Real Time Controller Functional Requirements

Don Gavel &
Marc Reinig

UCO Lick, laboratory for Adaptive Optics.

Keck NGAO Team Meeting
December 13, 2007
Videoconference

Outline

- Status
- Issues
- Plans

Status

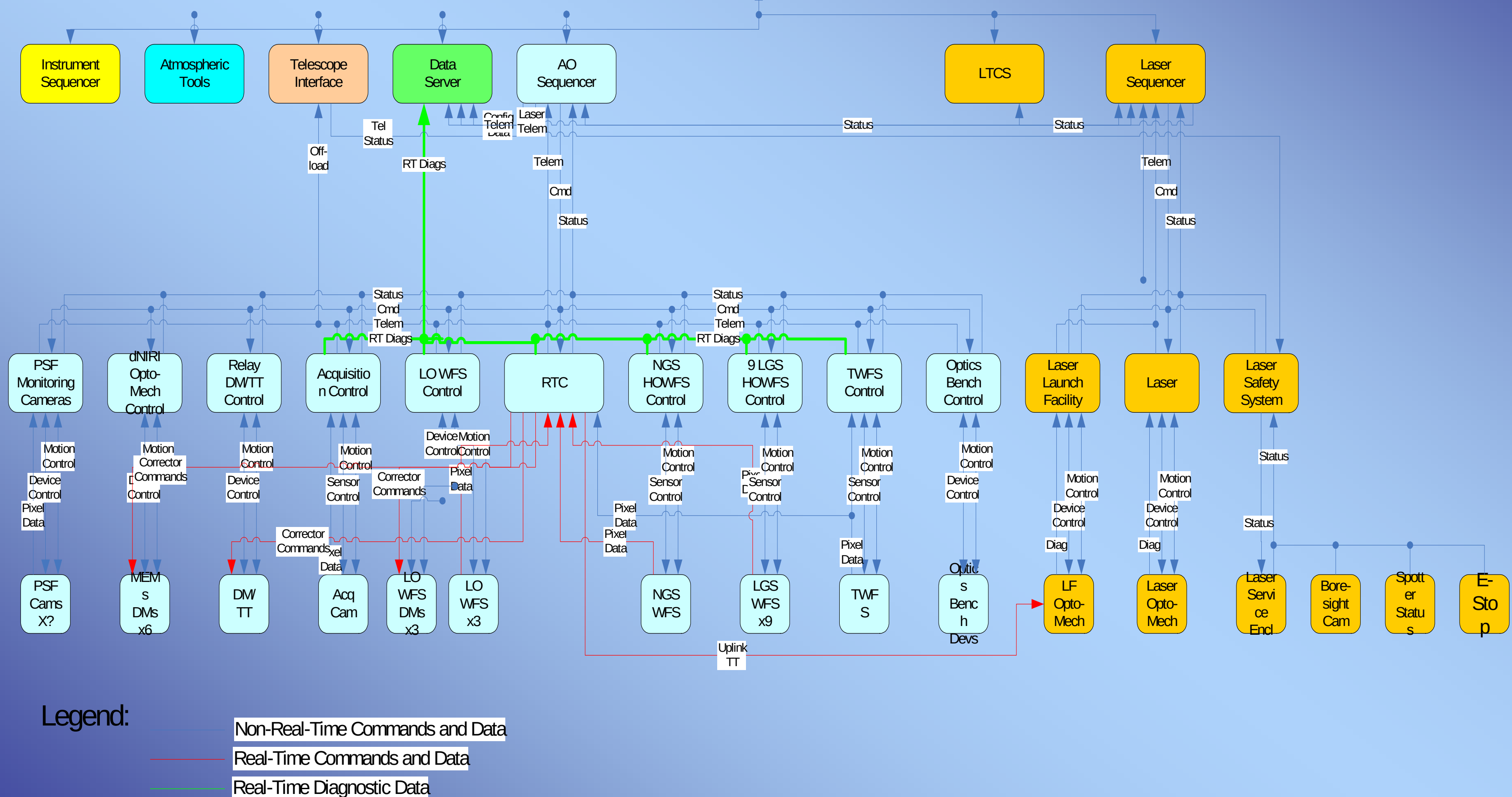
- We have an architecture that meets the performance requirements
  - 9 LGS/WFS (64x64)
  - 2 Tip/Tilt WFS
  - 1 T/T-focus-astigmatism WFS
  - 1 Truth WFS
  - 5 DM’s (with T/T) (64x64)
  - 1 Woofer
  - <500µSec from receipt of image to commands received at the DM
- Scalable to larger problems
- We have validated the software in the lab and through simulation

Architecture

– Hardware and Software –

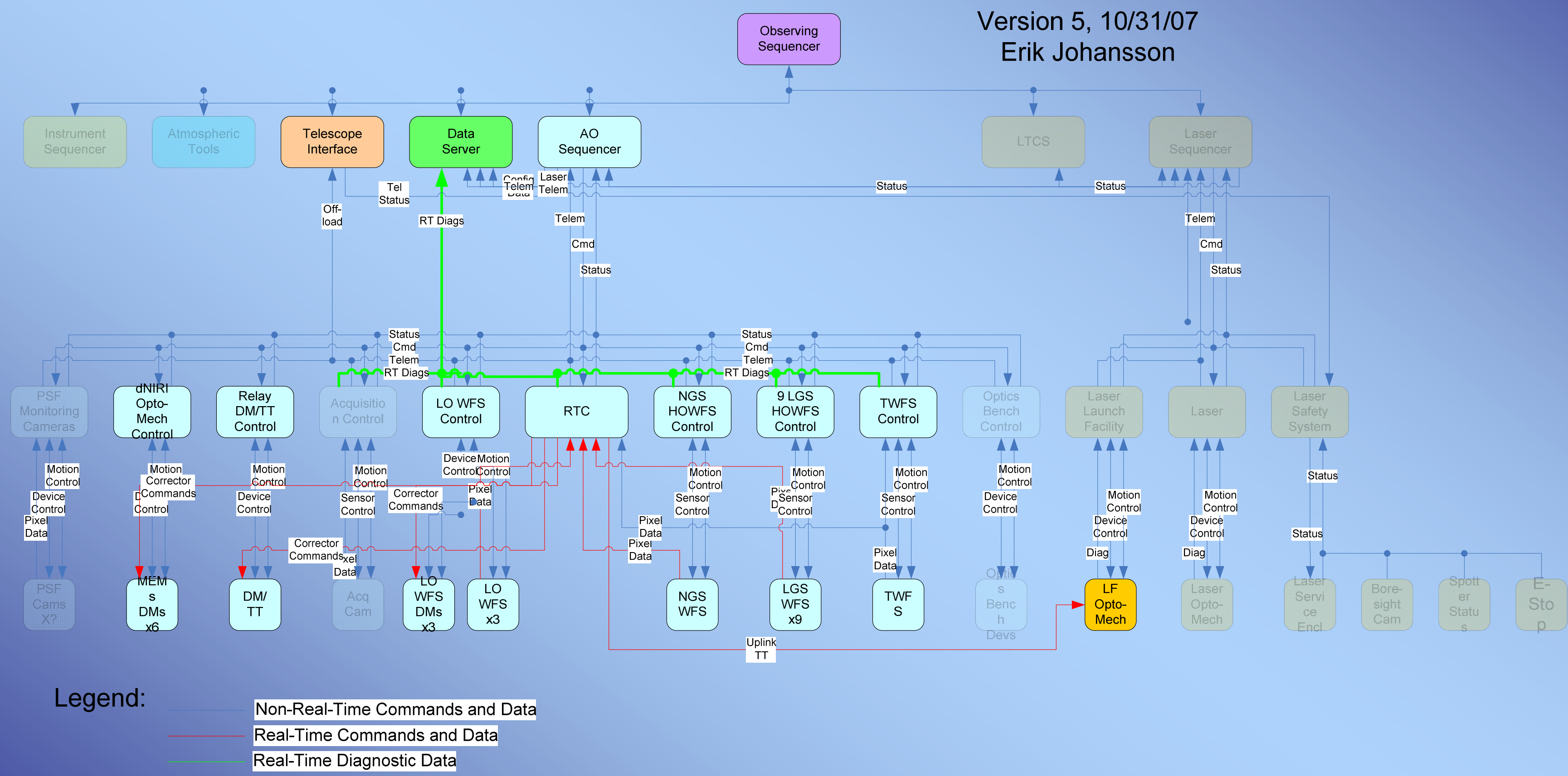
NGAO System Context Block Diagram
(DRAFT)

Version 5, 10/31/07
Erik Johansson



NGAO System Context Block Diagram

(DRAFT) cont.

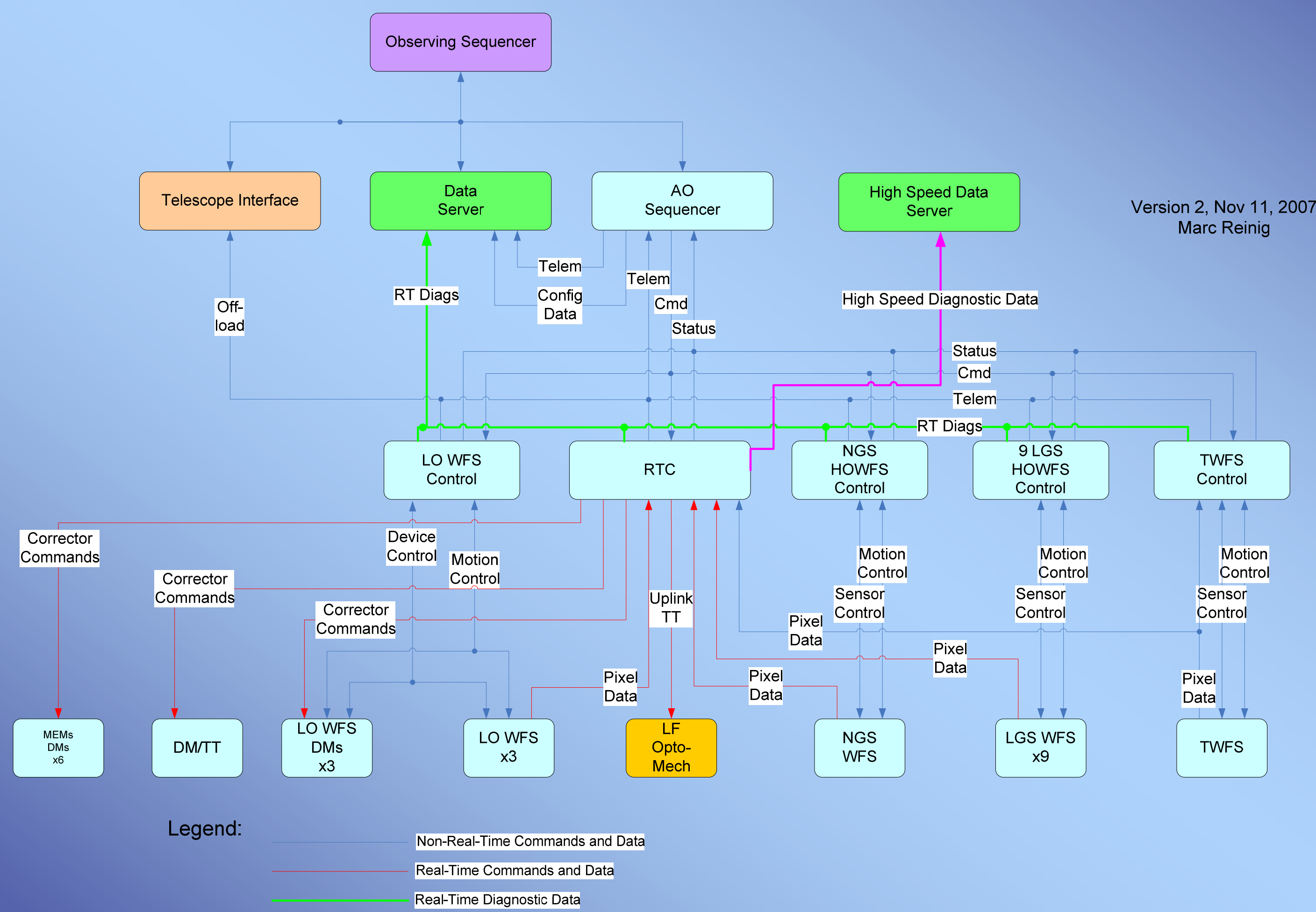


Interface to SRT System

- SRT controls all aspects of RTC - Ethernet
  - There will be a non-RT Linux PC controlling the RTC
  - Interface to the SRT will be Ethernet between the SRT and the RTC control PC
  - Load and read the RTC code for all operations
  - Load and read the RTC HW configuration
- Telemetry streams to monitoring, PSF, etc - Ethernet
- Interface between the RTC and the high-speed disk arrays for diagnostic data - Fiber Link or equivalent

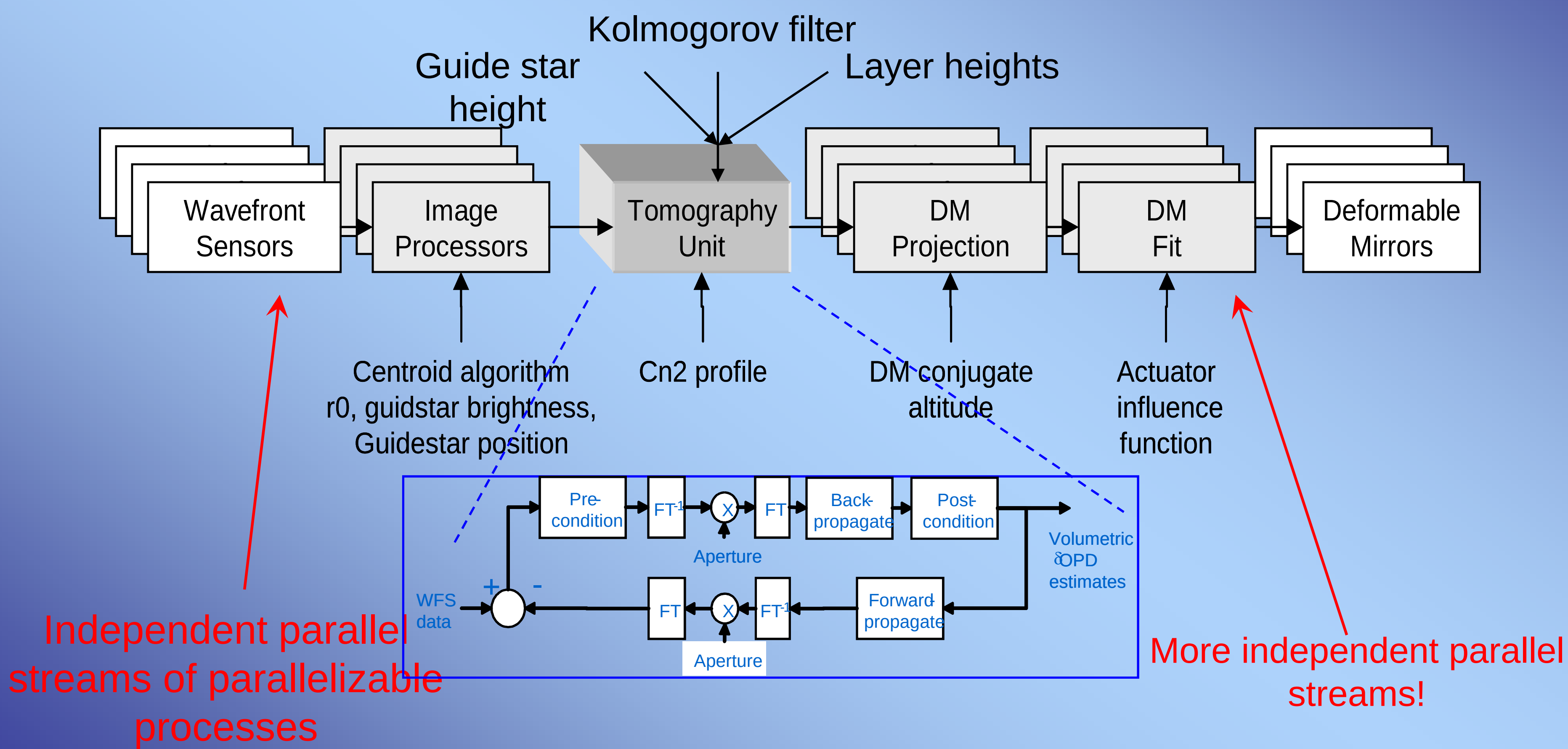
NGAO RTC System Context Block Diagram

(DRAFT)

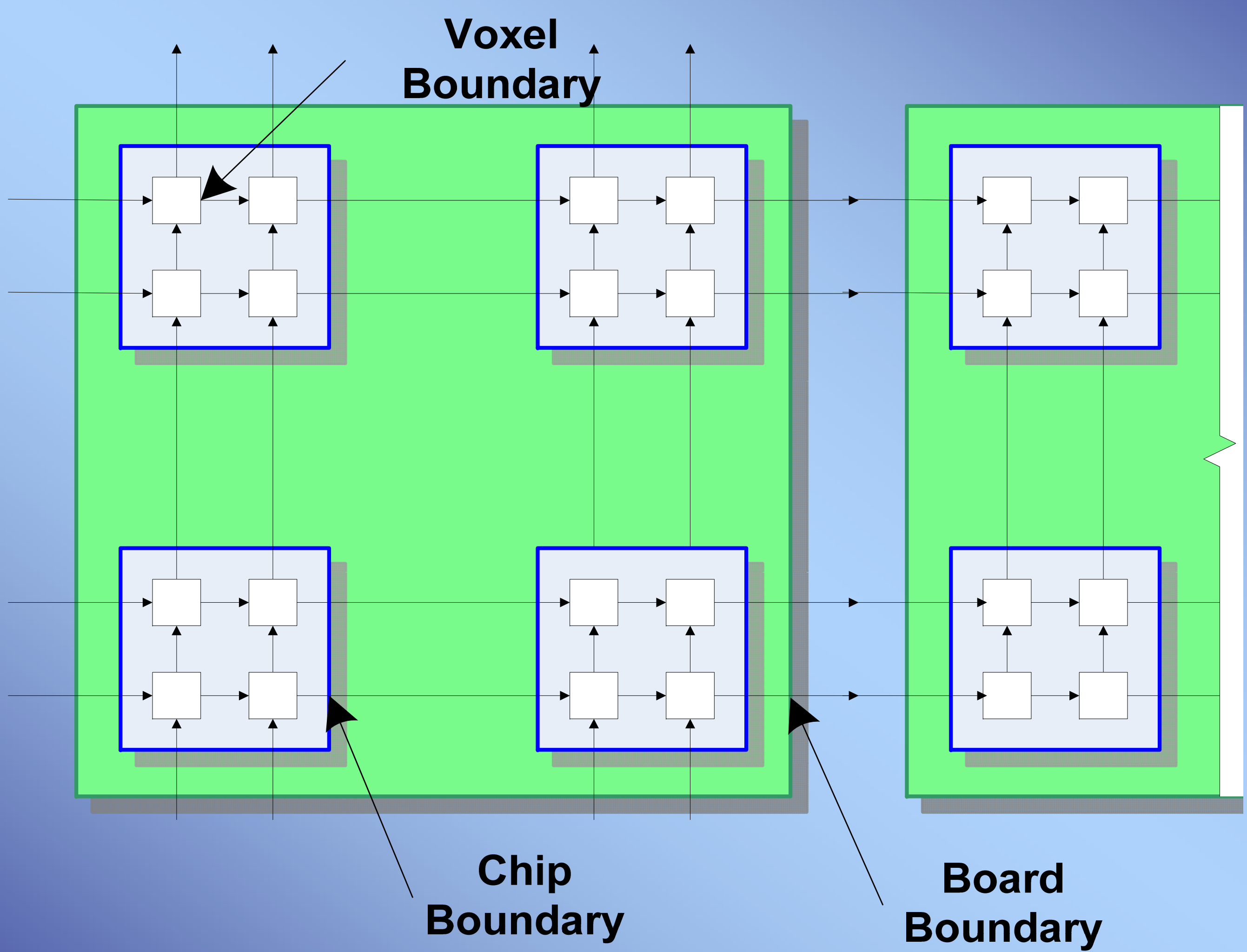


Tomography Is Part of a Larger System

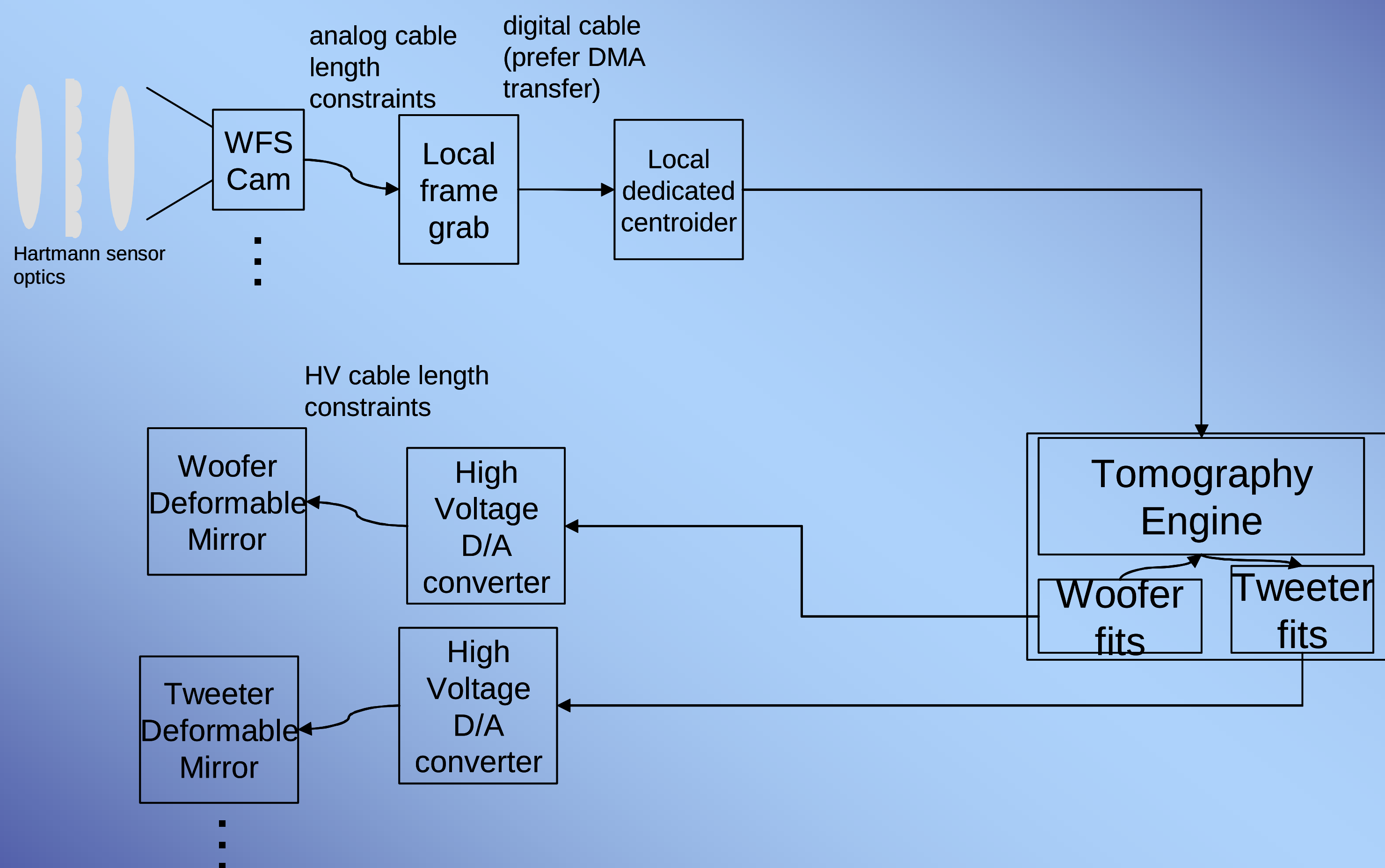
– With characteristics that lend themselves to a similar solution –



A Systolic Array



Data Rate and Location



Telemetry and Diagnostics Streams

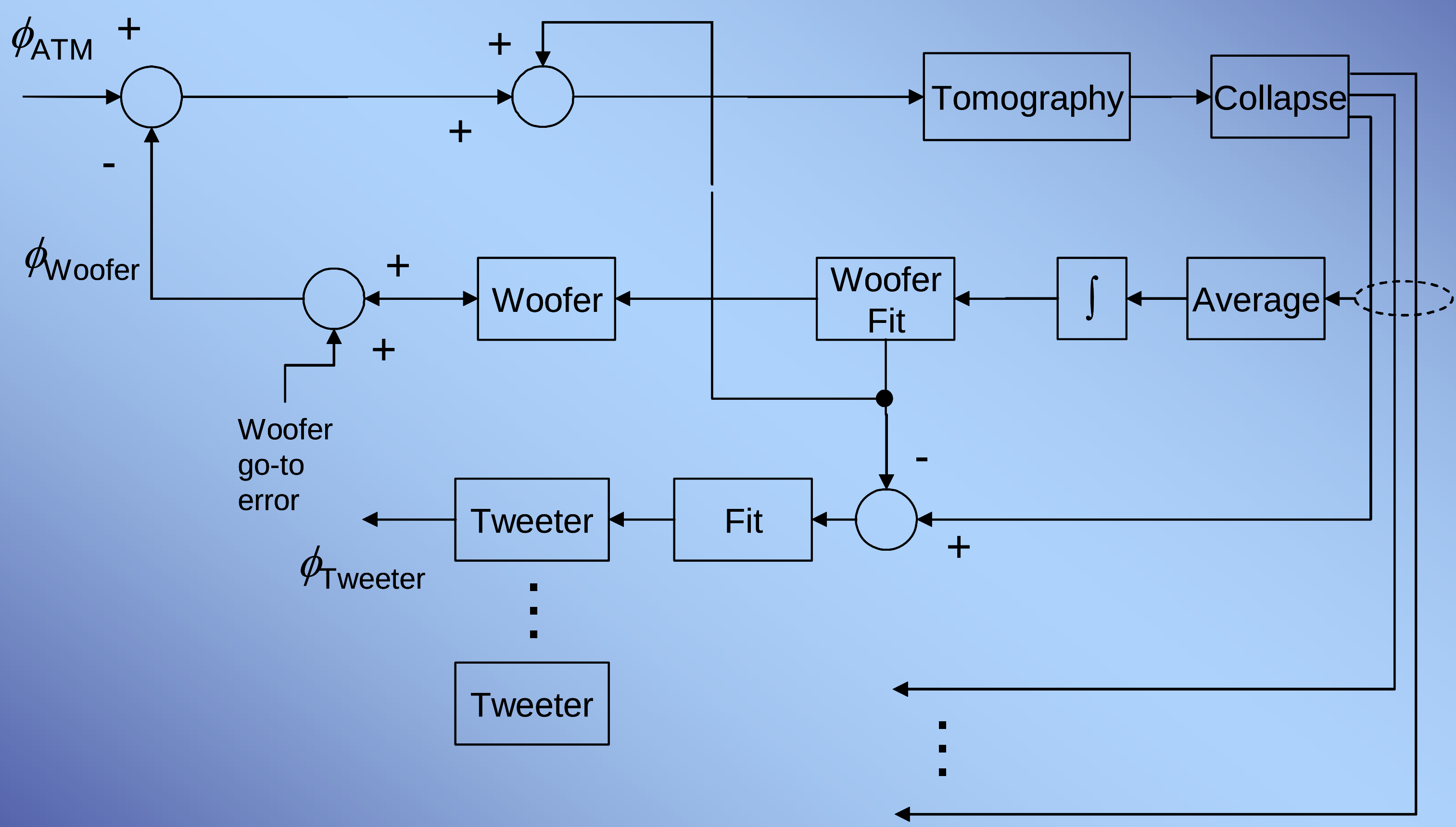
Telemetry (Low bandwidth, on-line monitoring) (>1Hz)

- Set and read the telemetry rate for each data stream
- Set and read the data streams that are enabled for telemetry
  - Centroids
  - Residual tomography error
  - DM commands, including uplink
  - Tip/Tilt commands

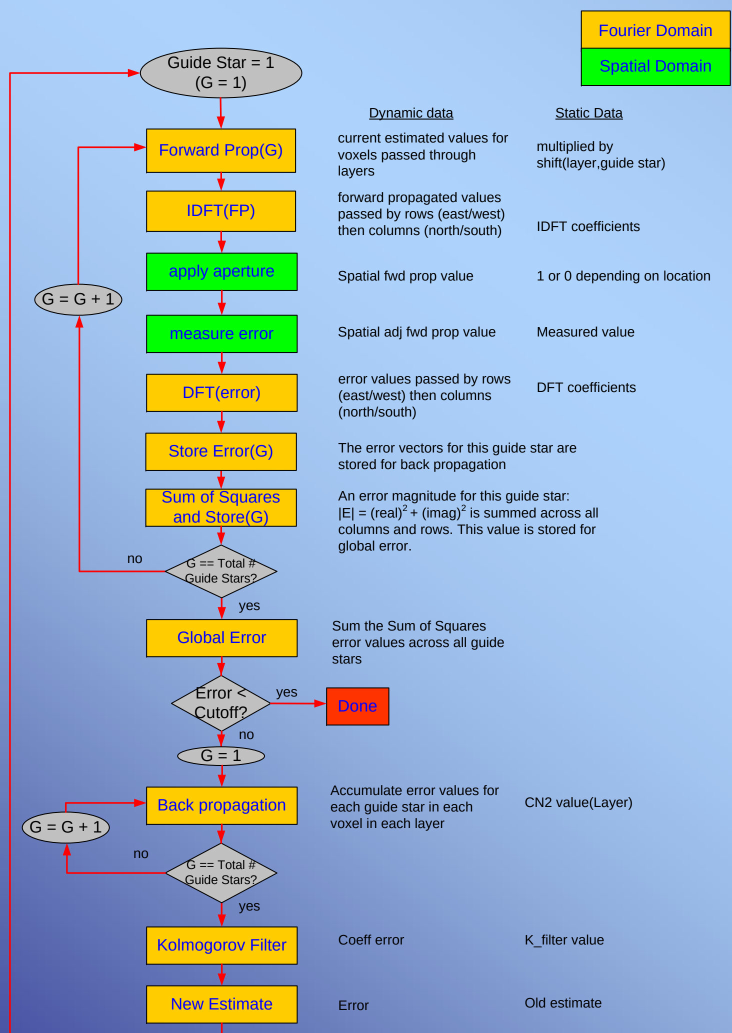
Diagnostics (High bandwidth, stored to disk) (full frame rate)

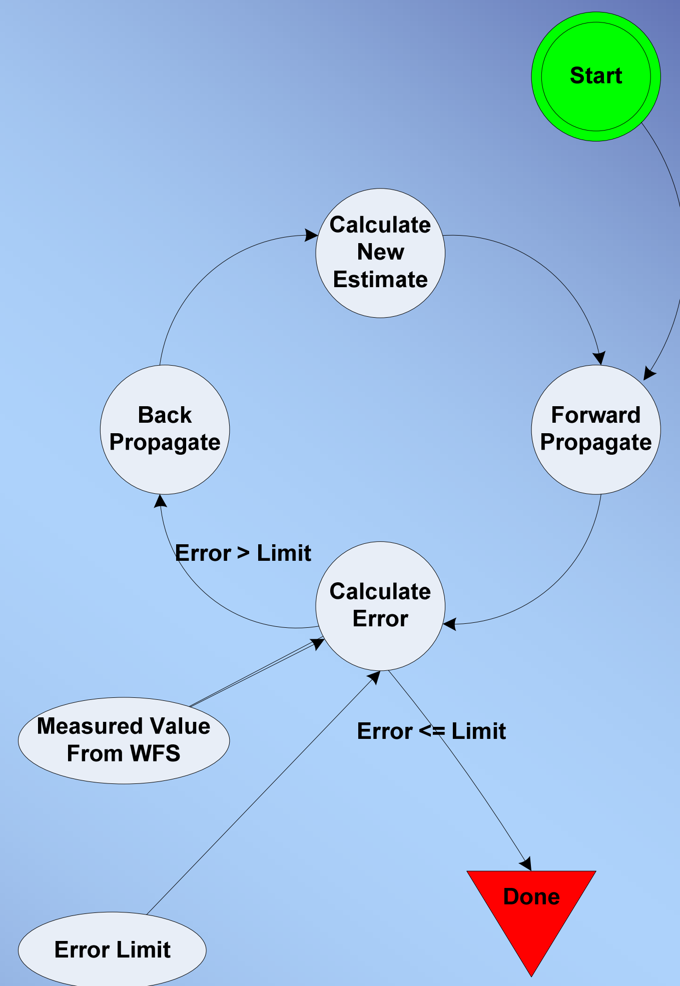
- Set and read the diagnostic rate for each data stream
- Set and read the data streams that are enabled for diagnostics
  - Centroids
  - Residual tomography error
  - Raw layer information
  - DM commands, including uplink
  - Collapsed layer information
  - Tip/Tilt commands

RTC Software Flow

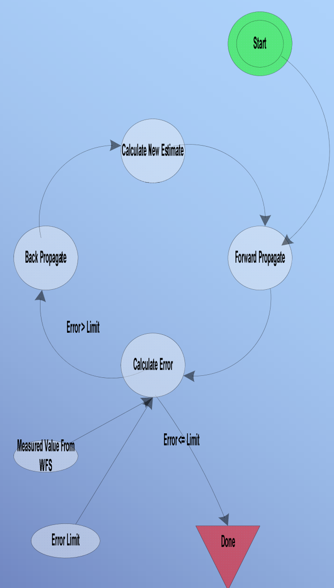


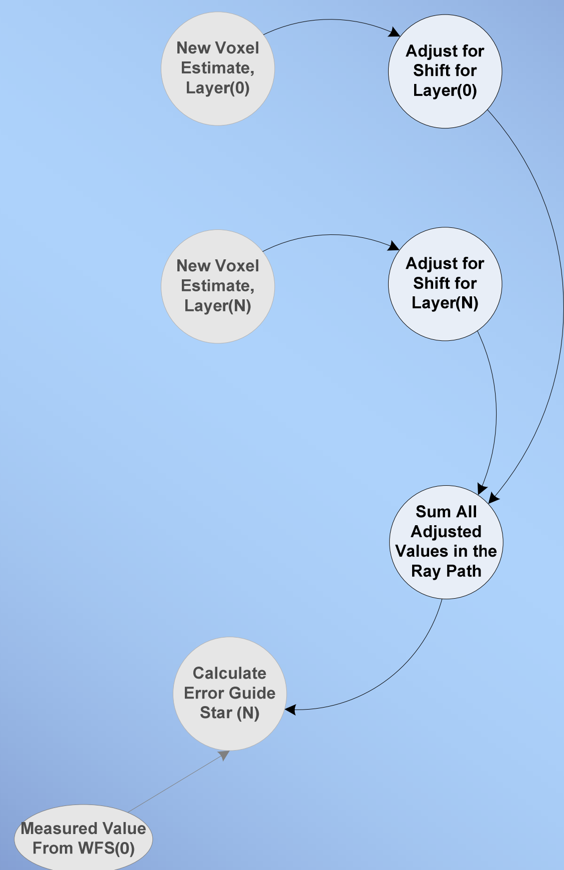
Tomography Basic Loop





Forward Propagation

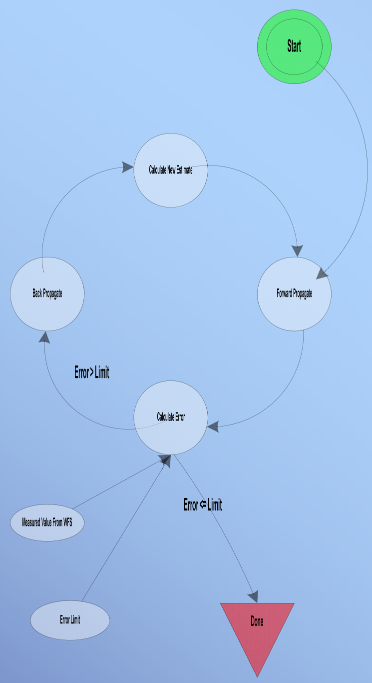


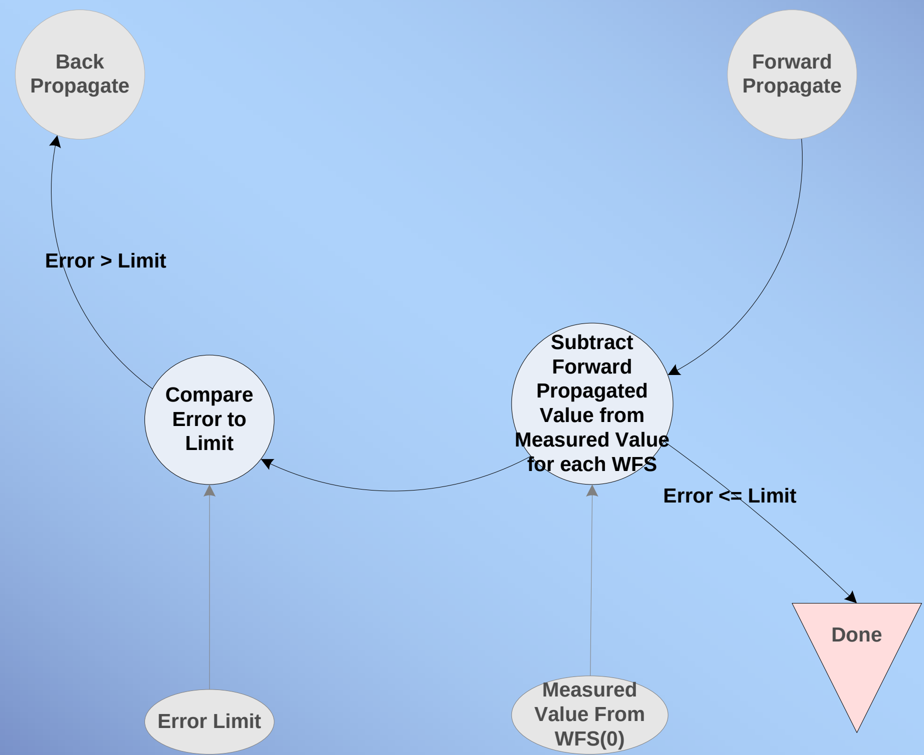


For each sub aperture, sum all the estimated Voxel values, along the path of a ray to each Guide Star.

In the Fourier domain, you must account for the spatial shift that occurs in the spatial domain. This will be the complex conjugate of the shift value used in the back propagation.

Calculate Error

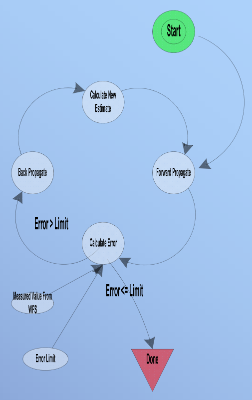


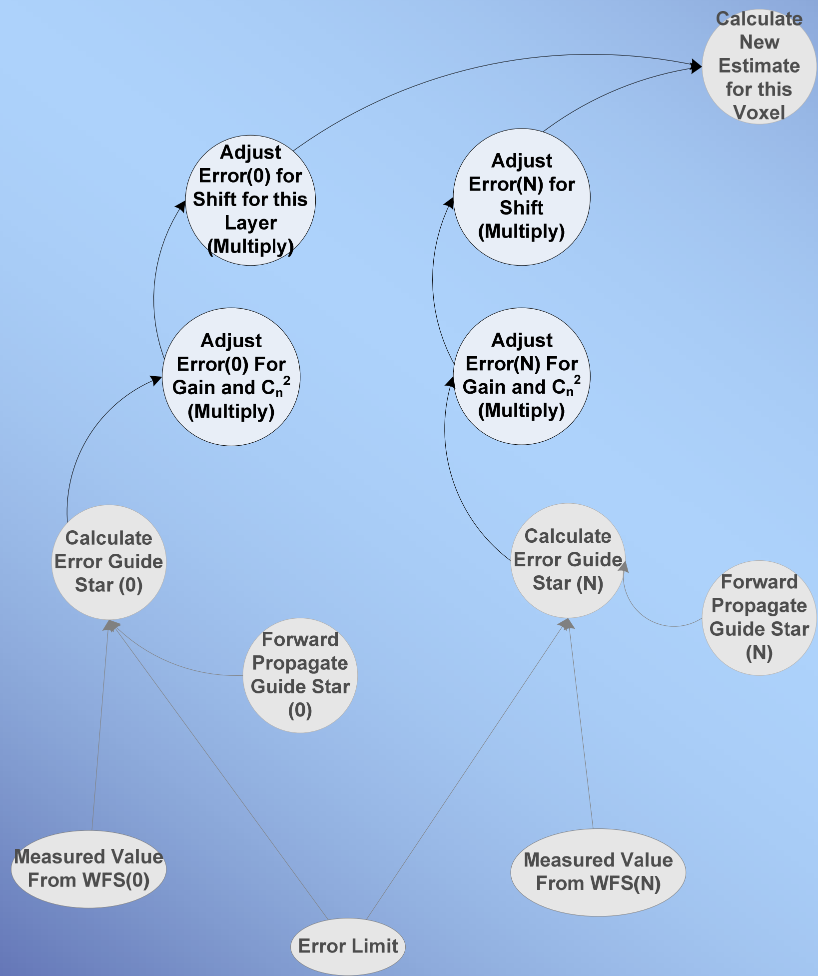


For each sub aperture, for each ray from a Guide Star, calculate the error between the forward propagated value and the measured value from the WFS for that Guide Star.

If you have reached the limit stop, else continue with another loop

Back Propagate



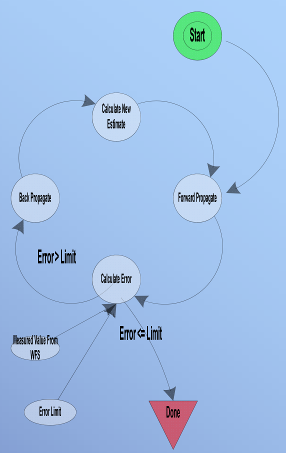


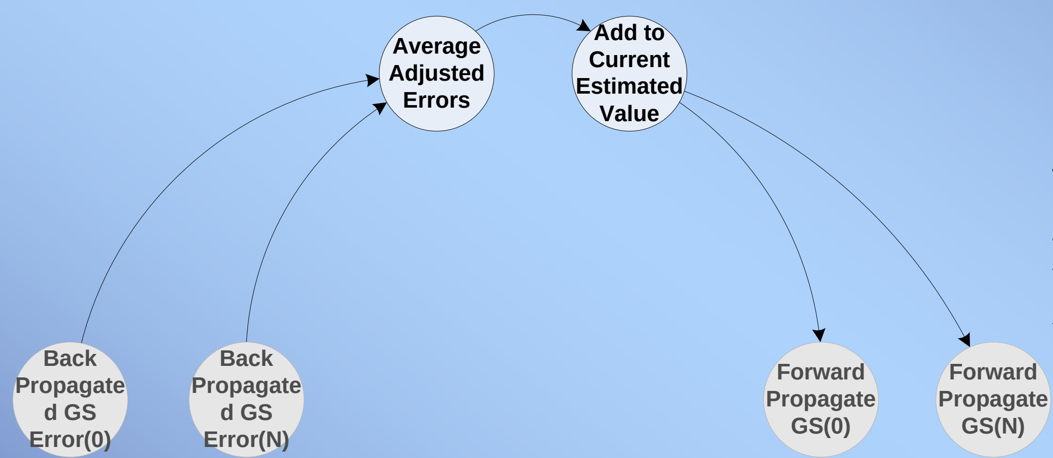
For each guide star, for each sub aperture, back propagate the error for a guide star to all the Voxels along the ray back to the Guide Star (These should be the same Voxels used in calculating the Forward Propagated value.)

Adjust the error for each Voxel for the C_n^2 value for its layer

In the Fourier domain, you must account for the spatial shift that occurs in the spatial domain. This will be the complex conjugate of the shift value used in the Forward propagation.

Calculate New Estimate





For each Voxel, calculate the average of the adjusted errors that were back propagated along the rays to various Guide Stars and add this to the estimated value for the Voxel to get a new estimated value.

WFS Control Commands

Camera

- Set and read the camera frame rate
- Set and read the camera pixel rate
- Set and read the camera gain

Centroiding

- Load the dark pattern
- Load pixel gain pattern
- Set and read the threshold level
- Set and read the centroiding algorithm
- Load and read the centroid weights
- Load and read the reference centroids
- Load and read the centroid offsets
- Set and read pixel offsets for the camera image
- Set and read the guidestar mode (NGS or LGS) for any WFS
- Load and read centroid non linearity tables

Tomography Control Commands

- Load and read the cone-effect scaling array for any guide star or target
- Load and read the sky position for any guidestar or target
- Load and read the Kolmogorov filter array
- Load and read the pre-conditioning arrays
- Load and read the tomography loop gain arrays
- Load and read the tomography bailout value
- Load and read the C_n^2 values for all layers
- Load and read the wind array for all layers

DM and Tip/Tilt Control Commands

- Load and read the layer-to-DM collapse arrays
- Load and read the DM command matrixes and non linearity lookup tables for open and closed loop control
- Set and read DM open/closed loop mode
- Set and read the Tip/Tilt command matrixes

Calibration Capabilities

- Create a system matrix for any DM
- Set the number of frames to use when creating the system matrix
- Clear the DM of any manually set values
- Calibrate science target DMs in MOAO
- Woofer
- Tweeter
- Force an arbitrary camera image for any WFS
- Force an arbitrary centroid output for any tomography WFS input
- Force an arbitrary layer pattern to be used by the DM command module
- Send an arbitrary pattern to any DM
- Save the current centroid positions as an reference centroid file (averaged over a settable number of frames)

System Diagnostics Capabilities

- Read all chip temperatures
- Read system temperatures at various predetermined locations
- Monitor chip and system fan operation
- Set and read temperature alarm trip levels

Reliability and Reparability

- Graceful degradation
- Spares requirement
- MTBF
- Self Healing (SEU)

Physical Characteristics

- Size
  - <20 m^3
- Weight
  - TBD
- Location
  - WFS and DM Drive are on the Nasmyth
  - RTC and High Speed Disk array are in the computer room
- Power
  - <20KW

Issues Remaining

- Determine Single Event Upset (SEU) rate from Gamma/Cosmic rays
- Verify detection and recovery scheme

Plans

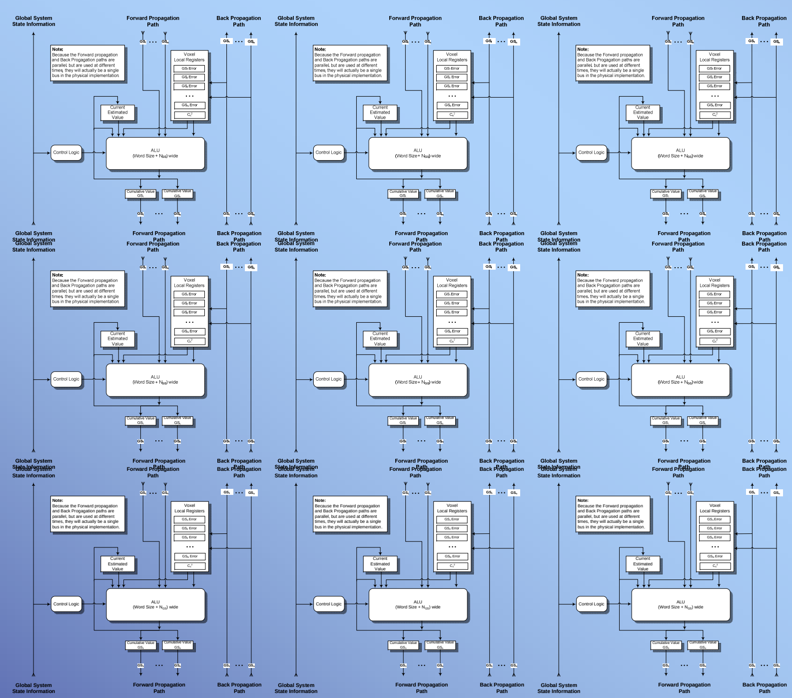
- Complete FRD
- Finish writing report
- Validate SEU issues

Backups

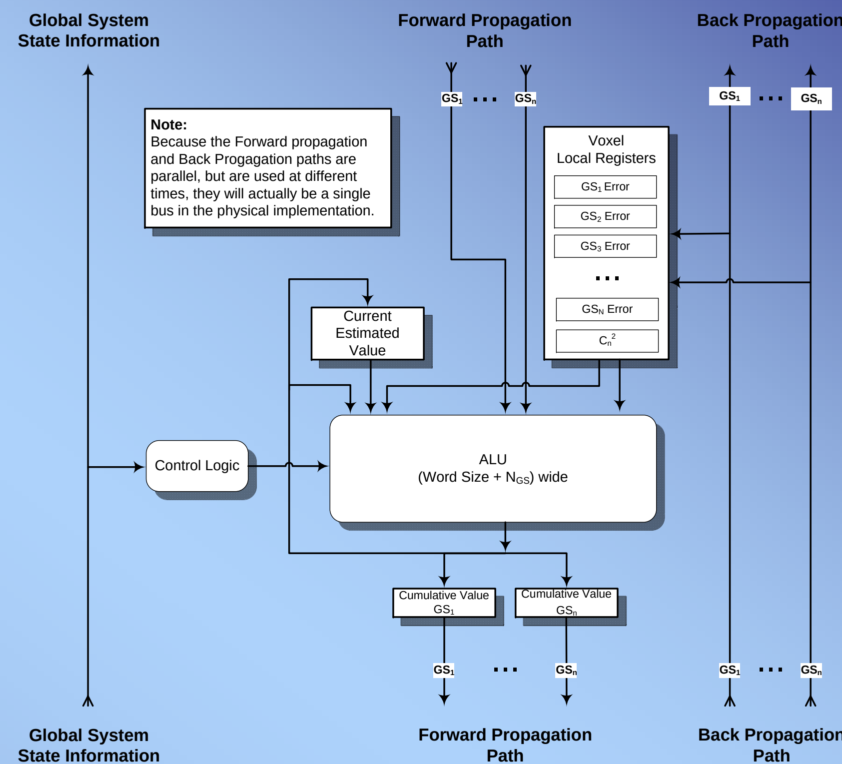
Sample Code to Configure the Hardware
– Simple ALU –

```
module alu4 ( input_a, input_b, output_c, ctrl_0, ctrl_1,
ctrl_2,ctrl_3, clr, clk );
  input ctrl_0, ctrl_1, ctrl_2, ctrl_3;
  input [15:0] input a, input b;
  output [31:0] output_c;
  input clr, clk;

  wire [3:0] ctrl;
  reg [31:0] output_c;

  assign ctrl = {ctrl_3, ctrl_2, ctrl_1, ctrl_0};
  parameter ADD = 4'b1001;
  parameter MUL = 4'b0001;
  parameter NOP = 4'b1000;

  always @(posedge clk or posedge clr)
  begin
    if (clr)
      output_c <= 0;
    else
      case (ctrl)
        ADD:
          output_c <= input_a + input_b;
        MUL:
          output_c <= input_a * input_b;
        default:;
      endcase
    end
endmodule
```

Processor Array Illustration



- An Array of Voxel Processors

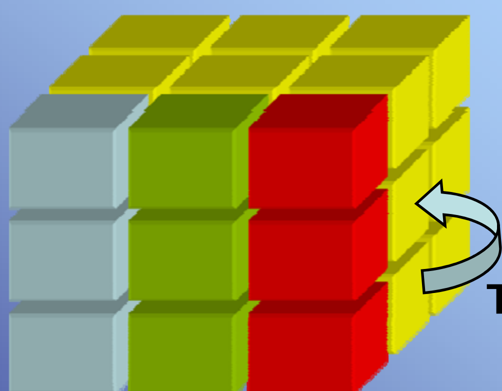


- A Single Voxel Processor

The Tomography Engine

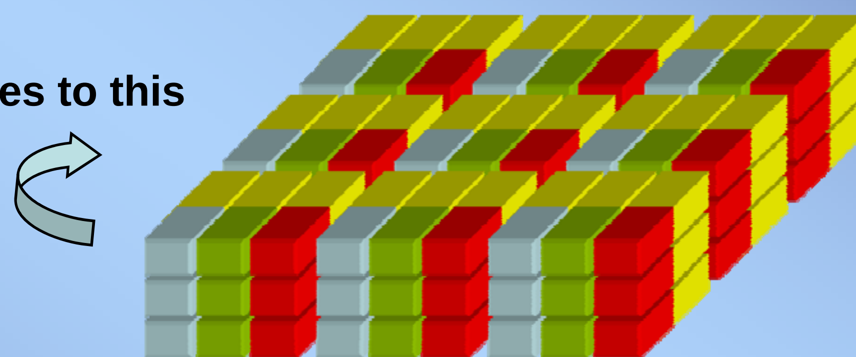
- A scalable, programmable, configurable processor -

- Scalable: System can be easily put together (by replication) to work for any sized problem
- Programmable: Changes in the algorithm can be loaded at run time
- Configurable: Parameters are easily changed (sub aps, guide stars, layers, **Cn2**; mixes and positions of natural and laser guidestars; etc.



This …

… scales to this



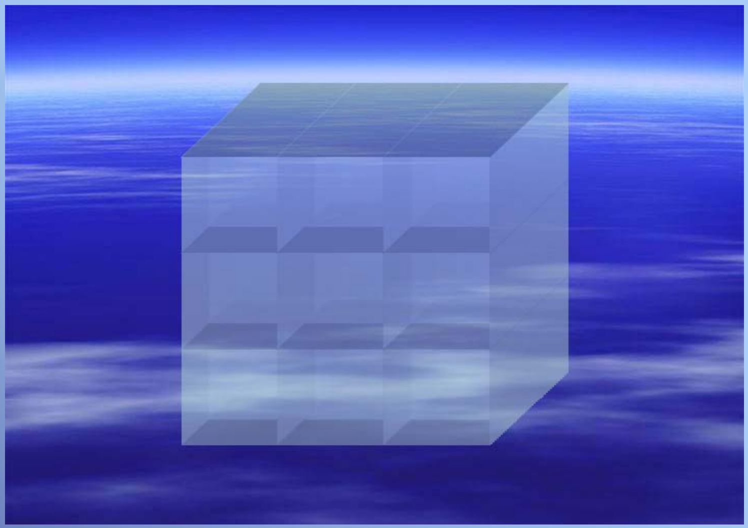
Multiple Voxels in each FPGA
(One column for each sub ap.)

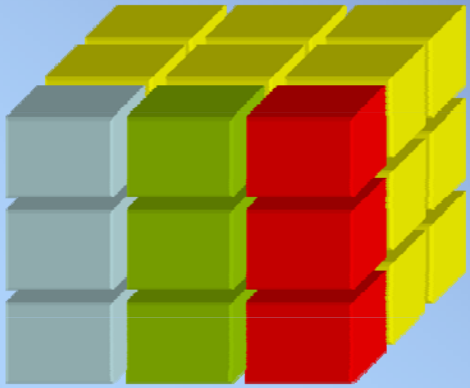
Multiple Chips on a Board(s)

A Single FPGA Can Hold Hundreds of Voxels

A Circuit Board Can Hold A Dozen FPGA's

Basing Our Architecture on Our Knowledge of Structure





We divide the atmosphere into regions called “voxels”

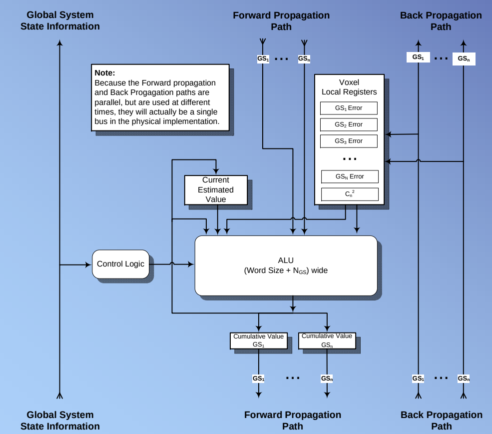
And use that model for our structure for the iterative solution

Using Our Knowledge of Structure

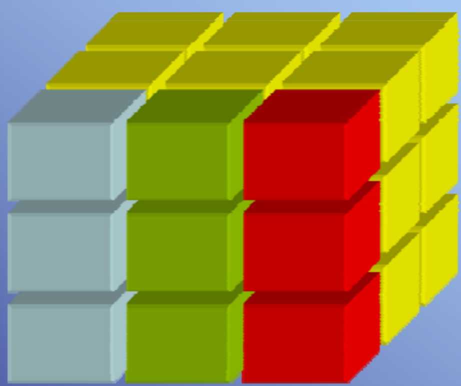
- <u>For a single Voxel</u>

- **Examining the problem, we find:**
- **All calculations for all voxels are identical**
- **Most operations on voxels are independent of other voxels**
- **Operations are simple**
- **Memory requirements are modest**
- **This is an "Embarrassingly" parallel problem**
- **So …**

We give each voxel its own simple processor

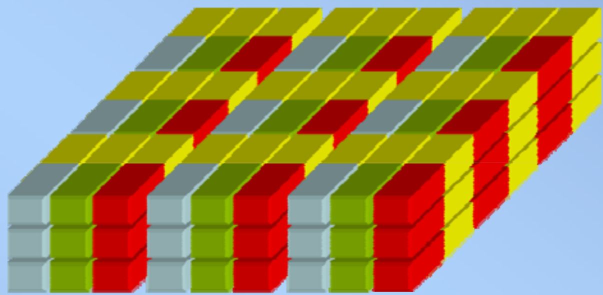


These Voxel processors are interconnected, modeled after the structure of our problem



This …

… scales to this



Each column of voxels is a complete sub aperture containing all layers in our model of the atmosphere. Multiple sub apertures are arrayed in each FPGA chip

Multiple Chips on a Board, and multiple boards to satisfy the size of the problem