Next Up Previous Contents FITSIO Home
Next: 4.5 Table I/O Routines **Up:** 4 CFITSIO Routines **Previous:** 4.3 HDU-level Routines **Contents**

4.4 Image I/O Routines

This section lists the more important CFITSIO routines which operate on FITS images.

```
int fits_get_img_type(fitsfile *fptr, int *bitpix, int *status)
int fits_get_img_dim( fitsfile *fptr, int *naxis,  int *status)
int fits_get_img_size(fitsfile *fptr, int maxdim,  long
*naxes,
                      int *status)
int fits_get_img_param(fitsfile *fptr, int maxdim,  int
*bitpix,
                       int *naxis, long *naxes, int *status)
```

Get information about the currently opened image HDU. The first routine returns the datatype of the image as (defined by the `BITPIX` keyword), which can have the following symbolic constant values:

```
BYTE_IMG     =   8   ( 8-bit byte pixels, 0 - 255)
SHORT_IMG    =  16   (16 bit integer pixels)
LONG_IMG     =  32   (32-bit integer pixels)
LONGLONG_IMG =  64   (64-bit integer pixels)
FLOAT_IMG    = -32   (32-bit floating point pixels)
DOUBLE_IMG   = -64   (64-bit floating point pixels)
```

(+)

The second and third routines return the number of dimensions in the image (from the `NAXIS` keyword), and the sizes of each dimension (from the `NAXIS1, NAXIS2`, etc. keywords). The last routine simply combines the function of the first 3 routines. The input `maxdim` parameter in this routine gives the maximum number dimensions that may be returned (i.e., the dimension of the `naxes` array)

```
int fits_create_img(fitsfile *fptr, int bitpix, int naxis,
        long *naxes, int *status)
```

Create an image HDU by writing the required keywords which define the structure of the image. The 2nd through 4th parameters specified the datatype, the number of dimensions, and the sizes of the dimensions. The allowed values of the `bitpix` parameter are listed above in the description of the `fits_get_img_type` routine. If the FITS file pointed to by `fptr` is empty (previously created with `fits_create_file`) then this routine creates a primary array in the file, otherwise a new IMAGE extension is appended to end of the file following the other HDUs in the file.

```
int fits_write_pix(fitsfile *fptr, int datatype, long *fpixel,
        long nelements, void *array, int *status);

int fits_write_pixnull(fitsfile *fptr, int datatype, long
*fpixel,
        long nelements, void *array, void *nulval, int
*status);

int fits_read_pix(fitsfile *fptr, int  datatype, long *fpixel,
```

(+)

```
long nelements, void *nulval, void *array,
int *anynul, int *status)
```

Read or write all or part of the FITS image. There are 2 different 'write' pixel routines: The first simply writes the input array of pixels to the FITS file. The second is similar, except that it substitutes the appropriate null pixel value in the FITS file for any pixels which have a value equal to `*nulval` (note that this parameter gives the address of the null pixel value, not the value itself). Similarly, when reading an image, CFITSIO will substitute the value given by `nulval` for any undefined pixels in the image, unless `nulval = NULL`, in which case no checks will be made for undefined pixels when reading the FITS image.

The `fpixel` parameter in these routines is an array which gives the coordinate in each dimension of the first pixel to be read or written, and `nelements` is the total number of pixels to read or write. `array` is the address of an array which either contains the pixel values to be written, or will hold the values of the pixels that are read. When reading, `array` must have been allocated large enough to hold all the returned pixel values. These routines starts at the `fpixel` location and then read or write the `nelements` pixels, continuing on successive rows of the image if necessary. For example, to write an entire 2D image, set `fpixel[0] = fpixel[1] = 1`, and `nelements = NAXIS1 * NAXIS2`. Or to read just the 10th row of the image, set `fpixel[0] = 1, fpixel[1] = 10`, and `nelements = NAXIS1`. The `datatype` parameter specifies the datatype of the C `array` in the program, which need not be the same as the datatype of the FITS image itself. If the datatypes differ then CFITSIO will (+)

convert the data as it is read or written. The following symbolic constants are allowed for the value of `datatype`:

```
TBYTE     unsigned char
TSBYTE    signed char
TSHORT    signed short
TUSHORT   unsigned short
TINT      signed int
TUINT     unsigned int
TLONG     signed long
TLONGLONG signed 8-byte integer
TULONG    unsigned long
TFLOAT    float
TDOUBLE   double
```

```
int fits_write_subset(fitsfile *fptr, int datatype, long
*fpixel,
        long *lpixel, DTYPE *array, > int *status)

int fits_read_subset(fitsfile *fptr, int  datatype, long *fpixel,
        long *lpixel, long *inc, void *nulval,  void *array,
        int *anynul, int *status)
```

Read or write a rectangular section of the FITS image. These are very similar to `fits_write_pix` and `fits_read_pix` except that you specify the last pixel coordinate (the upper right corner of the section) instead of the number of pixels to be read. The read routine also has an `inc` parameter which can be used to read only every `inc-th` pixel along each dimension of the image. Normally `inc[0] = inc[1] = 1` to read every pixel in a 2D image. To read every other pixel in the entire 2D image, set

(+)

fpixel[0] = fpixel[1] = 1
lpixel[0] = {NAXIS1}
lpixel[1] = {NAXIS2}
inc[0] = inc[1] = 2

Or, to read the 8th row of a 2D image, set

fpixel[0] = 1
fpixel[1] = 8
lpixel[0] = {NAXIS1}
lpixel[1] = 8
inc[0] = inc[1] = 1

Next Up Previous Contents FITSIO Home
Next: 4.5 Table I/O Routines **Up:** 4 CFITSIO Routines **Previous:** 4.3 HDU-level Routines **Contents**

(+)