NATIONAL
OPTICAL
ASTRONOMY
OBSERVATORY

MAJOR INSTRUMENTATION GROUP
950 N. Cherry Ave.
P. O. Box 26732
Tucson, Arizona 85726-6732
(520) 318-8000 FAX: (520) 318-8303

MONSOON
Data Handling System Interface

Status and Data Stream Description

NOAO Document ICD 1.1

Revision: 1.0

Authored by:
Nick C. Buchholz, Greg Chisholm, Phil N. Daly and Peter Ruckle
1/7/2002
Please send comments:
nbuchholz@noao.edu

Revision History

Version	Date Approved	Sections Affected	Remarks
0.0.1	3/11/2002	All	First release draft
0.0.2	4/12/2002		
1.0	9/28/2006	All	Reformat and edit - aro

Table of Contents

List of Figures

List of Tables

1.0 Introduction

1.1 Scope

The output of the Generic Pixel Server (GPX), the Instrument Control System (ICS) and the Generic Manager of Instruments (GMI) must all be combined at some single point to allow the data handling system to reduce and use the data. This document describes the message interface to this DHS interface program. It includes the Data/Status stream used by a Generic Pixel Server and the status and event messages produced by the ICS and GMI layers.

The Data and Status outputs from the **GPX** (ICD 4.0) have so far been left unspecified. It has been agreed to some extent that a system for publishing the data and status is required for all instruments. This document serves as a beginning of the discussion of the interface to the data and status streams from instruments.

In addition, while developing the ICD 4.0 description of the Data and Status outputs, the conclusion was reached that an ICD describing what the DHS Interface can expect to receive from the entire Observation Control System was more appropriate. It is assumed that the GPX will conform to this ICD completely.

The document is divided into the following sections:

- **Section 1.0 - Introduction.**
- **Section 2.0 - DHS Application Programmer Interface**
- **Section 3.0 - DHS Internals**

The intended audience for this document is:

- Anyone with an interest in the Data Handling System Interface.

1.2 Acronyms and Glossary

1.2.1 Abbreviations and Acronyms

AC	Acquisition Camera
ADC	Analog to Digital Converter
CCD	Charge Coupled Device
CCDACQ	CCD Acquisition Board
CDS	Correlated Double Sampler
CLKBRD	Clock and Bias Board
COTS	Commercial Off the Shelf
CPCI	Compact PCI
CPLD	Complex Programmable Logic Device
CTC	Command To Convert
DAC	Digital to Analog Converter
DCS	Detector Controller System (software)
DHE	Detector Head Electronics
DHS	Data Handling System

1.2.1 Abbreviations and Acronyms (Cont.)

DOP	Data Output Port
ECS	Enclosure Control System
EEPROM	Electrically Erasable Programmable Read Only Memory
EIDN	Electronic Identification Number
EM	Electromagnetic
EMI	Electromagnetic Interference
ES	Embedded System
FITS	Flexible Image Transport System
FP	Focal Plane
FPA	Focal Plane Array
FPDP	Front Panel Data Port -
FPGA	Field Programmable Gate Array
FPM	Focal Plane Module
GPX	Generic Pixel Server
IAS	Image Analysis System
IC	Integrated Circuit
ICD	Interface Control Document
ICS	Instrument Control System
ID	Identifier
IDPS	Image Data Preprocessor System
IR	Infrared
JTAG	The usual name used for the IEEE 1149.1 standard entitled ***Standard Test Access Port and Boundary-Scan Architecture***
LAN	Local Area Network
MCB	Master Control Board
MHz	MegaHertz
MONSOON	Not an acronym
MOP	
MSL	MONSOON Supervisory Layer
N/A	Not Applicable
NICD	NOAO Interface Control Document
NOCS	NEWFIRM Observation Control System
OCS	Observatory Control System
ODI	One Degree Imager
OTA	Orthogonal Transfer Array
PAN	Pixel Acquisition Node
PCB	Printed Circuit Board
PDF	Parameter Description File
PDT	Parameter Description Table
PWM	Pulse Width Modulated
QUOTA	Quad Orthogonal Transfer Arrays
ROI	Region of Interest
SCA	Sensor Chip Assembly
SUS	Status Update System
TBD	To Be Decided

1.2.2 Glossary

Attribute	An entity that describes some aspect of the configuration of a Pixel Server System, such as the level of a voltage or the state of a shutter. Some attributes are used by the Pixel Server System as command parameters. The OCS communicates with a science instrument by sending it sets of “attributes” and “values”.
Command	An instruction requiring a system to start some action. The action may result in a voltage changing or some internal parameters being set to particular values. A command may have command parameters (arguments) that contain the details of the instruction to be obeyed.
Data Array	The data, while it is stored in data processing memory, which resulted from one or more readouts of an IR array or CCD detector.
Data Set	A self-contained collection of data generated as a result of a Pixel Server obeying a *gpxStartExp* command. Each *gpxStartExp* command results in one and only one data set.
Detector Head Electronics	The lowest level hardware system. It is normally closely connected to the photon detector and coupled to the dewar in which the detector resides.
Exposure	The name used to describe the process and the data resulting from the process of resetting/clearing a detector, exposing it to photons and then reading one or more frames to determine the photon levels. These frames are processed into a data array, called an exposure, which may be further processed. (For example, an exposure would be the data array that results when a single Reset-Readout-Integrate-Readout cycle is performed on an IR detector or a single CCD Clear-Integrate-Readout cycle.)
Exposure Sequence	The process by which valid data is produced. Various levels of exposure sequencing occur during an observing run. At the lowest level there are the Reset-Readout-Integrate-Readout or Clear-Integrate-Readout cycles that result in a single IR or OUV exposure. At the highest level are the observing sequences that move the telescope, configure the instrument and take a series of exposures that create an observation.
Focal Plane	The geometrical plane where the image from an optical instrument is formed. This is the physical location of the detector device.
Focal Plane Segment	A collection of one or more detectors arranged to collect photons from an instrument. A Focal Plane Segment is controlled by a single Pixel Acquisition Node (PAN).

1.2.2 Glossary (Cont.)

Frame	The result of one or more readouts of an array averaged pixel by pixel. Each frame represents the signal values obtained from reading the entire ROI being read out of the detector. Multiple frames may be processed into a single exposure.
Generic Pixel Server	A pixel server that conforms to the GPX Interface description.
Guide Core	The software routines that calculate the centroids and image shifts required for controlling an Orthogonal Transfer Array (OTA).
Guide Map	An array of eight bytes that have a 1 in each position corresponding to an orthogonal transfer array (OTA) cell that will be used in the guide calculation.
Guide Region	A portion of an OTA guide cell as defined by the Guide Mapthat contains a guide star.
Image	The array of detector pixel and description data representing a science or diagnostic image or spectrum. An *image* is capable of being displayed or processed as a discrete entity. The values in the array may be stored in memory or on disk and are related to the data taken by the detector by some processing algorithm, (for example an *image* may consist of all the coadded and averaged exposures in one beam of a chop mode *gpxStartExp* command).
Image Acquisition System	A system of software and hardware capable of producing images from a focal plane on command.
Image Server	See Image Acquisition System.
MONSOON Image Acquisition System	A Generic Pixel Server. An extensible, modular Image Acquisition System. The system design is, to the extent possible, independent of the hardware being used in a particular implementation. Each component of the system should be capable of replacement by a similar component without having to redesign the rest of the system. Each component of the software is, as far as possible, independent of the underlying hardware and as modular as possible.
MONSOON Star Date	A date/time value that gives a unique ID to exposures in MONSOON systems. The MSD is formed using the JulianDay + TimeOfDay (to the nearest 86.4 ms .000001 of a day). The exposure ID is calculated to the nearest ms but on display is truncated to six decimal places.
Observation	The process of exposing the focal plane to photons in one or more exposures. The result of an observation is an image.

1.2.2 Glossary (Cont.)

Pixel Acquisition Node	The computer that handles the interface to the detector head electronics and the image pre-processing of the data stream from the *Detector Head Electronics.*
Pixel Server	A system which produces pixel values when requested to do so by some client system.
Pixel Server System	The combination of the *Detector Head Electronics* and a *Pixel Acquisition Node* which are coordinating the task of taking exposures and archive the resulting *data set.*
Read	When used as a noun to describe instrument data, this refers to a single read of a pixel on the detector. A read may consist of several A/D conversions of the pixel data that are averaged or processed in some other way to produce a single integer output value for the pixel. A Readout is made up of one read of each pixel in the detector ROI being read.
Readout	When used as a noun to describe instrument data, this refers to a single read of every pixel in the detector. One or more readouts can be averaged pixel by pixel to create a frame.
Region of Interest	A sub-array of the available detector area. There are two types of sub-arrays that can be defined. The Sequence ROI is on the active surface of the array used to increase the frequency of the Array readout. The Data Reduction ROI is an arbitrary rectangle of any size that fits on the Array. Data Reduction ROIs are defined to reduce the volume of data sent to the disk or DHS even when the entire array is being read out.
Supervisory Node	A computer capable of controlling multiple Image Acquisition systems. The computer that runs the software that conforms to the GPS interface.
Value	The value associated with an “attribute”.
Word	Four bytes or 32 bits.

1.2.3 Standard Terminology

To avoid confusion and to make very clear what the requirements for compliance are, many of the paragraphs in this standard are labelled with keywords that indicate the type of information they contain. The keywords are:

- RULE
- RECOMMENDATION
- SUGGESTION
- PERMISSION
- OBSERVATION
- REQUIREMENT
- GOAL

These keywords are used as follows:

RULE

<Paragraph Number> Subject Describing Text RULE

Rules form the basic framework of this draft standard. They are sometimes expressed in text form and sometimes in the form of figures, tables or drawings. All rules shall be followed to ensure compatibility between components. All rules use the “shall” or “shall not” words to emphasize the importance of the rule.

RECOMMENDATION

<Paragraph Number> Subject Describing Text RECOMMENDATION

Wherever a recommendation appears, designers would be wise to take the advice given. Doing otherwise might result in some awkward problems or poor performance. It is possible to design a system that complies with all the rules but has poor performance. Recommendations found in this standard are based on this kind of experience and are provided to designers to speed their traversal of the learning curve. All recommendations use the “should” or “should not” words to emphasize the importance of the recommendation.

SUGGESTION

<Paragraph Number> Subject Describing Text SUGGESTION

A suggestion contains advice that is helpful but not vital. The reader is encouraged to consider the advice before discarding it. Some design decisions that should be made are difficult until experience has been gained. Suggestions are included to help a designer who has not yet gained this experience.

PERMISSION

<Paragraph Number> Subject Describing Text **PERMISSION**

In some cases, a rule does not specifically prohibit a certain design approach, but the reader might be left wondering whether that approach might violate the spirit of the rule or whether it might lead to some subtle problem. Permissions reassure the reader that a certain approach is acceptable and will cause no problems. All permissions use the “may” word to emphasize the importance of the permission.

OBSERVATION

<Paragraph Number>Subject Describing Text **OBSERVATION**

Observations do not offer any specific advice. They usually follow naturally from what has just been discussed. They spell out the implications of certain rules and bring attention to things that might otherwise be overlooked. They also give the rationale behind certain rules so that the reader understands why the rules shall be followed.

REQUIREMENT

<Paragraph Number>Subject Describing Text **REQUIREMENT**

Requirements are expressed as a minimum acceptable value that will allow the performance characteristics of the system to be met. This term differs from the definition of a RULE in that a requirement specifies a ‘quantity’ that does not connect to anything (i.e. is not an interface) but rather provides a measure of the level of performance of a component of the interface.

GOAL

<Paragraph Number>Subject Describing Text **GOAL**

Goals are expressed as a desired improvement to a requirement that would enhance the system level performance in a significant way.

1.2.4 Reference Documents

SPE-C-G0037, "Software Design Description", Gemini 8m Telescopes Project.

"ICD/16 - The Parameter Definition Format", Steve Wampler, Gemini 8m Telescopes Project.

WHT-PDF-1, "FITS headers for WHT FITS tapes", Steve Unger, Guy Rixon & Frank Gribbin, RGO.

NOST 100-1.0, "Definition of the Flexible Image Transport System (FITS)", NASA Office of Standards and Technology.

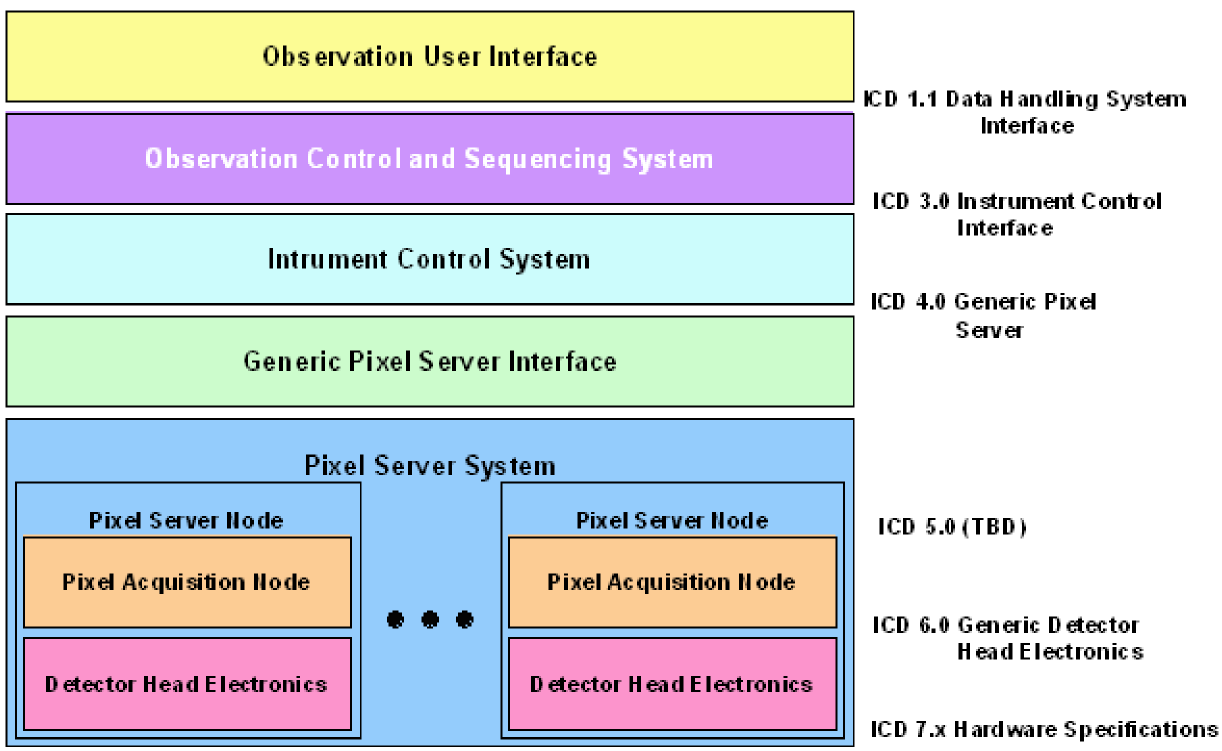
GEN-SPE-ESO-00000-794, "ESO Data Interface Control Document", Miguel Albrecht, ESO.

IEEE Std. 610.12-1990 - "IEEE Standard Glossary of Software Engineering Terminology", Standards Coordinating Committee of the IEEE Computer Society, USA, 19901210

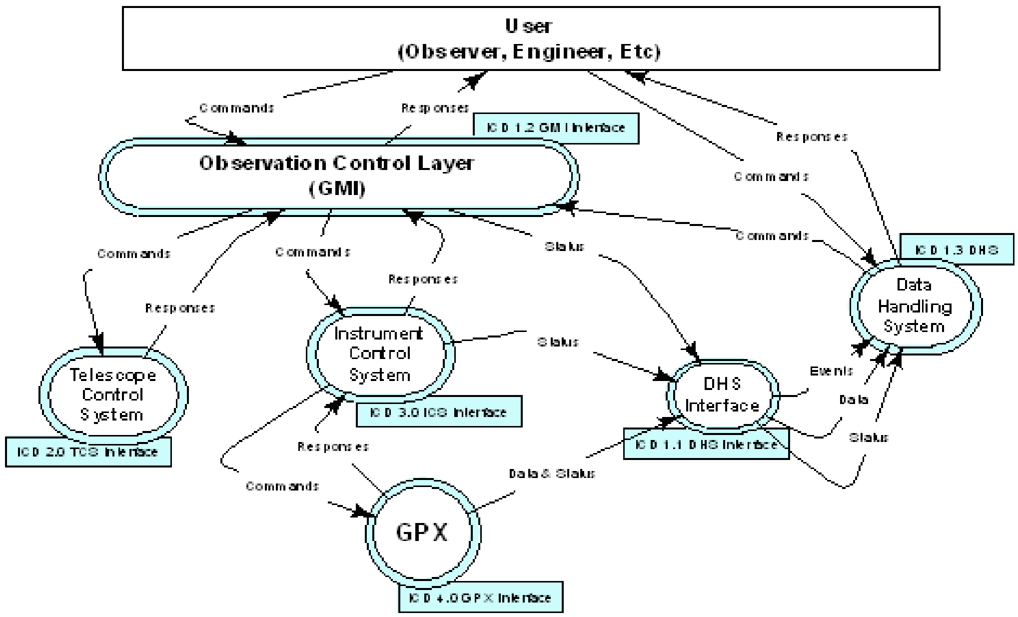
ANSI/IEEE Std 754-1985 - "IEEE Standard for Binary Floating-point Arithmetic" - Standards Committee of the IEEE Computer Society, USA 19850812

xxxx "XDR - Extended Data Representation Standard" ????

ICD 6.0 - "Generic Detector Controller - Command and Data Stream Interface Description", Nick C. Buchholz (NOAO), Barry M. Starr (NOAO), Version 0.1.2, dated: 200203011- NOAO Document Number MNSN-AD-01-0004



Observatory System Reference Model
Figure 1



System Context Diagram
Figure 2

2.0 DHS Application Programmer Interface

The DHS interface API will be included entirely within a single shared library loaded at runtime into the data publication process (panSaver) of the GPX. The library may use whatever data transfer method the DHS group deems appropriate. The nature of this protocol will be buried within the DHE API library.

2.1 Standard Calling Sequence for DHS API Functions

Each DHS API function will have a defined parameter set as follows:

*dhsOpen(long *status, char *response, dhsHandle dhsId, { additional parameters});*

2.1.1 Parameter 0 - long *status

This parameter performs two functions. First, it gives a status value which is inherited from earlier routines. Second, it provides a means for each routine to report its status to later routines. On entering each routine the routine should check the incoming status value and simply return if the Status indicates a previous error.

2.1.2 Parameter 1 - char *response

This parameter is a character buffer pointer that provides a means for status information to be returned to the upper levels of the process and eventually the user. The pointer may be null, however, if it is non-null it must point to a memory buffer that contains at least 4096 bytes. Each routine should detect the presence of this address and fill in the buffer only if the pointer is non-null.

2.1.3 Parameter 2 - dhsHandle dhsId

Except for the dhsOpen command, each dhs routine will pass in a handle parameter obtained by doing a dhsOpen. The dhsOpen routine will pass in a dhsHandle pointer (dhsHandle *) in which the routine will store the handle to a successfully opened DHS interface or an ERROR(-1) value.

2.2 DHS API Routines

2.2.1 openDhs -

*openDhs(long *status, char *response, dhsHandle *dhsID, TDBType *whoAmi, TBDType *configuration)*

2.2.2 openExp -

*openExp(long *status, char *response, dhsHandle *dhsID, TDBType *whoAmi, TBDType *configuration)*

2.2.3 closeDhs -

*closeDhs(long *status, char *response, dhsHandle dhsID)*

2.2.4 closeExp -

*closeExp(long *status, char *response, dhsHandle dhsID)*

2.2.5 dhsSendAVData-

*dhsSendAVData(long *status, char *response, dhsHandle dhsID, void *blkAddr, size_t blkSize, TBDType DescriptorStruct)*

2.2.6 dhsSendArrData-

*dhsSendArrData(long *status, char *response, dhsHandle dhsID, void *blkAddr, size_t blkSize, TBDType DescriptorStruct)*

2.2.7 dhsSendImage -

*dhsSendImage((long *status, char *response, dhsHandle dhsID, void *blkAddr, size_t blkSize, TBDType DescriptorStruct)*

2.2.8 dhsReadImage ???

2.2.9 dhsIoCtl

*dhsIoctl(long *status, char *response, dhsHandle dhsID, long ObsID, long ExpID, ...)*

2.3 DHS API Structure Definitions

2.3.1 Configuration Structure - used by dhsOpen

2.3.2 AVData Descriptor Structure

2.3.3 FITSData Descriptor Structure

2.3.4 ArrData Descriptor Structure

2.3.5 imageData Descriptor Structure

2.4 DHS Parser - What Does the DHS Know

2.4.1 ASCII Tables - FITS Header Data

2.4.2 ASCII Tables - Attribute-Value Pair Data

2.4.3 Binary Arrays - Like OTA Shift Lists

2.5 DHS API Behavior in the Face of DHES Failures

3.0 DHS Internals

The status and data stream message protocol to the DHS interface is defined in this section. The protocol uses messages passed over a socket stream to send pixel, status and event data to the outside world. The messages are considered to be a stream of bytes which are interpreted by the receiving process/client to be in one of the defined formats.

3.1 Byte Order RULE

All messages shall use network byte ordering. Bytes in a message are labelled from 0 to N (where N is the length of the message).

3.1.1 Creating Messages RECOMMENDATION

The sending software should build the message using htonl() or htons() or similar routines to convert from host ordering to network ordering. The receiving software SHOULD convert the message byte order into something which is usable locally using ntohl() or ntohs() or similar functions.

3.1.2 Strings RULE

Strings which are embedded in a message shall be inserted with the left most character of the string in the lowest order message byte. As an example, string “ABCD” will appear in message Byte N through N+3 with ‘A’ in byte n and ‘D’ in byte n+3.

3.1.3 Very Long Integers **RULE**

The protocol implementers shall make provision for 64-bit integers by using the network ordering decision used for long integers. In the messages the Most Significant Byte of a very long integer shall be sent first, that is, will be closest to the start of the message.

3.1.4 Floating Point and Double Values **RULE**

Real numbers represented by floats or doubles in 'C' shall be represented in the messages in IEEE floating point format (see Bold]Default ¶ Font). The byte ordering shall be as defined in the XDR/Network standard.

3.2 Message Structure **RULE**

DHS Interface messages shall have the following structure:

3.2.1 Message Header **RULE**

The message header shall have the structure described in Figure3.

Figure information required

Message Header Structure Description **RULE**
Figure 3

3.2.1.1 Message Type Field **RULE**

3.2.1.2 Protocol Version Field

3.2.1.3 Exposure ID Field

3.2.1.4 Message Body Length

3.2.1.5 Message Source IP Address

3.2.2 Message Body **RULE**

The message body shall be a number of bytes equal to the Message Body Length in the message header field. The contents of the message body are interpreted by the receiving system in accordance with the Message Type and Protocol Version stored in those message header fields.

3.2.3 Message Error Check **RULE**

The Message Error Check shall be a four byte CRC value calculated by doing a "exclusive or" operation of all other bytes in the message in groups of four bytes.

3.3 DHS Interface Message Types

3.3.1 Control Messages

3.4 Command/Response Communications Stream Definition RULE

3.5 Status and Data Stream Interface RULE

The status information and data output from the **GPX** shall be a stream of messages across a socket connection. Any client wishing to use the data produced by the **GPX** may connect to this socket and receive the message stream.

3.5.0.1 Data Output Minimum RULE

At a minimum, every **GPX** will have a mode that is able to produce a **FITS-**formatted image on a local disk for engineering and diagnostic purposes. This capability is in addition to the message based Data interface

3.5.0.2 Status Output Minimum RULE

At a minimum, every **GPX** will have a mode that allows the display of status information on a terminal-like display, as lines of status messages.

3.5.0.3 Local Client / Remote Client Message Formats RULE

3.5.0.4 Final Data Stream OBSERVATION

The final data stream from the **GPX** will be partially determined by local protocols.

3.5.0.5 Data Stream Minimum PERMISSION

It is possible for an implementation **GPX** to produce a stream of FITS images to export the data. This stream could then be transformed into the **GPX** standard format or into some local format such as Gemini DHS, NOAO PicFeed, IDL, and so forth, by an auxiliary process.

3.5.1 Keyword Messages RULE

Keyword messages are used to inform the final user of the data as to the value of an attribute of the data. A keyword message consists of

3.5.1.1

3.5.2 Interpretable Header Block Messages RULE

Interpretable Header block messages are strings that represent FITS header information to be included with the final image. This information is not examined or acted on by the **GPX** and may be interpreted and modified as required by the FITS standard or local DHS practices.

3.5.2.1

3.5.3 Inviolate Header block Messages RULE

Inviolate Header block messages are strings that represent FITS header information to be included with the final image. This information is required to be included in the form presented without modification.

3.5.3.1

3.5.4 Event Messages

GPX will indicate to the data consumers that a particular event has taken place.

3.5.5 Pixel Block Messages

Pixel Block messages are blocks of pixel data being sent. A short preamble in the message body describes the format and unscrambling needed by the data.

Appendix I Keywords Produced by a GPX System

This table is totally incomplete. We wanted to get a more defined basic structure before we went into these details so this can be ignored with the knowledge that most of the information about the state of the **GPX** will be preserved by the final system.

Table 1 – Keywords

Keyword Name	Usage/Explanation
numArrays	An integer value giving the number of arrays in the focal plane.
arrayDescriptor type rows columns outputsPerArray	A structure describing the characteristics of the array being controlled. The components are: a string giving the type of array, Two integers giving the size in rows and columns and an integer giving the number outputs on the array. Other elements may be needed for certain arrays.
outputArrangement picQueue baseR, baseC strideR, strideC chunkR, chunkC sizeR, sizeC	A structure which outlines how the array outputs are read. This includes a queue descriptor which tells where to place the pixels for processing and information on the structure of the pixel data block transferred. We describe the block of data as chunks of contiguous pixels separated by a intervening pixels from other blocks. The block is described by a number of integers giving the starting row and column of the block, a row and column stride (the number of pixels to skip when storing chunks) the row and column chunk size and the total size of the final block of data in rows and columns
spcDescriptor gain settingTime offset noiseFoM	A structure which describes the configuration of the signal processor chains in the system. The components of the structure are floating point arrays which describe the gain, settling time, and offset of the signal processing chain. Included is a noise figure of merit (TBD) which will allow the quietest set of chains to be chosen when that is important.
waveForms	A descriptor for the timing waveforms to be run when running the array. These will be an array of bytes which will either describe of define the timing of the array readout. It is expected that each system will have an idiosyncratic way of describing these waveforms.
DacValueN -float	An array of floating point voltage values which are to be loaded into any DAC settable voltages used to control the array. Each system will likely have a unique set of these voltages and an mapping from voltage name to DAC number should be provided in the **GPX**
Min Integration Time	A floating point number giving the minimum integration time achievable by the system
Base Readout Time	A floating point number giving the fastest possible readout time for the entire array.

Table 1 – Keywords (Cont.)

Keyword Name	Usage/Explanation
spdRoiDescriptor Row0 Col0 rowSize colSize	A structure which describes a “speed up” region of interest (ROI). This is provided so a system with a large array can describe a sub array which will be readout to provide faster readout and shorter integration times. (Mostly used for IR systems without an internal cold shutter) The ROI is described by four integers giving the First row and column to be read and the size of the ROI in rows and columns.
binning	An integer value giving the binning factor for the array readout. This may be two values if the row and column binning factors are different.
intTimeSecs	A floating point number giving the desired integration time to use in seconds.
digAvgs	An integer giving the Number of Digital Averages to use while reading out the pixels
numPics	An integer giving the Number of pictures to generate for each gpxStartExp
ROIdescrptors Row0 Col0 rowSize colSize	A list of structures defining the regions of interest (ROI) to readout and archive. The components of the structure are four values representing the first row and column included in the ROI and the row and column size of the ROI
Outputs to Read	An integer giving the number of outputs on the Array which will be used during the readout.
PreFlash	A Boolean Value determining if the exposure sequence will include a pre-flash step.
waveFormsToRun	A list of the Waveforms to run during this array readout.
shutterState	A Boolean Value determining if the shutter is to be opened during the Integration time.
arrayPowerState	A Boolean Value determining if the array will be activated/powered-up during the exposure
intraPixelDelay	A floating point number giving the amount of time to allow for settling while reading each pixel
idleProcess	An Integer tag describing how the array will be run during any Idle time in the observing run
Data Disposition -struct disposition - procedure name Arguments - filename, directory image format, data type, data stream/queue	
Pre-Processing Algorithm	
Unscrambling Algorithm	
Image Data Set ID	
coAdds – integer	

Table 1 – Keywords (Cont.)

Keyword Name	Usage/Explanation
fSamples – integer	
binning – integer	
intTimeSecs – float	
digAvgs – integer	
numPics – integer	
shutterState	
arrayPowerState	
Data Disposition	
Pre-Processing Algorithm	
Unscrambling Algorithm	
Image Data Set ID	
coAdds – integer	
fSamples – integer	
TriggerSource	
TriggerTimeOut	
Image Data Set ID	
None	
intTimeSecs – float	
intTimeSecs – float	
Current Shutter State	
Row or Y shift	
Column or X shift	
Units to Simulate	
Units to Test	
System Power State	
System Reset Level	

Appendix II - Data Stream Interface Questions

1. Will we need more than 255 message types?

   No, this should be enough.

2. Is one byte sufficient for version information?

   Yes, there is a suggestion that no version information be included. We'll include it although it's not needed.

3. Should messages be self describing or depend on defined structure?

   Defined structure.

4. Is the IP address sufficient for the message source ID?

   This is adequate.

5. Is a 4 Byte length too much? Would 2 bytes be enough?

   Two bytes is not quite enough and four bytes is probably too large but is a convenient size. Pixels will be sent as a series of 4-byte integers with a maximum message length in bytes of 4 x (NumPixels) + Preamble size or 4 x (65536) + Preamble. Assuming 65536 pixels per message, this will require 256 messages for a 4k x 4k image.

6. Should messages be grouped by time or by ID tag?

   Both! We will have a start association message and each message will include the association ID.

7. Should image IDs be strings or binary?

   Strings.

8. Should Association IDs be strings or binary?

   Strings.

9. How should pass through data be handled?

   As Header Blocks to be interpreted or not as required.

10. What formats should be allowed for output pixels?

    8-bit signed or unsigned bytes; 16-bit signed or unsigned integers; 32-bit signed or unsigned Integers; 32-bit IEEE Floating point numbers.

11. How should pixel ordering be described?

    A structure with fixed format.

12. How should pixel type be described?

    With text tags: byte, ubyte, short, ushort, int, uint, float.

13. Should the message stream include internal status messages such as housekeeping channels, state information and so forth?

    Yes.

14. What Event Classes have already been described?
15. What Events have already been described?
16. Can keywords be grouped into single messages?
17. Do we need a table data format?

    Yes.
18. Where should unscrambling take place? Should data alwaysbe presented in order?

    No, may or may not be assembled into an image.
19. Which will be more important, efficiency or human readability?

    Efficiency for pixels, human readability for everything else.