     0     0     0
c0-0c0s6n3    0332612070098
            0     0     0     0
=============================================
```

Figure 21. GPU OOB DBE/SBE Counts

The Node and the GPU must have power applied to allow HSS to read the data from the OOB debug path.

C. Intel In-Target Probe (ITP) Debug

The Intel In-Target Probe (ITP) [2] is a JTAG bus with some Intel-specific signals and protocol added. It is traditionally used to debug software and diagnose Intel processor (SandyBridge or IvyBridge) problems. Typically, this is done with an Intel ITP interface connected to the processor via the eXtended Debug Port (XDP). Cray has implemented an embedded ITP so that no external hardware needs to be connected to the Cray XC system. The embedded ITP is used as a processor hardware debug tool. Python bindings exist for the ITP library and several python scripts have been written to take advantage of this feature. These scripts reside on the SMW and are available via the SMW command *xtitp*.

Many of the scripts provide useful hardware debug information about the PCIe configuration and status, QPI configuration and status, processor information, MCA errors, MSR data, and the package Power Limit (turbo) registers. Executing this command on the SMW temporarily pauses the processor, until the data is read from the processor and resumes the processor once the read is complete.

```
xtitp -t c0-0c0s7 qpi-status 1

Socket 0
        QPI0:
                Link Speed: 8.0 GT/s
                Configured Tx Width: Full
                Configured Rx Width: Full
                Tx Lane Status: 0xfffff
                Rx Lane Status: 0xfffff
                Error Counter 0: 0
                Error Counter 1: 0
        QPI1:
                Link Speed: 8.0 GT/s
                Configured Tx Width: Full
                Configured Rx Width: Full
                Tx Lane Status: 0xfffff
                Rx Lane Status: 0xfffff
                Error Counter 0: 0
                Error Counter 1: 0

Socket 1
        QPI0:
                Link Speed: 8.0 GT/s
                Configured Tx Width: Full
                Configured Rx Width: Full
                Tx Lane Status: 0xfffff
                Rx Lane Status: 0xfffff
                Error Counter 0: 0
                Error Counter 1: 0
        QPI1:
                Link Speed: 8.0 GT/s
                Configured Tx Width: Full
Configured Rx Width: Full
Tx Lane Status: 0xfffff
Rx Lane Status: 0xfffff
Error Counter 0: 0
Error Counter 1: 0
```

Figure 22. Node OOB QPI Status and Error Counts Output

XII. Summary

Cray has provided a number of diagnostics, commands, and utilities that enhance the system diagnosability of the Cray XC system. The focus of system diagnosability has been on ensuring that each component is functioning properly by ensuring that each component can be validated and that all data is captured at the time of failure.

Each aspect of the tool chain has been enhanced on the Cray XC system to better ensure that the Cray XC system is performing as expected. Enhancements have included BIOS, SMW commands, utilities, and diagnostics, as well as, power and thermal data and event logs. Future enhancements are planned to continue to improve system diagnosability of the Cray XC system.

References

[1] Intelligent Platform Management Interface Specification, version 2.0, 2004, http://www.intel.com/design/servers/ipmi/spec.htm.

[2] ITP700 Debug Port, Design Guide, February 2004, http://download.intel.com/support/processors/xeon/sb/24967914.pdf