Software Requirements Specification

for

Offline MIPS-24μm Pipelines

Prepared by Frank Masci
(fmasci@ipac.caltech.edu)

Version 4.5, June 20, 2004

California Institute of Technology
SIRTF Science Center

National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

Deleted: 3

1. Revision History

Version	Description	Date
1.0	Initial version	June 20, 2002
2.0	• Modified pipeline flow diagrams in section 4 (replaced snestimator with sloperror in SUR pipelines, put flattrack in termination brach (no flow). Included caltrans in ensemble (cal) pipelines. • Tutorial examples in section 8 updated to reflect new –k <input_keyword_list> command-line parameter. • Section 9 environment variable SOS_REL changed to S6.3D. • Included new section on "required pipeline keywords" • Updated section 11 on pipeline input data constraints. • Updated section 13.3.2 (Non-linearity C-mask). • Sloperror tutorial and namelist parameters given. • Some namelist parameters updated. Included two new namelists: mips24_copy_to_sandbox.nl and w_bqd_files_to_copy_to_sandbox.nl.	July 22, 2002
2.1	• SURSIMSLOPE module removed from RAW-Science pipeline and replaced by IMAGEST which uses a more robust algorithm to estimate slope images from ramp data. A bmask is made in the output which is analygous to the dmask in SUR-mode processing. Namelists were updated. • RDOUTMOD keyword removed from required FITS keyword list (section 11). • B-Mask (output by imagest) bit definitions for RAW-science thread given in section 15.4	August 5, 2002
2.2	• Included flexibility of using "DCE dependent" namelist and calibration_data directories when executing pipelines on a list of images (section 10). • Included MIPL keyword "GROUPS" to required FITS keyword list (section 11) • Updated/replaced namelist parameters in section 17.	October 1, 2002
2.3	• B-Mask bit definitions for SUR-science thread given in section 15.3	October 18, 2002
2.4	• Updated installation instructions under section 7.0. Now a generic tar-ball exists which contains all mips24 dependencies.	November 13, 2002

Deleted: 3

3.0	• Included latent-image detection/flagging module to end of SUR-science pipeline. This can be run optionally using a command-line switch. • Removed old sections 16 and 17 on module tutorials and namelists since these change continuously. The user already has access to this information.	December 17, 2002
3.1	• Added two more output products to SUR-mode science thread: an uncertainty and mask image corresponding to the "slope-pixel" replacement image. This is now the "BCD" image. Modified pipeline flow diagram in section 4.1 to show this. • Updated the d-mask/b-mask definitions for SUR mode to account for this. • Included new section 5.5 which summarizes the pointing-transfer/final product generator step performed by automated SSC-pipelines only.	February 2, 2003
3.7 3.8	• Replaced "radhitmedfilt" module in SUR-mode science pipeline with "medfilter" and "detect_radhit" modules. • Modified automated output tutorial screen for RUN_SURSCI.pl	March 17, 2003 April 24, 2003
4.0	• Included pointing-transfer and Final Product Generation (FPG) in SUR-mode science pipeline.	May 13, 2003
4.1	• Made a new calibration file which contains the DN-to-flux conversion factors and uncertainties called mips24_fluxconv.txt	May 21, 2003
4.2	• Added ability to process "post-tranhead" FITS files directly (tranhead module output) to all pipeline scripts.	June 9, 2003
4.3 4.4	• Updated to include imfliprot, rowfluxcorr modules • Remove sanity check module from all pipelines • Run FPG on calibration products • Remove flattrack module from flatfield module • Included "-P" option to re-run pointing part of "SURSCI" pipeline on existing bcd products. • Updated RUN_SURSCI.pl tutorial section • Added code to ski DCENUM=0 dces in non-linearity and flatfield calibration pipelines. • Updated RAWSCI DCE mask (imagest radhit flagging) and SURSCI BCD mask descriptions (rowfluxcorr). Added documention in section 12 on a script "MakeMirrorDepFlats.pl"	February 2, 2004 February 11, 2004

Deleted: 3

	which automatically generates mirror-dependent flat -field products for use in SSC science pipelines.	
4.5	Updated d-mask and b-mask bit definitions for SUR-mode BCDs in section 14. Bit 13 is now exclusively for soft saturation, bit 2 is exclusively for hard saturation. Bit 4 is for pixel replacement in the occurrence of either bit 13 or bit 2 being set.	June 20, 2004

Deleted: 3

2. Table of Contents

Deleted: 3

3. Introduction

3.1 Purpose and Scope

This document describes the procedure for installing, running and testing the mips-24µm offline BCD pipelines. By offline, we mean a design which does not interact with the Science Operations Database (SODB) and in which data products are not stored using the same archive structure to be use in operations at the SSC (see section 5 for more details). It is purely for analysis, science validation, fine tuning of the on-line automated system and for generating "fallback" calibration products required by the on-line system.

3.2 Summary of MIPS-24 Offline Pipeline Threads

In all, there are six pipeline threads (hereafter referred to as threads). There are two science threads designed to process data taken in the SUR and RAW acquisition modes. There are four calibration pipelines which produce calibration products required by the two science threads (1 and 2 below).

1. SUR-MODE SCIENCE (*includes latent-image detection/flagging and pointing-transfer/fpg options*).
2. RAW-MODE SCIENCE
3. SUR-MODE DARK-CURRENT CALIBRATION
4. RAW-MODE DARK-CURRENT CALIBRATION
5. DETECTOR NON-LINEARITY CALIBRATION
6. FPA NON-UNIFORMITY (FLAT-FIELD) CALIBRATION

4. Pipeline Thread Flowcharts

4.1 SUR-Mode Science Thread

```
INPUT SUR-Mode 2-plane Science DCE
 ↓
SANITY_DATATYPE – FITS header keywords sanity check
 ↓               - Instrument/Channel/Mode/Data Missing
QATOOL_DCE – Quality assurance characterization/Statistics on DCE
 ↓
TRANHEAD – Translation of FOS keywords, derivation of exposure-time
 ↓          time keywords, insertion of required keywords
```

Deleted: 3

IMPLIPROT – Optionally flip and rotate post-tranhead image
- In on-line pipeline, write mirror position/rate keywords to header from database

CALTRANS – Acquire/transfer desired calibration data from database

CVTI2R4 – Conversion from I*2 to R*4 & 0.5 DN truncation correction
- Creates D-Mask and replaces missing data with NaNs

SATMASK – Saturated-pixel detection: Output to D-Mask

CVTDNSEC – Conversion from DN/sampling-time to units of DN/seconds

SLOPERROR – Estimate noise in slope image

ROWFLUXCORR – Row-flux or “read-2” correction

DESATSLOPE – Slope desaturation for robust droop correction

DROOPOP – Droop correction

ROWDROOP – Row-droop (mux-bleed) correction

CUBESUB_SUR – Dark-current subtraction from slope image

SLOPECORR – Non-linearity correction (both slope and difference data)

FLATAP – Flat-field, non-uniformity correction

DNTOFLUX – Conversion from DN/sec to μJy/arcsec^2

REPLACEPIXELS – Replace saturated (slope) pixels with pixels from difference image and output to new single plane image (called bcd.fits). Also make corresponding replacements in uncertainty and mask images called bcd_uncert.fits and bcd_mask.fits respectively.

SPLITSURIMAGE – Split two-plane science, uncertainty and d-mask images into single plane slope and difference counterparts. The single plane d-masks are renamed bmask_slope.fits and bmask_diff.fits (i.e. “bcd-mask”).

MEDFILTER – Compute (median) background subtracted image from bcd.fits for input into “detect_radhit”.

DETECT_RADHIT – Single frame rad-hit detection using median filter input, bcd_mask.fits is updated.

QATOOL_DP - Quality assurance characterization/Statistics on BCD

Deleted: 3

CHECKMASK_DP – Statistics on B-Mask of slope image

QALOADER_DP – Load output from QATOOL_DP into database

OUTPUT BCD (Basic Calibrated Data for archive) FITS images: slope, difference, replaced (main bcd.fits), and corresponding uncertainty and B-Mask images for each. Become input for latent-image detection/flagging step once ensemble of bcd.fits images are available.

ENSEMBLE (LATENT) PROCESSING INPUTS – List of pre-processed BCD's from above for input into latent-image detection/flagging thread.

LATIMDETECT – Perform latent detection for each "target" image (using its corresponding uncertainty image for thresholding) in ensemble given previous "history" images with fixed look-back-time.
- Update relevant mask-bit in corresponding bcd_mask.fits image for pixels in target image where a latent was found.

POINTING-TRANSFER/FPG INPUTS –Input single processed bcd.fits and all associated ancillary products listed in "**OUTPUT BCD**" step above.

GetPH-OFFLINE – Input Boresight Pointing History File(s) (BPHF) and return 2-Hz separated pointing samples over exposure time SCLK range for input BCD (main bcd.fits). Output saved in table file: "ptghistory.dat"

MIRRORSYNCH – Compute/synchronize scan-mirror angle at each boresight pointing sample time. Output to new table: "mirrorsynch.dat"

BORESIGHTTRAN – Transform each boresight pointing sample in mirrorsynch.dat to desired aperture FOV in array using: apertureNames.tbl namelist (to get the desired aperture name) and instrument_FOV.tbl namelist (to select correct boresight-FOV Euler angle offsets). Output saved in "fov.tbl".

ANGLEAVG – Compute average RA, Dec, Twist angle in desired FOV and write to header of bcd.fits.
- Compute average input mirror angles over time-span of BCD and use distortion/pixel scale calibration file to linearly interpolate CDELTS/distortion coefficients and write to bcd.fits header.

FPG – For MIPS-24, copy all pointing keywords from bcd.fits header to all ancillary product headers listed in "**OUTPUT BCD**" step above.
- Execute the final product generator module on bcd.fits and associated ancillary products to produce neatly formatted headers. Output products are named in the following format: *"<inputFITSimage>_fp.fits"*

Deleted: 3

4.2 RAW-Mode Science Thread

INPUT RAW-Mode (Multi-plane) Science DCE

SANITY_DATATYPE – FITS header keywords sanity check

- Instrument/Channel/Mode/Data Missing

QATOOL_DCE – Quality assurance characterization/Statistics on DCE

TRANHEAD – Translation of FOS keywords, derivation of exposure-time time keywords, insertion of required keywords

IMPLIPROT – Optionally flip and rotate post-tranhead image
- In on-line pipeline, write mirror position/rate keywords to header from database

CALTRANS – Acquire/transfer desired calibration data from database.

CVTI2R4 – Conversion from I*2 to R*4
- Creates D-Mask and replaces missing data with NaNs
- Flags pixels which are hard saturated after A-to-D
- Add DC baseline offset (32768 DN) to all desired planes

SNESTIMATOR – Estimate noise image

BASECAL – Correct all planes for non-zero baseline at t=0 intercept

IMAGEST – Ramp de-saturation for robust droop correction

DROOPOP – Droop correction

ROWDROOP – Row-droop (mux-bleed) correction

CUBESUB_RAW – Dark-current subtraction

LINEARIZ – Non-linearity correction

RADHIT – Multi-frame rad-hit detection

FLATAP – Flat-field, non-uniformity correction

IMAGEST_SLOPE_EST – Fit slope to RAW-mode ramp data (output image units: DN/sec). Produces single plane B-Mask

DNTOFLUX – Conversion from DN/sec to µJy/arcsec^2

QATOOL_DP - Quality assurance characterization/Statistics on BCD

CHECKMASK_DP – Statistics on B-Mask

QALOADER_DP – Load output from QATOOL_DP into database

Deleted: 3

OUTPUT BCD (Basic Calibrated Data) FITS images (slope, uncertainty and B-Mask images) ready for archiving.

POINTING-TRANSFER/FPG INPUTS –Input single processed bcd.fits and all associated ancillary products listed in "**OUTPUT BCD**" step above.
-**NOTE: This thread can only be run on post-RAW-mode science products in the automated on-line system.**

GetPH-OFFLINE – Input Boresight Pointing History File(s) (BPHF) and return 2-Hz separated pointing samples over exposure time SCLK range for input BCD (main bcd.fits). Output saved in table file: "ptghistory.dat"

MIRRORSYNCH – Compute/synchronize scan-mirror angle at each boresight pointing sample time. Output to new table: "mirrorsynch.dat"

BORESIGHTTRAN – Transform each boresight pointing sample in mirrorsynch.dat to desired aperture FOV in array using: apertureNames.tbl namelist (to get the desired aperture name) and instrument_FOV.tbl namelist (to select correct boresight-FOV Euler angle offsets). Output saved in "fov.tbl".

ANGLEAVG – Compute average RA, Dec, Twist angle in desired FOV and write to header of bcd.fits.
- Compute average input mirror angles over time-span of BCD and use distortion/pixel scale calibration file to linearly interpolate CDELTS/distortion coefficients and write to bcd.fits header.

FPG – For MIPS-24, copy all pointing keywords from bcd.fits header to all ancillary product headers listed in "**OUTPUT BCD**" step above.
- Execute the final product generator module on bcd.fits and associated ancillary products to produce neatly formatted headers. Output products are named in the following format: *"<inputFITSimage>_fp.fits"*

4.3 SUR-Mode Dark-Current Calibration Thread

PRE-PROCESSING INPUT SUR-Mode (2-plane) dark-current calibration image for single image processing

SANITY_DATATYPE – FITS header keywords sanity check
- Instrument/Channel/Mode/Data Missing

QATOOL_DCE – Quality assurance characterization/Statistics on DCE

TRANHEAD – Translation of FOS keywords, derivation of exposure-time time keywords, insertion of required keywords

IMPLIPROT – Optionally flip and rotate post-tranhead image

Deleted: 3

- In on-line pipeline, write mirror position/rate keywords to header from database

CALTRANS – Acquire/transfer desired calibration data from database

CVTI2R4 – Conversion from I*2 to R*4 & 0.5 DN truncation correction
- Creates D-Mask and replaces missing data with NaNs

SATMASK – Saturated-pixel detection: Output to D-Mask

CVTDNSEC – Conversion from DN/sampling-time to units of DN/seconds

SLOPERROR – Estimate noise in slope image

ROWFLUXCORR – Row-flux or "read-2" correction

QATOOL_CAL - Quality assurance on single pre-processed dark image

CHECKMASK_CAL – Statistics on D-Mask

QALOADER_DP – Load output from QATOOL_CAL into database

ENSEMBLE PROCESSING INPUT –List of pre-processed darks from above

CALTRANS – Acquire/transfer desired calibration data from database

DARKEST_SUR – Dark current estimation (ensemble processing of a list)

CALKEYWORDS – Compute average of input keyword values from all images and write to product header. Also store values in database

SPLITDARKSURIMAGE – Remove redundant second plane from product

QATOOL_DP - Quality assurance characterization on dark product

CHECKMASK_DP – Statistics on C-Mask of dark product

QALOADER_EP – Load output from QATOOL_DP into database

FPG - Execute the final product generator module on product and associated ancillary products to produce neatly formatted headers.

OUTPUT PRODUCT – "Canonical" SUR-mode dark current image to be Stored in calibration archive

4.4 RAW-Mode Dark-Current Calibration Thread

PRE-PROCESSING INPUT RAW-Mode (multi-plane) dark-current

Deleted: 3

calibration image for single image processing

SANITY_DATATYPE – FITS header keywords sanity check
- Instrument/Channel/Mode/Data Missing

QATOOL_DCE – Quality assurance characterization/Statistics on DCE

TRANHEAD – Translation of FOS keywords, derivation of exposure-time time keywords, insertion of required keywords

IMPLIPROT – Optionally flip and rotate post-tranhead image
- In on-line pipeline, write mirror position/rate keywords to header from database

CALTRANS – Acquire/transfer desired calibration data from database

CVTI2R4 – Conversion from I*2 to R*4
- Creates D-Mask and replaces missing data with NaNs
- Flags pixels which are hard saturated after A-to-D
- Add DC baseline offset (32768 DN) to all desired planes

SNESTIMATOR – Estimate noise image

BASECAL – Correct all planes for non-zero baseline at t=0 intercept

QATOOL_CAL - Quality assurance on single pre-processed dark image

CHECKMASK_CAL – Statistics on D-Mask

QALOADER_DP – Load output from QATOOL_CAL into database

ENSEMBLE PROCESSING INPUT – List of pre-processed darks from above

CALTRANS – Acquire/transfer desired calibration data from database

DARKEST_RAW – Dark current estimation (ensemble processing of a list)

CALKEYWORDS – Compute average of input keyword values from all images and write to product header. Also store values in database

QATOOL_DP - Quality assurance characterization on dark product

CHECKMASK_DP – Statistics on C-Mask of dark product

QALOADER_EP – Load output from QATOOL_DP into database

FPG - Execute the final product generator module on product and associated ancillary products to produce neatly formatted headers.

OUTPUT PRODUCT – “Canonical” RAW-mode dark current image to be Stored in calibration archive

Deleted: 3

4.5 Non-Linearity Calibration Thread

PRE-PROCESSING INPUT RAW-Mode (20-plane) calibration image for single image processing

SANITY_DATATYPE – FITS header keywords sanity check
- Instrument/Channel/Mode/Data Missing

QATOOL_DCE – Quality assurance characterization/Statistics on DCE

TRANHEAD – Translation of FOS keywords, derivation of exposure-time time keywords, insertion of required keywords

IMPLIPROT – Optionally flip and rotate post-tranhead image
- In on-line pipeline, write mirror position/rate keywords to header from database

CALTRANS – Acquire/transfer desired calibration data from database

CVTI2R4 – Conversion from I*2 to R*4
- Creates D-Mask and replaces missing data with NaNs
- Flags pixels which are hard saturated after A-to-D
- Add DC baseline offset (32768 DN) to all desired planes

SNESTIMATOR – Estimate noise image

BASECAL – Correct all planes for non-zero baseline at t=0 intercept

DROOPOP – Droop correction

ROWDROOP – Row-droop (mux-bleed) correction

CUBESUB_RAW – Dark-current subtraction

QATOOL_CAL - Quality assurance on single pre-processed image

CHECKMASK_CAL – Statistics on D-Mask

QALOADER_DP – Load output from QATOOL_CAL into database

ENSEMBLE PROCESSING INPUT – List of pre-processed cubes from above

CALTRANS – Acquire/transfer desired calibration data from database

LINCAL – Non-linearity model estimation(ensemble processing of a list)

CALKEYWORDS – Compute average of input keyword values from all images and write to product header. Also store values in database

Deleted: 3

QATOOL_DP - Quality assurance on non-linearity product

CHECKMASK_DP – Statistics on C-Mask of product

QALOADER_EP – Load output from QATOOL_DP into database

FPG - Execute the final product generator module on product and associated ancillary products to produce neatly formatted headers.

OUTPUT PRODUCT – Non-linearity model (quadratic) coefficient image to be stored in calibration archive

4.6 Flat-Field Calibration Thread

PRE-PROCESSING INPUT SUR-Mode (2-plane) stimulator calibration image for single image processing

SANITY_DATATYPE – FITS header keywords sanity check

- Instrument/Channel/Mode/Data Missing

QATOOL_DCE – Quality assurance characterization/Statistics on DCE

TRANHEAD – Translation of FOS keywords, derivation of exposure-time time keywords, insertion of required keywords

IMPLIPROT – Optionally flip and rotate post-tranhead image
- In on-line pipeline, write mirror position/rate keywords to header from database

CALTRANS – Acquire/transfer desired calibration data from database

CVTI2R4 – Conversion from I*2 to R*4 & 0.5 DN truncation correction
- Creates D-Mask and replaces missing data with NaNs

SATMASK – Saturated-pixel detection: Output to D-Mask

CVTDNSEC – Conversion from DN/sampling-time to units of DN/seconds

SLOPERROR – Estimate noise in slope image

ROWFLUXCORR – Row-flux or "read-2" correction

DESATSLOPE – Slope desaturation for robust droop correction

DROOPOP – Droop correction

ROWDROOP – Row-droop (mux-bleed) correction

CUBESUB_SUR – Dark-current subtraction from slope image

SLOPECORR – Non-linearity correction (both slope and difference data)

Deleted: 3

QATOOL_CAL - Quality assurance on single pre-processed stim image

CHECKMASK_CAL – Statistics on D-Mask

QALOADER_DP – Load output from QATOOL_CAL into database

ENSEMBLE PROCESSING INPUT –List of pre-processed stims from above

CALTRANS – Acquire/transfer desired calibration data from database

FLATFIELD – Flatfield estimation (ensemble processing of a list)

CALKEYWORDS – Compute average of input keyword values from all images and write to product header. Also store values in database

FLATTRACK – Refinement of flatfield product using iterative Kalman filtering

QATOOL_DP - Quality assurance characterization on flatfield product

CHECKMASK_DP – Statistics on C-Mask of flatfield image

QALOADER_EP – Load output from QATOOL_DP into database

FPG - Execute the final product generator module on product and associated ancillary products to produce neatly formatted headers.

OUTPUT PRODUCT – "Canonical" flat-field image to be stored in calibration archive

5. Differences Between Offline and Online (SSC Pipeline) Versions

5.1 Database Interaction

The automated SSC pipelines rely heavily on querying uplink parameters from the database and processing is done on an AOR or IER basis. Processing is performed under the Automated Pipeline Executive for SIRTF (APES) where incoming DCEs are processed sequentially in the order they are received from FOS. The SODB stores ancillary and commanded information for scheduled DCEs such as the exposure-type (photometry or scan science observation, calibration DCE), pointing, and identification tags: AOR number, DCE-ID etc... This information is used by APES to assign the appropriate pipeline with which the DCE should be processed with and to make unique archive directory structures for each processed DCE. APES also ensures that the pipelines are executed in the correct order. For instance, all calibration DCEs are processed first whose products are used in the science pipelines.

Deleted: 3

The offline pipelines run as isolated systems and are completely independent of any database. All information is acquired as input by the user from namelists and the command line for each pipeline module.

5.2 Calibration Server

In SSC operations, calibration data to be used in a particular science pipeline (or different calibration pipeline) is acquired by querying the database using the CALTRANS module (which stands for calibration transfer). CALTRANS is run with a specific set of rule s. An example of a rule may be where one desires to use calibration data acquired immediately after or before a science observation. The location of the calibration product in the archive, calibration type, acquisition times and other parameters are read fom the database and used to transfer the product to the local processing directory each time a call is made from within a pipeline.

The offline pipelines do not use the CALTRANS module. It is shown in the above flowcharts for completeness. Instead, a static set of calibration files are acquired from one single directory. If one wishes to use a different calibration file produced by one of the pipelines, then it will have to be manually copied into this directory with the correct filename format (see section 13 for more details).

When applying mirrpor-position/rate dependent flats to science data, the SSC pipelines query the database for the mirror parameters, write them to the science headers and then a CALTRANS query returns the appropriate flat to use. The offline pipelines assume that input science FITS image headers already contain these mirror-specific keywords (typically in the post-tranhead images). Matching between these mirror parameters is then performed against values in the calibration flat headers.

5.3 Quality Assurance (QA) Tables

Quality assurance characterization data is performed by computing image statistics via the QATOOL module. Each of the science pipelines executes QATOOL twice (on the raw DCE and later on the BCD product). Each calibration pipeline executes QATOOL three times (the raw DCE, the pre-processed DCE and the post-(ensemble)-processed calibration product. The SSC pipelines register the output statistics/text files into database tables for efficient querying and QA analysis later.

The offline pipelines do not do any *database* registering of QA information. Image statistics are output directly to log-files to be examined by the user.

5.4 Data Archiving

SSC pipeline operations use an elaborate archive structure for both processing and storage of data. Each BCD product is archived (in a "sandbox") using a unique directory structure, assembled from DCE identifiers in the SODB. These identifiers include INSTRUMENT NAME, CHANNEL# (band), AORKEY, DCE-ID, EXP-ID (Exposure ID), DCE-NUMBER etc...

The offline pipelines automatically make a directory for each DCE processed from an input list where the directory name contains the DCE filename. This directory contains all intermediate products of the pipeline. A final directory (named by the user) contains the final BCD products of all DCEs processed from a list. See section 12 for more details.

5.5 Pointing Transfer and Final Product Generation

Both the offline and online SUR-mode science pipelines schedule output BCDs for the Pointing-Transfer/Final Product Generator (PTG/FPG) pipeline. The pointing history (a 2-Hz sampled range spanning the DCE's integration) is transferred and averaged from a Boresight-Pointing History File (BPHF) to the appropriate instrument FOV. Thereafter, the pointing samples are averaged on the array to compute RA, Dec, Twist for the desired FOV and the relevant keywords specifying these values attached to the FITS headers of BCD products. The BCD FITS image then goes through the final product generator (FPG) which re-formats headers to make them more esthetic and readable.

The major deficiencies in the offline pointing transfer thread compared to the online version are as follows:

- The offline pipelines can only (optionally) execute the pointing transfer/FPG thread at the end of the SUR-mode science pipeline. The online version integrates pointing transfer in both the SUR-mode and RAW-mode science pipelines (as well as the pipe0 or "zip" science pipelines).
- The offline pipelines require the 2-Hz sampled pointing history spanning a DCE's integration to be read from BPHFs specified on the command-line. In the online system, this is retrieved using a pointing server connected to the database.
- The *mirrorsynch* module in the offline system requires all input scan-mirror parameters to be specified in a namelist. Since these are dependent on the exposure number (defined by the EXPID keyword value), the offline system will search for namelists in the format *mirrorsynch_offline_EXPIDval.nl* in the namelists directory when processing an input DCE list spanning multiple EXPIDs. It will default to using *mirrorsynch_offline.pl* if no file with the appropriate "*_EXPIDval*" pre-pended before the ".nl" is found. In the online system, all scan-mirror parameters are retrieved from a call to the database. Also, another minor difference is that the offline system does not attempt to read the *scanpos1* mirror parameter from the raw FITS header as in the online system.
- The offline system knows nothing about requested pointing since this is located in the database. The online system propagates all requested pointing information to the output FITS header following the *angleavg* program.
- Offline, the FPG cannot write additional identification/mode parameters to the output FITS header since these can only be retrieved from the database.

In the offline pipelines, if one wishes to re-run pointing-transfer/FPG again on pre-existing science products, the user can specify the "-p" option on the RUN_SURSCI.pl command line (see tutorial in section 8).

6. Operating System and Hardware Requirements

The perl scripts and modules called from within must be executed under a UNIX environment running Solaris 5.8. The software is not compatible with earlier versions of Solaris since the C, C++ and FORTRAN binaries are built using FORTE compilers optimized for SUN operating systems 5.8 and higher. Linux is not yet supported. Any platform which supports SunOS 5.8 or higher can be used: Sparc, Ultrasparc, Ultra or Sunblade. At minimum, a 500MHz CPU with 512MB memory is suggested. The pipeline software itself requires at most ≈130 MB of free disk space.

7. Installation

7.1 Location of Required Files and Scripts

On any SSC machine mounted on the /stage/ssc-pipe partition, the latest delivery (a tar-ball of all binaries, libraries, and dependencies) is contained under:

/stage/ssc-pipe/fmasci/MIPS24_pipeline/offline_pl/S*.*_delivery

The g-zipped tar file is called **S*.*_DL_Offline_MIPS24_vsn*.tar.gz ,**

where **S*.*** designates the segment delivery version ID and vsn* a sub-version ID. Directories/files containing the largest of these numbers refer to the latest software.

If you do not have access to the above directory, please email: fmasci@ipac.caltech.edu to request a copy of the tar file.

7.2 Set-up

Here are the instructions for installing the software on your system.

1. *cd* to a place which has >130 MB
2. *mkdir* MIPS24_pipeline (or what ever name you like).
3. *cd* MIPS24_pipeline/ (or what ever you made in 2.)
4. *cp* **S*.*_DL_Offline_MIPS24_vsn*.tar.gz** (from its location) to the directory in 3.
5. *gunzip* **S*.*_DL_Offline_MIPS24_vsn*.tar.gz**
6. *tar –xvf* **S*.*_DL_Offline_MIPS24_vsn*.tar**

The following six directories are extracted: /mips24, /bin, /lib, /include, /cdf, /informix_v93

7. *rm –rf* **S*.*_DL_Offline_MIPS24_vsn*.tar** (to clean up)

8. *cd* mips24/

9. *source* regular_env.csh

10. And you're ready to go. The default settings in regular_env.csh file are such that you can only execute the pipelines from within the mips24/ sub-directory. The header of the regular_env.csh file outlines the procedure if you wish to execute the pipelines from elsewhere. You're now ready to generate the tutorials (section 8) or run the examples (section 17).

7.3 Contents of the /mips24 Directory

- MIPS24_pipelines.doc and MIPS24_pipelines.pdf (this document)
- calibration_data/ (directory containing calibration data/images)
- namelists/ (directory containing pipeline control parameters)
- test_imgs/ (directory containing example FITS files for testing)
- regular_env.csh (environment variables for running on SSC machines)
- libmips24offline.pl (perl library required by pipeline scripts)
- ptg_fpg_offline_lib.pl (perl library for running pointing transfer/FPG)
- RUN_FLTCAL.pl (runs flat-field calibration pipeline)
- RUN_LINCAL.pl (runs non-linearity calibration pipeline)
- RUN_RAWDRK.pl (runs RAW-mode dark calibration pipeline)
- RUN_SURDRK.pl (runs SUR-mode dark calibration pipeline)
- RUN_SURSCI.pl (runs SUR-mode science pipeline)
- RUN_RAWSCI.pl (runs RAW-mode science pipeline)
- MakeMirrorDepFlats.pl (Makes scan-mirror position/rate dependent FITS file lists and optionally executes RUN_FLTCAL.pl on each).

Deleted: 3

8. Generating Tutorials

Ensure you have the environment variable PERL_PATH set to the directory containing the program "perl". This can be found by typing at the unix prompt "*which perl*". For a description on how to execute any one of the pipelines, execute the relevant perl script with no arguments on the command-line. An example is shown below.

Typing "*RUN_SURSCI.pl –h*" will generate the following on your screen:

```
MIPS-24 SUR-Mode Science Pipeline
Frank Masci (fmasci@ipac.caltech.edu)
Copyright (C) 2001 California Institute of Technology.
Last modified 06-09-2003

  Description:

  o Executes the SUR-mode pipeline thread on SIRTF MIPS images

  o Ensure you have the environment variable PERL_PATH set to the
    directory containing the program "perl".

  o Command-line options:
    -i <inp_list>     Required: List of input FITS images listed one per line.
                                 Full paths may be included. Images may be in
                                 raw-MIPL (flight data) or "tranhead" format.
    -d <out_dir>      Optional: Output directory containing final products.
    -k <keywords_file>Optional: For non-MIPL (or non-flight) data: list of
                                 keyword/value pairs to write to input
                                 FITS headers. Listed one per line.
    -v                Optional: Verbose switch - prints more verbose output.
    -l                Optional: Latent-image flagging switch: performed at
                                 end of pipeline on all final BCD products.
                                 Need at least two input images to trigger.
    -a <BPHF_1>       Optional: Perform pointing transfer on BCD products
                                 using Boresight Pointing History File #1.
    -b <BPHF_2>       Optional: Boresight Pointing History File #2. Needed if
                                 input DCE SCLK range falls on BPHF boundary.
    -h                Optional: Help switch - prints this description.
    -p                Optional: Run only pointing transfer if BCD products
                                 already exist from a revious run.

  o The input images (listed in the input list) need not reside in the
    execution directory.

  o Required environment variables (see regular_env.csh for example)
    - Pathname of executables:                      SIRTF_BIN
    - Pathname of ancillary data:                   SIRTF_ANC
    - Pathname of cspice kernals for tranhead:      CSPICE_LIBRARY_KERNELS
    - Pathname of input namelists:                  SIRTF_CDF
    - Pathname of input calibration data:           SIRTF_CAL
```

```
     - Pathname of current telemetry dictionary (CTD):   SIRTF_DAT
     - Pathname containing library "libmips24offline.pl" PERL_LIB

   o Outputs:
     - Directories containing intermediate products for each
       processed DCE named: "products_<input_fits_image_name>" (always
produced)
     - Optionally: a directory named <out_dir> specified by the -d flag
       containing final product BCD images.

   o Examples:
     - Ensure you have "sourced" your environment variable file.

     - For non-MIPL (or non-flight) data, update your "keywords_file".
       e.g. see the file $SIRTF_CDF/insert_keywords

     - Using an input list of FITS images named "inputlist" and
       all final products copied to directory "bcd_dir":

       RUN_SURSCI.pl -i inputlist -d bcd_dir -v

     - With latent-image flagging, pointing transfer and final product
       generation (header formatting) performed on all products (pointing
       transfer can only be performed if all input files are in raw
       MIPL-format, not post-tranhead):

       RUN_SURSCI.pl -i inputlist -d bcd_dir -l -a BPHF.ptg -v

     - Only re-run pointing transfer on an already existing set of BCD
products:

       RUN_SURSCI.pl -i inputlist -d bcd_dir -a BPHF.ptg -v -p
```

9. Running with DCE Dependent Calibration/Namelist Files

If one desires to use different calibration files and/or namelist parameter files **for each or any specific DCE(s)** in the input list, the user can make and populate DCE dependent calibration data and/or namelist directories in the *execution* directory in the format: “calibration_data***i***/ ” and “namelists***i***/ ”, where ***i*** = 1, 2, 3… designates the DCE order in the input list from the top.

The user can make calibration and namelist directories corresponding to any selected number of DCEs in the input list. For those DCEs where no DCE-dependent calibration or namelist directory exists, the calibration data and namelist files will default to those in the directories specified by the `SIRTF_CAL and SIRTF_CDF` environment variables in the …*/mips24/regular_env.csh* file.

It is important to note that the naming convention: “calibration_data***i***/ and namelists***i***/” be adhered to and that these be created in the directory where pipelines are executed.

The above capability also exists for the *pre-processing steps* of all ensemble (calibration) pipelines. In the *ensemble processing step* of these pipelines, only calibration data and namelist files in directories specified by the `SIRTF_CAL and SIRTF_CDF` environment variables respectively are used. Both the *latent detection and pointing-transfer/FPG* steps at the end of the SUR-mode science pipeline will also use calibration data/namelists in directories specified by these environment variables.

10. Required FITS Keywords

Each pipeline requires a specific set of keywords in the FITS headers of every input image for successful execution. The following is the list of required keywords (taken from the example file "insert_keywords" in the namelists/ directory of the installation):

Inserted by *Multi-mission Image Processing Lab* (MIPL) from uplink parameters:

AORKEY =	000000	/ DUMMY! AOR or IER key
EXPID =	0	/ DUMMY! Exposure ID
FILE_VER=	1	/ DUMMY! File version made by SIS
INSTRUME=	'MIPS'	/ DUMMY! Instrument id
CHNLNUM =	1	/ DUMMY! channel ID (always 1 for 24μm band)
DCENUM =	1	/ DUMMY! DCE number in sequence (starts at 0)
GROUPS =	9	/ DUMMY! Number of groups per telemetry channel *(9 for 10 mips-second exposure, 29 for 30 mips-second exposure).*

Inserted by *Flight Operations System* (FOS) from telemetry:

I1061D00=	84	/ DUMMY! DCE_FRMS
I1071D00=	4	/ DUMMY! FRMFLYBK
I1061E00=	84	/ DUMMY! DCE_FRMS
I1071E00=	4	/ DUMMY! FRMFLYBK
I1061U00=	84	/ DUMMY! DCE_FRMS
I1071U00=	4	/ DUMMY! FRMFLYBK

For *non-MIPL test data* or *non-flight data*, the "insert_keywords" file must contain the above keywords, otherwise, the pipeline will abort with a message sent to standard output. Attention should be paid when setting DCENUM since it depends on the type of pipeline you wish to run (see section 12 for constraints and rules). The insert_keywords file can be specified via the –k command-line option (see the example tutorial in section 8).

11. Generic Output from All Pipelines

12.1 Science Pipelines

For an input list of science DCEs, the two science pipelines will generate a directory for each processed DCE named in the format "*products_<input_fits_image_name>*" containing intermediate products from the processing steps and the final BCD products. Optionally, the user can specify an output directory (via the –d command-line option) into which all BCD products from every DCE-directory are copied with "<*input_fits_image_name*>" pre-pended in each product filename.

Latent-image detection/flagging is optionally run in the SUR-science pipeline only (RUN_SURSCI.pl) and is performed if the "–l" switch is specified on the command-line (see tutorial in Section 8). Latent-image flagging is an ensemble process, executed on a set of BCD output products generated by SUR-science pre-processing. A minimum of two BCDs is required for triggering latent-image processing. The output from the latent-image flagging step is an update of a bit-mask image (**_bmask_slope.fits*) in each product subdirectory. Bit-5 is set in each mask image if a pixel contains a latent.

Pointing-transfer and Final Product Generation is also optionally run in the SUR-science pipeline only and is performed if the "–a <BPHF.pntg>" option is specified on the command-line, where BPHF.pntg is the Boresight Pointing History File covering the time span of input DCEs. Pointing transfer/final product generation can only be performed if all input FITS are in raw-MIPL (flight data) format and **not** "post-tranhead" format. The final products (with appropriately formatted headers containing pointing information) are named in the format ""*<inputFITSimage>_fp.fits*", i. e. with an "_fp" pre-pended before the ".fits".

12.2 Calibration Pipelines

For an input list of calibration DCEs, the four calibration pipelines will generate a directory for each DCE named in the format "*products_<input_fits_image_name>*" containing intermediate products and final pre-processed DCE products for input into the ensemble processing stages (sections below dashed lines in thread flowcharts of section 4). The user must specify an output directory (via the –d command line option) into which calibration products from the ensemble processing stage will be copied into.

12.2.1 Advisory on Running Calibration (Ensemble) Pipelines

- Due to the presence of a boost and reset for readout-samples 1 and 2 respectively in the first DCE of each exposure sequence, the corresponding planes (read samples) will consist entirely of zeros. Subsequent DCEs in the sequence will be unaffected. To account for this in processing, it is advised that both RAW and SUR-mode dark calibration pipelines (RUN_RAWDRK.pl and RUN_SURDRK.pl respectively) be run with an input list consisting of **only** "first DCEs" from different exposure sequences, or **only** "non-first DCEs".

- The MIPL header keyword DCENUM indicates the DCE order in an exposure sequence, where DCENUM = 0 for "first DCEs".

- All RAW-mode darks in the input list must have the same exposure time (equivalently, the input cubes must have the same number of planes as specified by the NAXIS3 keyword). The pipeline will abort if this criterion is not met. Since the maximum allowable exposure time is 30 mips-seconds (NAXIS3=60), you can use this same dark product when processing raw-science DCEs with NAXIS3 < 60, obviating the need to make a new dark for every raw-science exposure time.

- It is suggested that all SUR-mode darks in the input list correspond to 10 mips-second exposures (implicitly derived from 20-frame samples on board).

- It is advised that the non-linearity calibration pipeline (RUN_LINCAL.pl) be run with an input list consisting entirely of 20-plane (10 mips-second) RAW-mode DCEs so that good time-sampling is achieved. The RUN_LINCAL.pl pipeline will automatically skip DCEs with DCENUM=0 in their header.

- The flatfield calibration pipeline (RUN_FLTCAL.pl) must have an input list consisting entirely of SUR-mode DCEs. As a compromise, it is suggested that only 10 mips-second DCEs be used. The RUN_FLTCAL.pl pipeline will automatically skip DCEs with DCENUM=0 in their header.

- To automatically generate mirror position/rate dependent flat-fields and G-Masks in SSC format, there is script which reads in a FITS image list of post-tranhead products, sorts the data into filelists according to specified mirror-DAC intervals and optionally runs the flat-field pipeline to create flat-field products. Input FITS headers must contain the 'CSM_PRED' and 'CSM_RATE' keywords. After setting your environment (source regular_env.csh in the mips24 tar-ball), you can type "**MakeMirrorDepFlats.pl**" for a command-line tutorial.

- The RAW and SUR-mode science pipelines (RUN_RAWSCI.pl and RUN_SURSCI.pl respectively) automatically select the correct dark to use from the calibration_data/archive directory according to values for the DCENUM keyword in the science header. When processing RAW-mode science images, you must ensure that there exists a RAW-dark cube with NAXIS3(RAW-dark) $\geq$ NAXIS3(RAW-science image) in the archive. The same flatfield/non-linearity calibrations are applied to RAW and SUR-mode science images irrespective of DCENUM.

13. Calibration Data File-Naming Format

All pipelines which depend on a calibration product(s) in processing must be named in a specific file-name format. These files (mostly FITS images) reside in the …/mips24/*calibration_data/* directory in the default installation and its complete path must be specified in the SIRTF_CAL environment variable (in *…/mips24/regular_env.csh*). If you intend to make new calibration products for use in other pipelines, ensure you re-name your products in the format below and copy them to the master calibration directory (under …/mips24 with default name: "/*calibration_data*"). A description of all calibration files and required FITS keywords if one desires to generate them using alternative tools offline is described in …/mips24/*calibration_data/Mips24CalFitsFormat.pdf* .

14. Masks

There are four types of masks used/produced by the pipelines. The P-Mask is expected as input to each pipeline and "treated" as a calibration product. The D-Mask is produced by the pipelines and accompanys each SUR-mode BCD product. The B-Mask specifically applies to the RAW-mode science pipeline and accompanys the BCD output (slope image derived from RAW ramp data). The C-Mask(s) are equivalent to the D-Mask except that they accompany each "calibration" product. Each is discussed in turn below.

15.1. P-Mask (Pixel Mask)

This is a FITS image used to store pixel status information representing the "hardware state" of the focal plane array. The pixels are 16-bit signed integers with the status of certain conditions assigned to the individual bits. The status bits are pre-defined as follows:

```
Bit #   Condition
-----   ---------------------------------------------------
  0
  1
  2
  3
  4
  5
  6
  7     Dark current too variable (dark calibration accuracy will be
                                   unacceptably low)
  8     Response to light too variable (photometric accuracy will be
                                        unacceptably low)
  9     Pixel response to light is too high (unacceptably fast saturation)
 10     Pixel dark current is too excessive (pixel is hot)
 11
 12
 13
 14     Pixel's response to light is too low (pixel is dead)
 15     <reserved: sign bit>
```

15.2. D-Mask (DCE Mask)

This is a FITS image whose number of planes matches that the DCE and is used to store information representing a summary of processing for each pixel through a pipeline. Both the RAW and SUR-mode science threads use a D-Mask. Only at the very end is this converted to a single plane B-Mask (BCD-Mask) image. The pixels are 16-bit signed integers with certain conditions assigned to the individual bits that are turned on if there is at least one occurance of that condition during processing. The status bits are pre-defined as follows:

```
Bit #   Condition
-----   -------------------------------------------------
  0     Incomplete or questionable row-droop correction (rowdroop-RAW, SUR)
```

1	No row-droop correction applied (rowdroop-RAW, SUR)
2	Radhit detection was done (radhit - for RAW mode only) Hard-saturation detected (satmask – for SUR mode only)
3	Digital saturation detected (cvti2r4 - for RAW mode only) Read-2 correction could not be applied (rowfluxcorr – SUR mode only)
4	Saturation corrected (imagest-RAW, desatslope-SUR) and slope value replaced by difference value (satmask – for SUR mode only)
5	Latent-image flag
6	Droop removed using questionable value (droopop)
7	Flat field applied using questionable value (flatap)
8	Flat field could not be applied (flatap)
9	Radhit detection (radhit/imagest - RAW, detect_radhit - SUR)
10	Baseline adjustment failed (basecal - for RAW mode only)
11	Data bad from initial dmask in (radhit - for RAW mode only). Pixel masked in P-Mask – bad hardware state (satmask – for SUR mode only).
12	Non-linearity correction could not be computed (slopecorr - SUR, lineariz - RAW)
13	Saturated - beyond correctable non-linearity (satmask - SUR, lineariz - RAW)
14	Data missing in downlink (cvti2r4-RAW, SUR)
15	<reserved: sign bit>

15.3. SUR-Science B-Mask (BCD - Mask)

This is a single-plane FITS image used to store information representing a summary of processing for each pixel in the SUR-science pipeline. Every SUR-mode science BCD product (slope and difference image) will be accompanied by two B-Masks – one for the slope image (bmask_slope.fits) and one for the difference image (bmask_diff.fits). For SUR-mode only, the B-Mask is a replicated from the D-Mask. The pixels are 16-bit signed integers with certain conditions assigned to the individual bits that are turned on if there was at least one occurance of that condition during processing. The status bits are pre-defined as follows:

Bit #	Condition
0	Incomplete or questionable row-droop correction (rowdroop)
1	No row-droop correction applied (rowdroop)
2	Hard saturated (satmask)
3	Read-2 correction could not be applied (rowfluxcorr)
4	Corrected for soft saturation and slope value replaced by difference value (desatslope and satmask respectively)
5	Latent-image flag
6	Droop removed using questionable value (droopop)
7	Flat field applied using questionable value (flatap)
8	Flat field could not be applied (flatap)
9	Radhit detection (detect_radhit)
10	
11	Pixel masked in pmask - bad hardware state (satmask)
12	Non-linearity correction could not be computed (slopecorr)
13	Soft saturated (satmask)
14	Data missing in downlink (cvti2r4)

Deleted: 3

```
 15      <reserved: sign bit>
```

15.4. RAW-Science B-Mask (BCD - Mask)

This is a single-plane FITS image used to store information representing a summary of processing for each pixel in a RAW data ramp (e.g. the RAW-science pipeline). Every RAW-mode science BCD product (slope image derived from RAW-ramp data) will be accompanied by a B-Mask. The pixels are 16-bit signed integers with certain conditions assigned to the individual bits that are turned on if there was at least one occurance of that condition along the ramp during processing. The status bits are pre-defined as follows:

```
Bit #  Condition
-----  -------------------------------------------------
0
1
2      Digital saturation detected in sample(s) along ramp
3      Radhit detection along ramp in sample(s) along ramp
4      Non-linearity correction could not be computed in sample(s) along ramp
5
6      Droop or rowdroop removed using questionable value in sample(s) along
       ramp
7      Flat field applied using questionable value (flatap)
8      Flat field could not be applied (flatap)
9      Baseline adjustment failed (basecal)
10     Saturated - beyond correctable non-linearity in sample(s) along ramp
11     Data missing in downlink in sample(s) along ramp
12     Only one usable plane
13     No usable planes
14     Pixel masked in pmask
15     <reserved: sign bit>
```

15.5. C-Mask (Calibration-Mask)

This is a FITS image used to store status information representing a summary of processing for each pixel through a "calibration" pipeline. Every calibration product will be accompanied by a C-Mask. Since we have three types of calibration product: Darks, Non-linearity and Flatfield, there is a different C-Mask assigned to each. The pixels are 16-bit signed integers with the status of certain conditions assigned to the individual bits. The status bits are pre-defined as follows for the three types of calibration product:

15.5.1. Dark Calibration C-Mask

```
Bit #    Condition
-----    --------------------------------------------------------------
  0
  1      Some samples were masked in dmask
  2      Masked in input cmask
```

Deleted: 3

```
  3
  4
  5
  6
  7
  8
  9
 10
 11      Two-by-two pixel block identified as too noisy (current pixel has
                                         smaller x,y coordinate in block)
 12      Masked in pmask
 13
 14
 15      <reserved: sign bit>
```

15.5.2. Non-Linearity Calibration C-Mask

```
Bit #    Condition
-----    --------------------------------------------------------------
  0      Goodness-of-fit compensation applied
  1      Some samples were masked in dmask
  2      Non-negative A & C constraint enforced (model 1)
  3      Model A coefficient is negative (model 1)
  4      Model C coefficient is negative (model 1)
  5      Saturation rejection code activated
  6      Fitting sigma is greater than SigMax
  7      Negative A or C (model 1) or zero C (model 2)
  8      Insufficient dynamic range (DynLo - DynHi)
  9      Not enough usable points to compute a fit
 10      Determinant in least-squares fit is zero
 11      CalTrans interpolation could not be performed
 12      Masked in pmask
 13
 14
 15      <reserved: sign bit>
```

15.5.3. Flat-field Calibration C-Mask

```
Bit #    Condition
-----    --------------------------------------------------------------
  0      One or more input rad-hits (outliers) present in the input data
  1      One or more outliers detected via multi-frame method in the input
data
  2      One or more NaNs present in the input data
  3
  4
  5
  6
  7
  8
  9
 10
 11
 12      Too many outliers present
```

Deleted: 3

```
 13      Too many NaNs present
 14      No input data available
 15      <reserved: sign bit>
```

15. Individual Pipeline Modules

Below is a complete list of the modules used in each of the six mips-24 pipeline threads. Beside each one is the pipeline(s) it is used in. Please note that due to SODB independency, the modules: CALTRANS and QALOADER are not used in the offline versions. A brief tutorial on each module's inputs/outputs/dependencies can be obtained by typing the module binary name as it appears below on the command line (provided you have "sourced" your *regular_env.csh* file so that they get put into your path). For reference, a brief summary for each module was given in the pipeline flow-charts of section 4. Documentation is also available upon request for each individual module in the form of an SDS (see section 16).

Here's a legend for the pipelines referenced in the table below:

SURSCI - SUR mode science pipeline

RAWSCI - RAW mode science pipeline

SURDRK - SUR mode dark calibration pipeline

RAWDRK - RAW mode dark calibration pipeline

LINCAL - Non-linearity calibration pipeline

FLTCAL - Flatfield estimation calibration pipeline

```
Module           Pipeline
-----------     ----------------------------------------
angleavg        -  SURSCI
basecal         -  RAWDRK,LINCAL,RAWSCI
boresightTran   -  SURSCI
calkeywords     -  SURDRK,RAWDRK,LINCAL,FLTCAL
caltrans        -  SURDRK,RAWDRK,LINCAL,FLTCAL,SURSCI,RAWSCI
cubesub         -  SURSCI,RAWSCI,LINCAL,FLTCAL
cvti2r4         -  SURDRK,RAWDRK,LINCAL,FLTCAL,SURSCI,RAWSCI
darkest         -  SURDRK,RAWDRK
desatslope      -  SURSCI,FLTCAL
detect_radhit   -  SURSCI
dntoflux        -  SURSCI,RAWSCI
droopop         -  SURSCI,RAWSCI,LINCAL,FLTCAL
flatap          -  SURSCI,RAWSCI
flatfield       -  FLTCAL
flattrack       -  FLTCAL
fpgen           -  SURSCI
```

Deleted: 3

```
getPH_offline   -  SURSCI
hdrupd8         -  SURDRK,RAWDRK,LINCAL,FLTCAL,SURSCI,RAWSCI
hdrupdate       -  SURSCI
imagest         -  RAWSCI
imfliprot       -  SURDRK,RAWDRK,LINCAL,FLTCAL,SURSCI,RAWSCI
imheader        -  SURDRK,RAWDRK,LINCAL,FLTCAL,SURSCI,RAWSCI
latimdetect     -  SURSCI
lincal          -  LINCAL
neariz          -  RAWSCI
medfilter       -  SURSCI
mirrorsynch     -  SURSCI
qaloader        -  SURDRK,RAWDRK,LINCAL,FLTCAL,SURSCI,RAWSCI
qatool          -  SURDRK,RAWDRK,LINCAL,FLTCAL,SURSCI,RAWSCI
radhit          -  RAWSCI
rowdroop        -  SURSCI,RAWSCI,LINCAL,FLTCAL
rowfluxcorr     -  SURSCI,SURDRK,FLTCAL
sanity_datatype -  SURDRK,RAWDRK,LINCAL,FLTCAL,SURSCI,RAWSCI
satmask         -  SURSCI,SURDRK,FLTCAL
slopecorr       -  SURSCI,FLTCAL
sloperror       -  SURSCI,SURDRK,FLTCAL
snestimator     -  SURDRK,RAWDRK,LINCAL,FLTCAL,SURSCI,RAWSCI
split2planecube -  SURSCI,SURDRK
tranhead        -  SURDRK,RAWDRK,LINCAL,FLTCAL,SURSCI,RAWSCI
```

16. Further Documentation

Subsystem Design Specification Documents (SDSs) for individual modules are only accessible to SSC staff with accounts mounted across the /ssc/pipe partition. They can be found in directories: */ssc/pipe/docs/sds/S*/* . Look at the most recent S* directory first, then search backwards. All up-to-date SDS's are available upon request in one compressed-tar file.

17. Example Test Runs

The directory *test_imgs*/ under *mips24*/ contains test data for testing each pipeline. Below we give the command line syntax used to test each pipeline. The examples below are run in the same directory where the pipeline scripts reside. Directories containing output products are made in the same (execution) directory. These examples are purely illustrative. You are free to specify the complete paths to the input data, or, the output product directory. Either *absolute or relative* pathnames (relative to the execution directory) must be specified in the input FITS file lists.

The example below assumes you are processing flight-(or MIPL) data, already containing the required keywords. If this is not MIPL data, you must modify the "insert_keywords" file prior to executing a specific pipeline (an example "insert_keywords" file is in the namelists/ directory) and include it's name (with path) on the command-line, e.g: "–k ./namelists/insert_keywords.

```
% source regular_env.csh
```

For no latent-image flagging and pointing-transfer/FPG in post-processing:

```
% RUN_SURSCI.pl -i ./test_imgs/inputlist_sursci -d ./bcd_dir_sursci –v
```

With latent-image flagging and pointing-transfer/FPG in post-processing:

```
% RUN_SURSCI.pl -i ./test_imgs/inputlist_sursci -d ./bcd_dir_sursci –l –a
                 ./test_imgs/BPHF.0734486400.10.pntg –v

% RUN_RAWSCI.pl -i ./test_imgs/inputlist_rawsci -d ./bcd_dir_rawsci –v

% RUN_SURDRK.pl -i ./test_imgs/inputlist_surdrk -d ./cal_dir_surdrk –v

% RUN_RAWDRK.pl -i ./test_imgs/inputlist_rawdrk -d ./cal_dir_rawdrk –v

% RUN_LINCAL.pl -i ./test_imgs/inputlist_lincal -d ./cal_dir_lincal –v

% RUN_FLTCAL.pl -i ./test_imgs/inputlist_fltcal -d ./cal_dir_fltcal –v
```

18. Glossary

AOR	Astronomical Observer Request
APES	Automated Pipeline Executive for SIRTF
BCD	Basic Calibrated Data
BPHF	Bore-sight Pointing History File
CSMM	Cryogenic Scan Mirror Mechanism
CTD	Command and Telemetry Dictionary
DCE	Data Collection Event
DN	Data Number
DSN	Deep Space Network
FOS	Flight Operations System
FOV	Field-of-View
FPA	Focal Plane Array
FPG	Final Product Generator
MIPL	Multi-mission Image Processing Laboratory

Deleted: 3

MIPS	Multi-band Imaging Photometer for SIRTF
IER	Instrument Engineering Request
IOC	In-Orbit Checkout
PTG	Short for “Pointing”
SCET	Spacecraft Ephemeris Time (data-time string from UT 0hrs 1/1/1980)
SCLK	Spacecraft Clock time (in seconds from UT 0hrs 1/1/1980)
SDS	Subsystem Design Specification
SIRTF	Space Infrared Telescope Facility
SIS	Software Interface Specification
SODB	Science Operations Data-Base
SUR	Sample Up the Ramp
SSC	SIRTF Science Center
TBD	To Be Determined
TBR	To Be Resolved