Thirty Meter Telescope
Narrow Field InfraRed Adaptive Optics System
Real-Time Controller Prototyping Results

Malcolm Smith*[a], Dan Kerley [a], Edward L. Chapin[a], Jennifer Dunn[a], Glen Herriot[a], Jean-Pierre Véran[a], Corrine Boyer[b], Brent Ellerbroek[b], Luc Gilles[b], Lianqi Wang[b]
[a]National Research Council Herzberg, 5071 W Saanich Rd, Victoria, V9E 2E7, Canada
[b]Thirty Meter Telescope, Suite 300 – 100 Walnut St, Pasadena, CA 9124, USA

ABSTRACT

Prototyping and benchmarking was performed for the Real-Time Controller (RTC) of the Narrow Field InfraRed Adaptive Optics System (NFIRAOS). To perform wavefront correction, NFIRAOS utilizes two deformable mirrors (DM) and one tip/tilt stage (TTS). The RTC receives wavefront information from six Laser Guide Star (LGS) Shack-Hartmann WaveFront Sensors (WFS), one high-order Natural Guide Star Pyramid WaveFront Sensor (PWFS) and multiple low-order instrument detectors. The RTC uses this information to determine the commands to send to the wavefront correctors. NFIRAOS is the first light AO system for the Thirty Meter Telescope (TMT).

The prototyping was performed using dual-socket high performance Linux servers with the real-time (PREEMPT_RT) patch and demonstrated the viability of a commercial off-the-shelf (COTS) hardware approach to large scale AO reconstruction. In particular, a large custom matrix vector multiplication (MVM) was benchmarked which met the required latency requirements. In addition all major inter-machine communication was verified to be adequate using 10Gb and 40Gb Ethernet. The results of this prototyping has enabled a CPU-based NFIRAOS RTC design to proceed with confidence and that COTS hardware can be used to meet the demanding performance requirements.

Keywords: NFIRAOS, real-time controller, RTC, adaptive optics, TMT, prototyping, benchmarking

1. INTRODUCTION

1.1 NFIRAOS

NFIRAOS (Narrow Field InfraRed Adaptive Optics System) is a first light adaptive optics system at the TMT (Thirty Meter Telescope). It is discussed in detail in other papers[1][2]. For the purposes of real-time control, NFIRAOS contains two deformable mirrors, one tip/tilt stage, six Shack-Hartmann LGS WFS (laser guide star wavefront sensors) and one natural guide star pyramid WFS. The RTC (real-time controller) also receives additional pixels from multiple instrument detectors. The computational demands on the RTC are primarily driven by the (up to) 800 Hz frame rate, the 7673 DM actuators and the 2896 subapertures on each of the six laser guide star wavefront sensors. Due to the irregular shape of the primary mirror, and obstructions such as the secondary support structure, approximately 15504 subapertures (2584 per LGS WFS) are sufficiently illuminated to be used for control. These illuminated subapertures are then used to directly control 6981 of the 7673 deformable mirror actuators. The directly controlled actuators are called active actuators. The remaining 692 actuators are slaved actuators in which their actuator positions are determined from an extrapolation of the active actuators.

The most demanding computational task of the RTC is the matrix vector multiplication which converts high-order LGS WFS gradients into a DM error vector for active DM actuators. This reconstructor control matrix consists of 6981 rows and 31008 columns (two gradient components per illuminated subaperture). Using four byte floating point coefficients, the full control matrix requires over 825 MiB of memory.

*Malcolm.smith@nrc-cnrc.gc.ca; phone 1-250-363-8380; fax 1-250-363-0045; www.nrc-cnrc.gc.ca/eng/rd/nsi/

Adaptive Optics Systems V, edited by Enrico Marchetti, Laird M. Close, Jean-Pierre Véran, Proc. of SPIE Vol. 9909, 99094N
 · doi: 10.1117/12.2233711

1.2 RTC Architecture

Figure 1 is a simplified block diagram of the RTC data flow. The major tasks of the RTC are shown along with data flows in and out of the RTC. The most demanding real-time path occurs in LGS MCAO (multi-conjugate adaptive optics) mode where the LGS WFS pixels are converted into gradients by the LGS pixel processing. The gradients are used as the input vector to the large matrix vector multiplication (high-order reconstruction block) and the high-order errors are added to the low-order errors before the final computation, clipping and scaling of DM actuator commands and TTS commands (wavefront correction block). Since the low-order processing shown below has only modest computational demands it is not considered in this paper. See Kerley et al.[3] for details on the low-order processing, and other RTC architecture details.

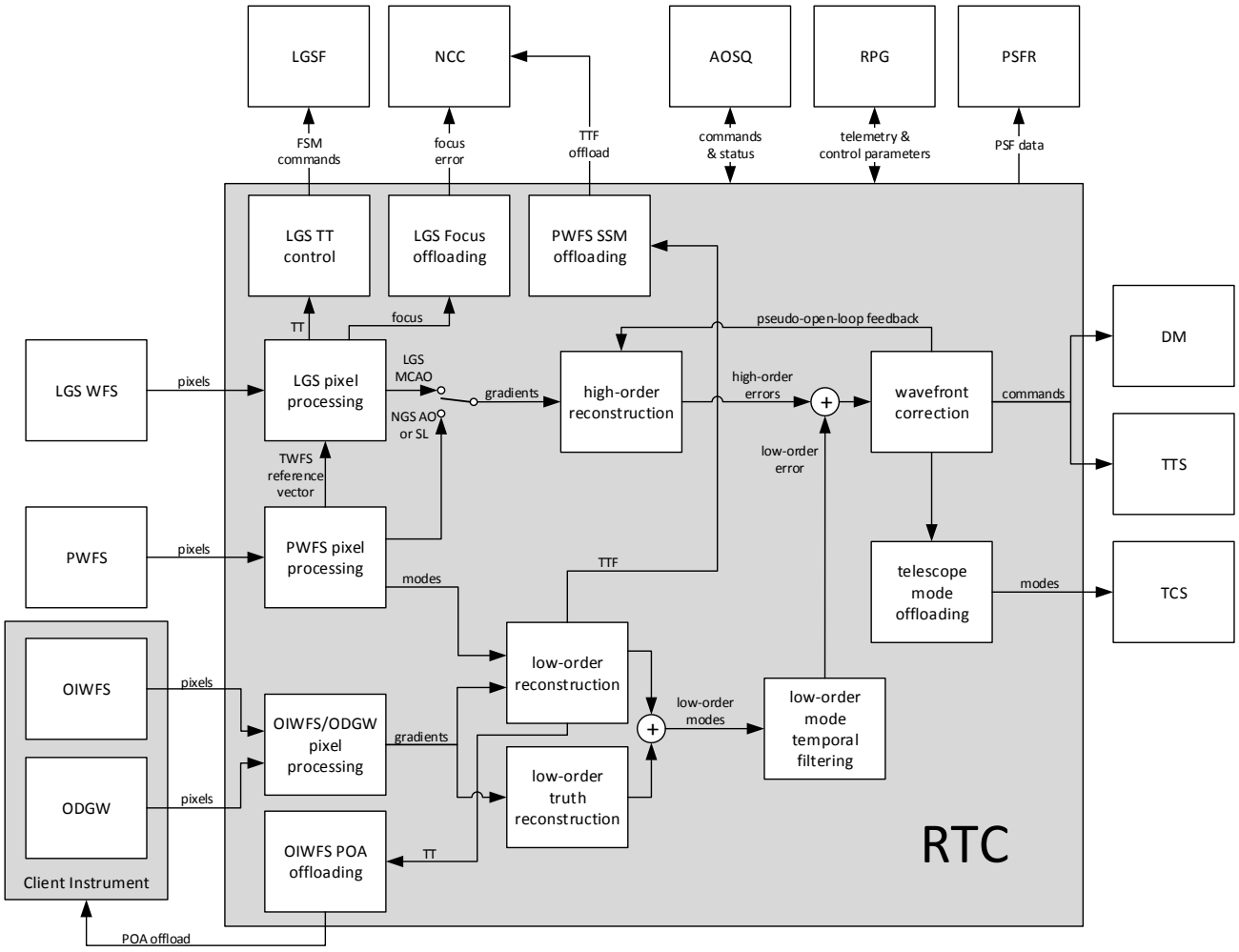

Figure 1 - Simplifed RTC Block Diagram

Figure 1 represents the RTC as an abstract collection of processing blocks. The actual RTC will consist of multiple servers, each executing multiple threads and operating in one of the following roles:

- HOP Server: The High-Order Processing server performs high-order pixel processing and high-order reconstruction. In LGS MCAO mode, up to six HOP servers may be active at one time to process the pixels from the six LGS WFS, and to perform the high-order reconstruction. In NGS AO mode, four HOP servers are used to process the PWFS pixels, and to perform the high-order reconstruction.

- WCC Server: The Wavefront Corrector Control server combines the individual high-order error vectors from the HOP servers, performs the low-order pixel processing, low-order reconstruction and filtering, and performs the final wavefront correction operations. The WCC server also computes various offloading parameters.
- TED Server: The Telemetry Engineering Display server provides both a web interface to the RTC for engineering purposes, and a unified interface between the RTC and other TMT systems. The TED server allows the RTC to appear as a single entity to most TMT systems.
- PTS Server: The Persistent Telemetry Storage server is used to store telemetry data required for point spread function (PSF) reconstruction, which is performed by an independent TMT server. The PTS also allows the storage of tagged data used for diagnostic purposes. The PTS contains fault tolerant storage to minimize the chance of data loss.
- Spare Server: One or more spare servers are included in the RTC so that there is an available replacement machine for each of the server roles.
- Test Server: The Test server allows standalone testing of the RTC. The Test server provides replacements for the data sources and data sinks that the RTC communicates with. Primarily, it sends pixels streams which mimic the LGS WFS and/or PWFS. It also sends the pixel streams for the low-order instrument sensors, accepts DM command vectors, and accepts Tip/Tilt stage commands. This allows the Test server to verify that the RTC has been correctly configured and is operating at an acceptable level of performance.

Figure 2 contains estimates of the execution time required for each of the various processing steps performed by the RTC in LGS MCAO mode. When two steps overlap in time, the steps are performed simultaneously as part of a pipelined set of processes. The three longest duration steps are reading the high-order pixels, processing the high-order pixels, and performing the high-order MVM. As these steps are discussed in this paper, keep in mind that they have a lower bound of 500 microseconds as set by the WFS readout rate. Pipelining allows the MVM to finish soon after the arrival of the last datagram of pixels. The next longest processing steps are transporting the DM error vectors from the HOP servers to the WCC, and the DM clipping process. These two steps are also discussed in this paper, and are shaded red to indicate that they are being investigated as possible areas of improvement for the RTC. The five steps discussed above have been benchmarked. The execution times of the remaining steps have been estimated based on small benchmarks such as a single threaded dense matrix vector multiply routine, and a single threaded sparse matrix multiply routine.

The RTC prototyping has been driven by the desire to reduce risk and uncertainty in the RTC design and to determine potential performance issues.

1.3 CPU Hardware

The hardware in Table 1 is a list of the CPUs used during prototyping. Testing of the E5-2643 V4 CPU has not yet occurred, but is planned for the remainder of 2016. The low core count E5-2643 V4 was selected as a prototype machine for the WCC role where a large number of cores and a large L3 cache are not required. Additionally, the higher clock speed of the E5-2643 V4 is an advantage for the WCC server.

The Resource Director Technology (RDT)[4], available on the E5-2643 V4, allows a partitioning of the L3 cache to ensure that high priority processes do not have the contents of their cache partition disturbed by other processes. This technology should allow more consistent execution times of the high priority real-time RTC tasks.

Each HOP server will use either four E7-8860 V4 CPUs (released on June 6, 2016, and similar to the tested E7-8870 V3 but operating at 2.2 GHz and including RDT) or a single Knights Landing Xeon Phi (expected later in 2016). The large L3 caches of the E7-8860 V4 allow the LGS MCAO high-order reconstructor control matrix to be fully cached when spread across the 24 CPUs of the six HOP servers. Each CPU must store approximately 35MB of control matrix. During regular operation, the control matrix will be replaced with a new control matrix every ten seconds. A quad socket E7 server has been chosen over a quad socket E5 server due to the higher memory bandwidth which reduces latency when a new control matrix is loaded from memory.

The Knights Landing Xeon Phi (KNL) is currently available only as pre-release developer machines. We are expecting the arrival of a developer system during the summer of 2016. According to publicly released information[6] from Intel, the Knights Landing Xeon Phi will support up to 72 cores, two vector units per core, and 16 GiB MCDRAM (high

bandwidth memory) which can be used either directly or as L3 cache. Each core can simultaneously execute up to four threads. Single threaded performance of the KNL Xeon Phi is expected to be lower than single core performance of a Xeon CPU, but for use within a HOP server, a single KNL chip should be able to replace four Xeon E7 CPUs due to the high memory bandwidth of the MCDRAM (greater than 400GB/s).

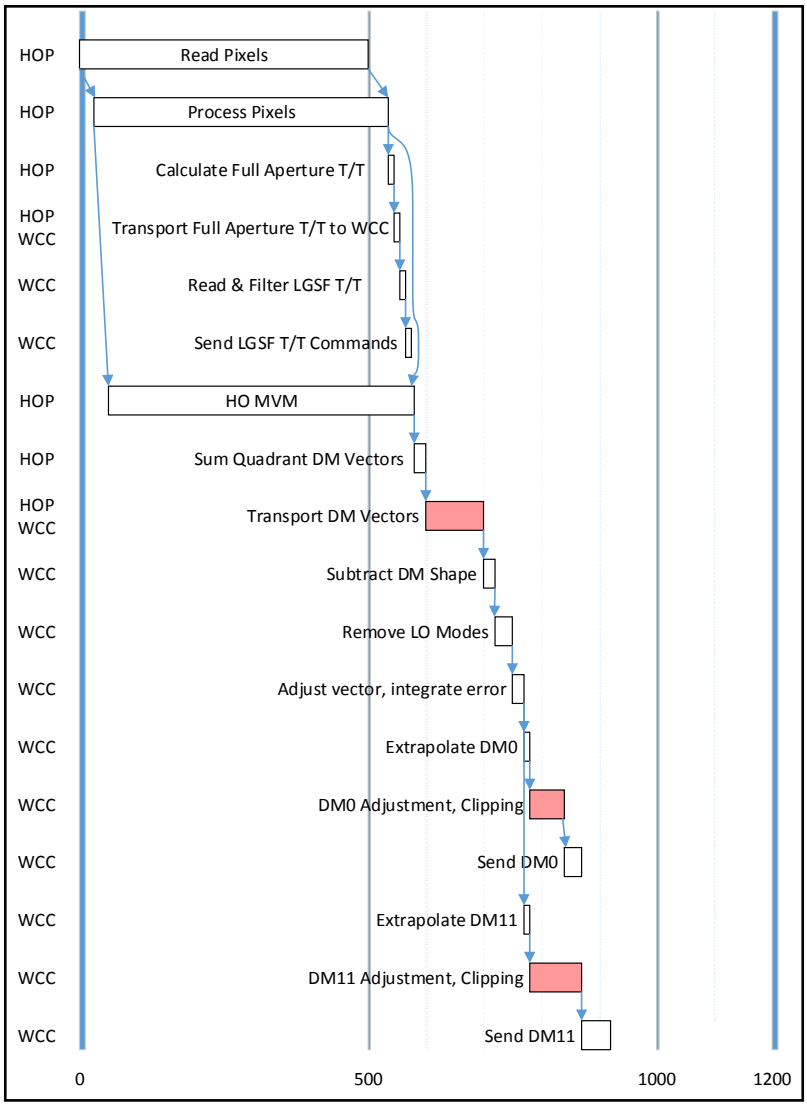

Figure 2 - Estimated Pipeline Stage Execution Times

The Knights Landing Xeon Phi can use sub-NUMA (non-uniform memory access) clustering to appear to the operating system as a four socket Xeon server. This configuration mode provides the lowest memory latency if the workload is highly NUMA optimized. The RTC software will be optimized for NUMA architectures. To make the HOP software less dependent upon hardware selection, time critical HOP server software will assume a four-way NUMA architecture. The HOP server software will use configuration files to allow control of the number of threads and their assignment to physical cores.

The use of the Knights Landing Xeon Phi is expected to increase system latency since the control matrix must be fetched from the MCDRAM for each frame (it is much too large to fit in the on-core caches), but is expected to significantly reduce system cost, power consumption, and size.

Table 1 - Specifications of CPUs used for RTC benchmarking

	E5-2670	E5-2697 V2	E5-2699 V3	E7-8870 V3	E5-2643 V4
Launch Date	2012-Q1	2013-Q3	2014-Q3	2015-Q2	2016-Q1
Lithography	32 nm	22 nm	22 nm	22 nm	14 nm
# Cores	8	12	18	18	6
CPU Clockspeed	2.6 GHz	2.7 GHz	2.3 GHz	2.1 GHz	3.4 GHz
L3 Cache (LLC)	20 MiB	30 MiB	45 MiB	45 MiB	20 MiB
Memory Bandwidth	51 GB/sec	60 GB/sec	68 GB/sec	102 GB/sec	77 GB/sec
Resource Director Technology (RDT)					✓

2. HOP PROTOTYPING

2.1 Pixel Processing

The first task of the HOP servers is to read and process the high-order pixel streams. Each LGS WFS provides approximately 200K pixels over a dedicated 10Gb Ethernet link. For each subaperture, the HOP server carries out the following steps:

- Read pixels from network adapter and convert bytes from network byte order to host byte order
- Convert 16 bit pixels into 32 bit floating point values
- Subtract background and scale by flat field
- Compute subaperture X and Y gradients via matched filters (i.e. dot product), compute subaperture flux
- Normalize subaperture gradients based on computed subaperture flux, subtract reference gradients

Tests during the RTC Architecture Trade Study showed that a single CPU core could easily perform the pixel calibration process (first three steps) and a second CPU core could perform the gradient computation (last two steps) for one LGS WFS quadrant. One CPU core was also dedicated to handling the Ethernet interrupts associated with the pixel stream. A total of nine CPU cores were therefore allocated to the pixel processing tasks for each HOP server.

The benchmarking done during the Architecture Trade Study used a simplified data stream that consisted solely of the LGS WFS pixels. The data stream from the actual LGS WFS includes meta-information which will increase the processing time. The possibility of using a KNL Xeon Phi, with a slower per thread performance, for the HOP servers also suggests that the pixel processing time could increase. In order to verify that the pixel processing would not need to be divided into more than two threads per LGS WFS quadrant, a pair of benchmarking programs was written to investigate the computational load of the pixel processing in more detail. An LGS WFS simulator was written to provide a realistic data stream (although pixel values were not realistic) and a standalone pixel processor was written to perform the pixel processing of the more complete data stream. The programs could be configured so that either a single quadrant was sent, or all four quadrants of the LGS WFS were sent. The pixel processing program used a single thread for reading pixels, and a second thread for pixel calibration and gradient computation. Since the pixels arrive as a UDP data stream, only performing pixel reading in the first thread reduces the chance of buffer overflow and lost datagrams.

Figure 3 shows the results of sending a single LGS WFS quadrant as quickly as possible. This allows us to estimate the amount of time required to read the pixels, perform the pixel processing and to store the calibrated pixels into RAM. The times in Figure 3 are all significantly lower than the 500 microsecond readout time of the LGS WFS. Both the sending server and the pixel processing server used E5-2697 V2 CPUs.

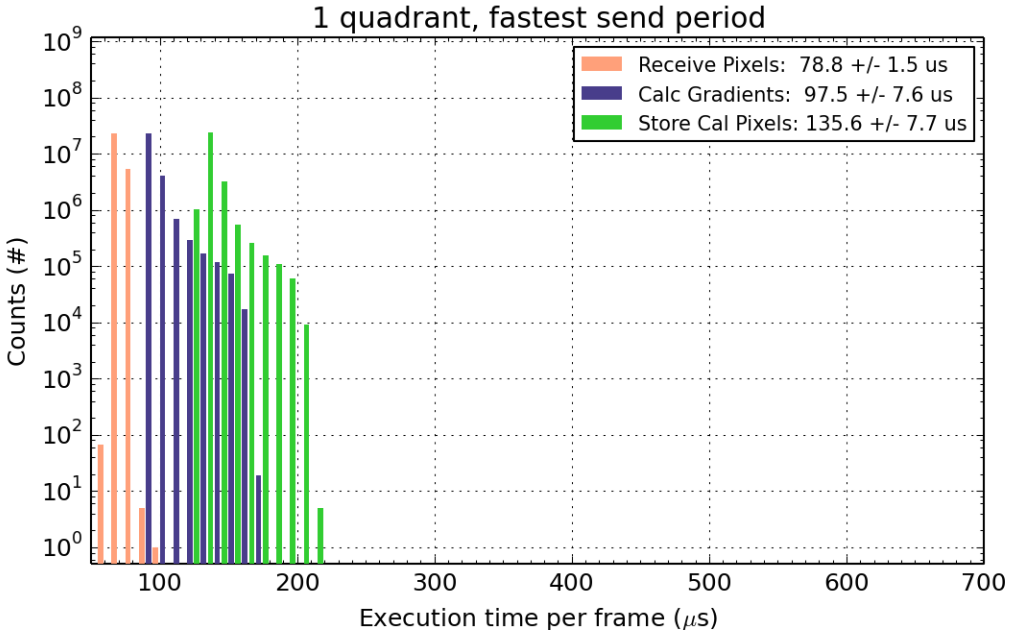

Figure 3 - Pixel processing times for single LGS WFS quadrant

In Figure 4, all four quadrants are sent by the LGS WFS simulator, but the UDP packets are delayed so that the pixels are sent out over an interval of approximately 500 microseconds. On average, the pixel reading required 487.6 microseconds, confirming the (approximate) 500 microsecond pixel sending interval. When the pixel datagrams are spread out over the 500 microsecond interval, the pixel processing thread, which computes the gradients, is able to keep up with the pixel datagram arrival rate, and a tight distribution of pixel processing times results. Similarly, the distribution of calibrated pixel storing times is also quite tight. It should be noted that the stored calibrated pixels are only saved for possible diagnostic use, and are not required by the RTC pipeline. The last gradient required as input for the MVM is therefore available at the end of the pixel processing, which occurs shortly after the arrival of the last set of pixels.

2.2 Standalone MVM

In order to allow a better understanding of the performance of the matrix vector multiply, a standalone MVM program was written. This program was developed so that MVM performance could be easily be measured on hardware without requiring optimized networking. The user specifies the number of MVM threads per CPU, and the number of frames as command line arguments. The MVM program then creates the appropriate number of MVM threads and measures the execution time of the MVM computation for each frame. Frames involving a swap of the control matrix are timed separately from the frames which use the cached control matrix.

The MVM is performed using a column-wise approach, with each column of the reconstructor control matrix being stored as contiguous memory aligned on a 64 byte cache line boundary. The 64 byte alignment, and the use of column major ordering, allows efficient vector operations and is consistent with optimizing for both KNL Xeon Phi and current Xeon CPUs. In this standalone program, all of the gradients are available at the start of each frame. In the operating RTC pipeline, the gradients will become available as X-Y pairs for each subaperture, at which point the MVM will scale two reconstructor columns by the new pair of gradient components and add them to the result vector. This standalone program is intended to measure the best possible performance case of the MVM, so the gradients used as the input vector are not generated over the 500 microsecond WFS readout interval.

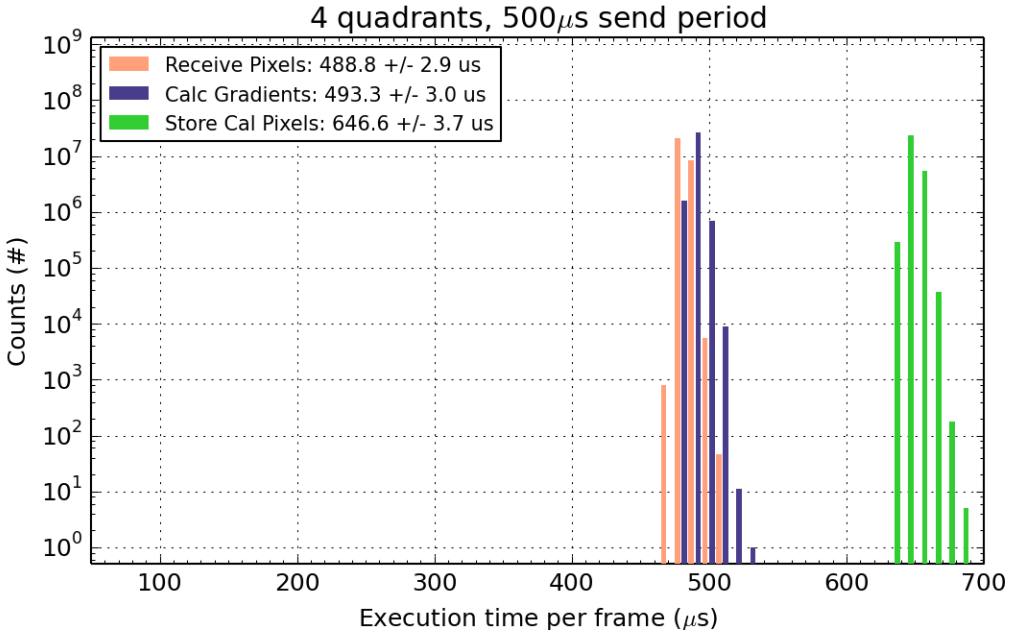

Figure 4 - Pixel processing times for full LGS WFS (~500 microsecond simulated readout)

Figure 5 shows MVM performance using both CPUs of a dual socket E5-2699 V3 server. Using 8 cores appears to minimize the worst case times at the expense of slightly longer mean execution time. Since the mean MVM execution time for 8 cores is only about 350 microseconds, there is no need to use more cores to speed up the average case. During the one hour of simulated operation, the worst case 8 core MVM execution times was 478 microseconds when the control matrix was in L3 cache, and 802 microseconds for an uncached control matrix. The increased tail of the distribution when using more threads is likely due to the requirement to wait for all threads to complete and the increased overhead of determining if the computation has completed. It should be noted, however, that the timings are not very sensitive to the number of CPU threads used above a minimum.

When only six CPU threads were used, the mean execution time increased to almost 410 microseconds and the tail of the distribution extends past 500 microseconds. The current RTC design assumes eight MVM threads per CPU if Xeon CPUs are used. It is expected that more threads will be required if the KNL Xeon Phi is used. The number of threads used with the delivered RTC hardware will be determined after benchmarking the final hardware.

Figure 6 shows the execution time histograms for a single CPU using 14 cores to perform the MVM corresponding to one LGS WFS quadrant. The blue bins are execution times using an E5-2699 V3 CPU running at 2.3 GHz, and the red bins are execution times using a Dell PowerEdge R930 (E7-8870 V3 CPU running at 2.1 GHz) borrowed from Intel. Due to a RAM shortage on the R930, only a single CPU could be used for the test. As expected, the higher clock speed of the E5 system results in faster execution when the control matrix is cached, but the high memory bandwidth of the E7 system results in better performance when a new control matrix is loaded from RAM. When the control matrix resides in the L3 cache, both systems are able to perform the MVM in less than the 500 microseconds required for WFS readout. Execution times for the single CPU are lower than the timings in Figure 5, since only one quadrant was processed and the code does not need to add the two intermediate DM error vectors together after completing the MVM for each quadrant.

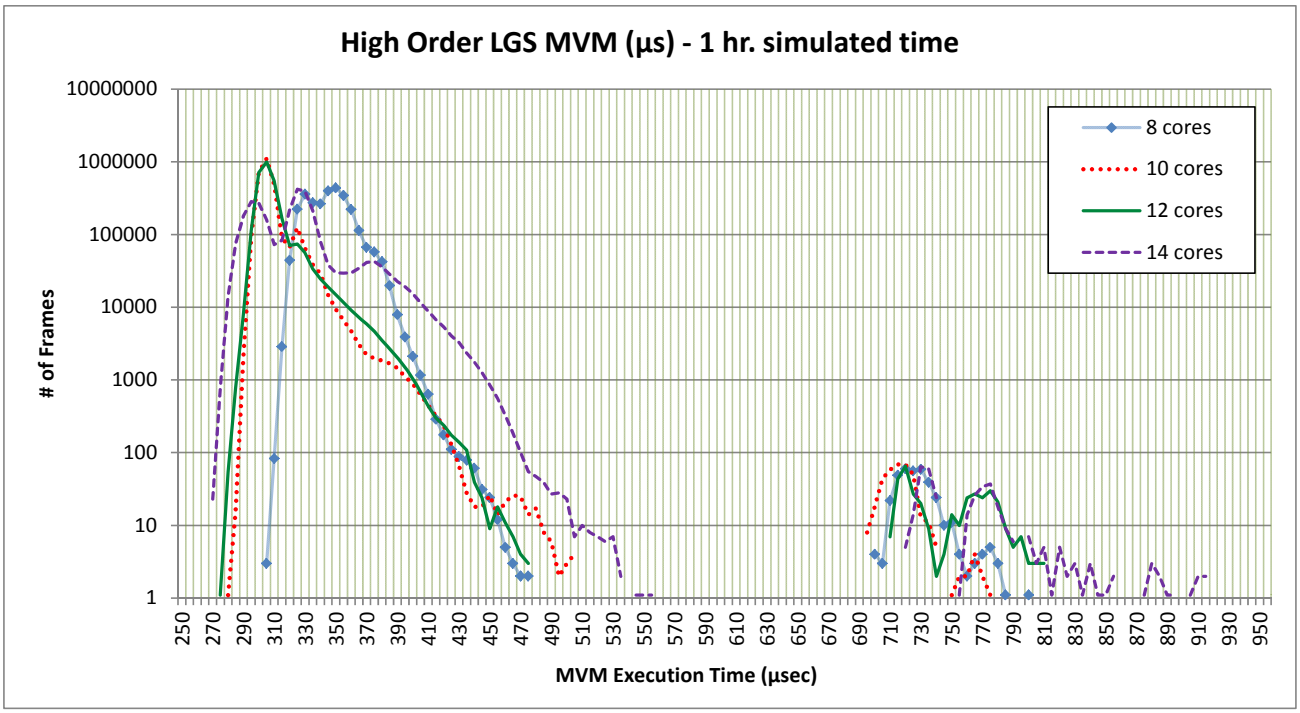

Figure 5 - Standalone MVM Time vs. # MVM Cores (control matrix swapped every 8000 frames)

2.3 Round Trip Timings

The timings in Figure 7 demonstrate the distribution of round-trip times for three different generation Xeon CPUs. The round trip time starts when a test server sends the first pixels of a frame to the HOP server, which performs the pixel processing and high-order MVM, and then sends back the DM error vectors. Each WFS frame was sent as a stream of pixels distributed over a 500 microsecond period. The frames were sent every 1250 microseconds (800 Hz) except when testing the 8 core E5-2670, in which case the frames were sent every 1600 microseconds. To mimic the delay caused by the arrival of a new control matrix, the MVM code would alternate between two control matrices every 8000 frames. Hyper-threading was disabled on all three servers, since enabling hyper-threading increases round trip time.[5] The servers were configured with the Linux real-time (PREEMPT_RT) patch, and CPU cores involved in the benchmarking were isolated using the isolcpus boot parameter.

Table 2 shows the average execution time for the pixels to DM commands round trip for three different generations of Intel Xeon E5 CPU. The original model was constrained by both number of cores and L3 cache size; only five cores were assigned to the MVM process, and the high-order reconstructor control matrix for one quadrant (~35MB) did not fit with the L3 cache. The second generation E5-2697 V2, with 12 cores, was only slightly constrained by core count for the benchmark code, but the control matrix still did not fit within the 30 MiB L3 cache. Finally, the third generation E5-2699 V3 allowed the entire control matrix to be stored within L3 cache, and the availability of 18 cores eliminated core sharing. It is expected that future CPUs will result in only modest performance gains since more cores and larger caches will have only a minor impact. On the E5-2699 V3 system, performing the MVM vector accumulation using double precision, adds approximately 250 microseconds to the MVM time when the control matrix is cached, and approximately 100 microseconds when the (single precision) control matrix is loaded from RAM. Future CPUs may allow the vector accumulation to be performed in double precision within 500 microseconds.

The discrepancy between round trip times for the cached and uncached control matrices shown in Figure 7 is reduced compared to the standalone MVM results, because in the round trip test, the pixels require 500 microseconds to be read (simulating WFS readout rate) and sent to the HOP server, thus creating a lower bound for the MVM execution time.

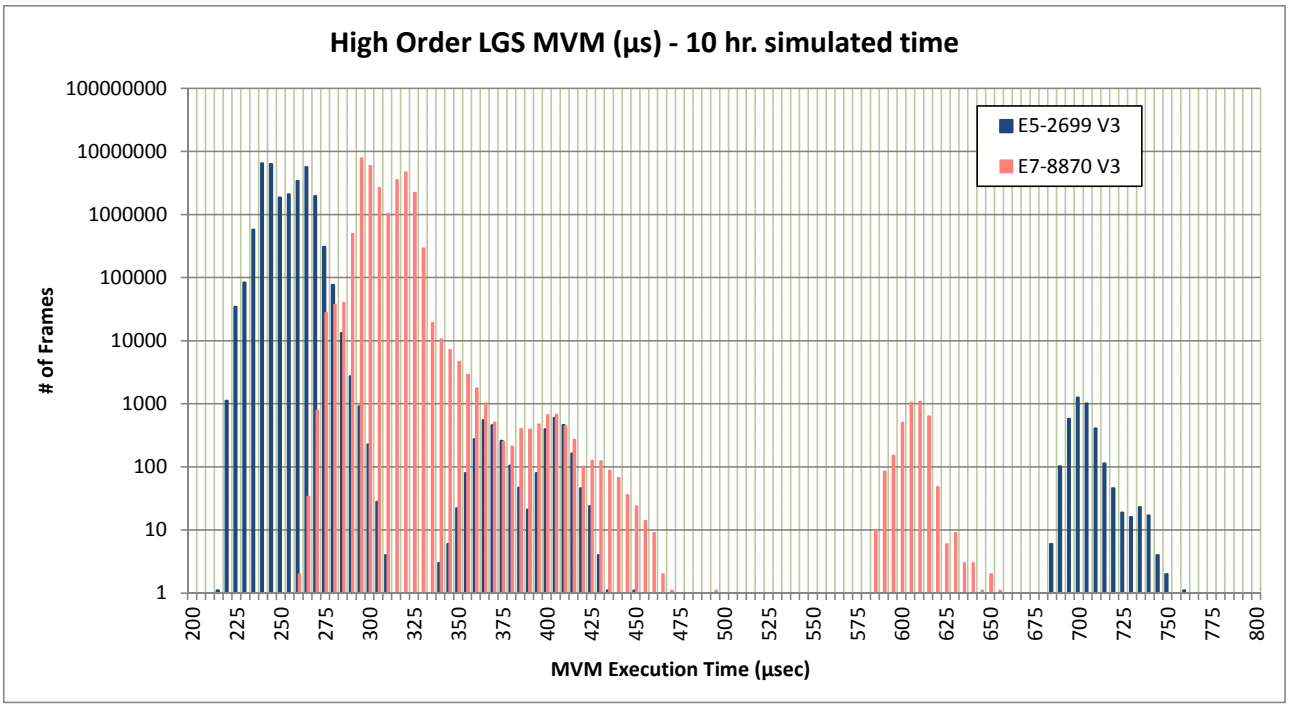

Figure 6 - Standalone MVM Execution Time

Table 2 - Round Trip Times vs. CPU Generation

CPU (Clock speed, cores, L3 cache)	Mean Execution Time (microseconds)	Maximum Execution Time (microseconds)
E5-2670 (2.6 GHz, 8c, 20MiB)	1144.8 ± 6.8	2212.3
E5-2697 V2 (2.7 GHz, 12c, 30 MiB)	895.4 ± 16.3	1223.3
E5-2699 V3 (2.3 GHz, 18c, 45 MiB)	602.7 ± 4.9	934.7

3. ADDITIONAL PROTOTYPING

3.1 Communication Performance

As seen in Figure 8, sending the DM error vectors from the six HOP servers to the WCC server requires approximately 75 microseconds, on average, using 40Gb Ethernet. This time estimate was arrived at during the preliminary design phase, and is based on simplified timings performed using 40Gb Ethernet hardware. In the pipeline timing diagram, Figure 2, a value of 100 microseconds was used since only a small number of frames required more than 100 microseconds for the DM vector transfer.

At the Preliminary Design Review, the design included a single 10/40Gb Ethernet switch, which provided communication links between the RTC servers. In order to reduce non-deterministic communication times, different data streams use physically distinct Ethernet links. This resulted in each RTC server requiring many Ethernet ports with the corresponding requirement on the switch.

During the preliminary design phase, the latency introduced by the prototyping switch (a Cisco Nexus 3172TQ containing 48 10GBASE-T ports and six 40Gb QSFP+ ports) was measured as 3.0-3.5 microseconds for 10GBASE-T links and 0.5-1.0 microseconds for QSFP+ ports.

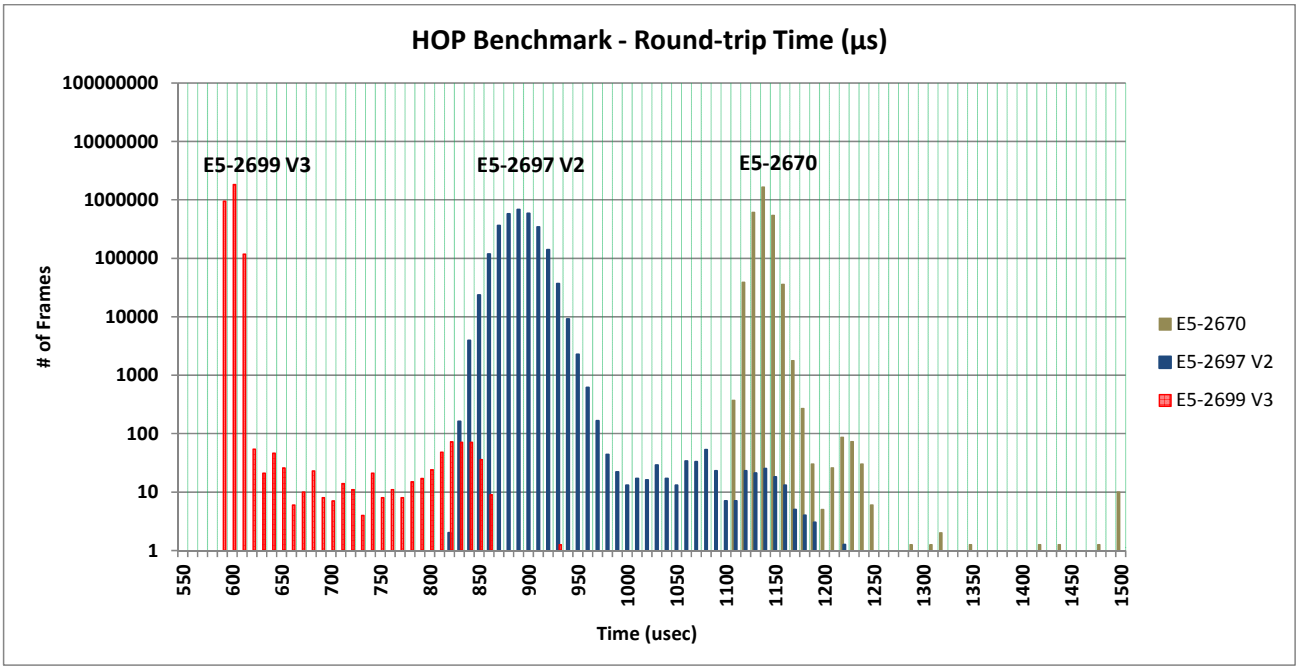

Figure 7 - HOP Benchmark - Round-trip Time

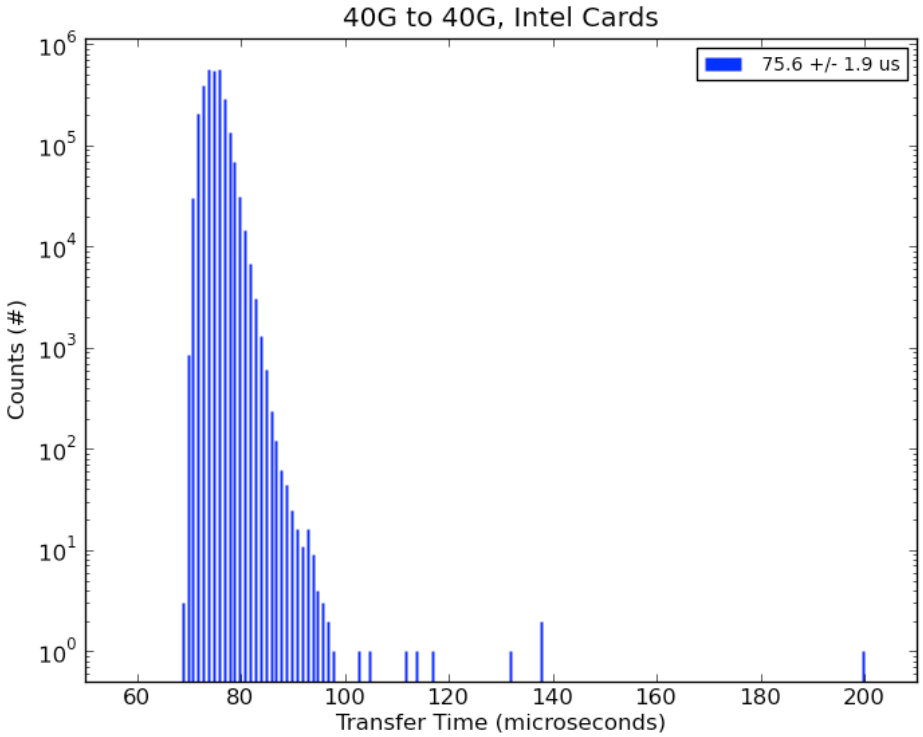

Figure 8 - HOP to WCC DM Error Vector Transfer Time

Late in 2015, Intel introduced Omni Path, a 100Gb fabric designed as a successor to Infiniband. This fabric (like Infiniband) provides lower latency and higher bandwidth than Ethernet. In addition, the fabric supports low level packet prioritization, so that high priority data streams can be immediately injected into the outgoing data stream. Using Omni Path will allow a reduction of the total number of communication ports used by the RTC.

3.2 Data Storage Tests

The PTS server provides fault tolerant storage for the data required by the PSF reconstructor. The PSF reconstructor is a separate TMT server that will read the required data from the PTS server and compute the point spread function for the evening's observations. The PSF reconstructor must complete the PSF computation prior to the next evening's start of observing.

In order to benchmark potential PTS performance, a Dell R730xd was purchased with ten 2TB spinning disks. These disks were configured as a RAID-6 array, with 12TB of usable storage, and two hot swap spares. Sequential write speed was measured by writing large data blocks to the RAID array. Initial sequential write performance was slightly over 1050 MB/sec, but as the disk was filled, the write performance decreased, reaching a minimum of about 625 MB/sec, when approximately 9 TB of data had been written. Subsequent writes exceeded 1 GB/sec, and peaked at 1100 MB/sec. This phenomenon was caused by the return of the disk heads from the inner tracks to the outer tracks in their search for unused space. After a disk reformatting, the first disk blocks used correspond to the fast outer tracks. If a large file is written and then deleted, future writes continue from the end of the large file, even though it has been deleted. This means that simply deleting all of the files on the disk will not guarantee that the next disk writes will use the fast outer blocks. Prior to the test illustrated in Figure 9, a 1 TB file had been created and deleted, shifting the start of free space 1 TB towards the center disk tracks. Worst case performance can be improved by short-stroking the drives (e.g. only using a portion of the disk corresponding to the outer tracks) but at the cost of unused disk space. It should be noted that the RAID-6 array provided an approximate speed up factor of 5.5 compared to a single disk. The hardware RAID controller provides good scaling since the maximum speed up possible for an eight disk RAID-6 array is six.

Disk tests on a single Intel DC S3510 SSD showed very consistent write performance of slightly greater than 450 MB/sec, even as the drive approached full capacity. The Intel specified write speed for the DC S3510 is 450 MB/sec. Given the superior performance and predictability of SSDs over spinning disks, their suitability to higher altitudes, and their expected decrease in cost per TB, the current RTC design assumes that SSDs will be used for all data storage.

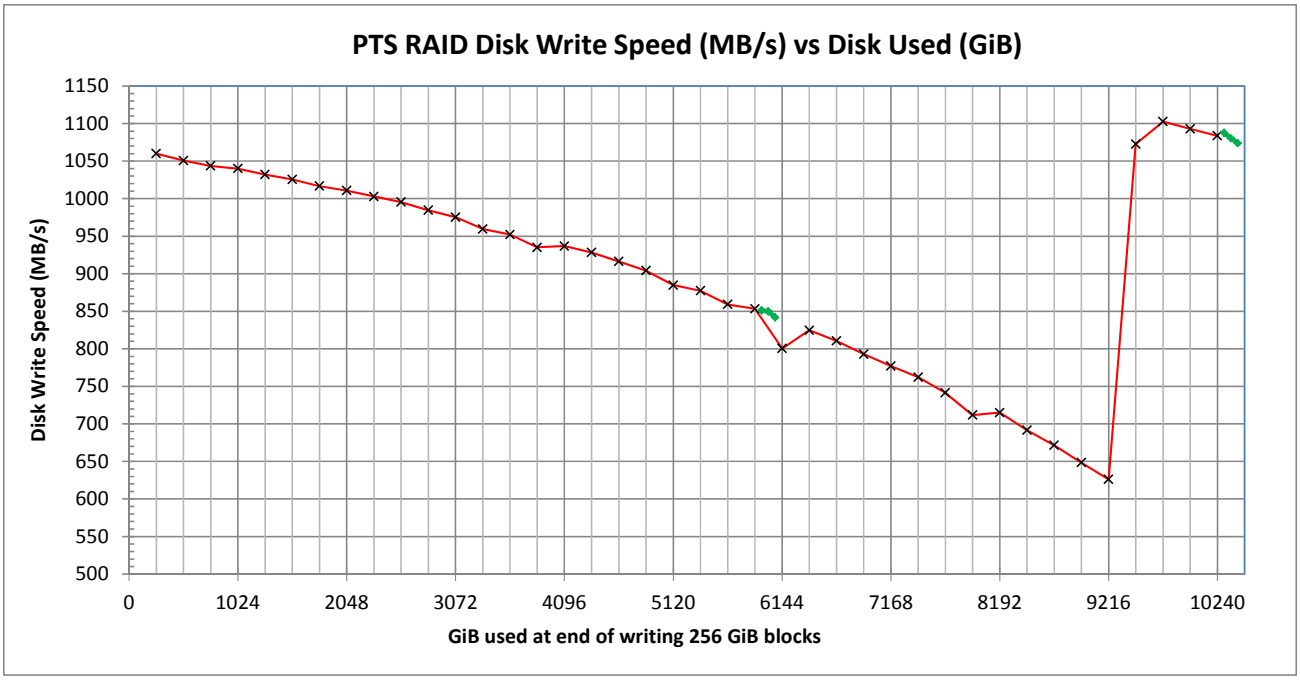

Figure 9 - RAID-6 performance of spinning disks

In addition to testing local performance of the RAID-6 array, a test was also performed which evaluated the impact of multiple disk failures on performance. Figure 10 shows the average and worst case sequential write performance during two simulated drive failures. Each second, the benchmark program writes 135 MB to the RAID-6 array using an NFS mount. The program performs a sync after each write, and times the duration of the write. The blue line shows the average write speed for each minute of writing, while the red line shows the slowest write speed within the one minute interval. The first two downward spikes, with write delays of approximately 10.0 and 11.4 seconds, respectively, correspond to the first and second simulated drive failures, that were simulated by physically removing a drive from the array. The last two downward spikes, with delays of approximately 4.7 and 6.9 seconds, respectively, correspond to when rebuilding of the first and second hot swap drives was completed. During the time between the first and last downward spikes, the RAID controller was actively rebuilding a failed (removed) drive onto a hot swap spare. The controller required approximately 4.5 hours to rebuild each disk. The controller does not try to rebuild the second failed disk until after it has completed the rebuild process for the first disk. This strategy reduces time spent in the vulnerable state of two failed disks.

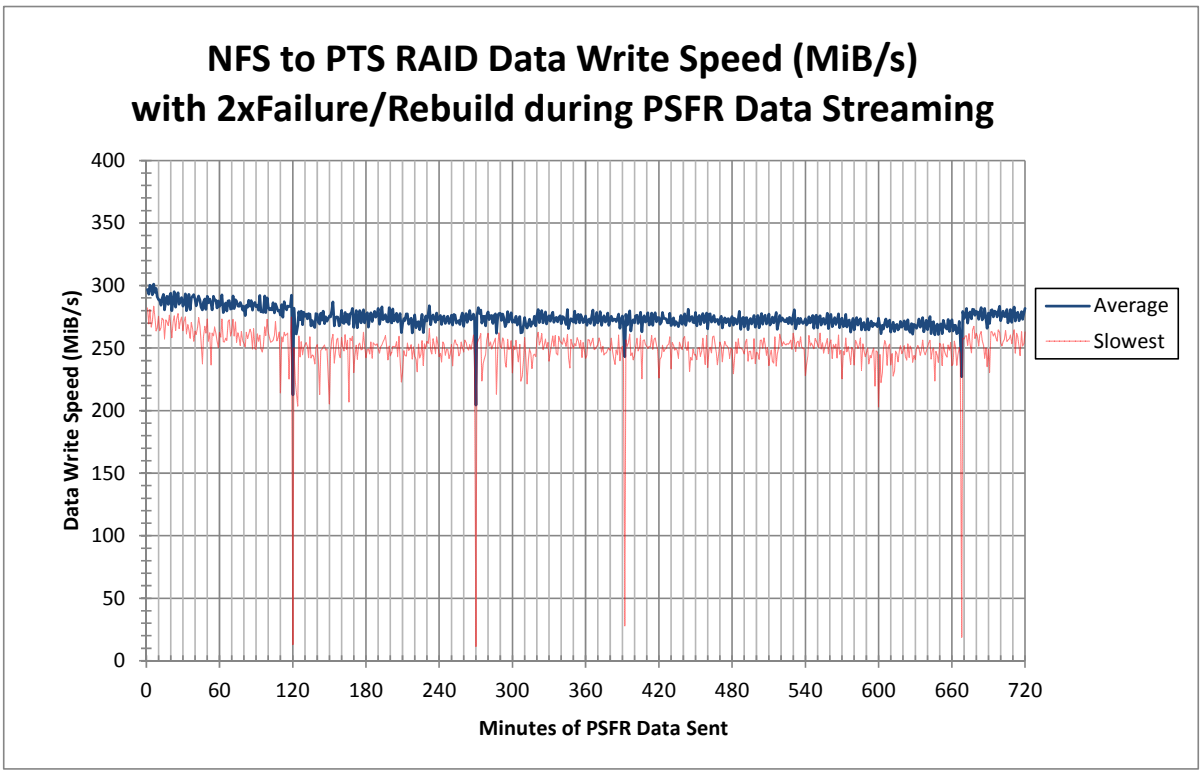

Figure 10 - RAID-6 array performance over NFS with two disk failures

3.3 Deformable Mirror Clipping

Prior to sending the deformable mirror actuator command values to the deformable mirror electronics, the actuator values are checked against absolute physical minimum and maximum limits. Actuator values that exceed the maximum (or minimum) allowable value are truncated to the maximum (or minimum) value. In addition, in order to avoid applying too much stress to the mirror face sheet, neighboring actuators must not be too far apart. Controlling the distance between neighboring actuators is called inter-actuator clipping. In this process, the actuators on each DM are scanned one row at a time. Each actuator is checked to see if it is too far away from any of the unscanned neighbors. If the difference between the actuator under consideration and one of its unscanned neighbors exceeds the allowable threshold, then both the actuator under consideration and the problematic neighbor have their commands adjusted towards each other so that their separation is just within the allowed separation. The scanning continues and adjusts any other actuator pairs that exceed the allowable separation. After scanning the entire DM, if any of the actuators were adjusted then the adjusted actuators are scanned again, checking all eight neighbors of each changed actuator. During the scan of

previously adjusted actuators, actuators positions are again adjusted as necessary. The scanning and adjusting process is repeated until no actuator position values are modified.

Table 3 lists the measured execution time required to perform the minimum/maximum and inter-actuator clipping for each of the two deformable mirrors under favorable conditions (no clipping). For the ground layer deformable mirror, DM0, the execution time required for worst case seeing is also measured. Although DM0 has fewer actuators than DM11, the higher altitude conjugation of DM11 results in less turbulence, and greatly reduced clipping.

The DM clipping process is applied independently to the two deformable mirrors by two separate threads. Converting the clipping algorithm into a more parallel algorithm has not been pursued.

Table 3 - Interactuator Clipping Execution Time

Scenario	Mean (μsec)	Max (μsec)	St. Dev. (μsec)
DM0 (3125 actuators) – no clipping	48.0	63.2	1.8
DM11 (4548 actuators) – no clipping	70.4	84.0	2.1
DM0 (3125 actuators) – worst case seeing	71.7	128.5	7.7

4. FUTURE WORK

4.1 HOP Prototype - Knights Landing Xeon Phi

We plan to benchmark a Knights Landing development system in late 2016. If a single Knights Landing Xeon Phi can be used in place of four Xeon E7 CPUs, then the total cost and power consumption of the RTC would be greatly reduced. It is expected that the end-to-end latency would slightly increase with the use of the Xeon Phi, so we will need to verify that the system still meets its performance requirements. The Knights Landing development system will be used to benchmark a full WFS MVM, and to benchmark sending DM vectors to the WCC server over Omni Path.

4.2 WCC Prototype - E5-2643 V4

A dual socket E5-2643 V4 system will be purchased in order to test the effectiveness of the new Resource Director Technology feature. This feature should reduce some of the timing variance in the RTC critical paths.

This system will also be used to measure the time required to perform the DM clipping. The DM clipping time should be lower for the new system due to the increased CPU clock speed. Since the new system will be the prototype WCC, it will also be used to measure the DM error vector transfer timings using an Omni Path adapter.

4.3 SSD RAID Storage

In addition to two high clock speed Xeon CPUs, the WCC prototype machine also includes three hardware RAID controllers and ~20 SSDs. This system will be used to investigate the feasibility of streaming all of the telemetry data to one or two servers. One of the Omni Path adapters will be installed in the WCC prototype machine to allow the streaming of telemetry data to the server while it, in turn, streams the data to one or more SSD RAID arrays.

The RTC must store approximately 120 TB of telemetry data per night. Prior to the next night's observing, any data not tagged for diagnostic purposes is deleted. The tagged data is copied to the PTS for later examination. During an observation, the RTC will be storing telemetry data at a rate of almost 3.5 GB/sec. Telemetry data storage details will be investigated further during the remainder of the final design phase. In particular, the WCC prototype machine will be used to the performance of streaming data to one or more SSD arrays.

4.4 Omni Path Fabric

NRC-Herzberg has purchased a pair of Omni Path adapters. Testing of the adapters was not possible prior to SPIE, however, due to the mismatch between the Omni Path software requiring version Red Hat 7.2 or equivalent, and our RTC prototyping machines currently running Scientific Linux 6.6. The Omni Path adapters will be put into the WCC

prototype and the KNL Xeon Phi developer system when they arrive. Later this year, our other prototyping machines will be upgraded to a newer operating system, to allow the use of the Omni Path adapters, if required.

The use of Omni Path is expected to significantly reduce the time required to send the DM error vectors from the HOP servers to the WCC server. This will be benchmarked by sending six DM error vectors to the WCC prototype server along with simultaneously streaming high volume low priority telemetry data. Sending low priority telemetry data will enable us to test the effectiveness of prioritization within the Omni Path fabric.

5. CONCLUSIONS

The NFIRAOS RTC is currently in the final design phase, and an important part of this design process is the prototyping and benchmarking of performance critical areas. Extensive prototyping also ensures that the design evolves in concert with verifications that it will meet its demanding performance requirements. With the exception of the KNL Xeon Phi, the current RTC design uses currently available COTS hardware and is expected to meet its performance requirements. The KNL Xeon Phi is expected to be available later this year (2016) and is expected to meet the performance requirements of the HOP server. Prototyping on both the KNL Xeon Phi developer system and the WCC prototype will allow a more educated refinement of the RTC design during the remainder of the final design phase.

6. ACKNOWLEDGMENTS

The TMT Project gratefully acknowledges the support of the TMT collaborating institutions. They are the California Institute of Technology, the University of California, the National Astronomical Observatory of Japan, the National Astronomical Observatories of China and their consortium partners, the Department of Science and Technology of India and their supported institutes, and the National Research Council of Canada. This work was supported as well by the Gordon and Betty Moore Foundation, the Canada Foundation for Innovation, the Ontario Ministry of Research and Innovation, the Natural Sciences and Engineering Research Council of Canada, the British Columbia Knowledge Development Fund, the Association of Canadian Universities for Research in Astronomy (ACURA) , the Association of Universities for Research in Astronomy (AURA), the U.S. National Science Foundation, the National Institutes of Natural Sciences of Japan, and the Department of Atomic Energy of India.

NRC-Herzberg gratefully acknowledges the generous support of the Intel Corporation for providing hardware access to allow our benchmarking to continue in a timely manner.

7. REFERENCES

[1] Herriot, G., Andersen, D., Atwood, J., Byrnes, P., Caputa, K., Fitzimmons, J., Hill, A., Lardière, O., Smith, M., and Stocks, J., "NFIRAOS AO for the Thirty Meter Telescope," in [*Adaptive Optics Systems V*], *Proc. SPIE* **9909** (2016).

[2] Herriot, G., Andersen, D., Atwood, J., Boyer, C., Byrnes, P., Caputa, K., Ellerbroek, B., Gilles, L., Hill, A., Ljusic, Z., Pazder, J., Rosensteiner, M., Smith, M., Spano, P., Szeto, K., Véran, J.-P., Wevers, I., Wang, L., and Wooff, R., "NFIRAOS: first facility AO system for the Thirty Meter Telescope," in [*Adaptive Optics Systems IV*], *Proc. SPIE* **9148**, 914810 (July 2014).

[3] Kerley, D., Smith, M., Dunn, J., Herriot, G., Véran, J.-P., Boyer, C., Ellerbroek, B., Gilles, L., and Wang, L., "Thirty Meter Telescope (TMT) Narrow Field Infrared Adaptive Optics System (NFIRAOS) real-time controller preliminary architecture," in [*Software and Cyberinfrastructure for Astronomy IV*], *Proc. SPIE* **9913-174** (2016).

[4] Khang, N. "Introduction to the Intel Resource Director Technology Features in Intel Xeon Processors E5 v4", https://software.intel.com/en-us/articles/introduction-to-the-intel-resource-director-technology-features-in-intel-xeon-processors-e5.

[5] Smith, M., Kerley, D., Herriot, G. and Véran, J.-P., "Benchmarking hardware architecture candidates for the NFIRAOS real-time controller," in [*Adaptive Optics Systems IV*], *Proc*. *SPIE* **9148**, 91484K (July 2014).

[6] Gardner, E. "What public disclosures has Intel made about Knights Landing?", https://software.intel.com/en-us/articles/what-disclosures-has-intel-made-about-knights-landing