

HIRDLS

SW-UCB-087

HIGH RESOLUTION DYNAMICS LIMB SOUNDER

Originator: Tom Lauren

Date: 21 May 04

Subject/Title:

**The HIRDLS Science Investigator-led Processing System (SIPS)
Operations Manual**

Description/Summary/Contents: Provides guidance on the operations of the HIRDLS SIPS.

Keywords:

Purpose of this Document:

**Oxford University
Atmospheric, Oceanic &
Planetary Physics
Parks Road
OXFORD OXI 3PU
United Kingdom**

**University of Colorado, Boulder Center
for Limb Atmospheric Sounding
3085 Center Green Dr.
Boulder, CO 80301**

EOS

The HIRDLS Science Investigator-led Processing System (SIPS) Operations Manual

OVERVIEW

The High Resolution Dynamics Limb Sounder (HIRDLS) Science Investigator-led Processing System (SIPS) is an application designed for operational, event-driven, instrument data processing. Its primary goal is to process HIRDLS data in a prompt and efficient manner and subsequently deliver science products (data files) to the Goddard Earth Sciences Distributed Active Archive Center (GES DAAC) for eventual use by the scientific community. This document provides guidance on the operation of the HIRDLS SIPS.

1. INTRODUCTION

HIRDLS data will arrive at the GES DAAC in the form of telemetry packets. The data are formed into 2-hour granules (L0 data) and distributed to the SIPS along with any associated ancillary data such as engineering housekeeping data and satellite orbit/attitude values. As distributed data arrives, it is archived locally in near line storage (NLS). Once enough data has accumulated, the SIPS uses three major processing stages in moving from telemetry to geophysical parameters: L1, L2 and L3. A science data processor is associated with each product level and these processors are delivered from the HIRDLS Science Computing Facility (SCF). After data products are generated, stored in NLS, and quality-checked, they are delivered to the GES DAAC for wider distribution to the scientific community. Figure 1 shows an overview of the HIRDLS data flow.

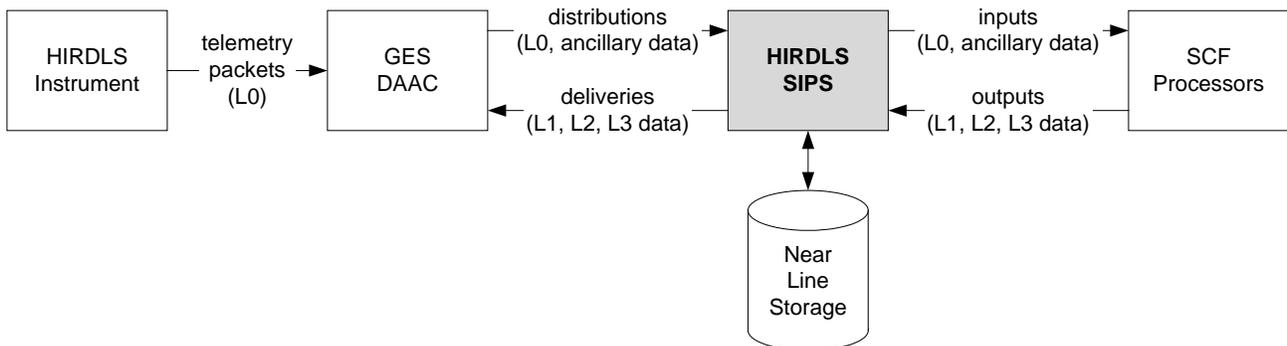


Figure 1 The HIRDLS data flow.

The SIPS consists of several different elements. *Products* model the data in the system. *Near line storage* is a file system where data products are stored locally. *Distributions* represent the transfer of data from the DAAC. *Processors* are software executables that transform the data. *Jobs* consist of one or more processors chained together to form a processing event. *Sandboxes* are the directories where data processing takes place. *Deliveries* represent the transfer of data to the DAAC.

The application provides a Web-based user interface that allows an operator to monitor distributions, define and search for products, install and configure processors, run and control jobs, and make deliveries.

Two major hardware systems are used to perform the processing. *hir1* is an SGI Origin 3900 with 12 500MHz R14k Processors and 20 600MHz R14k Processors including 26 GB of shared memory running Irix 6.5. *hal* is an SGI Altix 3300 with 12 1.3 GHz Itanium 2 processors including 23 GB of shared memory running Redhat Enterprise Linux version 2.1 with SGI modifications.

2. PRODUCTS

An operator must use the SIPS GUI to configure SIPS to understand the various file types that are involved in the HIRDLS production environment. SIPS classifies each data file as belonging to a certain type, called a file type. An example of a file type is HIRDLS1 data. The operator must associate certain information with a file type, such as a short name, file suffix, description, granule length, time reader, and file storage path. This information provides SIPS with the knowledge that it needs to store the file, provide completeness information, and extract data times.

File types that are closely related are grouped into a product type. An example of a product type is HIRDLS1 data, HIRDLS1 metadata, and HIRDLS1 production history.

An Earth Science Data Type (ESDT) is the DAAC's view of a product type. The operator must use the SIPS GUI to define the ESDTs that are specific to the HIRDLS instrument. The most important aspects of an ESDT are its short name and version which appear in distribution notices and deliveries. File types may be associated with ESDTs to form a bridge between the SIPS view of a data product and the DAAC's view of that product.

A particular instance of a file type is called a file instance. An operator may query for, view information about, and download any file instance in near line storage. The operator can view such details as the time that the file was produced, the file size, the arrival method (distribution, upload, or processor-generated), the data time coverage for the file, and any related file instances such as metadata.

File coverage summaries are provided in the SIPS GUI as well. An operator can view a graphical display of time coverage for all files of a single day. Additionally, the application allows an operator to define any group of file types as a file view. Once a file view is defined, the operator can view a summary of file time coverage for that file view over an entire year.

File instances may be uploaded by the operator into SIPS directly from the user interface.

Near line storage is where files are stored locally. Currently near line storage resides on hir1 below /sips/ftp/nls. A file in NLS is stored and named according to the path template that is associated with that file's file type. Path templates are defined using the SIPS GUI.

If a product-related failure occurs during operation of the SIPS, report the problem to a SIPS contact person. If the problem is related to a particular processor run, an SCF contact person may also need to be notified.

3. DISTRIBUTIONS

SIPS obtains data from the DAAC through the ECS subscription service. A subscription is an advance request to the ECS subscription service for data of a particular ESDT. Subscriptions are sent to the DAAC in a properly formed e-mail message. When the DAAC grants the subscription request, a subscription event notification is returned via e-mail. The format of a subscription request and event notification message can be found in the ECS-SIPS ICD (423-41-57). Once a subscription has been established, regular distributions of that ESDT will be sent to the SIPS as data is archived at the DAAC.

We currently have subscriptions for the following ESDTs: AURATTH, AURATTN, AUREPHMH, AUREPHMN, AURGBAD1, AURGBAD4, AURGBAD8, D4FAPCHM, D4FAPMIS, HIR0BENG, HIR0BSCI, HIR0BSCX, HIR0ENG, HIR0SCI, HIR0SCIX, LeapSecT, SAGHGTH, SAMOISTH, SATEMPH, SAWINDSH, UTCPOleT. An up-to-date list of the distributed product types is viewable in the SIPS GUI.

The DAAC distributes data to the SIPS via secure copy (SCP) to a predetermined distribution directory on hir1: /sips/gdaac/distributions.

After the data has been sent, the DAAC sends a distribution notice via e-mail to the SIPS. A distribution notice is a notification that data has been distributed from the DAAC to the SIPS. Information in a distribution notice includes

whether or not the distribution attempt was successful, the method of file transfer (ftp pull or ftp push), and the directories and filenames of the distributed data.

SIPS automatically monitors its email inbox and detects the arrival of new distribution notices. Once a distribution notice has been successfully parsed, SIPS locally archives the data files into NLS. SIPS also updates its database with a new distribution notice object containing information about the distribution. The operator can use the SIPS GUI to view such details as the arrival time, granules, granule files, and any failures that may have occurred.

SIPS must be configured to be able to understand the various distribution notices that are sent from the DAAC. An operator must use the SIPS GUI to create distribution granule maps and file maps. These maps use a combination of ESDT information and filename regular expressions to map distributed files to SIPS file types.

The operator can use the SIPS GUI to query for and view information about any distribution notice from the DAAC. The operator can also view a completeness page that provides a summary of completed, overdue, and missing distributions.

Missing data may be reordered using the machine-to-machine gateway. Information on how to use the machine-to-machine gateway is provided on the HIRDLS SIPS Wiki.

If a failure prevents the DAAC from making a distribution, the DAAC will send a distribution notice specifying the failure. In this scenario the DAAC must be contacted to resolve the problem. If the problem is due to a firewall or host service problem, the HIRDLS system administrator will need to be contacted. If a failure occurs after a successful distribution notice arrives, notify a SIPS contact person.

4. PROCESSORS

Processors are the software executables that transform the distributed data into higher level products. Processors are provided to the SIPS operator from the SCF. To install an SCF-provided processor into SIPS, an operator must use the SIPS GUI to create an installed processor. At this time the operator specifies information such as a processor type, architecture, installation version, directory, and executable. SIPS will then copy the processor directory to a processor storage area on `hir1: /sips/processors`.

Once a processor is installed into SIPS, it must be configured. This requires an operator to use the SIPS GUI to create a configured processor. At this time an operator specifies information such as a configuration name, an optional configuration template file, arguments to the processor, whether or not the processor is production quality, and configurations for the input, output, and log files related to the processor. The template file may contain tokens for filenames and operator-definable values that are replaced by SIPS at job start time. An input file configuration represents input for a processor. To configure a processor's input, the operator must specify a file type, a template token or a fixed filename, and required time coverage information. This gives SIPS the information it needs to find a best-fit set of file instances to use as input to the processor at job start time. An output file configuration represents an output of a processor. To configure a processor's output, the operator must specify a file type and either a template token or regular expression that SIPS uses to identify the file(s) in the processor sandbox. A log file configuration identifies a log file that a processor produces. The operator may define error and warning tokens that SIPS will look for and take action on when the processor finishes its run. Multiple configurations of the same installed processor may be created. The ability to install and configure new processors at run-time results in the SIPS application being loosely coupled to the individual processors that are actually run by SIPS.

To keep track of the various processor installations and configurations, versions are required for installed processors and configuration names are required for configured processors. Installation versions will be provided by the SCF developers. Configuration names are created by the operator. Meaningful installation versions and configuration names are critical in being able to identify and run the appropriate processors for a given job run.

When a configured processor is run by SIPS, a processor instance is produced. An operator may use the SIPS GUI to query for and view information about any processor run. The operator can view such details as the data time range,

when the processor started, how long it has been running, its current state and status, who started the processor, and all of the input, log, and output file instances associated with the processor instance. For a running processor, the operator may view platform-specific process information for that run. The operator may also view the sandbox for a particular processor instance. The sandbox view shows the filenames and sizes of all files in the sandbox. The operator may view or download any sandbox file.

If a failure occurs during the installation or configuration of a processor, notify a SIPS contact person.

5. JOBS

The SIPS notion of a job is one or more configured processors connected together by files. A job may be one self-contained processor, or a combination of many processors, each with a smaller set of responsibilities.

To run one or more processors the operator must use the SIPS GUI to create a configured job. An operator specifies information such as a job type, configuration name, and description. The operator may then add configured processors to the job in the order in which they should run. As each processor is added to a configured job, the operator must decide if each input for that processor should come from NLS, the output of a previous processor, or be the same as the input of a previous processor in that job.

To keep track of the various job configurations, configuration names are required. Configuration names are created by the operator. Meaningful configuration names are critical in being able to identify and run the intended job.

The operator uses the SIPS GUI to start a job. A configured job is selected and then data start/end times are chosen. The operator may request SIPS to try to automatically find a best-fit set of input files for all of the processors of the job. A best-fit set of input files is a set of file instances in which the required time coverage is met without exceeding the maximum gap and maximum overlap times that are defined in that file type's input file configuration. If multiple file instance sets meet the criteria, the most recently produced set of file instances is chosen. If SIPS can't find a set of input files for a particular processor, then it provides a message explaining why. The operator must then modify the processor input file configuration for that input type or query for those files manually.

When the operator starts a configured job, SIPS updates its database with a new job instance containing information about this run. SIPS creates a sandbox for the first processor of the job below the sandbox root directory on hir1: /sips/ftp/sandbox. The sandbox directory name will be a combination of the processor type's name and the current time. SIPS copies the installed processor from the storage area to the sandbox. SIPS then copies the input files for the processor from NLS to the sandbox. Finally, SIPS will send a message to a Perl process manager that resides on the processing machine. The process manager will exec the processor's executable.

An operator may query for and view information about any job run. The operator may view such details as the job's data time range, when the job was started, how long it has been running for, its current state and status, who started the job, and the current processor instance. The operator may use the GUI to stop a running job.

If a job fails, it is the operator's responsibility to resolve the problem. Ideally the problem should be resolved swiftly because unused CPU time is not recoverable. Two different types of errors are reported by SIPS when it detects a job failure. If a system error occurs, the problem is likely due to the SIPS software itself. In this case a SIPS contact person should be notified. If a validation error occurs, the problem may be due to the processor itself, or the configuration of the processor. Determining the cause of the problem may involve a combination of looking at the SIPS error message, analyzing the processor's sandbox, viewing log files, and viewing SIPS configuration information. If the problem is discovered to be due to improper configuration, the operator may fix the problem and then either restart or recover the job. If the problem is due to the processor software then an SCF contact person must be notified.

6. DELIVERIES

Before a set of file instances can be delivered to the DAAC, the operator must be sure that the appropriate delivery file types are assigned to the types of files being delivered. The current delivery file types are HDF-EOS, PRODHIST,

SCIENCE, METADATA, BROWSE, and QA. Additionally, the file types must be associated with ESDTs. The current deliverable ESDTs are HIR1CAL, HIRDLS1, HIRDLS2, and HIRDLS3.

To send a delivery to the DAAC, the operator must use the SIPS GUI. First, an ESDT is chosen. Then, the files to deliver must be queried for and selected. Only files that are production quality are selectable. If multiple data files are selected, non-metadata child file instances may be sent as linkage files after the first set of files is delivered. Linkage files are handled automatically by SIPS and do not require operator intervention.

Once the operator uses the SIPS GUI to submit a delivery request, SIPS creates a subdirectory for each Product Delivery Record (PDR) below the delivery staging area on hir: /sips/gdaac/deliveries. SIPS copies the files to deliver into each subdirectory. SIPS then creates the PDRs and places them in the delivery staging area. Finally, SIPS updates its database with a delivery instance that records information about the delivery event. The operator may use the SIPS GUI to view such details as the delivery creation time, its current state, the current Product Delivery Record (PDR) instance, and who sent the delivery. The operator may query for and view information about any delivery that was sent to the DAAC. The operator can also view a completeness page that provides a summary of completed, pending, and missing deliveries.

When the DAAC receives a delivery, it responds with the e-mail of a Product Delivery Record Discrepancy (PDRD) or Production Acceptance Notification (PAN). SIPS automatically monitors its email inbox and detects the arrival of new PDRDs and PANs. If a successful PAN arrives, the delivery was successfully received by the DAAC. If a PDRD arrives, it means that the PDR could not be successfully validated. A SIPS contact person may need to be notified to resolve the problem. If a failure PAN arrives, it means that the DAAC successfully parsed the PDR but could not otherwise retrieve the data files. This may mean that the DAAC could not access the files in the delivery staging area. A DAAC contact person, the HIRDLS system administrator, or both may need to be contacted to resolve the problem.

7. NORMAL OPERATIONS

A nominal job will be configured for normal operations. The job will be started daily when all granules have been received from the DAAC or at 4:00 p.m. whichever comes first.

The nominal job will consist of the following eight configured processors.

1. L1 pre-processor

Inputs: HIR0SCI
 Outputs: HIR0SCI-GOOD
 Host: hir1
 Expected runtime: < 10 minutes

2. L1 processor

Inputs: AURATTN, AUREPHMN, HIR0SCI-GOOD, HIR1CAL
 Outputs: HIR1CAL, HIRDLS1, metadata
 Host: hir1
 Expected runtime: 3.0 +/- 0.5 hours

3. L2 pre-processor

Inputs: AURATTN, AUREPHMN, HIRDLS1
 Outputs: HIRRAD
 Host: hir1
 Expected runtime: 3.0 +/- 0.5 hours

4. L2 processor, CLP/LWP subsystem

Inputs: HIRRAD
 Outputs: HIR2CLCA, HIR2CLCC, HIR2LSGW
 Host: hal
 Expected runtime: < 15 minutes

5. L2 processor, retrieval subsystem

Inputs: HIR2CLCA, HIR2CLCC, HIR2LSGW, HIRRAD
 Outputs: HIRPROF
 Host: hal
 Expected runtime: 20 +/- 3 hours

6. L2 processor, build output subsystem

Inputs: HIRPROF
 Outputs: HIRDLS2, metadata
 Host: hal
 Expected runtime: < 15 minutes

7. Browse processor

Inputs: HIRDLS2
 Outputs: HIR2BRWSE
 Host: hir1
 Expected runtime: < 10 minutes

8. QA processor

Inputs: HIRDLS2
 Outputs: HIR2QA
 Host: hir1
 Expected runtime: < 10 minutes

8. OFF-NOMINAL OPERATIONS

There will be times when recovery from a data outage or hardware downtime will be required. If SIPS hardware is down for two days, then SIPS must be turned back on and allowed to ingest the distribution notices that were sent during the down time. The job completeness page, product completeness page, and/or the product day view may be used to determine what jobs can be run. The latest job that can be started should be started first on hal. The latest job is always the top priority unless otherwise directed. Catch-up jobs can be started next on hir1 using up to two sets of eight processors.

If hir1 is down for two days, the subscriptions may have to be restarted at the DAAC. This would involve communication with the DAAC to get the subscriptions turned on again. Additionally, missing data would have to be re-ordered. The product completeness and/or product day view pages will show where data is missing. The machine-to-machine gateway would be used to order the missing data. Once missing data has been ingested by SIPS, the job completeness page, product completeness page, and/or the product day view may be used to determine what jobs can be run. The latest job that can be started should be started first on hal. Catch-up jobs can be started next on hir1 using up to two sets of eight processors.

If hal is down for two days, normal processing of jobs would proceed using only hir1.

Special testing runs may be required to be run while normal production runs are occurring. A production run occurs when all of the configured processors of the job are marked as production quality. Additionally the job run itself must be set to be a production run. To perform a test run, the job must be run in test mode. If one or more of the configured

processors in the job are not production quality, then the job is automatically in test mode. If all of the configured processors in the job are production quality, the operator is still allowed to run the job in test mode. All of the output that is generated by a job in test mode is marked as being test quality. The SIPS GUI prevents test quality files from being delivered to the DAAC.

Some special tests may require data to be uploaded directly into SIPS using the GUI. To upload a file into SIPS, select the appropriate file type on the file upload page. Enter a path for the file as well as a comment to help identify the file after the file is ingested. Choose a time source option for determining the start/end times of the data in the file. Enter any remaining times that must be specified manually.

9. POINTS OF CONTACT

Here are the people that can be contacted in the event of a failure.

SIPS:	Tom Lauren (lauren@ucar.edu) or Greg Young (gyoung@ucar.edu)
L1 pre-processor:	Vince Dean (vdean@ucar.edu)
L1 processor:	Soji Oduleye (oduleye@atm.ox.ac.uk) or Charles Cavanaugh (cavanaugh@ucar.edu)
L2 pre-processor:	Brent Peterson (bpeterse@ucar.edu) or Cheryl Craig (cacraig@ucar.edu)
L2 processor:	Cheryl Craig (cacraig@ucar.edu) or Alyn Lambert (alambert@ucar.edu)
DAAC and SCF interfaces:	Vince Dean (vdean@ucar.edu)
HIRDLS system administrator:	Dan Packman (pack@ucar.edu)