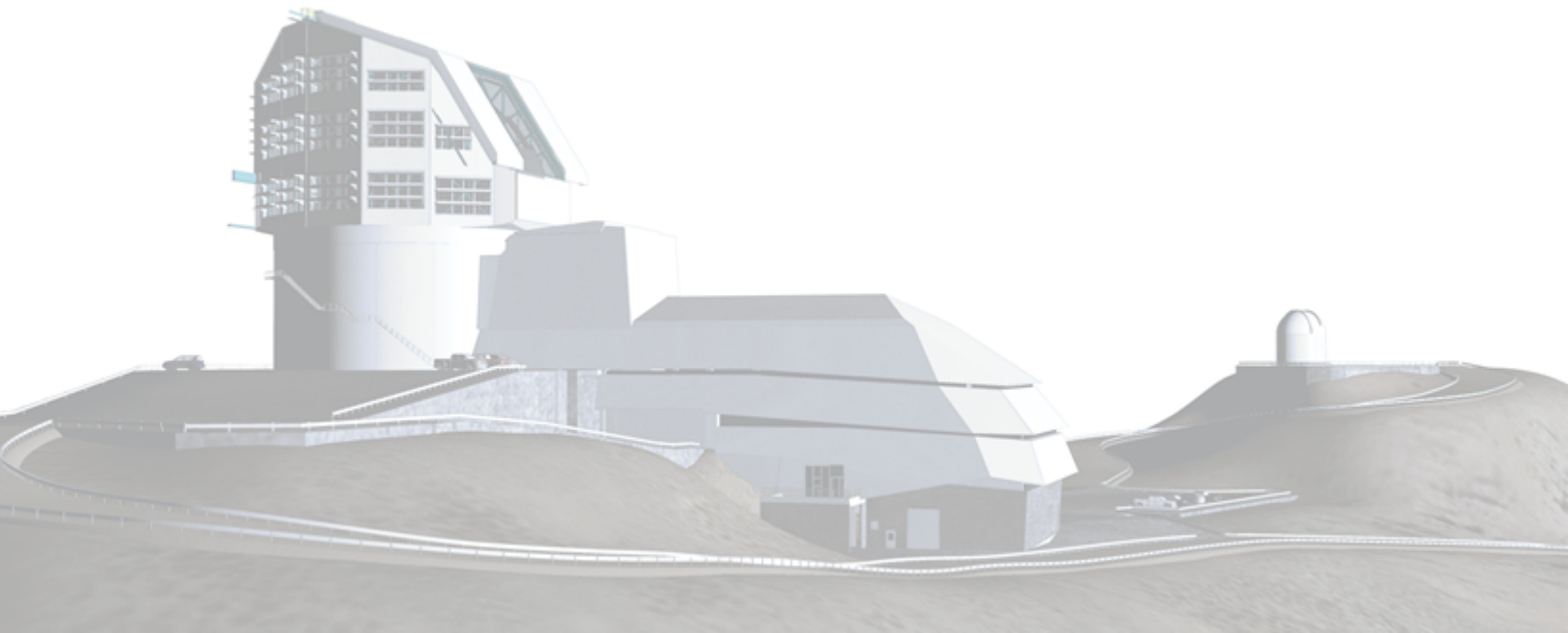

Vera C. Rubin Observatory
Data Management

Convention for identifying bits in a mask/flags image in FITS

Gregory Dubois-Felsmann

DMTN-252

Latest Revision: 2023-02-28

Abstract

Describes a convention used by the Vera C. Rubin Observatory and other projects for the identification of individual bits in a ”flags image” or ”mask image” - an integer-valued image in which individual bit planes are assigned to represent a set of Boolean values associated with individual pixels in an accompanying main image. The convention supplies a symbolic name for each bit plane, and optionally a description string. This convention applies to the serialization of such a ”flags image” in FITS.

Change Record

Version	Date	Description	Owner name
1	YYYY-MM-DD	Unreleased.	Gregory Dubois-Felsmann

Document source location: `https://github.com/lsst-dm/dmtn-252`

Contents

Convention for identifying bits in a mask/flags image in FITS

It is a common convention in astronomical imaging pipelines to annotate the main images[1] being processed with ancillary pixel arrays of the same shape as the main image, conveying additional information about each pixel. These additional arrays can convey quantitative information such as a variance or weight for each pixel, but it is often also useful to provide Boolean "flags" on a per-pixel basis.

Such flags may provide a variety of information about the progress and success or failure of the processing of the main image. Common meanings of such flags range from error states conveying "this pixel's value in the main image is bad and should not be used for further processing or science" to purely informational states such as "this pixel is part of a detected extended source". Multiple flags may be associated with a single pixel by bit-packing them into an integer at designated bit positions. Note that, because of the nature of thie representation, if compression is used for such an image, it must in general be lossless.

In some cases this "flags image" or "mask image" (both terms are in general use in the community) is provided as an additional extension in a FITS serialization of the main image, while in other cases it may be delivered as a separate file.

The data documentation for projects that provide such data will generally provide a list of flags by bit position and purpose or interpretation.

Well-established conventions of software engineering suggest that the specific bit positions not be "hard-coded" into project software as bare constants, either as a bit position (e.g., 7) or as a mask (e.g., `0x80`). Symbolic names should be used instead. This can be, and has been, done in a variety of ways on different projects, from the use of C-style preprocessor constants to the use of dictionary-like mechanisms.

Even if this is done, however, there is still the likelihood that the persisted file artifacts for such "mask images" cannot be readily understood by an end user without recourse to the documentation for the table of bit number mappings to meanings. This can be made more

[1] This term is used herein to mean whatever image is the principal concern of the processing — a raw instrumental image, a calibrated instrumental image, a derived data product, etc. Often the pixel values of a "main image" may represent calibrated or uncalibrated fluxes, but that is not relevant to the discussion in this paper.

complicated by the likelihood that different categories of images created by a project (e.g., single-epoch images, coadded images, and image-like calibration data products) will use different sets of flags and therefore have different mappings.

Furthermore, when images are displayed, it is highly desirable for image visualization tools to be capable of displaying the flags, for instance as semi-transparent overlays on the associated main images, and to allow users to interrogate and manipulate the visualization using the symbolic names for the flag bit positions, not just their bit numbers.

Within several projects, enumerated below, it has therefore been seen as valuable to have a convention for encoding the symbolic names for the flag bit positions in the persisted image files.

This document describes a convention for doing so in FITS persistence, and was written expressly in order to document this in the FITS community's registry of conventions. The authors would welcome recommendations for the representation of equivalent information in other persistence formats for astronomical images, e.g., in ASDF. A corresponding convention for the annotation of bit positions in integer-valued table columns would also be of value to the community.

This convention has its origins in internal data products produced within the Sloan Digital Sky Survey (SDSS) imaging pipelines (though to our knowledge it never appeared in publicly released SDSS images). It has been in use in the Vera C. Rubin Observatory's imaging pipelines since at least 20xx, and is available in the Rubin "Data Previews" that were opened to the community in 2021–2022.

The convention is also in use in the publicly released data from the Hyper-Suprime-Cam (HSC) project, and is being used by the NASA SPHEREx space telescope (to be launched in 2025) image processing pipelines.

The open-source Firefly astronomical data visualization package from Caltech/IPAC, used in the NASA Infrared Science Archive (IRSA), NASA Extragalactic Database (NED), and the NASA Exoplanet Archive, as well as in the Rubin Science Platform, supports the display and manipulation of "mask images" based on the convention presented herein.

We are registering this convention in the hope that other projects and missions in the inter-

national astronomy community will choose to adopt it for their own data, and that the community will come to support it in a wider variety of visualization tools and software packages.

1 Explanation of the Convention

The principal purpose of this document is to describe a method for including symbolic names for bit positions in the headers of an individual integer-pixel-value image extension in FITS. We believe this alone to be of significant value and widespread applicability in the community. We also describe below a convention for how to structure a multi-extension FITS file to indicate the association of a "mask image" extension, with such headers, with one or more other image extensions, but this is of secondary importance.

The core of this convention is the representation as FITS headers of a mapping between bit positions and short symbolic names for those bit positions. We also describe an optional mechanism for associating a longer documentation string with each symbolic name.

An example of a mapping is shown in Table X.

X

In this example, the convention would then lead to the inclusion of the following headers in the FITS extension containing the bit-packed integer image containing the flags:

```
MP_X     = 0

HIERARCH MP_CROSSTALK = 1
```

The convention incorporates the use of the ESO `HIERARCH` registered FITS convention, relying on the statement in that document that "The tokens may, however, be longer than the 8 character limit of formal FITS keywords" in order to support symbolic names that are longer than 5 characters. The convention does not mandate the use of `HIERARCH` for *all* `MP_*` keywords, despite the superficial consistency that might offer, in order to allow the present convention to be used even in contexts in which `HIERARCH` is not supported, if the names are limited to 5 characters.

In the existing practice for the use of this convention, e.g., in the Rubin Observatory software, the same mapping applies to every image in the same category — e.g., all calibrated single-epoch images would share the same flag bit positions. However, note that that is not a requirement of this convention, as the entire point is to allow each such image to be understood in isolation, based only upon the headers actually present.

1.1 Optional Support for Documentation Strings

1.2 Suggestions for Implementation

Community-facing libraries wishing to provide support for this convention should permit users to query the status of individual planes based on their symbolic names alone. Note that as the name-to-bit-position mapping is image metadata, in such a library the mapping cannot be represented by compile-time constants (e.g., C preprocessor macros).

Image visualization software supporting this convention should provide at least the following features:

Display the mapping itself.

Allow a user to specify, by their symbolic names, which flags (i.e., which bit positions) should be visible in the visualization.

Display the names of the flags which are ”true” (bit value 1) for a given pixel in the same contexts in which the software displays what the pixel value of a non-flags image is.

Support the visualization of a “mask image” in conjunction with an associated main image (e.g., a flux image), including both in the image display itself (perhaps as a semi-transparent overlay), and in whatever UI element the tool uses to display the ordinary pixel values (e.g., fluxes) of the main image.

2 Formal Statement of the Convention

This section describes the behavior of "conforming implementations"; i.e., software that supports either reading or writing bit-packed Boolean flags as image pixel data and that provides behavior aligned with this convention. The standards-vocabulary terms "shall", "should", "may", "should not", and "must not" in this section should be interpreted as relative to that goal.

2.1 Writing

A conforming implementation shall support the input and/or output of data in the FITS image

A conforming implementation **must not** output negative bit position numbers in the `MP_*` headers, or bit position numbers exceeding the available number of bits in the extension's pixel values.

A conforming implementation **must not** use lossy compression on a FITS extension containing bit-packed pixel flags.

2.2 Reading

A conforming implementation shall support the input of data in the FITS image extension formats (image or compressed image) ...

A conforming implementation shall treat an extension as containing bit-packed pixel flags if a) it is an integer image or compressed image extension, b) it has one or more headers beginning with `MP_` or `HIERARCH MP_` with FITS-header-key conformant names and non-negative integer values, c) the integer values of such headers do not exceed the number of bits in the integer pixel values of the extension, and d) any compression used internal to the extension is lossless.

A conforming implementation **may** process extensions in this way even if `MP_` headers are present with integer values exceeding the bit width of tthe pixel values, but **should** issue a warning in such situations.

A conforming implementation **must** issue an error if a compressed image extension otherwise meeting the above criteria was compressed with a lossy algorithm.

3 Survey of Existing Practice

A References

B Acronyms

Acronym	Description
DM	Data Management