

NGAO

Real Time Controller Processing Engine

Design Document

|  |  |  |  |
| --- | --- | --- | --- |
| Revision History | | | |
| Version | Author | Date | Details |
| 1.1 | Marc Reinig and Donald Gavel | May 8, 2009 | First draft version. Derived from MR’s draft version 4.17 |
| Ver. 1.1.134 |  | August 9, 2009 | Updated Draft |

Table of Contents

[1. Introduction 1](#_Toc237599526)

[1.1 Background and Context 1](#_Toc237599527)

[1.2 Functions of the RTC System 2](#_Toc237599528)

[1.3 RTC Architecture 4](#_Toc237599529)

[1.3.1 Wave Front Sensors (WFS) 4](#_Toc237599530)

[1.3.2 Tomography Engine 6](#_Toc237599531)

[1.3.3 DM Command Generator 6](#_Toc237599532)

[1.3.4 Control Processor (CP) 6](#_Toc237599533)

[1.3.5 Disk Sub-System 7](#_Toc237599534)

[1.4 Global Compute Engine System Timing 7](#_Toc237599535)

[1.5 Diagnostics and Telemetry 7](#_Toc237599536)

[1.6 Disk Sub-System 7](#_Toc237599537)

[2. Overall Latency, Data Rates and Timing 9](#_Toc237599538)

[2.1 Latency 9](#_Toc237599539)

[2.2 Non-Pipelined vs. Pipelined Architectures 10](#_Toc237599540)

[2.2.1 Non-Pipelined Architectures 10](#_Toc237599541)

[2.2.2 Pipelined Architectures and Latency 11](#_Toc237599542)

[2.3 Latency Calculations 13](#_Toc237599543)

[2.3.1 Transfer Latency 13](#_Toc237599544)

[2.3.2 Non-Transfer Latencies 14](#_Toc237599545)

[2.3.3 Total Latency 15](#_Toc237599546)

[2.4 Telemetry and Diagnostics Data Rates 15](#_Toc237599547)

[2.5 Timing, Control and Orchestration of States 15](#_Toc237599548)

[3. RTC Control Processor (CP) 16](#_Toc237599549)

[3.1 Interfaces and Protocols 16](#_Toc237599550)

[3.2 Data Flow and Rates 16](#_Toc237599551)

[3.3 State Machines Used by the RTC 17](#_Toc237599552)

[4. Cameras 18](#_Toc237599553)

[4.1 Interfaces and Protocols 18](#_Toc237599554)

[4.2 Data Flow, Rates and Latency 18](#_Toc237599555)

[4.2.1 Camera Frame Time 18](#_Toc237599556)

[4.2.2 CCD Read Time 18](#_Toc237599557)

[4.2.3 Camera Transfer Latency 18](#_Toc237599558)

[4.3 Master Clock Generation for Camera Synchronization 18](#_Toc237599559)

[4.4 AO Control 18](#_Toc237599560)

[5. Wave Front Sensors (WFS) 19](#_Toc237599561)

[5.1 WFS Interfaces 19](#_Toc237599562)

[5.2 Data Flow and Rates 19](#_Toc237599563)

[5.3 Centroid and Wave Front Reconstruction Engine 19](#_Toc237599564)

[5.3.1 FPGA implementation 20](#_Toc237599565)

[5.3.2 GPU implementation 20](#_Toc237599566)

[5.4 LOWFSs (Low Order Wave Front Sensors) 20](#_Toc237599567)

[5.4.1 Interfaces and Protocols 22](#_Toc237599568)

[5.4.2 Data Flow, Rates and Latencies 22](#_Toc237599569)

[5.4.3 Low Order Reconstruction Engine 22](#_Toc237599570)

[5.5 HOWFSs (High Order Wave Front Sensors) 23](#_Toc237599571)

[5.5.1 Tomography HOWFS 23](#_Toc237599572)

[5.5.2 Point-and-Shoot HOWFS 24](#_Toc237599573)

[6. Tomography Engine 27](#_Toc237599574)

[6.1 Systolic Arrays 27](#_Toc237599575)

[6.2 Voxels 27](#_Toc237599576)

[6.3 Processing Elements (PEs) and their Functions 28](#_Toc237599577)

[6.4 Interfaces and Protocols 28](#_Toc237599578)

[6.5 Data Flow and Rates 28](#_Toc237599579)

[6.6 Timing and Events 28](#_Toc237599580)

[6.7 Control and Orchestration of Processor States 28](#_Toc237599581)

[7. DM Command Generation 29](#_Toc237599582)

[7.1 Interfaces and Protocols 29](#_Toc237599583)

[7.2 Data Flow and Rates 29](#_Toc237599584)

[7.3 DM Control Engine 30](#_Toc237599585)

[7.3.1 Open Loop 30](#_Toc237599586)

[7.3.2 Non Linear Look-Up Tables 31](#_Toc237599587)

[8. DMs 32](#_Toc237599588)

[8.1 Types of DMs 32](#_Toc237599589)

[8.1.1 Woofer 32](#_Toc237599590)

[8.1.2 Tweeters 32](#_Toc237599591)

[8.2 Interfaces and Protocols 32](#_Toc237599592)

[8.3 Data Flow and Rates 32](#_Toc237599593)

[9. Tip/Tilt Mirrors 33](#_Toc237599594)

[9.1 Interfaces and Protocols 33](#_Toc237599595)

[9.2 Data Flow and Rates 33](#_Toc237599596)

[9.3 Tip/Tilt Control Engine 33](#_Toc237599597)

[10. Telemetry Data and Diagnostics Transmittal 34](#_Toc237599598)

[10.1 Interfaces and Protocols 35](#_Toc237599599)

[10.2 Data Flow and Rates 35](#_Toc237599600)

[10.3 Timing and Events 35](#_Toc237599601)

[10.4 Synchronization of AO Data Time Stamps with the Science Image Time Stamps 35](#_Toc237599602)

[10.5 An Important Note on the Telemetry Rates 36](#_Toc237599603)

[11. RTC Disk Sub-System 37](#_Toc237599604)

[12. Timing Generation and Control 38](#_Toc237599605)

[13. Requirements for all Custom System Components 39](#_Toc237599606)

[13.1 Acceptance Testing 39](#_Toc237599607)

[13.2 Diagnostics Capability 39](#_Toc237599608)

[13.2.1 Board ID, S/N, location reporting 39](#_Toc237599609)

[13.2.2 Power supply monitoring 39](#_Toc237599610)

[13.2.3 Clock monitoring 39](#_Toc237599611)

[13.3 Trouble shooting capability 39](#_Toc237599612)

[13.4 Cable Attachment Verification 39](#_Toc237599613)

[13.5 Temperature 40](#_Toc237599614)

[14. Other Hardware and Software not Otherwise Covered 41](#_Toc237599615)

[15. Power Distribution and Cooling 42](#_Toc237599616)

[15.1 WFS 42](#_Toc237599617)

[15.2 Tomography Engine 42](#_Toc237599618)

[15.2.1 FPGA Power Dissipation 43](#_Toc237599619)

[15.2.2 Tomography Engine Cooling 43](#_Toc237599620)

[15.3 DM command Generation 43](#_Toc237599621)

[15.4 Disk Sub-System 43](#_Toc237599622)

[15.5 Overall Power and Cooling Requirements 44](#_Toc237599623)

[16. Appendices 45](#_Toc237599624)

[Appendix A Time-related keywords in Keck Observatory FITS files 46](#_Toc237599625)

[Appendix B Wavefront Error Calculations Due to Latency 50](#_Toc237599626)

[Appendix C Split Tomography Issues 51](#_Toc237599627)

[Appendix D RTC Test Bench 52](#_Toc237599628)

[Appendix E System Synchronization 53](#_Toc237599629)

[Appendix F RTC Sensors and Actuators 54](#_Toc237599630)

[Appendix G Tomography Array Size and Cost Estimate 55](#_Toc237599631)

[17. References 56](#_Toc237599632)

**Table of Figures**

[Figure 1 Top-level view of the command and data environment 3](#_Toc237599633)

[Figure 2 RTC Overview 4](#_Toc237599634)

[Figure 4 HOWFS/LOWFS Pair 5](#_Toc237599635)

[Figure 5 Overall RTC Latency Components (not to scale) 9](#_Toc237599636)

[Figure 6 Non-Pipelined Latency Issues (not to scale) 11](#_Toc237599637)

[Figure 7 Pipelined Architecture Latency Issues (not to scale) 12](#_Toc237599638)

[Figure 8 RTC Control Processor (CP) 16](#_Toc237599639)

[Figure 9 RTC states viewable by the SRT 17](#_Toc237599640)

[Figure 10 LOWFS 21](#_Toc237599641)

[Figure 11 LOWFS Latency (not to scale) 21](#_Toc237599642)

[Figure 12 Tomography HOWFS 24](#_Toc237599643)

[Figure 13 Tomography WFS Latency (not to scale) 24](#_Toc237599644)

[Figure 14 Point-and-Shoot HOWFS 25](#_Toc237599645)

[Figure 15 Point-and-Shoot HOWFS Latency (not to scale) 25](#_Toc237599646)

[Figure 16 RTC Boards 27](#_Toc237599647)

[Figure 17 DM Command Generation Latency (not to scale) 29](#_Toc237599648)

[Figure 18 Science Object DM Command Generation 30](#_Toc237599649)

[Figure 19 Woofer Command Generation 30](#_Toc237599650)

[Figure 20 RTC Diagnostics and Telemetry 34](#_Toc237599651)

[Figure 21 Camera Synch and System Timing Generation 38](#_Toc237599652)

[Figure 22 Power Distribution 42](#_Toc237599653)

[Figure 23 RTC Test Bench 52](#_Toc237599654)

**Table of Tables**

[Table 1 NGAO RTC System Parameters. 13](#_Toc237599655)

[Table 2 Transfer Latencies and Rates 13](#_Toc237599656)

[Table 3 Non-Transfer Latencies 15](#_Toc237599657)

[Table 4 Total Latency Calculations 15](#_Toc237599658)

[Table 5 Preliminary FPGA Power Dissipation Analysis 43](#_Toc237599659)

[Table 6 List of sensors and actuators connected to the RTC 54](#_Toc237599660)

[Table 7 Estimate of the Tomography Engine Array Size and Cost 55](#_Toc237599661)

# Introduction

This document covers the design of the Keck Next Generation Adaptive Optics Real Time Controller (NGAO RTC).

This section is an overall introduction and summary of the RTC architecture and implementation.

Section 2 covers the latency and timing requirements that drive the architecture and implementation decisions.

Sections 3 on cover these areas in detail.

For details on the algorithm used, see [ref NGAO Algorithm Document].

## Background and Context

The design of the Real-Time Processor for the Keck Next Generation Adaptive Optics system represents a radically new computational framework that blazes new territory for astronomical AO systems. The need for a new approach is due to the considerable step up in the amounts of data to be processed and controls to be driven in a multiple guide star tomography system. Furthermore, KNGAO has set ambitious new goals for wavefront correction performance that puts further pressure on processing rates and latency times that traditional compute paradigms have trouble scaling up to. Current processing requirements are in the Terra Operations per second region. Instead, the KNGAO RTC is structured around a massively parallel processing (MPP) concept, where the highly parallelizable aspects of the real-time tomography algorithm are directly mapped to a large number of hardware compute elements. These compute elements operate separately and simultaneously. With this approach, increasing problem size is handled roughly by increasing the number of processor elements, rather than processor speed.

In short, in this design, the structure of the algorithm drives the structure of the hardware. Therefore, it is necessary to understand the algorithm itself to understand the hardware architecture, so please review the companion volume [ref Algorithms Design Document], which presents the technical details of the massively parallel algorithm for AO tomography.

For the purpose of introducing the hardware design, the algorithm and hardware have three main top-level component steps: wavefront sensing, tomography, and DM control. These physically map to component blocks of hardware, see . The actual physical layout of boards and connecting cables mimic this geometry. Inside the blocks, further parallelization is achieved by use of Fourier domain processing, where each spatial frequency component is processed independently. Fourier-domain algorithms have been developed in recent years for wavefront phase reconstruction [ref Poyneer] and for tomography [ref Tokovinin, Gavel].

As stated, the real-time processing takes place on small compute elements, either Field-Programmable Gate Array (FPGA) logic chips, or Graphical Processing Units (GPUs).

FPGA devices have been used in the digital electronics industry since the 1980’s when they were first used as alternatives to custom fabricated semiconductors. The arrays of logic were “field-programmed” to produce any digital logic function. They grew in a role as support chips on board computer motherboards and plug-in cards. Recent versions incorporate millions of transistors and some also incorporate multiple on-chip conventional arithmetic processors. Their use is now common in many products from high performance scientific add in boards to consumer portable electronics including cell phones. Today these chips represent a fast growing, multi-billion dollar industry.

Graphical Processing Units were first introduced to offload the video processing for computer displays from the computer’s CPU. They have since developed as a key processor for digital displays, which have demanding applications in computer-aided design and video gaming. GPUs specialize in geometric rotations, translations, and interpolation using massive vector-matrix manipulations. Beyond graphics, GPUs are also used in analyzing huge geophysical signal data sets in oil and mineral exploration and other scientific calculations requiring extremely high performance. GPUs can be adapted to the wavefront reconstruction problem through these means and, at this point, they remain a possible low-cost option in AO for wavefront sensor front-end video processing, but they are still under evaluation for this purpose.

The RTC uses a mix of GPUs and FPGAs to best match their individual strengths to the RTC’s needs.

## Functions of the RTC System

The RTC System is a real-time digital control system that operates on a fixed clock cycle. This is referred to as the Frame Clock or Frame Cycle. The same program runs repeatedly on every frame. The Processing Engine interacts with its environment via several classes of interface:

1. Ethernet is used two external interfaces

Command, status, and diagnostics are communicated between the system supervisory processor and the RTC control processor over Gbit Ethernet, using sockets and a simple protocol (see \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_).

Access to the disk sub-system is also through Gbit Ethernet.

1. Camera Link™

Wave front sensor cameras communicate to the wave front processors through Camera Link (an industry standard high-speed camera interface).

1. LVDS

Communication to the DM DAC hardware is through LVDS, an industry standard hardware interface.

The RTC controller includes a Control Processor (CP) based on a conventional Linux box. It acts as the synchronizing interface between asynchronous requests coming from the supervisory control system and the synchronized parameter loading to the interconnected Processing Engine components.

The “state” of the RTC is affected by the supervisory processor via a parameter loading process. This loading occurs in the background to the real-time cycles. Parameter updates to the PE’s [define] program occur synchronized to the frame clock. The companion document [ref Processing Engine to Supervisory Control Interface Design Document] describes the available control parameters and the update process in detail.

In order to accommodate differing guide star’s brightness, the rate of the camera clocks can be set differently on each wavefront sensor with respect to the tomography engine. This is useful to allow overall system optimization by trading individual wavefront or tip/tilt/focus sensor signal-to-noise ratio with sensor bandwidth. All sensor sample periods however must be less than the tomography sample period in order for the PE to remain synchronized. The master frame clock is derived from one of the HOWFS cameras.

Figure **1** shows a top-level view of the Processing Engine System in its computational and data flow environment.

**Database**

**Query /**

**Return**

***PE System***

**RTC**

**Control**

**Processor**

**RTC**

**Processing**

**Engine**

**Cameras**

**DMs/TT**

**Mirrors**

**Bit**

**-**

**Stream**

**Recording**

**Disk Farm**

**SC/RTC**

**Asynch**

**Interface**

***Super***

***-***

***visory***

***Control***

**Parameter**

**Load /**

**Status**

Figure 1 Top-level view of the command and data environment



Figure

Figure

Figure 4 RTC Overview

## RTC Architecture

The Processing Engine is physically laid out in a manner that reflects the real-time signal flow and the algorithmic structure. Figure **2** RTC Overview depicts the overall design, emphasizing real-time signal flow. Figure **19** RTC Diagnostics and Telemetry, highlights the flow of telemetry data and Figure **7** RTC Control Processor (CP), highlights the flow of parameter loads and status feedback to and from the control processor.

### Wave Front Sensors (WFS)

A short overview: There are 2 types of WFS in the NGAO RTC: low order (LOWFS) and high order (HOWFS).

LOWFS use an IR natural guide star (NGS) to estimate tip/tilt, focus and astigmatism for the science field.

A low-order wavefront sensor (LOWFS) processor is responsible for converting raw camera pixel data into a tip/tilt signal, plus focus and astigmatism numbers for the one TTFA sensor. Algorithm details are given in the [Algorithm Design document]. Each LOWFS wavefront sensor has an associated wavefront sensor processor. These operate in parallel then feed their aggregate results to the science tip/tilt actuator on the Woofer and to the tomography engine.

a

Figure HOWFS/LOWFS Pair

A high-order wavefront sensor processor is responsible for converting raw Hartmann WFS camera pixel data into an array of wavefront phase measurements spaced on a regular grid in a coordinate system that defined as common throughout the AO system. HOWFS use a laser guide star (LGS) or an NGS to probe the atmosphere and determine the high order atmospheric distortion in that direction. There are 2 types of HOWFS:

The Point-and-shoot HOWFS first generates a wave front to correct light in the direction of the LGS. This wave front is then converted to DM commands that are used to determine the shape of a DM. This DM is used to sharpen the associated LOWFS NGS.

Tomography HOWFS provide probing of the atmospheric volume containing the science object and generate a wave front in the direction of the LGS. These wave fronts are processed by the tomography engine and used to estimate the volume in the direction of the science object. The estimate of the volume is in turn used to provide a wave front to correct light in that direction. This wave front is passed to a DM command generator to place the desired shape on the science DM.

Each HOWFS wavefront sensor has an associated wavefront sensor processor. These operate in parallel then feed their aggregate results to the tomography engine.

The architecture for the hardware for all WFSs follow the massively parallel approach. First of all, each wavefront sensor has an associated wave front sensor processor card, operating in parallel with but otherwise independent of (other than the frame synchronization clock) every other processor card.

At the present time, it is not established whether the WFSs will be implemented using FPGAs or with GPUs.

### Tomography Engine

The tomography engine’s processors are mapped “horizontally” over the aperture. A given computational unit on this map is assigned a piece of the aperture and alternates processing a portion of the spatial or Fourier domain of it.

All computational elements run the identical program, albeit with different parameters and data. Each processor in the tomography engine is connected to its four neighboring elements (representing the next spatial frequency over in both directions) because is it necessary to shift data to neighbors in order to implement the Fourier transform and interpolation steps in the tomography algorithm.

### DM Command Generator

Finally, the tomography engine results are fanned out to the deformable mirror command generators, which are dedicated units, one per DM assigned the job of taking into account actuator influence functions and nonlinearities of each DM, in short, converting desired mirror shape (phase) into actuator command voltages.

Processing Engine Physical Layout

Processing Engine flow of telemetry data

Processing Engine flow of parameter loads

### Control Processor (CP)

#### Synchronization of the RTC with the Supervisory Processor

### Disk Sub-System

## Global Compute Engine System Timing

Sub msec accurate time stamping of data

## Diagnostics and Telemetry

All major data signals can be sent to the Telemetry/Diagnostic data path at the request of the supervisory processor. Any data in this path can be sent to either-or-both the supervisory processor, for diagnostics, or the RTC disk sub-system, for storage. See: Section 7.

Data that can be saved through Telemetry or viewed through Diagnostics include:

1. Raw Camera Data
2. Centroids
3. Wave Fronts
4. Tomographic Layers
5. Science on-axis High Order Wave Front
6. Science on-axis Woofer Wave Front
7. Science on-axis High Order Commands
8. Science on-axis Woofer Commands
9. RTC Current Parameter Settings

All Diagnostic information is time stamped accurate to 1 Frame Time.

## Disk Sub-System

The RTC Disk Sub-System is an extremely high performance, large storage system that is designed to capture system Telemetry data that can be generated by the RTC. This data can be generated at over 36 GBytes/sec. This system is designed to be able to capture this data for an extended period, but it is not intended to be an archival system for long term storage or access. The system has a capacity to store up to 60 terra bytes of data which could be filled in a matter of days of heavy use.

Likewise, no data base facilities are provided beyond normal directory services.

# Overall Latency, Data Rates and Timing

## Latency

The most significant element to overcome in an AO system is the latency between sampling the atmospheric turbulence and applying the compensation for it. See for calculations on the impact of latency components on the wavefront error.

It and the spatial sampling determine the rate at which calculations and data transfers must be made to achieve a given level of compensation.

These in turn determine the size of the computation engine that is needed to handle the problem. For the current NGAO specification, this is a terra operation per second problem, which directly affects and limits the choices of algorithms, architectures, and implementations.

The total RTC latency is made up of several components, illustrated below (not to scale).

b

Figure

Figure

Figure 8 Overall RTC Latency Components (not to scale)

The total latency is calculated from the midpoint of the camera stare time to the midpoint of the subsequent DM hold-time. Computational Latency includes only the portion of time due to our actual processing of the data.

Description of each element

## Non-Pipelined vs. Pipelined Architectures

Processing system architectures can be divided into two major categories: non-pipelined and pipelined.

### Non-Pipelined Architectures

In a non-pipelined system, data is brought in, processed, and sent out with latency, L, and an effective cycle time of L. New data is brought in at time 0 and corrections are made at time L, whereupon new data is brought in again. The update rate is 1/L.



Figure

Figure

Figure 11 Non-Pipelined Latency Issues (not to scale)

In a non-pipelined system, the frame rate is determined by the total latency of handling the data. In this case, it is the time to read the CCD plus the computational latency. The total latency is 2 frames (1 frame to handle the data, plus ½ a frame on the front end to account for the integration time of the camera and ½ a frame at the end to account for the integrated effect of the DM hold time).

### Pipelined Architectures and Latency

In a pipelined system, the processing is divided amongst several units. Each unit processes the data and passes it on to the next unit with the total time still being L, as above (assuming transfer times between units are negligible). This means the computational latency is the same, L. However, it can be seen that new data can be brought in as soon as the first unit is through processing its last data, and the output can be updated as soon as the last unit has processed its data. The update rate here is 1/M where M is the largest time spent by any individual unit. This rate is considerably faster than that in the non-pipelined case.



Figure

Figure

Figure 14 Pipelined Architecture Latency Issues (not to scale)

In this system, each element of the data processing is handled by separate hardware. This allows us to have several frames of camera data being processed somewhere in the pipe at the same time, just not in the same piece of hardware. The maximum frame rate is determined by the longest time it takes for any single element to process its data.

In our situation, the frame rate is determined by the CCD read time, which happens to be 500 µsec. All other individual computational elements are less than this. The time to handle the data is the time to read the CCD plus the Computational Latency (~3 Frames). The total latency is 4-2KH frames (3 frames to handle the data, plus ½ a frame on the front end to account for the integration time of the camera and ½ a frame at the end to account for the integrated effect of the DM hold time).

While in Figure **6**, the computational Latency ends on a frame boundary, it is actually asynchronous to the frame clock. This means the DM operates on a clock that is the same frequency as the frame clock, but shifted by a not necessarily integral amount. See Figure \_\_\_\_\_\_\_.

Assuming the time to handle the data is the same for both the pipelined and the non-pipelined case, the pipelined architecture will provide less total latency. It can require more hardware, however.

For the NGAO RTC, we use a pipelined architecture.

## Latency Calculations

Latency can be divided into three categories: transfer, compute, and fixed latencies.

Transfer latency is the result of moving data from one place to another, usually between physically distinct hardware.

Compute latency is due to the time it takes to apply an algorithm on the data, and includes any minor transfer latency that may occur during that process.

Fixed latency is due to system architecture issues that are separate from the above. In the case of the RTC, it is the result of the value of the frame clock and the time read the data out of the CCD subsequent from the actual time to transfer it to the WFS frame grabber.

shows the system parameters used in the following latency analysis.



Table 1 NGAO RTC System Parameters.

The frame rate of 2 KHz is driven by the maximum tomography error allowed (see Appendix B and [Requirements Doc]).

There are 64 subapertures across the primary aperture.

In order to avoid artifacts due to wrapping while processing Fourier data, the number of apertures used internally is 88.

### Transfer Latency

Table **2** shows the transfer latencies that impact the total latency of the RTC. Also shown are the data rates needed to support the required Telemetry of system data during operation.



Table 2 Transfer Latencies and Rates

**Notes:**

1. Note that Operational transfer times must take place in a small part of a frame (~50 µsec ~10% of a frame) whereas Diagnostic/Telemetry transfers may take an entire frame (~500 µsec). This leads to a much higher rate for Operational transfers.

The two key parameters calculated are the “Total Diagnostic/Telemetry Rate” and the “Operational Transfer Time”. The former determines the characteristics of the RTC Disk Sub-System and the later is part of the Total RTC latency calculation.

1. This is the time for a single camera to transfer the camera data to the Frame Grabber. A time of 50 µsec has been allocated which is consistent with a transfer using Camera Link full configuration. Also shown is the total data rate needed to save the camera data for all 7 cameras through the Diagnostic/Telemetry port.
2. If the frame grabber is a separate piece of hardware from the rest of the WFS, this is the time to transfer the camera data from it to the next stage of the WFS. Since camera data has already been saved if desired, there is no load indicated for the Diagnostic/Telemetry port.
3. If it is desired to save the centroids, this number is the amount of time to transfer the data over the Diagnostic/Telemetry port. No additional operational transfer load is required.
4. After the WFS has calculated the wave front, these numbers are the amount of time needed to transfer it to the tomography engine and the Diagnostic/Telemetry Port.
5. After the Tomography Engine has estimated the atmospheric volume, the data needs to be transferred to the DM Command Generator. If it is also desired to save the tomographic layer information, the amount of time to transfer the data over the Diagnostic/Telemetry port is also given.
6. The DM information from the Tomography Engine is in a spatial measure and the DM Command Generator generates the correct DM actuator voltages to best match the desired DM shape. These numbers are the amount of time it takes to transfer the correct shape to the DM. Additionally, the data rate for the Diagnostic/Telemetry Port is shown if the data is to be saved.

### Non-Transfer Latencies



Table 3 Non-Transfer Latencies

### Total Latency



Table 4 Total Latency Calculations

## Telemetry and Diagnostics Data Rates

In addition to the latency associated with processing the AO data and controlling the actuators, the NGAO system needs to be able to save key system telemetry data. This feature allows subsequent processing of the science data, atmospheric and system analysis, and the display real time diagnostics information.

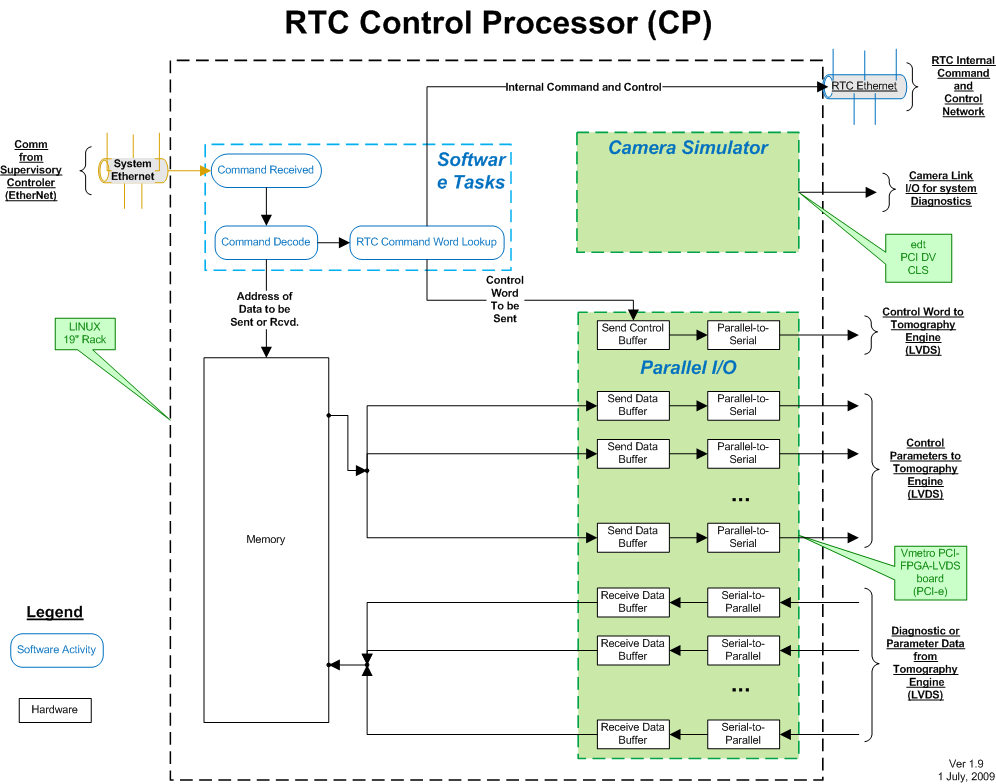
The amount of data it is possible to save is huge (many Giga Bytes per second) and a very large, fast disk sub-system is needed to capture this data prior to analysis.

## Timing, Control and Orchestration of States

# RTC Control Processor (CP)

The RTC system provides an asynchronous interface to the AO Supervisory Processor through the RTC Control Processor.

The interface is over Ethernet using sockets and the details of this interface are covered in a separate document.



Figure

Figure

Figure 17 RTC Control Processor (CP)

## Interfaces and Protocols

The CP’s interface to the rest of the RTC is through Gbit Ethernet, LVDS, or Camera Link busses.

## Data Flow and Rates

## State Machines Used by the RTC



Figure RTC states viewable by the SRT

# Cameras

The following applies to both the LOWFS and HOWFS cameras.

## Interfaces and Protocols

The interface between the camera sub-systems and the RTC will be through Camera Link™ using the Camera Link™ Full configuration.

## Data Flow, Rates and Latency

The following are the specifications the camera systems must meet in order for the RTC to meet the required error budget.

### Camera Frame Time

The camera must be able to sustain a 2 KHz frame rate

### CCD Read Time

After each frame is exposed, the CCD transfers the frame data to holding registers and starts exposing the next frame. The CCD must be able to transfer this data to the camera controller at a 2 KHz rate without affecting the simultaneous science object acquisition.

### Camera Transfer Latency

The data latency between the last byte transferred from the CCD and the last byte sent to the frame grabber must be less than 50 µsec. See:

## Master Clock Generation for Camera Synchronization

Cameras must be able to be synched to an external Frame Clock or be able to generate a Frame Clock to which other cameras can be locked.

## AO Control

The following controls must be supported be each camera:

Gain

Frame rate: 2 KHz, 1 KHz, 500 Hz, 100 Hz,

Frame Transfer rate

\_\_\_\_\_\_\_\_\_

…

# Wave Front Sensors (WFS)

## WFS Interfaces

The interfaces to the WFSs are shown in Figure **9**. Input will be via the machine-vision industry’s standard Camera Link cabling and communications protocol using the full configuration, unless otherwise noted. Output will connect to the tomography engine also via Camera Link communication. The 24 bits data width in this standard can represent two camera pixels each with a dynamic range of up to 2^12 = 4096 counts per pixel, which is above the maximum counts anticipated in NGAO wavefront sensing. The computed result from the WFSP does not increase the measurement limited signal-to-noise ratio so this 24 bit word width is also sufficient for transfer to the tomography engine, even though the tomography engine will ultimately be using a 36 bit internal word width to maintain computational accuracy.

The Camera Link interface cable is serialized low-voltage differential signaling (LVDS) pairs on an engineered standard cable (e.g. 3M’s Mini D Ribbon). The base configuration allows a throughput of up to 2.04 Gbit/s. For the 256x256 WFS chip running at 2 kHz frame rate, and 12 bits per pixel, the pixel data rate is 1.573 Gbit/s, which is under the base configuration limit and so does not demand an enhanced bandwidth configuration. Since the bit rate of Camera Link is generally faster than the processor clock (in the case of FPGAs), standard serializer/deserializer (SERDES) transceivers are needed to terminate the connections on each board.

Wavefront sensor processor cards are assigned one per wavefront sensor, while the tomography engine processors are mapped over the aperture, it is necessary for the four tomography WFSPs (each containing a full aperture’s worth of data) to distribute their results over the tomography engine processors. The distribution is accomplished using a specialized interface located along the left side of the tomography engine array, as shown in Figure **15**. This interface makes sure the correct data are shifted in along each row of tomography processors. The cable connection from the WFSPs to the tomography engine uses the same LVDS / Camera Link 2 Gbit/s standard as used for the input to the WFSPs, only this time there is far less data to transfer. The full-frame transfer of 64x64x2x12bit numbers (an extra factor two for complex Fourier coefficients) is accomplished in 50 microseconds, about 10% of a 2 KHz frame cycle. A doublewide Camera Link interface can be used if this is deemed to take too much of the latency budget.

## Data Flow and Rates

## Centroid and Wave Front Reconstruction Engine

### FPGA implementation

Using the Xilinx Vertex 5 chip, it is possible to fit 64 or more subapertures per chip, as established using the Xilinx design suite and simulation package. The entire 88x88-subaperture array (64 subapertures across with 12 on either side for zero padding to avoid excessive wrapping during deconvolution processing) can be fit onto a 12x12 array of chips. These can be engineered to fit on a 4x4 array of multilayer boards.

### GPU implementation

A 64 subaperture across (88 across for zero padding) reconstructor has been implanted on a single Nvidia Graphics Processor card. This presents a simple solution for the WFSPs that would not require the manufacturing costs of additional FPGA boards.

The GPU would reside on a PCI-E backplane slot on a CPU motherboard. The CPU can be used to configure and run the GPU. Additional PCI-E slots would house Camera Link interface cards, which in turn connect to the WFS camera for input and to the tomography engine for output.

We are currently analyzing whether we can utilize the GPU in the system at a 2 KHz frame rate.

## LOWFSs (Low Order Wave Front Sensors)

The LOWFSs take inputs from the 3 IR Tip/Tilt/Truth cameras. The cameras feeding the LOWFSs are focused on natural guide stars (NGS). They generate low order control for the on-axis correction of the light for both science and AO paths.



Figure

Figure

Figure 21 LOWFS



Figure

Figure

Figure LOWFS Latency (not to scale)

### Interfaces and Protocols

#### Inputs

The Inputs are from 3 IR cameras over Camera Link base configuration:

* Two tip/tilt cameras (size)
* One truth camera (Size)

#### Outputs

The outputs are:

* An analog X/Y controls to the on-axis Tip/Tilt stage of the Woofer.
* 3 analog X/Y controls to the tip/tilt mirrors associated with the 3 LOWFS camera.
* Digital Tip/tilt, focus, and astigmatism information sent to the tomography engine.

### Data Flow, Rates and Latencies

#### Camera Data

Each LOWFS camera has an image size of \_\_\_\_\_\_ and a data width of \_\_\_\_\_\_. This yields a data rate of \_\_\_\_\_\_ B/frame. This data is sent to a frame grabber in the LOWFS and alternatively to the Telemetry/Diagnostic signal path.

How long is the LOWFS camera stare time?

Do we update the Tip/Tilt actuator on the Woofer at this rate?

What is the impact on error budget of the shorter latency on LOWFS information to tomography and science tip/tilt?

#### Tip/Tilt Signals

The tip/tilt signals are analog voltages < ±10 V, with a rise/fall time of < 10 µsec, and a frame rate of 2 KHz.

#### Astigmatism and Focus

The astigmatism and Focus data are are sent to the tomography engine as digital data over 1 LVDS port.

### Low Order Reconstruction Engine

See the \_\_\_\_\_\_ document for the details on the algorithms for extracting tip/Tilt, focus, and astigmatism from the camera data.

## HOWFSs (High Order Wave Front Sensors)

Seven HOWFSs generate seven wave fronts used by the AO RTC system.

* Four HOWFSs generate tomographic information that corrects for atmospheric aberration affecting the science object(s).
* Three HOWFSs compensate for atmospheric aberrations affecting a corresponding LOWFS NGS.

To generate these wave fronts, the HOWFSs take inputs from the tomography or point-and-shoot cameras. These cameras can be focused on either natural or laser guide stars (NGS or LGS).

The images are first corrected for dark current, thresholded, and then processed to produce centroids corrected for reference centroids and offsets.

Tip/Tilt is then extracted from the centroids and subtracted from them.

The resulting centroids are then processed to produce a corresponding tip/tilt removed wavefront.

The Tip/Tilt extracted by each HOWFS is used to control the tip/tilt mirror associated with that particular HOWFS camera.

There is a small difference between a Tomography HOWFS and a Point-and-Shoot HOWFS.

### Tomography HOWFS

The Tomography HOWFS sends the reconstructed wave front to the Tomography Engine. See Figure **11**.

The latency contributed to the total RTC latency in Table **3** is the computational Latency shown in Figure **12**.



Figure

Figure

Figure 27 Tomography HOWFS



Figure

Figure

Figure 30 Tomography WFS Latency (not to scale)

### Point-and-Shoot HOWFS

The Point-and-Shoot HOWFS, however, further processes the reconstructed wave front to generate DM commands which are sent to a DM that corrects the laser spot associated with its corresponding camera, see Figure **13** and Section 7. Figure **13** illustrates the Point-and-Shoot HOWFS latency.



Figure

Figure

Figure 33 Point-and-Shoot HOWFS



Figure

Figure

Figure 36 Point-and-Shoot HOWFS Latency (not to scale)

# Tomography Engine

The tomography engine is implemented as a systolic array mapped to the physical volume of the atmosphere above the telescope.



Figure

Figure

Figure 39 RTC Boards

## Systolic Arrays

## Voxels

## Processing Elements (PEs) and their Functions

One PE can process data for multiple sub apertures. This allows us to trade off processing speed for system size.

## Interfaces and Protocols

## Data Flow and Rates

## Timing and Events

## Control and Orchestration of Processor States

# DM Command Generation

There are three types of DMs in the AO system:

* Science Object DM (4K Boston Micromachines)
* HOWFS DM (1K Boston Micromachines)
* Woofer (400 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)



Figure

Figure

Figure 42 DM Command Generation Latency (not to scale)

## Interfaces and Protocols

## Data Flow and Rates

## DM Control Engine



Figure

Figure

Figure 45 Science Object DM Command Generation



Figure

Figure

Figure 48 Woofer Command Generation

### Open Loop

### Non Linear Look-Up Tables

# DMs

There are three types of DMs in the AO system:

* Science Object DM
* HOWFS DM
* Woofer

## Types of DMs

### Woofer

### Tweeters

## Interfaces and Protocols

## Data Flow and Rates

# Tip/Tilt Mirrors

The RTC controls 10 tip/tilt mirrors (one for each WFS) and 1 on-axis tip/tilt actuator on the Woofer:

* 3 T/T mirrors associated with the 3 LOWFS cameras
* 7 T/T mirrors associated with the 7 HOWFS cameras
* 1 T/T actuator on the Woofer

## Interfaces and Protocols

## Data Flow and Rates

## Tip/Tilt Control Engine

# Telemetry Data and Diagnostics Transmittal

All major data signals can be sent to the Telemetry/Diagnostic data path at the request of the supervisory processor. Any data in this path can be sent to either-or-both the supervisory processor, for diagnostics, or the RTC disk sub-system, for storage. See: 7.

These signals include:

* Raw Camera Data

4 tomography, 3 point-and-shoot, 2 T/T, and 1 Truth sensor

* Centroids
* Reconstructed Wave Fronts

4 tomography and 3 point-and-shoot wave fronts

* Tomography Layers
* Science DM, Woofer, and TT commands
* Currently it is not planned to save the tomography or point-and-shoot DM commands



Figure

Figure

Figure 51 RTC Diagnostics and Telemetry

Diagnostic data is captured at rates of approximately 1 sample per second, as determined by the supervisory processor.

A separate frame grabber associated with the disk sub-system captures camera data for diagnostics and telemetry. This data is a parallel stream supplied by a Camera Link splitter, shown in Figure **19** RTC Diagnostics and Telemetry.

## Interfaces and Protocols

## Data Flow and Rates

## Timing and Events

## Synchronization of AO Data Time Stamps with the Science Image Time Stamps

The RTC system will have a sub millisecond accurate clock, GPS based to time stamp the AO data for telemetry and diagnostics and provide to the SRT.

* How will we use atmospheric data collected during the capture of science images for post processing (e.g. PSF)?
* To what accuracy must the time stamps on the science image and the AO data match?
* How to determine what time slice of the atmospheric data corresponds to a science image?
* Currently, science image shutter open and close times are time stamped to a precision nominally of a small fraction of a second but may be as much as ±30 seconds in error (see Appendix A, Time-related keywords in Keck Observatory FITS files).
* To what do we reference the RTC data and how?
* How to relate the AO data time stamps to the times in the science data at which the shutter was actually opened and closed?

## An Important Note on the Telemetry Rates

Since the total storage rate can exceed 36GB/sec if all possible data sets are captured, care should be taken to insure there is enough space remaining in the disk sub-system to save the data. The disk sub-system can support multiple TB of data, but after a number of nights of heavy use, a great deal of the storage may be used.

The disk sub-system is not meant to archive data, merely to store it temporarily until the desired data can be archived to another location and the undesired data can be deleted.

# RTC Disk Sub-System

All major data signals can be sent to the Telemetry/Diagnostic data path at the request of the supervisory processor. Any data in this path can be sent to either-or-both the supervisory processor, for diagnostics, or the RTC disk sub-system, for storage.

Since the total storage rate can exceed 36GB/sec if all possible data sets are captured, care should be taken to insure there is enough space remaining in the disk sub-system to save the data. The disk sub-system can support multiple TB of data, but after a number of nights of heavy use, a great deal of the storage may be used.

The disk sub-system is not meant to archive data, merely to store it temporarily until the desired data can be archived to another location and the undesired data can be deleted.

# Timing Generation and Control



Figure 52 Camera Synch and System Timing Generation

# Requirements for all Custom System Components

## Acceptance Testing

All components must have an established acceptance test requirement as part of their individual design package.

## Diagnostics Capability

Each board must have the ability to have a minimum set of diagnostics performed on it while in the system and operating normally. The following are required for all boards. Requirements for specific board types are specified in their section.

### Board ID, S/N, location reporting

#### Board ID

#### S/N

#### Location

If there are multiple identical board of identical types installed (i.e., Tomography boards or WFS boards) there must be a means of identifying in which position the board sits in the arrangement of identical boards. To insure that it this cannot be incorrectly set, this should be set automatically by its position in the installation.

### Power supply monitoring

Sampling rate, sample depth, trigger capabilities, threshold capabilities.

### Clock monitoring

Clock monitors

## Trouble shooting capability

Basic trouble shooting procedures will need to be performed occasionally: voltage or waveform viewing with a meter or scope, …

If an extender is to be used to perform these operation, the operation of the board in a system during normal operations must be demonstrated using such an extender.

## Cable Attachment Verification

Cables and connectors should be designed in such a manner that a device at one end can determine whether the cable is plugged in to the receptacle at the other end.

Circuitry will be included in each board for each cable to perform this analysis and report its result during diagnostics.

**Note:** If this feature cannot be implemented, then this requirement must be specifically waived for the specific board type and connector in question.

## Temperature

# Other Hardware and Software not Otherwise Covered

# Power Distribution and Cooling

Power to the various sub-systems will be controlled by a networked power control system such as a Pulizzi. These are controlled by commands through the CP by the system supervisory processor. The power to the CP and the Disk Sub-System will be controlled directly by the supervisory processor.

Figure **21** shows the power control for the RTC.



Figure 53 Power Distribution

## WFS

## Tomography Engine

### FPGA Power Dissipation

The majority of the power dissipated by the tomography engine is in the FPGA’s. There will be approximately 150 FPGAs, each dissipating a maximum of approximately 15 Watts for a total of 2.25 KW.

#### Sample Chip and Configuration

The initial evaluation of the FPGA design was based on the following estimated parameters in Table 5:

| **Item** | **Value** | **Comment** |
| --- | --- | --- |
| Chip: | VSX95T |  |
| System Clock: | 100 MHz |  |
| LVDS Clock: | 600 MHz |  |
| Sub apertures per chip: | 100 (4 sub apertures per PE) |  |
| Process Parameters | Nominal |  |
| Ambient Air Temp | 30° C |  |
| Air Flow | 250 LFPM (moderate air flow) |  |
| Heat Sink | High Profile (I like cool junctions) |  |
| Junction Temp (θJ) | 52° C (Nice) |  |
| I/O utilization: | 100% |  |
| DSP-48 Mult/Acc utilization | 40% |  |
| Logic utilization | 56% |  |

Table Preliminary FPGA Power Dissipation Analysis

#### Initial Power Estimates per Chip

It results of the preliminary power estimate per chip was a worst case of < 15 Watts.

### Tomography Engine Cooling

The initial results of the pre design power estimator tool look good. We’re not pushing any limits yet. No exotic cooling requirements seem indicated. A more detailed examination will be done later with a sample design actually synthesized and routed into a chip.

## DM command Generation

## Disk Sub-System

## Overall Power and Cooling Requirements

# Appendices

* 1. Time-related keywords in Keck Observatory FITS files

[[[1]](#endnote-1)]

As an overview of the time-related keywords found in FITS files from the optical instruments at Keck Observatory please, see this [computer-generated documentary list](http://spg.ucolick.org/cgi-bin/Tcl/runRpt.cgi?target=KeckObsTime&type=MemeByNam).

The two meanings of "keyword" at Keck

In the subsequent discussion, it is important to distinguish between KTL keywords and FITS keywords.

KTL keywords are obtained by interprocess (and often intermachine) communication when a client process indicates interest. The values of the KTL keywords communicate information about the state of the various subsystems of the Keck telescopes and instruments. Depending on the nature of the quantity described by the keyword it may be continuously varying during an observation or it may never change. Each KTL keyword is associated with a "service" that corresponds to some subsystem of the operations at Keck. Some KTL services are the responsibility of the Keck staff and describe observatory-wide systems (e.g., DCS describes the telescope pointing and dome, ACS describes the primary mirror). Other KTL services are associated with the particular instrument (e.g., CCD subsystems, motor control subsystems, etc.).

FITS keywords are recorded in the headers of the files, which are produced by the instrument systems when they obtain an image. In many cases, there is a one-to-one correspondence between KTL keywords and FITS keywords. There are, however, a number of FITS keywords inserted into every image that do not correspond to any KTL keyword. There can also be FITS keywords, which correspond to KTL keywords, but where the FITS keyword name does not match the KTL keyword.

Finally, it is relevant to note that the FITS DATE keyword indicates a time stamp related to the construction of the FITS file itself. There should be no expectation that the value of the FITS DATE keyword is related to the time at which the data in the FITS file were acquired.

Historical review of Keck timing keyword systems

During the development of the keyword handling systems at Keck Observatory, there was no requirement for a method of obtaining a precise time stamp. None of the specifications for the initial optical instruments at Keck (HIRES, LRIS, ESI, DEIMOS) included any requirements for precise timing. As a result, the ability of these instruments at Keck to indicate the time for any exposure-related event is haphazard.

The Keck telescope pointing control system (DCS) uses commercial GPS receivers as the time base. The time provided by these receivers has experienced "interesting" events during the GPS W1K rollover in 1999 and during theater-level GPS jamming experiments conducted by DoD. It is not clear what the DCS system reports during a leap second. Other than in these exceptional conditions, the time base used by DCS is relatively reliable.

Any of the Keck instruments is able to ask DCS for its current values of its timing keywords. Most notable among these KTL keywords are DATE-OBS, UTC, and MJD. Reading the KTL DATE-OBS keyword from DCS provides a FITS Y2K-agreement-compliant value of the calendar date according to Coordinated Universal Time. Reading the KTL UTC keyword from DCS provides a sexagesimal character string representation of the value of Coordinated Universal Time. It is important to note that these are two separate keywords. It is not possible to request both keywords simultaneously, nor to receive values, which are guaranteed to refer to the same instant. As a result, it is possible that around 0 hours UTC the values of the KTL DATE-OBS and UTC keywords from DCS can refer to different calendar days -- thus resulting in a one-day discrepancy in the available notion of the time. (Given that 0 hours UTC occurs around mid-day in Hawaii this will probably never be an issue for observational data.)

The manner by which DCS can be queried for date and time is a KTL keyword request. In the current scheme, the KTL keyword read generates a EPICS request that is sent over the network to the DCS systems. After that request is received and processed in DCS the resulting value is sent back over the network to the KTL requestor. The design of the system provides no guarantees about how long this round trip takes. Under normal circumstances the response to the KTL read is received within a small fraction of a second, but under adverse circumstances the DCS system has been observed to take as long as 30 seconds. In these adverse circumstances, it is not clear how to ascertain what point in the transaction is represented by the time value.

Gathering the time-related keywords at Keck

The initial optical instruments at Keck (HIRES, LRIS, ESI, DEIMOS) monitor the sequence of events of an exposure using a process named watch\_ccd. The watch\_ccd process is responsible for gathering most of the KTL keywords, which will be written, into the FITS file along with the image data of an exposure. The watch\_ccd process has a list of KTL keywords, which indicates when each keyword should be gathered during the exposure sequence. The three points during the exposure are erase, shutter open, and shutter close.

In these instruments, the computer, which has command of the CCD, is called the CCD crate. The CCD crate issues signals to the CCD controller to initiate all operations of the CCD electronics and the shutter. When the CCD crate issues exposure-related signals to the CCD controller, it also transmits a MUSIC broadcast message to the traffic process. The traffic process then re-transmits that MUSIC broadcast message to every other process, one of which is watch\_ccd. Each of these hops between processes on the network takes time.

When watch\_ccd receives an exposure-related event it proceeds to issue KTL read requests for each of the keywords whose value is desired at that event. These KTL read requests are issued sequentially. Each request must complete before the next KTL read is done. A successful KTL read involves a round trip between watch\_ccd and the system(s), which know(s) the value(s) of each keyword. Usually each KTL read succeeds in a small fraction of a second, but under adverse circumstances, the servers for the keywords may respond slowly, or not respond at all.

Each KTL keyword read has a timeout. If the server does not respond within that timeout a KTL read will fail, and watch\_ccd will proceed to read the next keyword in its list. There are often dozens, even hundreds of keywords in the lists that watch\_ccd has to acquire. The total time to gather the keywords, even under ideal circumstances can be several seconds, and under adverse circumstances, it has been monitored to be as many as 30 seconds or more. If the delay is long then the values of the KTL DATE-OBS and (more importantly) UTC keywords from DCS may not bear much relation to the instant at which the shutter events happened.

Strategies employed to get the best possible time stamps

These issues regarding the validity of the time-related keywords came to the attention of the UCO/Lick Scientific Programming Group during the retrofit, which added the exposure meter to HIRES. Starting with that deployment the CCD readout software has been tailored to provide the best possible indication of event times given the constraints of systems, which were not specified with that as a design requirement. These principles are now in use with HIRES and DEIMOS. They will be applied to LRIS during the Red Mosaic upgrade.

The first remedy was to carefully arrange the order of the keywords in the lists provided to watch\_ccd. In the more recent deployments, the list has the KTL UTC keyword as the first in the list. There can be no quicker way of getting that keyword from DCS.

Another remedy was to enhance watch\_ccd to distribute the keyword collection over three events rather than two. Initially the keywords were only gathered in response to shutter open and shutter close. Those lists were rather large, and not optimally arranged. In the more recent versions of watch\_ccd, it is possible to gather keywords in response to the initiation of the erase of the CCD prior to opening the shutter. The objective is to gather keywords whose value tends not to change during erase, and thus reduce the length of the lists of keywords desired at shutter open and shutter close.

Another remedy was to enhance watch\_ccd to be able to rename a KTL keyword when writing it to the FITS file. This renaming is often done when it is desired to sample the value of a KTL keyword more than once during an observation. In the case of HIRES the KTL UTC and DATE-OBS keywords are read in response to both shutter open and shutter close. For shutter open, they are written to the FITS file with the same name, and for shutter close, they are written to the FITS file with the names UTC-END and DATE-END. This has been in use for all instruments deployed or upgraded since the HIRES exposure meter. (Note that in the unusual case of an exposure very near to 0 h UTC the combination of all four of these keywords can probably be used to ascertain whether the pair of keywords for one of the shutter events may have straddled the day boundary.)

The most definitive remedy was to create an independent method for watch\_ccd to give some indication of the times of the shutter events. In recent versions when watch\_ccd receives the shutter events, it immediately queries the system to get the UNIX system time. The values of these queries are inserted into the KTL keywords DATE\_BEG (for shutter open) and DATE\_END (for shutter close). These keywords are only as precise as one second. Furthermore, these keywords rely on the UNIX system clock being set correctly. In normal operation, the Keck systems should be using NTP to keep their clocks on time, but this is not guaranteed.

Interpretation of time keywords in Keck FITS files

In any instrument, which supplies the FITS DATE\_BEG and DATE\_END keywords along with the UTC and UTC-END keywords, it is prudent to inspect their values. If the various subsystems were operating nominally then the keywords for shutter open will be in reasonable agreement, as will the keywords for shutter close. Furthermore, the difference between the times of shutter close and shutter open should be in reasonable agreement with the various keywords that indicate the duration of the exposure. If all of these agree, then their values are probably reliable. If they do not agree, then their discrepancies give some indication of how unreliable the time stamps may be.

Steve Allen <sla@ucolick.org>

$Date: 2008/12/08 $

* 1. Wavefront Error Calculations Due to Latency
  2. Split Tomography Issues
  3. RTC Test Bench



Figure 54 RTC Test Bench

* 1. System Synchronization
  2. RTC Sensors and Actuators



Table 6 List of sensors and actuators connected to the RTC

* 1. Tomography Array Size and Cost Estimate



Table 7 Estimate of the Tomography Engine Array Size and Cost

# References

1. Steve Allen, <http://www.ucolick.org/~sla/fits/kecktime.html>, Jan 28, 2009

   PAGEREF [↑](#endnote-ref-1)