

IS/IC

"Instrument Sequencer to Instrument Controller"

???

September 28, 2001

Version: 0.0

Issued By:

Internal NICI Interface Document

Revision Control

1.Revision Version 0.0

Date: September 28,2001

Revised by: William Rambold

Reason for / items changed:

Initial draft for discussion.

2. Revision Version 0.01

Date: February 20, 2001

Revised by: Jim Hinds

Reason for / items changed:

Addition of text for section 10: AO System Commands

3. Revision Version 0.03

Date: March 14, 2001

Revised by: Jim Hinds

Reason for / items changed:

Addition of text for section 8 and 9:Array and Observation commands

4. Revision Version 0.04

Date: March 17, 2001

Revised by: Jim Hinds

Reason for / items changed:

Addition of text for section 11 and section 12 :Mechanism control and Thermal monitor

Table of Contents

1.	DESCRIPTION	5
1.1.	Purpose	5
1.2.	Scope	5
1.3.	Stylistic Conventions	5
2.	RELATED DOCUMENTS AND DRAWINGS	5
2.1.	References	5
2.2.	Abbreviations and Acronyms	5
2.3.	Glossary	6
3.	NICI SOFTWARE ARCHITECTURE OVERVIEW	6
4.	IS / IC INTERFACE PROTOCOL	6
4.1.	Overview	6
4.2.	Sending a command	6
4.3.	Acknowledging command receipt	7
4.4.	Acknowledging command completion	7
4.5.	Checking a command for validity	7
4.6.	Requesting status information	8
4.7.	Receiving unsolicited messages	8
5.	COMMAND PROTOCOL SCENARIOS	8
5.1.	Configuration Command Sequence	8
5.2.	Action Command Sequence	9
5.3.	Process Control Command Sequence.	10
5.4.	Rejected Command Sequence	10
5.5.	Failed Command Sequence	11
5.6.	Command Checking Sequence	12
6.	COMMAND OBJECT SUMMARY	13

7.	COMMAND SUMMARY	13
7.1.	Acquire control commands	14
7.2.	Acquire configuration commands	14
7.3.	Adaptive optics system control commands	14
7.4.	Adaptive optics system configuration commands	15
7.5.	Mechanism control commands	15
7.6.	Mechanism configuration commands	15
7.7.	Sensor control commands	15
7.8.	Sensor configuration commands	16
7.9.	Telescope data configuration commands	16
8.	ARRAY COMMANDS	16
9.	OBSERVATION COMMANDS	18
10.	AO SYSTEM COMMANDS	18
11.	MECHANISM COMMANDS	21
12.	ENVIRONMENT COMMANDS	24
13.	STATUS SUMMARY	26
13.1.	Adaptive optics system status – “AO configure <i>attribute</i> ”	26
13.2.	Mechanism configuration – “ <i>mechanism</i> configure <i>attribute</i> ”	26
14.	SCENARIOS	26
14.1.	Scenario List	26
14.2.	Initialization sequence	27
14.3.	Normal exposure sequence	27
14.4.	Stopping an exposure in progress	27
14.5.	Aborting an exposure in progress	27
14.6.	More!!!	Error! Bookmark not defined.

1. Description

1.1. Purpose

This Interface Design Description describes the communications protocol and command set used between the NICI Instrument Sequencer and the NICI Instrument Controller. It is aimed principally at the developers of the NICI work package.

1.2. Scope

This document covers, and is limited in scope to, interactions between the NICI Instrument Sequencer and Instrument Controller.

All interactions take place via the NICI single remote command socket as described in this document.

Since EPICS provides unrestricted engineering access to all records in the system this mode of operation is not covered in this document.

1.3. Stylistic Conventions

Text within angle brackets like this < > is used to indicate a range of possible values. Discrete choices are separated by a vertical bar like '|', while a range of choices are separated by a pair of periods like '..'. Units or type are presented in square brackets like this [].

2. Related Documents and Drawings

2.1. References

- [1] *Channel Access Reference Manual*, J. O. Hill, Los Alamos National Laboratory
- [2] gscg.bdg.004.grrm *Gemini Record Reference Manual*, Bret Goodrich and Andy Foster.
- [3] *ICD 1.9./3.1 The OCS to NICI Interface*, Maune Kea Infrared
- [4] *EPICS Input Output Controller Record Reference Manual*, Janet B. Anderson and Martin R. Kraimer, Argonne National Laboratory, Dec 1, 1994

2.2. Abbreviations and Acronyms

CC	Component Controller
EPICS	Experimental Physics and Industrial Control System
IS	Instrument Sequencer
N/A	Not applicable here.
SIR	Status and Information Record
TCS	Telescope Control System

2.3. Glossary

VxWorks

A Real Time Operating system from Wind River

3. NICI software architecture overview

- Does not follow the standard Gemini model
- Standalone server
- Except for DHS all interactions with Gemini systems are handled by the Instrument sequencer
- Command line operation
- Enhanced with command checking and command completion

4. IS / IC Interface Protocol

4.1. Overview

There are three types of command messages sent from the Instrument Sequencer to the Instrument Controller:

- Configuration commands that directly set or query internal variables. Setting or reading the variables is an instantaneous operation. Configuration commands are considered complete as soon as the variable has been set or read.
- Discrete action commands that change the position or state of internal components. This change may take time to complete and may encounter errors along the way. Discrete action commands are considered complete when the new position or state has been reached or an error has been detected.
- Process control commands that start or stop an ongoing process. Starting or stopping the process may take time to complete and may encounter errors along the way. Process control commands are considered complete when the process has been started or stopped or an error has been detected.

In response to these commands there are three types of messages sent from the Instrument Controller back to the Instrument Sequencer:

- Acknowledgment response messages that indicate acceptance or rejection of an action or configuration command.
- Completion response messages that indicate the eventual success or failure of an action or process control command.
- Spontaneous messages that result from a state change of an Instrument Controller component.

4.2. Sending a command

An Instrument Controller command message consists of a series of space-delimited tokens terminated by a carriage return. The tokens define the name of the object to be acted upon and the action to be performed, followed by an optional action attribute and value. The generic command format is:

object action [attribute [value]]<cr>

The commands, attributes and values associated with each object are described in section TBD.

4.3. Acknowledging command receipt

Upon receiving a command an object will check the request and current object state to see if the command can be executed. After checking the command the object always returns an immediate acknowledgement string to the IS. This string can have one of two forms:

Command is valid and safe to execute given the current state of the object:

object accept: {action [attribute [value]]} state<cr>

Command is not valid or the object is not in a state where it is safe to execute the command:

object reject: {action [attribute [value]]} "reason" state<cr>

4.4. Acknowledging command completion

After accepting an action command an object will try to perform the requested action. When the action has been completed (either successfully or otherwise) the object will return a completion string to the IS. The return string can have one of two forms:

Command completed successfully:

object {action [attribute [value]]} stable <cr>

Command failed during execution:

object {action [attribute [value]]} "reason" error <cr>

4.5. Checking a command for validity

Occasionally it is necessary to test the validity of a command without having the Instrument Controller perform the actions associated with the command.

The only difference between issuing a command and testing a command is that the command is not executed if it is accepted.

This feature is used when a number of actions need to be performed simultaneously if, and only if, all of the actions can be performed.

A validity check is requested by adding the "test" tag to the command to be tested:

object test action [attribute [value]]<cr>

If the command is valid and it is safe to execute the command given the current state of the object then a string with the following format is returned and no further action takes place:

object accept: {test action [attribute [value]]} state <cr>

If the command is not valid or it is not safe to execute the command when it is received a string with the following format is returned and no further action takes place:

object reject: {test action [attribute [value]]} "reason" state<cr>

4.6. Requesting status information

Configuration commands can also be used to request the current state of an attribute by specifying the attribute without the value as follows:

```
object configure attribute <cr>
```

The object will respond by filling missing value field(s) with the current state of that attribute:

```
object {configure attribute currentValue} state <cr>
```

Object configuration commands can also be used to request the current state of all associated attributes by issuing the command without either attributes or values as follows:

```
object configure <cr>
```

The object will respond by sending a list of all the attributes associated with the command and the current value of each:

```
object {configure {attribute1 value1 ... attributeN valueN ...}} state <cr>
```

4.7. Receiving unsolicited messages

Under certain circumstances the instrument controller will send unsolicited messages to the instrument sequencer. These messages are generated in response to changes in the instrument state. Spontaneous messages are identified by the “transient” tag appended to the end of the message.

The generic form of these messages is:

```
object {configure attribute value} transient <cr>
```

5. Command Protocol Scenarios

The following sequences serve to illustrate the command protocol described above. The following convention is used throughout:

- **IS:** Indicates strings that are sent from the Instrument Sequencer to the Instrument Controller.
- **IC:** Indicates strings that are sent from the Instrument Controller to the Instrument Sequencer.
 - Indicates an action that takes place in the Instrument Controller

5.1. Configuration Command Sequence

Since this is a configuration command only one type of message is sent back to the Instrument Sequencer:

- Acceptance message to indicate that the configuration variable was changed or returned.

Set the AO system loop gain:

- **IS: AO configure loopgain 10.7 <cr>**

- AO object checks to see if attribute is the name of a valid internal variable and that the value is within the acceptable limits for that variable.
- AO object determines that the request is valid so updates the variables value.
- AO object indicates that the command was accepted.
- IC: **AO accept: {configure loopgain 10.7} stable <cr>**
 - Filter wheel object remains in *idle* state until the next command is received.

Request the position of the neutral density filter wheel:

- IS: **NDFW configure position <cr>**
 - Filter wheel object remains in *idle* state
 - Filter wheel object checks the filter wheel position sensors to see which position it is in and updates the internal position variable.
- IC: **NDFW accept: {configure position 4} stable <cr>**
 - Filter wheel object remains in *idle* state until the next command is received.

5.2. Action Command Sequence

Since this command involves a motion that will take some time to complete Instrument Controller will sent three types of messages to the Instrument Sequencer:

- Acceptance message to indicate that the command was accepted and motion has started.
- Transient message when the filter wheel passes a valid position.
- Completion message to say that the command has finished successfully.

Request the valid filter ND3 while the flux is low enough for this filter to be used safely:

- IS: **NDFW move ND3 <cr>**
 - Filter wheel object checks filter name and flux level without leaving the *idle* state.
 - Filter wheel object decides that it is safe to put the requested filter in to the AO wfs light path so accepts the command.
- IC: **NDFW accept: {move ND3} stable <cr>**
 - Filter wheel object transitions from *idle* to *active* state.
 - Filter wheel starts moving to the ND3 position.
 - Filter wheel passes the RED position.
- IC: **NDFW {configure name RED} transient <cr>**
 - Filter wheel passes the ND4 position.
- IC: **NDFW {configure name ND4} transient <cr>**
 - Filter wheel reaches the ND3 position.
- IC: **NDFW {configure name ND3} transient <cr>**

- Filter wheel transitions from *active* to *idle* state.
- IC: **NDFW {move ND3} stable <cr>**

5.3. Process Control Command Sequence.

Since process control commands do not complete in the same manner as action commands the protocol for these commands is different.

When starting the loop the object knows that it is safe to start so it accepts the command. There is no guarantee that the servo loop will eventually stabilize, however, so a form of completion message must be generated when the loop eventually stabilizes. The action is ongoing, however, so the completion message has the transient state tag (or error if the loop failed to stabilize).

When stopping the loop the action is instantaneous and can not fail so the acceptance message serves as the completion message as with the configuration commands.

Start the AO system servo loop:

- IS: **AO pause off <cr>**
 - AO object checks to see if it is safe to start the AO servo loop.
 - AO object determines that the AO system can be started at this time.
- IC: **AO accept: {pause off} stable <cr>**
 - AO transitions to the *active* state
 - AO object attempts to start the servo loop.
 - AO servo loop settles and starts correcting within the allotted time.
- IC: **AO {???) transient**
 - AO servo loop continues correcting until commanded to stop.

Stop the AO system servo loop

- IS: **AO pause on<cr>**
 - AO object checks to see if it is safe to stop the AO servo loop.
 - AO object stops the servo loop.
- IC: **AO accept: {pause on} stable <cr>**
 - Accumulated tilt exceeds bandwidth.

5.4. Rejected Command Sequence

Since no action is possible the Instrument Controller will only send one type of message to the Instrument Sequencer:

- Rejection message to indicate that the command can not be executed and the reason why it was rejected.

Set the AO system loop gain to 1000.7:

- IS: **AO configure loopgain 1000.7** <cr>
 - AO object checks to see if attribute is the name of a valid internal variable and that the value is within the acceptable limits for that variable.
 - AO object determines that the requested gain value is outside the legal range.
 - AO object indicates that the command was rejected.
- IC: **AO reject: {configure loopgain 1000.7} “gain value too high” stable** <cr>
 - Filter wheel object remains in *idle* state until the next command is received.

Request the valid filter ND3 while the flux is too high for this filter to be used safely:

- IS: **NDFW move ND3** <cr>
 - Filter wheel object checks filter name and flux level without leaving the *idle* state.
 - Filter wheel object determines that putting the requested filter into the AO wfs light path would result in too much flux on the wfs detector and so rejects the command.
- IC: **NDFW reject: {move ND3} “flux too high for requested filter” stable**<cr>
 - Filter wheel object remains in the *idle* state until the next command is received.

5.5. Failed Command Sequence

Since this command involves motion the Instrument Controller will send three types of messages back to the Instrument Sequencer:

- Acceptance message to indicate that the command was accepted and motion has started.
- Transient message when the filter wheel passes a position.
- Completion message to say that the command failed during execution.

Request the valid ND3 filter while the flux is low enough to allow this filter to be used. The wheel, however, jams and does not reach its destination within a reasonable time:

- IS: **NDFW move ND3** <cr>
 - Filter wheel object checks filter name and flux level without leaving the *idle* state.
 - Filter wheel determines that putting the requested filter into the AO wfs light path will not result in too high a level of flux on the wfs detector so accepts the command.
- IC: **NDFW accept: {move ND3} stable** <cr>
 - Filter wheel object transitions from *idle* to *active* state.
 - Filter wheel starts moving to the ND3 position.
 - Filter wheel passes the RED position.

- IC: **NDFW {configure name RED} transient <cr>**
 - Filter wheel jams and does not move.
 - Filter wheel motion timeout expires.
 - Filter wheel transitions from *active* to *error* state.
- IC: **NDFW {move ND3} “motion timeout” error <cr>**
 - NDFW object remains in the *error* state until the recover command is received.

5.6. Command Checking Sequence

Since this command does not involve any action only one type of message is sent back to the Instrument Sequencer:

- Acceptance message to indicate that the command would be accepted or rejected.

.

Check to see if the ND3 filter can be inserted:

- IS: **NDFW test move ND3 <cr>**
 - Filter wheel object checks filter name and flux level without leaving the *idle* state.
 - Filter wheel object determines that it would be safe to put the requested filter into the AO wfs light path without overloading the wfs detector so indicates that the command would be accepted.
- IC: **NDFW accept: {test move ND3} stable<cr>**
 - NDFW object remains in the *idle* state until the next command is received.

Check to see if the FOO filter can be inserted:

- IS: **NDFW test move FOO <cr>**
 - Filter wheel object checks filter name and flux level without leaving the *idle* state.
 - Filter wheel object determines that the requested filter is not one of the currently installed filters so indicates that the command would be rejected.
- IC: **NDFW reject: {test move FOO} “invalid filter name” stable<cr>**
 - NDFW object remains in the *idle* state until the next command is received.

Check to see if the ND3 filter can be inserted while flux level is too high :

- IS: **NDFW test move ND3 <cr>**
 - Filter wheel object checks filter name and flux level without leaving the *idle* state.
 - Filter wheel object determines that the flux level on the AO wfs detector would be too high if the requested filter were inserted into the wfs light path so indicates that it would reject the command.

- IC: NDFW reject: {test move ND3} “flux level too high for ND3 filter” stable <cr>
 - NDFW object remains in the *idle* state until the next command is received.

6. Command Object Summary

This section contains a summary of the command objects that make up the NICI instrument controller. How these objects are used to control the instrument functions is the subject of the following sections.

<i>Object</i>	<i>Name</i>	<i>Description</i>
Data acquisition	acquire	Controls the array readout and data saving functions
Adaptive optics	AO	Controls the adaptive optics sub-system
Beam splitter	BSW	Controls the beam splitter wheel motion
Channel one filter	C1FW	Controls the first detector channel filter wheel motion
Channel two filter	C2FW	Controls the second detector channel filter wheel
Fibre-optic calibration source	FOCS	Controls the fibre-optic calibration source and calibration slide motion
Neutral density filter	NDFW	Controls the wavefront sensor neutral density wheel motion.
Pupil imager	PI	Controls the pupil imager position
Pupil mask	PMW	Controls the pupil mask wheel position
Instrumentation sensors	Sensor	Sends current sensor information to the interface layer
Spider mask	SMRW	Controls the spider mask rotator position
Telescope state	Telescope	Receives current telescope information from the interface layer
Tip/tilt steering mirror	TTSM	Controls the position of the tip/tilt steering mirror
Temperature Monitor	LC1, LC2 and LS	Monitors temperatures in dewar

Table 0

7. Command Summary

This section contains a summary of the command set provided by each of the instrument

controller's command object.

7.1. Acquire control commands

Each acquisition control command has the format: “**acquire action [value]**”

<i>Action</i>	<i>Value</i>	<i>Description</i>
abort	-	Stop a data acquisition set immediately
go	-	Start the data acquisition set defined by the current configuration attributes
stop	-	Stop a data acquisition set after the current acquisition has completed

Table 0

7.2. Acquire configuration commands

Each Array system configuration command has the format: “**acquire configure -attribute value**”

<i>Attribute</i>	<i>Value</i>	<i>Description</i>
subarray	specification	Define a sub-array
mode	type	Define the type of readout to be performed for the next <i>go</i> command
ndr	count	Define the number of non-destructive reads to add before reset
rtime	delay	Define the delay time between reset and pedestal readout
integrate	time	Define the exposure time for each image scan
exposures	count	Define the number of exposures to perform for the next <i>go</i> command
sequence	count	Define the starting data file suffix for the exposure set
destination	list	Define the data destination(s) for the next <i>go</i> command

Table 7.2

7.3. Adaptive optics system control commands

Each AO system control command has the format: “**AO action [value]**”

<i>Action</i>	<i>Value</i>	<i>Description</i>
focus	state	Enable/disable offload of focus errors
pause	state	Enable/disable AO correction
tilt	state	Enable/disable offload of tip/tilt errors

Table 0

7.4. Adaptive optics system configuration commands

Each AO system configuration command has the format: “AO configure *-attribute value*”

<i>Name</i>	<i>Value</i>	<i>Description</i>
focuslimit	deadband	Define the focus offloading deadband
loopgain	gain	Define the AO correction loop gain
stroke	stroke	Define membrane mirror stroke
tiltlimit	deadband	Define the tip/tilt offloading deadband

Table 7.2

7.5. Mechanism control commands

Each mechanism control command has the format: “*mechanism action [value]*” where *mechanism* can be one of: FOCS, TTSM SMRW, PMW, BSW, C1FW, C2FW or PI.

<i>Action</i>	<i>Value</i>	<i>Description</i>
abort	-	Abort motion and put device in error state
initialize	-	Move to home position
move	position	Move to given position
query	attribute	Return current state of device attribute
recover	-	Try to bring device out of error state

Table 7.5

7.6. Mechanism configuration commands

Each mechanism configuration command has the format: “*mechanism configure attribute value*” where *mechanism* can be one of: FOCS, TTSM SMRW, PMW, BSW, C1FW, C2FW or PI.

Each mechanism has its own set of configuration parameters, see section TBD for details.

7.7. Sensor control commands

Each sensor control command has the format: “*sensor action [value]*” where *sensor* can be one of: TBD.

<i>Action</i>	<i>Value</i>	<i>Description</i>
start	-	Start reporting current sensor state at the configured rate
stop	-	Stop reporting current sensor state

Table 7.5

7.8. Sensor configuration commands

Each environment configuration command has the format: “*Sensor configure attribute value*”

<i>Attribute</i>	<i>Value</i>	<i>Description</i>
interval	time	Time in seconds between current state messages

Table 7.6

7.9. Telescope data configuration commands

Each environment configuration command has the format: “*Telescope configure attribute value*”

<i>Attribute</i>	<i>Value</i>	<i>Description</i>
azimuth	angle	Current telescope azimuth angle
altitude	angle	Current telescope altitude angle
rotator	angle	Current instrument rotator angle

Table 7.6

8. Array Commands

The commands to the IC’s acquire Command Object are of two purposes. First, to describe the region of interest, integration time, capture mode or other parameters. These are described in this section. The commands to actually capture an observation are also sent to the “acquire object, and are described in the next section.

At the level of the Instrument Sequencer, the ‘acquire’ command object is the main tool for getting images. There is to be one Command Object for each imaging array, Acquire1 and Acquire2 and are otherwise identical. The command set for these objects is described below. The details of the information flow within NICI are described in the “router” section.

8.1. Subarray Specification

```
acquire configure -subarray {list of subarray specifications }
```

Example:

```
Acquire1 configure -subarray { {0 0 20 40} {200 100 16 16} }
```

This sets two arrays starting at with upper left corner at (0,0) and lower right corner at (20,40), the other with upper left corner at (200,100) and lower right corner at (216, 116). The X dimension increments from left to right, the Y dimension increments from top to bottom.

8.2. Exposure Mode Selection

```
acquire configure -mode <modetype>
```

The legal modetypes are: arc_s (single image acquisition after array reset) or arc_d (image acquired after array pedestal noise subtracted off)

Example:

```
Acquire1 configure -mode arc_d
```

8.3. Set Non-Destructive Reads

`acquire -configure ndr <ndrcount>`

- `ndrcount`: the number of times to acquire data (adding into current image) between array resets.

Example:

```
Acquire2 -configure ndr 3
```

To set 3 non-destructive reads.

8.4. Set Reset To Pedestal Time

`acquire -configure rtime <time>`

-specifies the time in milliseconds to wait after the array reset before the pedestal data is read from the array.

Example:

```
Acquire2 -configure rtime 20
```

For a 20 millisecond delay after reset before a pedestal scan is to be taken.

8.5. Set Integration Time

`acquire configure -integrate <time>`

-specify the time in ms to expose array for each image scan

Example:

```
Acquire1 configure -integrate 1500
```

To integrate the arrival of photons on the imaging device for 1.5 seconds.

8.6. Set Number Of Exposures

`acquire configure -exposures <count>`

-specify the number of exposures to perform for next "go" command.

Example:

```
Acquire2 configure -exposures 10
```

Specifies that a sequence of 10 complete reset-integrate-capture cycles is to be taken.

8.7. Set DHS Sequence Attribute

`acquire configure -sequence <identifier>`

specify the DHS identifier for this data set.

Example:

```
Acquire2 configure -sequence A333
```

The DHS sequence identifier is generated by a request for a cookie from the DHS system. This is done by the Instrument Sequencer (IS) epics interface processor as part of its normal operation.

8.8. Set DHS Mode

`acquire configure -destination <destination list>`

-specify the destination for the following image acquisition. The list of legal destinations are: quicklook, data dest, {fitsfile <file prefix>} {fitssocket <legal socket specification> }

Example:

```
Acquire1 configure -destination quicklook
```

Specify the image is to be sent to the quicklook system (a DHS subscriber)

9. Observation Commands

9.1. Start Exposures

`acquire go`
start acquiring data

Example:

`Acquire2 go`
This will start taking the images. Images will be collected using the number of exposures, reset times, etc that have been set with the 'configure' commands.

9.2. Terminate Exposures After Current Exposure

`acquire stop`
stop acquisition after current image scan.

Example:

`Acquire1 stop`
Stop getting data after the current integration. If the integration time is excessive, this may take a while.

9.3. Terminate exposures immediately

`Acquire abort`
stop acquisition sequence immediately. This will result in a user level error condition. The NICI instrument must be reset before further commands will be accepted.

Example:

`Acquire2 abort`
Stop. Do not pass go. Do not collect \$100.
The NICI instrument will stop collecting data immediately and enter an error

10. AO System Commands

10.1. Off load of focus

As the telescope focus drifts the NICI AO system will correct for this wavefront curvature with the DM. When the focus correction in the AO system exceeds a preset limit a command will be issued from NICI to GEMINI to offset the focus by a certain amount in a certain direction. How often this happens will be determined by the focus drift rate of the telescope. Corrections should be expected at a rate comparable to what has been seen with the Hokupaa system. It is expected that these corrections will be made ??? times/hour.

Command:

`AO configure -focuslimit <limit value>`

The response will be:

`AO {configure -focuslimit <limit value>} stable`

Indicating that the AO system is using this value as the focus limit.

The command

`AO configure -focuslimit`

Will respond with the currently know value as:
AO {configure focuslimit <value>} stable

Additional responses are:

AO {configure focuslimit <limit>} error
The limit is outside of the legal values for the AO system.

10.2. Offload of tilt

- As the telescope position drifts the NICI AO system will correct for this wavefront tilt with the DM and the tip/tilt stage behind the DM. When the tilt correction in the AO system exceeds a preset limit a command will be issued from NICI to GEMINI to offset the telescope by a certain amount in a certain direction. How often this happens will be determined by the tracking rate error of the telescope. Corrections should be expected at a rate comparable to what has been seen with the Hukupaa system. It is expected that these corrections will be made ???? times/hour.

Command:

AO configure tiltlimit <limit value>

The response will be:

AO {configure tiltlimit <limit value>} stable

The command

AO configure tiltlimit

Will respond with the currently known value as:

AO {configure tiltlimit <limit value>} stable

Indicating that the AO system is to use this value as the tilt/tip limit.

Additional responses are:

AO {configure tiltlimit <limit>} error

The limit is outside of the legal values for the AO system.

10.3. Enable/Disable NICI control of tilt and focus

AO configure tilt on

The response will be:

AO {configure tilt on} stable

Indicating that the AO system is active and controlling the image. The AO system will stay in the transient state until an error occurs or is shut off with the following command.

To disable the NICI AO system, the command:

AO configure tilt off

The response will be

AO {configure tilt off} stable

Indicating the AO system tilt correction is turned off.

AO configure focus on

The response will be:

AO {configure focus on} transient

Indicating that the AO system is active and controlling the image. The AO system will stay in the transient state until an error occurs or is shut off with the following command.

To disable the NICI AO system, the command:
AO configure focus off

The response will be
AO {configure focus off} idle

Indicating the AO system is turned off.

Any configuration values will remain until the AO system is turned back on or the NICI instrument is shut down.

10.4. Commanding membrane mirror stroke

AO configure stroke <value>

Returns
AO {configure stroke <value>} stable

To query this value:
AO configure stroke

Returns
AO {configure stroke <value>} idle
Where idle is the currently known value for the stroke parameter.

10.5. Commanding loop gain

AO configure loopgain <value>

Returns
AO {configure loopgain <value>} stable

To query this value
AO configure loopgain

Returns
AO {configure loopgain <value>} stable
Where 'value' is the last known value for loopgain.

10.6. Enabling AO corrections

AO configure pause on

Returns
AO {configure pause on} stable
And the OA system will suspend its mirror corrections.

AO configure pause off

Returns
AO {configure pause off} transient

And resumes AO mirror deform operations.

10.7. Querying APD counts

The command
AO configure counts

Returns
AO {configure counts {list of values of APD counts} } transient

11. Mechanism Commands

This list summarizes the command syntax for each of the real mechanisms.

11.1. Test:

Every command object in the NICI system supports a “test” variant. This will parse the command for spelling and check if the command could be performed in the current state of the system. No movement or other state changes are done by the test variant.

Example:

```
FOCS TEST MOVE "Grism 1"
```

Would respond:

```
FOCS {TEST MOVE "Grism 1"} {No position "Grism 1"} reject
```

11.2. Status:

Every command object in the NICI system supports a “cget” variant. This will report on internal state. The command objects for mechanisms all have the position attribute. No movement or other state changes are done by the cget command. Example:

```
C1FW cget -position
```

Would reply

```
C1FW {cget -position} {K Methane Off} accept
```

11.3. Movement:

To cause the mechanism to position itself:

```
<CO> move <position>
```

Where <CO> is the command object and <position> is the new position requested.

Fiber Optic Calibration source

```
FOCS move grid
```

Would respond:

```
FOCS {move grid} accept
```

Further messages are possible, for example, if the mechanism moved through the “open” and “corner” positions, the FOCS would respond with intermediate positional information:

```
FOCS position open transient
```

Followed shortly by

FOCS position corner transient

And finally,

FOCS position grid stable

These status messages are broadcast to all user connections (ex. Engineering gui and thin-client). These status messages are similar for all of the mechanisms and will not be mentioned in the summaries below.

11.3.1.FOCS

FOCS move in

FOCS move out

FOCS move grid

FOCS move corner

The fiber optic calibration system will report its position as above.

11.3.2.TTSM

TTSM move X Y

Where X and Y are floating point values between -10 and 10. For example:

TTSM move 1.23 -3.545

The tip/tilt mechanism will report it's position as above.

11.3.3.NDFW

NDFW move ND2

NDFW move ND3

NDFW move ND4

NDFW move RED

NDFW move OPEN

NDFW move OFF

The Neutral density Adaptive optics filter wheel will report the final position on completion.

11.3.4.FPMW

FPMW move <position>

Where position is an rotational position in degrees.

Example

FPMW move 45.2

The focal plane mask wheel will report the final position on completion.

11.3.5.SMR

SMR move <position>

Where position is an rotational position in degrees.

Example

SMR move 45.2

The focal plane mask wheel will report the final position on completion.

11.3.6.PMW

PMW move BLANK
PMW move OPEN
PMW move 2
PMW move 3
PMW move 4
PMW move 5
PMW move 6
PMW move 7

The pupil mask wheel will report intermediate positions as described above.

11.3.7.DW

DW move "<position>"

Where <position> can be any one of: H/K, L/M, K-Methane, L-Methane, 50/50 Short, 50/50 Long, Hole, Mirror, Methane, [FeII], H2 1-0 s(1), Bracket Gamma. For example:

DW move "50/50 Long"

The dichroic beam splitter wheel will report intermediate positions as described above.

11.3.8.C1FW

C1FW move "<position>"

Where <position> can be any one of: OFF, J, H, K', L', M', H methane On, H Methane Off, K Methane On, K Methane Off, L Methane On, L Methane Off, J age Filter 1, J age filter 2, [FeII], [FeII]cont., H2 1-0 s(1), Br gamma, NB Continuum, Grism 1, Grism 2. For example:

C1FW move "K Methane Off"

The channel 1 filter wheel will report intermediate positions as described above.

11.3.9.C2FW

C2FW move "<position>"

Where <position> can be any one of: OFF, J, H, K', L', M', H methane On, H Methane Off, K Methane On, K Methane Off, L Methane On, L Methane Off, J age Filter 1, J age filter 2, [FeII], [FeII]cont., H2 1-0 s(1), Br gamma, NB Continuum, Grism 1, Grism 2. For example:

C2FW move "K Methane Off"

The channel 2 filter wheel will report intermediate positions as described above.

11.3.10.DOFF -- Digital Offset Command Object

The virtual command object Digital Offset will be used during normal NICI operation. The Instrument sequencer (IS) will compute information about the position of the telescope, relative rotation of the sky, and position of the instrument as these data items are contained in a database directly accessible to the IS, but not to the other portions of the NICI instrument. The IS will pass needed corrections to the instrument controller (IC) as commands to the Digital Offset (DOFF) command object. The DOFF will contain variables corresponding to the dithering offset and the compensation which may be read back in the standard way by:

```
DOFF cget -offset
And
DOFF cget -compensation
```

If the sum of the compensation and the dithering offset would exceed the allowable range of the Tip/Tilt Steering Mirror, the command will be rejected with an appropriate message.

The command summary will be:

```
DOFF NULL
```

The current position of the Pupil Mask Wheel and tip/tilt mirror (and hence the star on the imaging array) will be considered to be in the primary position. This position will be the origin for dithering offsets.

```
DOFF moveto <position>
```

The TTSM and the pupil mask wheel PMW are moved together to move the image of the star to <position> relative to the NULL position of the image array. The position is a floating point number.

```
DOFF move <position>
```

The TTSM and the pupil mask wheel PMW are moved together to move the image of the star to <position> relative to its current position on the image array. The position is a floating point number.

```
DOFF compensate <x> <y>
```

The correction of the star position also depends on the atmospheric wavelength correction (since the AO steering is done by visible light and the imaging array is sensitive to infrared), as well as the instrument flexure. Both of these depend on information known to a database accessible to the IS. The IS will precompute the vector sum of the compensation required. That compensation will be added current dithering position (vector sum) and sent to the TTSM by the Digital Offset.

The intended use of the DOFF is as follows:

Close off optical path:

Move DOFF to zero.

Turn off AO control of system.

Slew telescope to star.

Turn on AO control of system.

Move Pupil Mask wheel to desired position.

Apply positional correction with DOFF compensate command

Null the star position with DOFF NULL

Open the optical path

The TTSM should not be controlled directly by the IS during normal operation.

12. Environment Commands

The NICI instrument contains thermal devices manufactured by Lakeshore. The IC software will initialize the lakeshore controllers making them appear as simple data elements of the temperature command object. There will be a range of legal temperature values for all of the sensors. When the value goes outside of the range, the system may be forced into an error mode. This may force an automatic recovery (if the terminal interface controller is down, for example) or may shut down one or both of the arrays and signal a serious temperature situation.

The sensors and controllers will be known as software controlled objects with the following names: LC1, LC2, LS for Lakeshore Controller 1 and 2 and Lakeshore Sensor.

Software Names for Temperature Control and Sensing access -- NCI

Equipment position	CO name	Range list	Error?
Array #1 mount sensor	LC1::S1	LC1::R1 = {lowval, hival}	Array #1
Array #2 mount sensor	LC2::S1	LC2::R1 = {lowval, hival}	Array #2
Cold Structure Sensor	LS::S4	LS::R4 = {lowval, hival}	Entire instrument

The Command Objects LC1, LC2 and LS will have the local variables (attributes) as above. Setting the values of the ranges will be done by the 'configure' subcommand. The values for the sensors will be automatically scanned from the Lakeshore instrument controller in the following sequence and pacing:

LC1::S1, LC2::S1, LS::S4 (and again from start). Each value will be read sequentially at intervals of 5 seconds. The total scan will take 15 seconds. Each time the value is updated, it will be compared with the valid range. If the range is exceeded an error will be signaled.

When the scan is taking place, if the last value requested has not been sent by the time the 5 second polling interval has expired, the Terminal interface controller will be marked as suspect. If this happens twice, the terminal interface controller will be sent the 'recover' command. Scanning will continue. If the terminal interface controller is down for an entire scan (15 seconds), the system will report an error. If the range of valid temperatures is set to the empty set, the comparison will not take place and the value, however weird will not cause an instrument error condition.

The information from the scan will be similar to the user interface issuing the command: <CO> cget - <variable>. For example:

```
LS cget -S4 32.05
```

Would be a valid message indicating that LS::S4 currently reads at 32.05 degrees (the trailing command accept/reject word is not present, since this message is NOT generated as a direct result of a user command)

The specific examples for the temperature subsystem are for setting ranges or reading values as follows.

To set the range values for the command objects, use the following syntax:

```
LC1 configure -R1 { lowval, hival }
```

```
LC2 configure -R1 { lowval, hival }
```

```
LS configure -R4 { lowval, hival }
```

To disable the value range use the following example syntax:

```
LS configure -R1 { }
```

The values read back will look like:

```
LC1 cget -S1 <value>
```

```
LC2 cget -S1 <value>
```

```
LS cget -S4 <value>
```

13. Status Summary

This section contains a summary of the command set provided by the component controller. Detailed information on each command can be found in section TBD below.

The commands available listed alphabetically:

13.1. Adaptive optics system status – “AO cget attribute”

<i>Name</i>	<i>Description</i>
counts	Return the current APD counts
focus	Return current focus offloading state
focuslimit	Return current focus offloading deadband
loopgain	Return current AO loop gain
pause	Return current AO correction state
stroke	Return current membrane mirror stroke
tilt	Return current tip/tilt offloading state
tiltlimit	Return current tip/tilt offloading deadband

Table 13.1

13.2. Mechanism configuration – “mechanism cget attribute”

<i>Name</i>	<i>Description</i>
position	Get the mechanism target position

Table 13.2

14. Scenarios

This section provides a number of scenarios that attempts to describe the sequence of actions that would achieve some desired results. The aim is to provide a clearer insight into how the IS to IC interface is expected to be used.

14.1. Scenario List

1. Initialization sequence
2. Normal exposure sequence
3. Stopping an exposure in progress

4. Aborting an exposure in progress
5. Nodding sequence
6. More!

14.2. Initialization sequence

14.3. Normal exposure sequence

14.4. Stopping an exposure in progress

To stop an exposure in progress at the earliest convenient moment, prior to starting a new scan:

Acquire stop

14.5. Aborting an exposure in progress

To terminate immediately an exposure in progress:

Acquire abort