Technical Report MESA 2018-001

Procedures for Submitting Calibration Reference Files

Matt McMaster, Rosa Diaz, and the ReDCaT Team
June 04, 2018

ABSTRACT

This document gives more detail regarding what instrument teams need to do prior to and during the submission of calibration reference files to CRDS via the ReDCAT Team. Where appropriate, the information will be split according to telescope, since the procedures can vary slightly for each. Also included is information regarding what to do when a new reference file type is needed, or when making a significant change to a reference type (new modes, different number of extensions, new columns in tables, etc.). Finally, a section giving an overview of the varies mapping files.

Introduction

The generation of calibration reference files is the sole responsibility of the instrument team whose data will be calibrated by them. As part of this process, testing and assessment of the files need to be performed before they are sent to the ReDCaT (Reference Data for Calibration and Tools) Management Team, henceforth, ReDCaT Team, for ingest into DMS (Data Management System), which includes CRDS (Calibration Reference Data System) and DADS (Data Archive Distribution Services), or the archive. Ingestion into the archive allows both local and offsite users to retrieve them. Note that users can also obtain references using CRDS tools. Information on these tools can be found at: https://hst-crds.stsci.edu/static/users_guide/basic_use.html, https://jwst-crds.stsci.edu/static/users_guide/basic_use.html, and http://jwst-pipeline.readthedocs.io/en/stable/jwst/introd uction.html#crds-reference-files.

The main function of CRDS is to correctly select references for a given observation mode of the data. The selection of references is based on information contained in CRDS rmaps (rules or reference mapping files – see Section 7.3). For the HST instruments, the selection rules are based on what is contained in ICD-47 (Interface Control Document), located at: http://newcdbs.stsci.edu/doc/ICD47/index.html. For JWST instruments, the selection rules are generated from information given in the calibration reference files document, located at: http://jwst-reffiles.stsci.edu/intro.html.

Checking and quality assessment of reference files is necessary for the following reasons:

- to ensure that the required keyword content and syntax are correct
- to ensure that file structure and format are correct
- to ensure that the reference files work properly with the calibration software
- to ensure that the reference files are properly selected by CRDS
- to verify scientific validity
- to ensure proper documentation of the reference files

The instrument teams are responsible for all of the points listed above, while during delivery, the ReDCaT Team performs checks to verify compliance of the first four points, and to a lesser extent, the last point. Note that reference files are the sole responsibility of the instrument teams and that no changes to the files, regardless of how seemingly insignificant they may be, will be made by the ReDCaT Team.

1 Information on File Preparation

HST reference files are all in FITS format, while JWST references can be FITS, ASDF, or JSON files. Regardless of the format, all references must be compliant to the standards for that format type. If you are using python when generating a file, you will receive a warning or error upon closing the file if it is not compliant. If this happens, the file must be corrected or it will be rejected by CRDS and the ReDCaT Team will be unable to deliver the file. Along with having your references format compliant, there are several keywords that are also required. There are keywords that are mission specific, and keywords that are reference specific.

1.1 HST

For HST, the mission specific keywords are: USEAFTER, COMMENT, DESCRIP, PEDIGREE, and HISTORY.

1.1.1 USEAFTER

MMM DD YYYY HH:MM:SS the date and time after which the reference will be used. All of the fields are required. If there is no meaningful time for a reference (e.g., a reference that covers data throughout the life of the instrument, or when it was first switched on), it should be set to 00:00:00. Note, if the reference you're delivering replaces one currently in use, the USEAFTERs have to match exactly; not doing so will result in either an orphaned reference (an active reference that will not be used on any data), or incomplete coverage (some data will not be calibrated with the new reference). If the reference being replaced should no longer be used for any reason, please see the explanation for item 10b in Section 5.

1.1.2 COMMENT

Who created the file, can also include any script that was used in generating the file (e.g., "Reference file created by N. Miles." or "Reference file created by M. Bourque using sriptname.py"). Note that some FITS files contain the default COMMENT section which gives the literary reference about these types of files; this section should be removed before submitting your files.

1.1.3 DESCRIP

This keyword should be a concise sentence that makes it easy for a user to determine the type of reference file and the changes being made; it should be a one-line explanation of why the file is being delivered. For example:

DESCRIP = "ACS cte corrected dark for data taken after May 21 2016."
DESCRIP = "New STIS MAMA blaze function corrections and sensitivities for echelle modes."

More complete information about any changes and how the file was created should be provided in the HISTORY section.

1.1.4 PEDIGREE

The type of data that was used in generating the file, it can be DUMMY, MODEL, GROUND, or INFLIGHT.

1.1.4.1 DUMMY

A placeholder file that contains all ones or all zeroes, depending on whether the file is multiplicative or additive, respectively.

1.1.4.2 MODEL
A reference that was generated using only model or mathematical information.

1.1.4.3 GROUND
A reference that was created using data from ground tests.

1.1.4.4 INFLIGHT
A reference that was generated by utilizing data that was taken on-orbit. It has the format: INFLIGHT DD/ MM/YYYY DD/MM/YYYY which represents the range in dates of the data used in generating the file.

1.1.5 HISTORY

The HISTORY section should contain a complete, but concise description of the process used in generating the file. This would include things like any scripts used in generating the file. Also include any pertinent numbers users might find useful: the total exposure time, the timeframe over which the exposures used in creating the reference were taken, the value above which hot pixels were flagged, etc. The names of the raw datasets used in generating the reference, pointers to relevant ISRs and TIRs, why the file is being delivered and/or an explanation of what was updated, etc. As much as possible, the information should pertain to just that particular file and not the reference type as a whole.

1.2 JWST

The JWST mission specific keywords are: REFTYPE, DESCRIP, AUTHOR, USEAFTER, PEDIGREE, and HISTORY.

1.2.1 REFTYPE

The type of reference being delivered. The acceptable values depend on the reference file being used. These are listed in http://jwst-reffiles.stsci.edu/source/all_steps.html

1.2.2 DESCRIP

This keyword gives a summary of the file content and the reason for delivery; it should be a concise sentence. For example:

DESCRIP = “New NIRISS pixel flat for the F356W filter updated with cycle 1 observations.”
DESCRIP = “New MIRI MIRIMAGE readnoise array updated with cycle 2 observations.”

1.2.3 AUTHOR

The person(s) who created the file. For example:

AUTHOR = "M. Cracraft"
AUTHOR = "Michael A. Wolfe and Kevin Volk"

1.2.4 USEAFTER

USEAFTER has the format YYYY-MM-DDTHH:MM:SS The date and time after which the reference is to be used; the T is required. If there is no meaningful value for the time, it should be set to 00:00:00.

1.2.5 PEDIGREE

PEDIGREE has the options MODEL, GROUND, DUMMY, or INFLIGHT.

1.2.5.1 MODEL
Use MODEL when a file that has been generated using modeling or simulation software.

1.2.5.2 GROUND
Use GROUND for a file that has been derived from data taken during ground testing.

1.2.5.3 DUMMY
Use DUMMY for a file containing all zeroes (additive) or ones (multiplicative) and used as a placeholder.

1.2.5.4 INFLIGHT
Use INFLIGHT for a reference file derived from data taken on-orbit, having the format: INFLIGHT YYYY-MM-DD YYYY-MM-DD which represents the range of dates of the data used to generate the file.

1.2.6 HISTORY

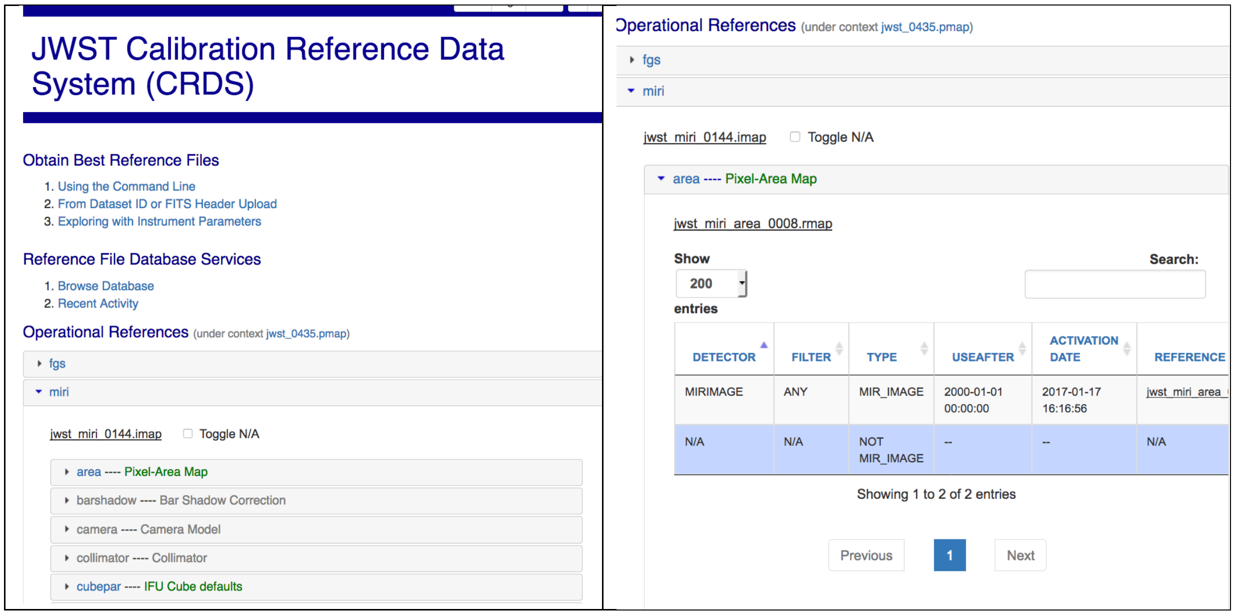
This section contains a detailed description of the creation of the reference file, including documentation which describes the strategy and algorithms used to generate the file, software used, the names of the raw data files used, the any differences with the file it is replacing (if applicable). As much as possible, the information contained in the HISTORY section should pertain to that particular reference and not the reference type as a whole. The HISTORY section should look something like the following:

HISTORY Date corresponding to the following description
HISTORY Description of how the reference was created.
HISTORY 'DOCUMENT: Title of the document describing the strategy and algorithms used HISTORY in generating the file.

HISTORY SOFTWARE: The description, version number, and location of the software that
HISTORY   was used in creating the file.
HISTORY DATA USED: Names of the data files used to create the file.
HISTORY ‘DIFFERENCES: If applicable, how is this version
HISTORY different than the one that is being replaced.

1.3 Other Important Keywords

There are other keywords, specific to instrument and reference type, that are also required. There are a couple of ways in which to find them. One is to examine the rmap (details about rmaps can be found in Section 7.3). For HST references, go to *https://hst-crds.stsci.edu*; similarly, for JWST instruments, go to *https://jwst-crds.stsci.edu*. Click on your instrument and then on the reference type. This is shown in Figure 1-1



JWST Calibration Reference Data System (CRDS)

Obtain Best Reference Files

1. Using the Command Line
2. From Dataset ID or FITS Header Upload
3. Exploring with Instrument Parameters

Reference File Database Services

1. Browse Database
2. Recent Activity

Operational References (under context jwst_0435.pmap)

- fgs
- miri

jwst_miri_0144.imap   Toggle N/A

- area ---- Pixel-Area Map
- barshadow ---- Bar Shadow Correction
- camera ---- Camera Model
- collimator ---- Collimator
- cubepar ---- IFU Cube defaults

Operational References (under context jwst_0435.pmap)

- fgs
- miri

jwst_miri_0144.imap   Toggle N/A

- area ---- Pixel-Area Map

jwst_miri_area_0008.rmap

Show 200 entries   Search:

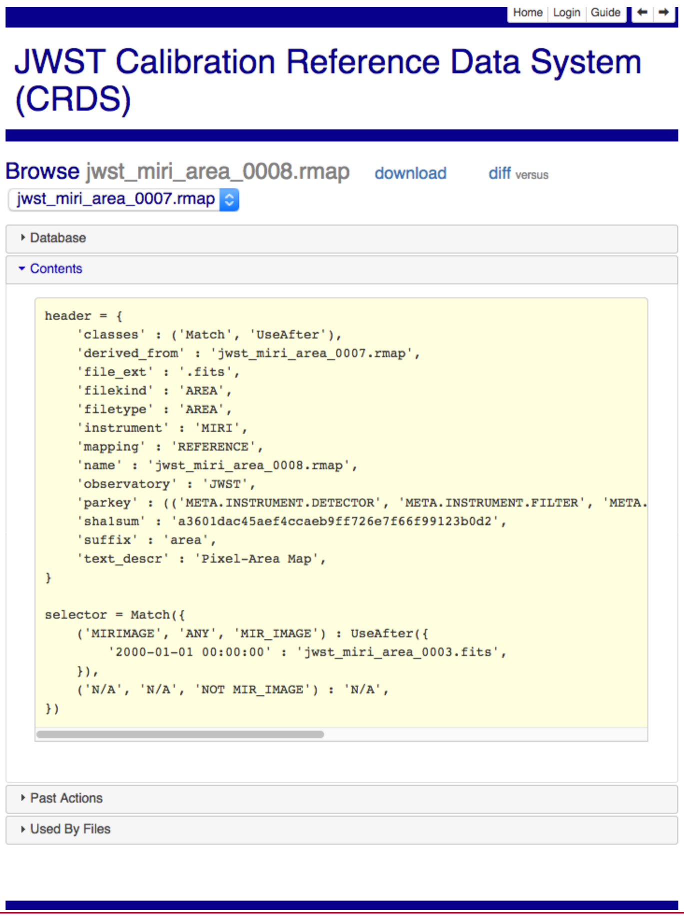
DETECTOR	FILTER	TYPE	USEAFTER	ACTIVATION DATE	REFERENCE
MIRIMAGE	ANY	MIR_IMAGE	2000-01-01 00:00:00	2017-01-17 16:16:56	jwst_miri_area_
N/A	N/A	NOT MIR_IMAGE	--	--	N/A

Showing 1 to 2 of 2 entries

Previous 1 Next

Figure 1-1 Sample view from jwst-crds.stsci.edu

At the top of the listing, you will see a link to the rmap (e.g., hst_wfc3_darkfile_0359.rmap or jwst_miri_area_0008.rmap). Clicking on that link will take you to another page with four dropdowns; select (open) the one marked Contents. Partway down, you will see an entry called ‘parkey’, short for parameter keyword.


Home | Login | Guide
JWST Calibration Reference Data System (CRDS)
Browse jwst_miri_area_0008.rmap download diff versus
jwst_miri_area_0007.rmap
Database
Contents
Past Actions
Used By Files

```
header = {
    'classes' : ('Match', 'UseAfter'),
    'derived_from' : 'jwst_miri_area_0007.rmap',
    'file_ext' : '.fits',
    'filekind' : 'AREA',
    'filetype' : 'AREA',
    'instrument' : 'MIRI',
    'mapping' : 'REFERENCE',
    'name' : 'jwst_miri_area_0008.rmap',
    'observatory' : 'JWST',
    'parkey' : (('META.INSTRUMENT.DETECTOR', 'META.INSTRUMENT.FILTER', 'META.
    'sha1sum' : 'a3601dac45aef4ccaeb9ff726e7f66f99123b0d2',
    'suffix' : 'area',
    'text_descr' : 'Pixel-Area Map',
}

selector = Match({
    ('MIRIMAGE', 'ANY', 'MIR_IMAGE') : UseAfter({
        '2000-01-01 00:00:00' : 'jwst_miri_area_0003.fits',
    }),
    ('N/A', 'N/A', 'NOT MIR_IMAGE') : 'N/A',
})
```

Figure 1-2 Sample view options for rmap files

The view option above does not show the full `parkey` entry. In this case it is:

```
parkey' : (('META.INSTRUMENT.DETECTOR', 'META.INSTRUMENT.FILTER',
'META.EXPOSURE.TYPE'), ('META.OBSERVATION.DATE', 'META.OBSERVATION.TIME')),
```

These are the keywords that CRDS pulls from the data in order to determine the best reference to use, they are also the other required keywords for that particular reference type. Note that in the `selector = Match` section, that DATE-OBS and TIME-OBS are combined to generate a date which is used for comparison with the USEAFTERs of the references. For JWST references, it's the meta keywords that are listed (e.g., META.SUBARRAY.NAME, META.INSTRUMENT.DETECTOR) but their header keyword counterpart (using the previous example: SUBARRAY, DETECTOR) is the one that is required in the reference.

1.4 Checksum

At times, an instrument team will use an existing file in the creation of an updated version of that file. The existing file may have CHECKSUM (and DATASUM) keywords that were added by CRDS during the delivery process. When CRDS checks the updated version of the file, it generates an internal CHECKSUM and compares it with any other CHECKSUMs it finds in the header. If the CHECKSUM it generated does not match with the CHECKSUM in the header, CRDS will issue an error and the delivery of the file will fail. Therefore, it is important that any CHECKSUMs in the header be updated.

To update the CHECKSUM and DATASUM keywords, do the following:

First, update your environment to the latest version

```
$bash
$source activate astroconda
$conda update --all
```

then, update the checksum of your file(s)

```
$crds checksum /directory/where/files/reside/*.fits
```

Alternatively, you can use a file list containing the files you want to be updated (note, that, the complete directory path must be included with each file). The last step would then be:

```
crds checksum @filelist
```

For more information, please see:
(https://jwst-crds.stsci.edu/static/users_guide/command_line_tools.html#crds-checksum).

2 FILE NAMES

There are no restrictions on what the references are named when they are submitted to he ReDCaT Team for delivery, the files will receive a new name based on their FILETYPE (HST) or REFTYPE (JWST) keywords.

2.1 HST

HST references are renamed by the ReDCaT Team during the delivery process by means of a script which calls crds.uniqname. A reference is named based on the years since HST launch, date and time (UT), instrument, and reference type, and has the form:

```
ymdhhmmsi_type.fits
```

where

y = year (a-z, 0-9, references delivered in 2018 all begin with 2)
m = month (1-9, a-c)
d = day (1-9, a-s, [t], [u], [v])
hh = UT hours (00-23)
mm = UT minutes (00-59)
s = UT seconds, in 2 second intervals (0-9, a-t)
i = instrument (j = ACS, l = COS, o = STIS, i = WFC3)
type = reference type extension (e.g., bia, drk, gsag).

2.2 JWST

For JWST references, the names are sequentially generated by CRDS during the file certification process, and have the form:

```
jwst_instrument_reference_version.ext
```

where extension can be .fits, .asdf, or .json (e.g., jwst_nirspec_readnoise_0015.fits).

3 CERTIFYING FILES

This is not optional, instrument teams need to do this in order to prevent problems with the delivery.

First, you need to make sure you update your conda and astroconda environments, as your CRDS version might be outdated. For this do the following:

```
mycomputer: bash
mycomputer: conda update conda
bash$ source activate astroconda (or whatever your conda environment is called)
bash$ conda update --all

bash$ source activate jwst_latest_version
bash$ crds certify /full/path/to/yourfiles/*.fits (or *.json, or *.asdf)
```

Activating the latest JWST environment is optional. For more information on how to load the JWST environment refer to the JWST Documents (https://jwst-docs.stsci.edu/display/JDAT/Obtaining+and+Installing+Software).

Note, that, there still may be some back-and-forth with the ReDCaT Team if you find problems with your files, or if we find something later in the delivery process.

3.1 Fitsverify the FITS Files

Whilte certify will also verify your fits files, you may also want to do a separate check. The easiest way to do this is by using **fitsverify**, which is included in astroconda.

Start a bash session and make sure your conda environment has been updated (see above). Then type the following within the directory where your files reside:

```
bash$ fitsverify *.fits
```

The results can be sent to a file for easier viewing. Any errors found need to be fixed**.**

4 TESTING FILES

During the delivery process, the ReDCaT Team will double check that your references are FITS (or ASDF, or JSON) compliant, contain the required keywords and they have valid values, and that the file will be properly selected, given the criteria. However, we do not check the scientific/technical validity of a file (i.e., we don't make sure the data within the file makes sense nor whether it properly calibrates the data). Therefore, it is highly recommended that reference files be thoroughly checked before submission. It is also recommended that you verify that the new reference files have the desired effect on the data they'll be used on. Keep notes on how you determined that the files were good, since this information will be needed when you submit them for delivery.

To ensure that there are no problems within the calibration pipeline, you are also required to test the reference using the version of the calibration software currently being used within the pipeline. For JWST references, you can directly use the Calibration Pipeline step you need or use the Reference File Testing Tool (RFTT). The documentation on how to use this tool can be found at http://reference-file-testing-tool.readthedocs.io/en/stable/index.html.

If your references are for HST, are going into the TEST system, and there is a new version of the calibration software for your instrument which will also be tested, this must be noted on the delivery form.

5 DELIVERY FORM

The delivery form is used by the ReDCaT Team to determine what it is an instrument team is trying to deliver, why the files are being delivered, whether or not files are being replaced, and if so, whether they need to be marked as bad. The more information contained in the delivery form, the easier it is for the ReDCaT Team to spot (potential) trouble with a delivery, or possibly help in diagnosing a problem during the processing of a delivery.

Below is a sample of the delivery form:

```
1. Name of deliverer: a. (other e-mail addresses)

2. Date of Delivery:

3. Instrument:

4. Type of file(s) (example: Bias, Dark, etc.):

5. Has HISTORY section in the primary header been updated to describe in
detail the reason for delivery and how the file(s) was(were) created?
(yes/no):

6. Have USEAFTER, PEDIGREE, DESCRIP, COMMENT, and HISTORY (for HST files),
or REFTYPE, DESCRIP, AUTHOR, USEAFTER, PEDIGREE, and HISTORY (for JWST
files) been checked?.
a. Was the DESCRIP keyword updated with a summary of why the file was
updated or created? (yes/no):
b. If the reference files are replacing previous versions, do the new
USEAFTER dates exactly match the old ones?

7. Verification for compliance complete? (.fits, .json, .asdf compliant,
certify, etc.):

8. Should these files be ingested in the DMS, Archive and CRDS databases?(If
not clear or needs to go to CRDS-TEST first,indicate here):
a. If files are pysynphot files, should they be delivered to ETC?
b. Are these JWST ETC files?
```

9. Files run through the current version of the calibration software being used by the pipeline or PYSYNPHOT and ETC? (yes/no and list the version used):

10. Does it replace an old reference file? (yes/no):
a. If yes, which one?
b. If the file being replaced is bad, and should not be used with any data, please indicate this here:

11. Was a JIRA issue filed in regard to the references being delivered and/or their rmap? (yes/no):
a. if yes, please include the JIRA issue number here

12. What is the level of change of the file? (e.g. compared to old file it could be: SEVERE, MODERATE, TRIVIAL – initial delivery of a reference file type is always SEVERE):
a. If files are tables (but no throughput tables), please indicate exactly which rows/columns have changed:

13. Please indicate which modes (e.g. all the STIS, FUVMAMA, E140L modes) are affected by the references:

14. Description of how the files were tested for correctness:

15. Additional Considerations:

16. Disk location and name of files:

17. Reason for delivery:

Some detailed notes on the delivery form:

Item
1. The name of the person submitting the delivery form.
1a: The email addresses of other people who should be notified after the completion of the delivery.

2. The date the delivery form is being submitted to the ReDCaT Team.

3. The instrument for which the references will be used to calibrate data.

4. The type(s) of file(s) being submitted. Note that you can submit more than one type of file, but each reference type has to be accounted for in the form.

5. See Section 1.1.5 (HST) or Section 1.2.6 (JWST) for the details on how the HISTORY section should be updated.

6. For HST files, the USEAFTER, PEDIGREE, DESCRIP, and COMMENT keywords need to be checked and, if necessary, updated; similarly, REFTYPE, DESCRIP, AUTHOR, USEAFTER, and PEDIGREE need to be done for JWST references. See Section 1.1 or 1.2 for more information.
6a. The DESCRIP keyword should be a clear and concise reason for why the file is being delivered. See Section 1.1.3 or 1.2.2 for more details.
6b. If the files you are submitting will be replacing currently active files, the USEAFTER(s) of the file(s) have to match exactly, to the second. See Section 1.1.1 or 1.2.4 for more details.

7. Your files must be format (FITS, JSON, ASDF) compliant. If you've generated them using python, you will receive warnings or errors if they are not compliant. If you've used something other than python to generate your references, you would need to use the method described in section 3 (certify not only checks that required keywords are present and have valid values, it also checks for format compliance). Note that certify will not work on pysynphot, ETC, and CALSPEC files.

8. As long as the files being delivered are for calibration pipeline use and are NOT going to HST TEST, then you should answer YES to this question. Note that while JWST references do not presently go to DMS or the Archive, they do go the CRDS bestref database.
8a. Generally speaking, HST pysynphot files are delivered to the ETC. HST ETC releases occur once or twice per year.
8b. If you are delivering JWST ETC files, please indicate that here, otherwise, answer NO.

9. As mentioned in Section 4, your references need to be run through the current version of the calibration software running in the pipeline. For HST references, go to http://astroconda.readthedocs.io/en/latest/releases.html#pipeline-install and follow the instructions for installing the most recent HSTDP (HST Data Processing) build. For JWST references follow Section 4. Similarly, if your references are going to the HST TEST system and the software that uses them will also be installed there, please note it here.

10. Are the files you're submitting replacing currently active files (note that if this is the case, the USEAFTER dates need to exactly match, though there may be times when the USEAFTER dates for the files being replaced and those replacing them intentionally don't match – see below)? There may be instances where some of the files you're submitting are new, while others are replacements. If this is the case, answer YES and make some sort of note: "YES, some files are being replaced." "YES, the bias files are replacements." "YES, three superbias and two gain references are replacements."
10a. List the files that are being replaced, one per line. If possible, give the name of the replacement file and the file that it's replacing: newfile_1.fits replaces activefile_1.fits This is important because before a delivery is confirmed, the ReDCaT Team member doing the delivery will examine the processing results. Not only will files being replaced be noted, but the files replacing them will also be given. If we see that a file is to be replaced but isn't, we can then contact you for clarification. If necessary, the delivery can be cancelled.
10b. In here, "bad" is a reference that will cause the calibration to fail or the result of applying it during calibration would make the data unusable. For example, a dark that is replaced by one with better noise statistics would not be considered bad, just non-optimal. However, if, say, the calibration code were changed in such a way that using the current dark would cause the software to crash or the output file to be unusable for analysis, then that file would be considered bad. For cases where the USEAFTER dates don't match, please indicate that the files need to be removed (they will be deactivated, but otherwise still available).

11. Was a JIRA issue filed in relation to either the references being submitted or their respective rmap? If you are introducing a new reference type or this is the initial delivery of a reference type, a JIRA issue should have been filed. If the rmap for a reference was updated (say a new search criteria was added) and the submission is a result of the change, a JIRA issue should have been filled.
11a. If a JIRA issue was filed, please list it here (e.g., JIRA CRDS-102).

12. If the files you're submitting are replacements, what is the level of change compared to the files they're replacing. If the files are not replacements, then the level of change should be set to SEVERE.
12a. If the references you're submitting are tables, and not ETC or pysynphot files, list the rows/columns that were updated (e.g., "All rows in the DETECTOR column." or "All FUVA rows in the CENWAVE column.").

13. Which observational modes are affected by the references you're submitting ("All STIS FUV MAMA, E140L modes." "NRC_IMAGE data taken in either the NRCA1 or NRCA2 detectors after July 1 2017."). Note that there may be more than one mode covered by the references being submitted. In this case, you would need to give the information for each affected mode. This is particularly important for table reference files that might have one row per observation mode.

14. Describe what was done in order to make sure that the references you're submitting work with the version of the calibration software, and that the results are as expected.

15. Include any additional comments that would make it easier for the ReDCaT Team to process the delivery or help in diagnosing a problem. Examples would include: initial delivery of a reference type; a new selection criterion has been added to the rmap; the file should go to the TEST system.

16. This should be left blank, it is automatically filled in by the ReDCaT Team's reference submission script.

17. This is the most important part of the delivery form. The information as to why the references are being delivered will not only go out to users, but it will also eventually find its way to GSFC, so the more detail that is added, the better. For example, "New NIRCAM darks." or "Files are being delivered at the request of so-and-so." are not acceptable. Instead, "New ACS WFC darks, cte corrected darks, and biases created from on orbit data taken between April 22 2011 and May 5 2011, and generated using <scriptname>.py. These files are for the calibration of ACS WFC data taken after May 1 2011." or "New NIRCam SUB640 darks for the NRCB1, NRCB2, and NRCB3 detectors. These new darks were generated using (CV2,CV3, simulated, etc....) data and are being delivered for use with the Build 7.1 calibration pipeline." would be more appropriate. Note that the only grammatical marks that should be used are commas (,), underscores (_), and periods (.). Any other characters, such as semi-colons (;), colons (:), dashes (-), etc., will cause CRDS to crash and the delivery to fail.

6 DEFINING A (NEW) REFERENCE TYPE

Occasionally, a new reference file type is needed, either as a result of a change to the calibration software or as a mission requirement; this also refers to significant updates to an existing reference type (new modes, new table columns, new/different extensions, etc.). In order for CRDS to properly handle the new/updated type, please follow the steps below.
If you are defining a new reference type, please file a JIRA issue against CRDS (https://jira.stsci.edu/browse/CRDS), assign it to Matt McMaster, and share it with redcat@stsci.edu. Use the following template when filing your issue:

```
Basic Information
=================

1- JIRA ISSUE number
--------------------------------------------
2- FILE DESCRIPTION (For rmap and web submission documentation)
---------------------------------------------------------------
3- INSTRUMENT
-------------------
4- TYPE NAME (CRDS filekind, no underscores, letter first; e.g.ipc, mask)
----------------------------------------------------------------------------
5- LONG FORM TYPENAME (CRDS filetype? for website context display;e.g.
   “Interpixel Capacitance”, “Bad Pixel Mask”)
----------------------------------------------------------------------------
6- EXTENSION / FILE FORMAT (.asdf, .fits, .json)
-------------------------------------------------------------
7- SELECTION PARAMETERS (e.g. DETECTOR, FILTER)
-----------------------------------------------------------------------
8- REQUIRED (YES / NO?)
-------------------------------
9- DIRECTORY PATH WHERE WE CAN FIND THE FILE(S)
---------------------------------------------------------------------------
10-   INTIAL REFERENCE FILENAMES (e.g. jwst_fgs_ipc)
--------------------------------------------------------------------
11-   FILE CREATOR
----------------------
12-   TABLE (YES / NO?)
--------------------------
13-   PARAMETER NAME FOR CALIBRATION STEP (eg. HST BIASCORR)
----------------------------------------------------------------------------
14-   OBSERVATORY
-----------------------
15-   WHEN NOT USED (N/A):
---------------------------------
16-   NEW EXP_TYPE CAL pipeline .cfg? (JWST only)
--------------------------------------------------------------

More Advanced
=============

17-   EXPRESSION DEFINING N/A (rmap_relevance expression e.g. DEADCORR!=
   "OMIT", OPTIONAL)
----------------------------------------------------------------------------

18-   PARAMETERS DEFINING N/A (cases with N/A as filename due to parameter
   value combinations which should be defined as having N/A results,
   OPTIONAL)
----------------------------------------------------------------------------

19-   PARAMETER VALUE SUBSTITUTIONS (e.g. GENERIC --> N/A, reference +
   displays as GENERIC, CRDS matches as N/A, OPTIONAL)
---------------------------------------------------------------------------

20-   CONDITIONAL PARAMETER VALUE SUBSTITUTIONS (e.g. WFC3 APERTURE group
   translations, rmap update time, "LRFWAVE == '3774.0'" : 'between 3710
   3826’, "DETECTOR=='HRC' and CCDGAIN=='-1'": '1.0|2.0|4.0|8.0’, OPTIONAL)
----------------------------------------------------------------------------
```

```
21-   UNIQUE ROW COLUMN NAMES (for table difference checking, OPTIONAL,
   tables)
----------------------------------------------------------------------------

22-   TABLE ROW LOOKUP COLUMN NAMES (for reprocessing optimization,
   OPTIONAL, tables)
----------------------------------------------------------------------------

23-   TABLE ROW LOOKUP ALGORITHM (for reprocessing optimization, OPTIONAL,
   tables)
----------------------------------------------------------------------------
```

Items:

1- JIRA Issue: The JIRA ISSUE that is being filed for the new reference type (e.g., CRDS-122). You might not know this the when you are filing the JIRA issue, so you can add it after submission.

2- FILE DESCRIPTION: A short description of the new file type for rmap and web submission documentation. For example, NIRSpec IRS2 Reference Pixel File

3- INSTRUMENT: Name of the instrument.

4- TYPE NAME: The reference file type; this will be the REFTYPE keyword for JWST references and will also be used as the 'filekind' entry in the rmap. Examples of this are: 'dark' or 'mask' for the Dark Frame or the Bad Pixel Mask, respectively.

5- LONG FORM TYPENAME: for website context display; e.g. "Interpixel Capacitance" for the ***_ipc_***.fits file or "Bad Pixel Mask" for the ***_mask_***.fits . This is the FILETYPE keyword in HST references. It will also be used as, or as part of, the name of the reference type that is seen on the CRDS website (hst-crds.stsci.edu or jwst-crds.stsci.edu).

6- EXTENSION / FILE FORMAT: The file format of the reference. CRDS can handle FITS, JSON, ASDF, and YAML formats (see https://jwst-crds.stsci.edu/static/users_guide/reference_conventions.html or https://hst-crds.stsci.edu/static/users_guide/reference_conventions.html).

7- SELECTION PARAMETERS: The list of keywords you want the reference to be selected on; e.g., DETECTOR, EXP_TYPE, FILTER, PUPIL.

8- REQUIRED: This is almost always YES. What's being asked is if a reference can't be found based on the values of the keywords used in the selection parameters provided in item 7, do you want CRDS to issue an error and stop any further processing? If YES, then CRDS will issue an error and exit from any further processing of the data. If NO, then there won't be an error issued and processing will continue.

9- DIRECTORY PATH: The directory where samples of the new reference type can be found.

10- INTIAL REFERENCE FILENAMES: The names of the files in the directory above that can be used to test CRDS. These do not need to be the references that will be delivered, but their headers, etc. need to be set up so that we can ensure that they will be accepted by CRDS.

11- FILE CREATOR: The person who generated the files above, or someone who can be contacted in case of questions.

12- TABLE (YES / NO?): Is the new reference a table? This is particularly important if the table has several rows from which the Calibration Pipeline will select depending on the observatioin mode.

13- PARAMETER NAME FOR CALIBRATION STEP (eg. HST BIASCORR): The keyword which will be used to trigger the calibration step where this reference will be used (e.g., BIASCORR, SINKCORR). This is for HST only.

14- OBSERVATORY: HST or JWST (at some point, WFIRST)

15- WHEN NOT USED (N/A): If the new reference type makes another reference type obsolete, list the obsolete reference. For example, the when the COS Team introduced their XWLKFILE and YWLKFILE, the WALKTAB reference was no longer needed and was removed from the COS imap (see Section 7.2 for more information on the imap files). In this case, WALKTAB would have been entered.

16- NEW EXP_TYPE CAL pipeline .cfg? (JWST only): The fundamental drivers for the CRDS reprocessing code are EXP_TYPE and CAL code version (CALVER); eventually, context may matter as well. When a new EXP_TYPE is introduced, CRDS reprocessing needs to know the calibration pipeline .cfg files expected to be used for routine calibration. This enables CRDS reprocessing to determine which CAL steps run, and from the steps, which reference types should be checked for potential reprocessing effects. If you are unsure what to put, please answer: Unknown.

    For more information about the calibration steps, refer to 'Stages of Processing' at: https://jwst-docs.stsci.edu/display/JDAT/Stages+of+Processing

17- EXPRESSION DEFINING OMMISION (rmap_relevance expression e.g. DEADCORR!= "OMIT", OPTIONAL): This is generally for HST references and is a single expression that defines when a reftype should be omitted and is generally used by HST to incorporate logic related to CORR keywords that was not included in the CDBS reference tables. There is only one expression per rmap, and it should be relatively simple. Examples:
    - DETECTOR == WFC and PCTECORR != OMIT (the DETECTOR has to be WFC and PCTECORR cannot be OMIT in order for the reference to be used).
    - DETECTOR == FUV (the reference can only be used for observations taken in the FUV detector).
    - LIFE_ADJ in ["3.0", "4.0"] (the LIFE_ADJ keyword has to be either 3.0 or 4.0 in order for the reference to be used).

18- PARAMETERS DEFINING OMMISSION (OPTIONAL): This is usually for JWST and is where an ordinary match case can return that it is not applicable for that step, and hence should not return a specific reference file. It can be as simple or complicated as need be, there can be more than one, and are specified using match case notations. They can also be seen in the operational context display on jwst-crds.stsci.edu.

    For example, for the NIRSPEC wavelengthrange reference type, the file is selected on the value of EXP_TYPE and the date and time of the observation. However, this reference is not applied to those data types that are AUTOFLATS (NRS_AUTOFLAT) and darks (NRS_DARK). These cases are denoted within the rmap as:

- 'NRS_AUTOFLAT' : 'N/A'
- 'NRS_DARK' : 'N/A'

which tells CRDS that this reference is omitted for these types of data.

19- RMAP PARAMETER VALUE SUBSTITUTIONS: For a given keyword used in selecting the reference, if CRDS sees this value, what should it match as. For example, one of the selection criteria for JWST refpix references is the value of the READPATT keyword. For NIRSpec, if the value of the keyword in the refpix reference is ALLIRS2, that file will be used for the cases where the READPATT of the data is either NRSIRS2 or NRSIRS2RAPID. The entry in the rmap for this looks like:

```
'substitutions' : {
      'META.EXPOSURE.READPATT' : {
         'ALLIRS2' : 'NRSIRS2|NRSIRS2RAPID',
      },
   },
```

Where the bar (|) indicates ‘or’. This is known as an unconditional substitution because any instance of it has exactly that translation, regardless of any other parameter values. This feature has the advantage that it can be used without changes to CRDS code, modifying only rmaps. It is limited to translating simple string values, so is not usable in all cases. Note that defining substitutions may require coordination with the CAL code data models and changes to schema.

For the most part, the above has been replaced by use of P_ keywords. In the example above, the following would be in the header (note the trailing bar):

```
P_READPA = ‘NRSIRS2|NIRSIRS2RAPID|’
```

20- CONDITIONAL PARAMETER VALUE SUBSTITUTIONS: The set of conditions one or more header keywords must meet in order for the reference to be applied. For example:

“DETECTOR==‘NRCA1’ and EXP_TYPE==‘NRC_IMAGE’”
“FILTER==‘F606W’”

21- UNIQUE ROW COLUMN NAMES (for table difference checking, OPTIONAL, tables): If the reference you’re defining is a table, list the column names you’d like CRDS to use in it’s comparison to the previous version of the reference (note, you do not need to list all of the column names). CRDS attempts to check for duplicate rows or rows accidentally deleted from a new version of a table. These column names are used to give rows an identity that can be used to discern if there is more than one instance, or if some combination that was present in the prior table is not present in the new version. Not all columns are used to define identity so that, for example, coefficients can be tweaked between versions of a table without making it appear that something has been deleted. This also is information that the ReDCaT Team will use to ensure that the reference table has all of the expected value combinations and also validating for completeness and duplication.

22- TABLE ROW LOOKUP COLUMN NAMES (for reprocessing optimization, OPTIONAL, tables): The name(s) of the column(s) that are most likely to change and would warrant the reprocessing of affected data.

    This information will be used to determine what data needs to be reprocessed, as well as avoid unnecessary reprocessing of data that was not affected by the delivery of the reference file.

23- TABLE ROW LOOKUP ALGORITHM (for reprocessing optimization, OPTIONAL, tables): The table row lookup algorithm can be used by CRDS reprocessing to determine which rows in a table affect a particular dataset parameter combination. CRDS will use the algorithm and dataset parameters to attempt to select the same table rows that would be used in calibration. This enables CRDS to determine if a table update affects a particular dataset based on the rows that were changed or if reprocessing is not needed (this is currently unused).

Once the above form is complete, attach it to your JIRA issue. While the issue is being worked, you may be contacted for details regarding the new reference file. During this time, the ReDCaT Team will also make any necessary changes to the scripts used in checking references.

Please be ready to provide samples of the files (location is included in the form). These do not have to be the actual files you plan to deliver, just something that can be used to ensure that the references will be accepted by CRDS.

Please wait until you are contacted by the ReDCaT Team before sending in your delivery request.

7 MAPPING FILES

This is a brief overview of the mapping files used by CRDS. For more, in depth information, please see the CRDS rules section of the CRDS users guide located at: https://jwst-crds.stsci.edu/static/users_guide/overview.html#crds-rules or https://hst-crds.stsci.edu/static/users_guide/overview.html#crds-rules

CRDS uses a series of mapping files to 'drill down' in order to find the best reference for a given reference using a set of rules provided by the instrument teams. The progression goes from the pmap, also known as the context (or pipeline or processing map), to the imap (or instrument map), to the rmap (rules or reference map).

7.1 The PMAP

The pmap contains the list of imaps associated with it. There can only be one pmap active within the calibration pipeline, but you can point to any existing pmap using one of the methods below. Note that you must be in a bash shell using the latest version of astroconda.

For JWST:

```
mycomputer: bash
$ source activate astroconda
$ conda update —-all
$ export CRDS_CONTEXT=jwst_XXXX.pmap
```

where XXXX is the version of the pmap you want to use. As your data are calibrated, CRDS will use the references associated with that pmap.

Here is an example of the JWST pmap:

```
header = {
    'derived_from' : 'jwst_0424.pmap',
    'description' : 'Hand edited to add FGS, MIRI, NIRCAM,
NIRISS PATHLOSS as N/A',
    'mapping' : 'PIPELINE',
    'name' : 'jwst_0425.pmap',
    'observatory' : 'JWST',
    'parkey' : ('META.INSTRUMENT.NAME',),
    'sha1sum' : 'ffbee5abae53de1cd8c027db0514ea0f10d51a2b',
}
selector = {
    'FGS' : 'jwst_fgs_0064.imap',
    'MIRI' : 'jwst_miri_0143.imap',
    'NIRCAM' : 'jwst_nircam_0093.imap',
    'NIRISS' : 'jwst_niriss_0094.imap',
    'NIRSPEC' : 'jwst_nirspec_0155.imap',
    'SYSTEM' : 'jwst_system_0015.imap',
}
```

For HST:

```
mycomputer: bash
$ source activate astroconda
$ conda update —-all
$crds bestrefs --new-context hst_XXXX.pmap --files *raw.fits --
update-bestrefs
```

where XXXX is the version of the pmap you want to use. This will update your raw files with the references associated with the pmap, after which you can run your calibration software.

Here is an example of the HST pmap:

```
header = {
    'derived_from' : 'hst_0599.pmap',
    'mapping' : 'PIPELINE',
    'name' : 'hst_0600.pmap',
    'observatory' : 'HST',
    'parkey' : ('INSTRUME',),
    'sha1sum' : 'eec507efea63760834aa257e80626e72c656e087',
}
selector = {
    'ACS' : 'hst_acs_0369.imap',
    'COS' : 'hst_cos_0291.imap',
    'NICMOS' : 'hst_nicmos_0250.imap',
    'STIS' : 'hst_stis_0294.imap',
    'WFC3' : 'hst_wfc3_0395.imap',
    'WFPC2' : 'hst_wfpc2_0250.imap',
}
```

In both examples, the 'parkey' value, INSTRUME for HST and META.INSTRUMENT.NAME for JWST, given as INSTRUME keyword in the data header, determines which imap CRDS needs to use.

7.2 The IMAP

The imap (instrument map) files contain a list of the rmaps for that particular instrument. For JWST, the imaps contain all reference file types, but only have rmaps listed for those types actually being used to calibrate data from that instrument, If a reference type is not used or needed for a particular calibration pipeline step, it will be listed as 'N/A' in the imap. For example, the BARSHADOW reference type rmap is listed as 'N/A' for NIRCAM. The imaps for the HST instruments contain only those rmaps relevant to that instrument (i.e., none of the listings are set to 'N/A').

NIRCAM imap (note how some reference types have rmap listings, while others are set to 'N/A'):

```
header = {
    'derived_from' : 'jwst_nircam_0092.imap',
    'instrument' : 'NIRCAM',
    'mapping' : 'INSTRUMENT',
    'name' : 'jwst_nircam_0093.imap',
    'observatory' : 'JWST',
    'parkey' : ('REFTYPE',),
    'sha1sum' : '576665acb77212e5b39535604b7acdf29a434be1',
}
selector = {
    'AREA' : 'jwst_nircam_area_0006.rmap',
    'BARSHADOW' : 'N/A',
    'CAMERA' : 'N/A',
    'COLLIMATOR' : 'N/A',
    'CUBEPAR' : 'N/A',
    'DARK' : 'jwst_nircam_dark_0012.rmap',
    'DFLAT' : 'N/A',
    'DISPERSER' : 'N/A',
    'DISTORTION' : 'jwst_nircam_distortion_0018.rmap',
    'DRIZPARS' : 'jwst_nircam_drizpars_0001.rmap',
    'EXTRACT1D' : 'N/A',
    'FFLAT' : 'N/A',
    'FILTEROFFSET' : 'N/A',
    'FLAT' : 'jwst_nircam_flat_0008.rmap',
    'FORE' : 'N/A',
    'FPA' : 'N/A',
    'FRINGE' : 'N/A',
    'GAIN' : 'jwst_nircam_gain_0008.rmap',
    'IFUFORE' : 'N/A',
    'IFUPOST' : 'N/A',
    'IFUSLICER' : 'N/A',
    'IPC' : 'jwst_nircam_ipc_0003.rmap',
    'LASTFRAME' : 'N/A',
    'LINEARITY' : 'jwst_nircam_linearity_0011.rmap',
    'MASK' : 'jwst_nircam_mask_0006.rmap',
    'MSA' : 'N/A',
    'MSAOPER' : 'N/A',
    'OTE' : 'N/A',
    'PATHLOSS' : 'N/A',
    'PERSAT' : 'jwst_nircam_persat_0004.rmap',
    'PHOTOM' : 'jwst_nircam_photom_0005.rmap',
    'READNOISE' : 'jwst_nircam_readnoise_0005.rmap',
    'REFPIX' : 'N/A',
    'REGIONS' : 'N/A',
    'RESET' : 'N/A',
    'RESOL' : 'N/A',
    'RSCD' : 'N/A',
    'SATURATION' : 'jwst_nircam_saturation_0006.rmap',
    'SFLAT' : 'N/A',
    'SPECWCS' : 'jwst_nircam_specwcs_0007.rmap',
    'STRAYMASK' : 'N/A',
    'SUPERBIAS' : 'jwst_nircam_superbias_0006.rmap',
    'THROUGHPUT' : 'N/A',
    'TRAPDENSITY' : 'jwst_nircam_trapdensity_0003.rmap',
    'TRAPPARS' : 'jwst_nircam_trappars_0003.rmap',
    'V2V3' : 'N/A',
    'WAVECORR' : 'N/A',
    'WAVELENGTHRANGE' : 'jwst_nircam_wavelengthrange_0005.rmap',
    'WCSREGIONS' : 'N/A',
}
```

As your JWST data progresses through the various calibration steps, CRDS will note the REFTYPE needed for that step. If the reference type listing contains an rmap, CRDS will open it and select the best reference to use based upon the rules it has. If the reference type is set to 'N/A', CRDS will not search for a reference file and the calibration step will not be performed.

The imap for ACS; notice that none of the reference types are listed as ’N/A':

```
header = {
    'derived_from' : 'hst_acs_0368.imap',
    'instrument' : 'ACS',
    'mapping' : 'INSTRUMENT',
    'name' : 'hst_acs_0369.imap',
    'observatory' : 'HST',
    'parkey' : ('REFTYPE',),
    'sha1sum' : 'abb4651030ff093f5f05d94141eeb41d4e80ff67',
}
selector = {
    'atodtab' : 'hst_acs_atodtab_0250.rmap',
    'biasfile' : 'hst_acs_biasfile_0314.rmap',
    'bpixtab' : 'hst_acs_bpixtab_0250.rmap',
    'ccdtab' : 'hst_acs_ccdtab_0250.rmap',
    'cfltfile' : 'hst_acs_cfltfile_0250.rmap',
    'crrejtab' : 'hst_acs_crrejtab_0250.rmap',
    'd2imfile' : 'hst_acs_d2imfile_0252.rmap',
    'darkfile' : 'hst_acs_darkfile_0322.rmap',
    'dgeofile' : 'hst_acs_dgeofile_0250.rmap',
    'drkcfile' : 'hst_acs_drkcfile_0335.rmap',
    'flshfile' : 'hst_acs_flshfile_0260.rmap',
    'idctab' : 'hst_acs_idctab_0254.rmap',
    'imphttab' : 'hst_acs_imphttab_0251.rmap',
    'mdriztab' : 'hst_acs_mdriztab_0252.rmap',
    'mlintab' : 'hst_acs_mlintab_0250.rmap',
    'npolfile' : 'hst_acs_npolfile_0252.rmap',
    'oscntab' : 'hst_acs_oscntab_0251.rmap',
    'pctetab' : 'hst_acs_pctetab_0253.rmap',
    'pfltfile' : 'hst_acs_pfltfile_0251.rmap',
    'shadfile' : 'hst_acs_shadfile_0250.rmap',
    'snkcfile' : 'hst_acs_snkcfile_0007.rmap',
    'spottab' : 'hst_acs_spottab_0250.rmap',
}
```

During generic conversion (creation of the raw FITS files, *raw.fits), CRDS will first check to see if a calibration step is to be done (####CORR set to PERFORM). If so, it will populate the filetype keywords (BIASFILE, DARKFILE, etc.) with the name of the best reference file to use based on the rules it has. If a calibration step will not be done (####CORR set to OMIT), CRDS will set the filetype keyword to 'N/A' (e.g., FLATFILE = 'N/A'), indicating that the particular reference type will not be used in calibrating the data.

7.3 The RMAP

The rmaps (rules or reference maps) are the heart of the CRDS reference file selection process (https://hst-crds.stsci.edu/static/users_guide/rmap_syntax.html#reference-mappings-rmap or https://jwst-crds.stsci.edu/static/users_guide/rmap_syntax.html#reference-mappings-rmap. They contain the parameter keywords CRDS uses in reference file selection, as well as the matching rules (combinations of selection keyword values); there could potentially be dozens of matching rules, depending on the number of valid values for a given keyword. rmaps can also be changed or updated in lieu of resubmitting a reference. For example, if a reference file has an incorrect USEAFTER date, it can be corrected within the rmap and the rmap delivered by the ReDCaT Team without having to redeliver a corrected file then deactivating the file with the problem. Note, changes to rmaps ARE NOT reflected in the reference files contained within them. Therefore, when delivering a reference that replaces another, it's best to check the rmap to ensure the new file is actually going to replace the file you want it to.

Example - NIRCAM gain rmap (note the number of matching rules due to the detector):

```
header = {
    'classes' : ('Match', 'UseAfter'),
    'derived_from' : 'jwst_nircam_gain_0007.rmap',
    'filekind' : 'GAIN',
    'instrument' : 'NIRCAM',
    'mapping' : 'REFERENCE',
    'name' : 'jwst_nircam_gain_0008.rmap',
    'observatory' : 'JWST',
    'parkey' : (('META.INSTRUMENT.DETECTOR', 'META.SUBARRAY.NAME'),
('META.OBSERVATION.DATE','META.OBSERVATION.TIME')),
    'sha1sum' : '38d644fe62a25698fc51ffe4492255d9c19a407f',
    'substitutions' : {
        'META.SUBARRAY.NAME' : {
            'GENERIC' : 'N/A',
         },
     },
}

selector = Match({
    ('NRCA1', 'GENERIC') : UseAfter({
        '1900-01-01 00:00:00' : 'jwst_nircam_gain_0019.fits',
        '2015-10-01 00:00:00' : 'jwst_nircam_gain_0045.fits',
    }),
    ('NRCA2', 'GENERIC') : UseAfter({
        '1900-01-01 00:00:00' : 'jwst_nircam_gain_0001.fits',
        '2015-10-01 00:00:00' : 'jwst_nircam_gain_0043.fits',
    }),
    ('NRCA3', 'GENERIC') : UseAfter({
        '1900-01-01 00:00:00' : 'jwst_nircam_gain_0020.fits',
        '2015-10-01 00:00:00' : 'jwst_nircam_gain_0046.fits',
    }),
    ('NRCA4', 'GENERIC') : UseAfter({
        '1900-01-01 00:00:00' : 'jwst_nircam_gain_0021.fits',
        '2015-10-01 00:00:00' : 'jwst_nircam_gain_0039.fits',
    }),
    ('NRCALONG', 'GENERIC') : UseAfter({
        '1900-01-01 00:00:00' : 'jwst_nircam_gain_0022.fits',
        '2015-10-01 00:00:00' : 'jwst_nircam_gain_0041.fits',
    }),
    ('NRCB1', 'GENERIC') : UseAfter({
        '1900-01-01 00:00:00' : 'jwst_nircam_gain_0023.fits',
        '2015-10-01 00:00:00' : 'jwst_nircam_gain_0038.fits',
    }),
    ('NRCB2', 'GENERIC') : UseAfter({
        '1900-01-01 00:00:00' : 'jwst_nircam_gain_0024.fits',
        '2015-10-01 00:00:00' : 'jwst_nircam_gain_0042.fits',
    }),
    ('NRCB3', 'GENERIC') : UseAfter({
        '1900-01-01 00:00:00' : 'jwst_nircam_gain_0025.fits',
        '2015-10-01 00:00:00' : 'jwst_nircam_gain_0047.fits',
    }),
    ('NRCB4', 'GENERIC') : UseAfter({
        '1900-01-01 00:00:00' : 'jwst_nircam_gain_0026.fits',
        '2015-10-01 00:00:00' : 'jwst_nircam_gain_0040.fits',
    }),
    ('NRCBLONG', 'GENERIC') : UseAfter({
        '1900-01-01 00:00:00' : 'jwst_nircam_gain_0027.fits',
        '2015-10-01 00:00:00' : 'jwst_nircam_gain_0044.fits',
    }),
})
```

Example - COS gsag table rmap:

```
header = {
    'derived_from' : 'hst_cos_gsagtab_0252.rmap',
    'filekind' : 'GSAGTAB',
    'instrument' : 'COS',
    'mapping' : 'REFERENCE',
    'name' : 'hst_cos_gsagtab_0253.rmap',
    'observatory' : 'HST',
    'parkey' : (('DETECTOR', 'CENWAVE'), ('DATE-OBS', 'TIME-OBS')),
    'reffile_format' : 'TABLE',
    'reffile_required' : 'NONE',
    'reffile_switch' : 'NONE',
    'rmap_relevance' : '(DETECTOR == "FUV")',
    'sha1sum' : '5ecbecf43a9c202f536a4817b39c02319c87f87a',
}
selector = Match({
    ('FUV', 'BETWEEN 1055 1097') : UseAfter({
        '2009-05-11 00:00:00' : 'zbn1927fl_gsag.fits',
    }),
    ('FUV', 'N/A') : UseAfter({
        '2009-05-11 00:00:00' : 'zbn1927gl_gsag.fits',
    }),
})
```

As can be seen above, an rmap is separated into two sections: the header, which contains information about the rmap and the reference type it refers to, and the selector, which has the matching rules and the reference files associated with them.

In the header, filekind is the reference file type, instrument and observatory are self explanatory, and mapping is always set to REFERENCE. The most important entry in the header is ‘parkey’, which is the set of parameter keywords used in selecting a reference file. The parkeys are set up differently for each of the telescopes. For JWST, the parkeys are given with their data model (metadata) names:

```
'parkey' : (('META.INSTRUMENT.DETECTOR', 'META.SUBARRAY.NAME'),
('META.OBSERVATION.DATE', 'META.OBSERVATION.TIME')),
```

While for HST, the header keywords themselves are given:

```
'parkey' : (('CCDAMP', 'CCDGAIN', 'CCDOFFST', 'BINAXIS1', 'BINAXIS2'),
('DATE-OBS', 'TIME-OBS'))
```

The parkeys `META.OBSERVATION.DATE` and `META.OBSERVATION.TIME` for JWST, and `DATE-OBS` and `TIME-OBS` for HST are combined by CRDS to form a date-time value. The date-time from the data is compared to the USEAFTERs of the reference files. The reference file with the USEAFTER that is closest to, but less than, the date-time, will be the one selected, provided the other matching criteria are met.

The selector area within the rmap contains the matching rules and their associated reference files, and the information here can be changed or updated, provided CRDS has knowledge of it. For example, a reference file can be replaced with one that was previously delivered. In the NIRCAM gain rmap shown above the entry

```
 ('NRCBLONG', 'GENERIC') : UseAfter({
        '1900-01-01 00:00:00' : 'jwst_nircam_gain_0027.fits',
        '2015-10-01 00:00:00' : 'jwst_nircam_gain_0044.fits',
```

could be changed to

```
    ('NRCBLONG', 'GENERIC') : UseAfter({
        '1900-01-01 00:00:00' : 'jwst_nircam_gain_0030.fits',
        '2015-10-01 00:00:00' : 'jwst_nircam_gain_0044.fits',
```

provided jwst_nircam_gain_0030.fits had previously been delivered.

You could also add a new matching rule. Say for the COS gsag rmap above, you wanted to add an entry for CENWAVE = 1250 using the reference file for CENWAVE BETWEEN 1055 1097. The rmap would become

```
header = {
    'derived_from' : 'hst_cos_gsagtab_0252.rmap',
    'filekind' : 'GSAGTAB',
    'instrument' : 'COS',
    'mapping' : 'REFERENCE',
    'name' : 'hst_cos_gsagtab_0253.rmap',
    'observatory' : 'HST',
    'parkey' : (('DETECTOR', 'CENWAVE'), ('DATE-OBS', 'TIME-OBS')),
    'reffile_format' : 'TABLE',
    'reffile_required' : 'NONE',
    'reffile_switch' : 'NONE',
    'rmap_relevance' : '(DETECTOR == "FUV")',
    'sha1sum' : '5ecbecf43a9c202f536a4817b39c02319c87f87a',
}

selector = Match({
    ('FUV', 'BETWEEN 1055 1097') : UseAfter({
        '2009-05-11 00:00:00' : 'zbn1927fl_gsag.fits',
    }),
    (‘FUV’, ‘1250’) : UseAfter({
        ‘2009-05-11 00:00:00’ : 'zbn1927fl_gsag.fits',
    }),
    ('FUV', 'N/A') : UseAfter({
        '2009-05-11 00:00:00' : 'zbn1927gl_gsag.fits',
    }),
})
```

provided 1250 is a valid value for CENWAVE for the gsag reference type (regardless whether it’s valid for any other reference type). Remember, any changes made to an rmap ARE NOT refelected in the reference file, so make sure to double check the rmap before sending in a replacement. To update an rmap, file a JIRA issue (https://jira.stsci.edu) against CRDS, assign it to Matt McMaster, and either update the rmap yourself and attach it to the issue, or explain in detail what needs to be changed and the ReDCaT Team will update the rmap. Please make sure you give as much detail as possible as to why the rmap needs to be updated.

If it turns out that CRDS does not know of the value of a given parameter keyword for the reference type, file a JIRA issue against CRDS, assign it to Matt McMaster and share it with the full ReDCaT team (redcat@stsci.edu), and either explain in detail what needs to be changed in CRDS (USEAFTERs need to match a format, so this doesn't apply here). Once the necessary changes have been made, the rmap can be updated either by opening a separate JIRA issue, or as an addition to the one requesting the change to CRDS.

Occasionally, you might see a list of parameter values separated by the '|' symbol (barred-or). Examples of this are:

```
        ('WFC','AC|BD','1.0|2.0|4.0|8.0','N/A',4144,2068,24.0,0.0, 'N/A',
'N/A','N/A') : UseAfter({
                '1991-01-01 00:00:00' : 'kcb17346j_bia.fits',
        }),
```

and

```
        ('NRCA2', 'MASKA210R|SUB640A210R') : UseAfter({
                '2015-06-01 00:00:00' : 'jwst_nircam_dark_0073.fits',
        }),
```

In these cases, the reference file will be used for any of values listed. In the ACS example above, that particular bias would be used for the AC or BD subarrays, with a gain of 1 or 2 or 4 or 8. The NIRCAM dark would be used for either the MASKA210R or SUB640A210R subarrays. The barred-or values are generated differently depending on the telescope. For HST, the barred-or values are produced when CRDS encounters a wildcard value for a keyword. The wildcard value is expanded according to rules within CRDS based on instrument, keyword, and reference type. JWST barred-or values come about from the use of P_ keywords, where the barred-or values are listed as the value of the keyword, with a trailing bar ('|') being used to designate the end of the entry.

For example, the NIRISS SUB256 subarray superbias for the NIS detector can be used for either the NIS or NISRAPID read patterns. The matching rule and reference entry look like the following:

```
    ('NIS', 'NIS|NISRAPID', 'SUB256') : UseAfter({
          '2015-10-01 00:00:00' :'jwst_niriss_superbias_0009.fits',
          '2015-11-01 00:00:00' :'jwst_niriss_superbias_0081.fits',
    }),
```

To get the barred-or value for READPATT, the references had to have the following in their headers:

```
P_READPA = NIS|NISRAPID|
```

When the rmap is generated, the trailing bar is omitted. As long as each of the individual values within the barred-or is valid, CRDS will OK the file. In other words, there are no set values for the P_ keywords that CRDS uses for comparison, it only checks the individual values to ensure they are valid. However, during rmap certification, CRDS compares the new rmap to the current one, and if it finds any differences, a warning will be issued and you will be contacted by the ReDCaT Team if this difference hasn't been explained in your delivery form.

You may also see the value of a keyword set to ‘N/A’ within a matching rule. For example, the following is the first matching rule for MIRI dark files, which get selected on DETECTOR, READPATT, SUBARRAY, and USEAFTER:

```
selector = Match({
    ('MIRIFULONG', 'N/A', 'FULL') : UseAfter({
        '1900-01-01 00:00:00' : 'jwst_miri_dark_0029.fits',
        '2015-08-01 00:00:00' : 'jwst_miri_dark_0050.fits',
    }),
```

In CRDS, ‘N/A’ within a matching rule is known as a ‘weak match’, while a specific value is known as a ‘strong match’ and would take precedence in reference file selection. In the above example, the ‘N/A’ means that given the values of DETECTOR, SUBARRAY, and USEAFTER, if the value of READPATT within the data is not applicable to any of the other matching rules contained within the rmap, use this one.

However, if the filename associated with a matching rule is set to ‘N/A’, CRDS will not search for a reference file and that particular calibration step will be skipped. For example, the NIRCam SPECWCS reference is selected on PUPIL, MODULE, EXP_TYPE, and USEAFTER. The entry in the rmap for PUPIL=GRISMC, MODULE=A, and EXP_TYPE=NRC_TSGRISM is:

```
('GRISMC', 'A', 'NRC_TSGRISM') : 'N/A',
```

Which means for these data, the SPECWCS reference will not be used, and that calibration step will be skipped.

ACKNOWLEDGEMENTS

The author would like to thank Todd Miller for the information on CRDS he provided, as well as members of the ReDCaT Team for their suggestions and for proofreading this document.