

5.6. *CBFlib*: an ANSI C library for manipulating image data

BY P. J. ELLIS AND H. J. BERNSTEIN

5.6.1. Introduction

CBFlib is a library of ANSI C functions providing a simple mechanism for accessing crystallographic binary files (CBFs) and image-supporting CIF (imgCIF) files (see Chapters 2.3, 3.7 and 4.6). The *CBFlib* application programming interface (API) consists of a set of low-level functions and a set of high-level functions. The low-level functions are loosely based on the *CIFPARSE* (Tosic & Westbrook, 2000) API for mmCIF files. As in *CIFPARSE*, the low-level functions in *CBFlib* do not perform any semantic integrity checks and simply create, read, modify and write files conforming to the CIF syntax, with additional functionality for working with binary sections. These basic functions are independent of the details of the CBF/imgCIF dictionary and can be used to manage any CIF data set. In contrast, the high-level functions are based on the CBF/imgCIF dictionary and facilitate writing or reading commonly used entries to or from CBF and imgCIF data files.

External to a program, a CBF/imgCIF data set ‘lives’ in a file. Internally, when managed by *CBFlib*, a CBF or imgCIF data set has a simple tree structure pointed to by a ‘handle’ (Fig. 5.6.1.1). At the highest level are named data blocks. Each data block may contain a number of named categories. Within each category, the actual data entries are stored in tabular form with named columns and numbered rows. The numbers of rows in different columns of a given category are constrained by the software to be the same.

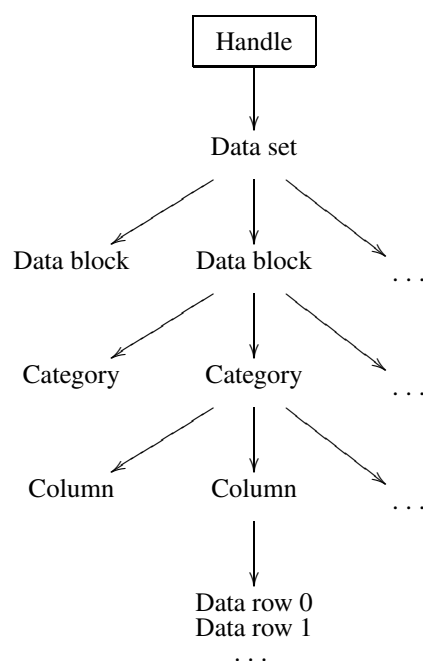


Fig. 5.6.1.1. Schematic data structure of a CBF or imgCIF data set accessed through a handle.

Table 5.6.1.1. Return values from *CBFlib* functions

Error code	Hexadecimal value	Meaning
CBF_FORMAT	1	Invalid file format
CBF_ALLOC	2	Memory allocation failed
CBF_ARGUMENT	4	Invalid function argument
CBF_ASCII	8	Value is ASCII (not binary)
CBF_BINARY	10	Value is binary (not ASCII)
CBF_BITCOUNT	20	Expected number of bits does not match the actual number written
CBF_ENDOFDATA	40	End of the data reached before end of array
CBF_FILECLOSE	80	File close error
CBF_FILEOPEN	100	File open error
CBF_FILEREAD	200	File read error
CBF_FILESEEK	400	File seek error
CBF_FILETELL	800	File tell error
CBF_FILEWRITE	1000	File write error
CBF_IDENTICAL	2000	A data block with the new name already exists
CBF_NOTFOUND	4000	Data block, category, column or row does not exist
CBF_OVERFLOW	8000	Number read cannot fit into the destination argument; destination has been set to the nearest value
CBF_UNDEFINED	10000	Requested number not defined
CBF_NOTIMPLEMENTED	20000	Requested functionality is not yet implemented

CBFlib provides functions to create a corresponding data structure in memory; to copy a data set from an external file to the data structure or from the data structure to an external file; to navigate the tree; to scan, add and remove data blocks within data sets, categories (tables) within data blocks, and rows or columns within categories; to read or modify data entries; and finally to delete the structure from memory.

As is common in C programming, all functions return an integer equal to 0 for success or an error code for failure. The *CBFlib* error codes are given in Table 5.6.1.1.

CBFlib is thread-safe, re-entrant and able to operate on multiple data sets at a time. This means that the library maintains no static data and that the object to be operated on must be passed to each function. In *CBFlib*, this is accomplished by referring to each data set in memory with a unique handle of type `cbf_handle`. The handle maintains a link to the data structure in memory as well as the current location on the tree (data block, category, column and row). Before reading or creating a data set, the handle is created by calling the `cbf_make_handle` function. When the data set is no longer required, the resources associated with it can be freed using `cbf_free_handle`. Most functions in the library expect a handle as the first argument.

CBF binary data files and imgCIF ASCII data files may have one or more large images or other data sections as values for CIF tags. The focus of *CBFlib* is to handle large data sections efficiently.

The basic flow of an application reading CBF/imgCIF data with the low-level *CBFlib* functions is shown in Fig. 5.6.1.2.

Affiliations: PAUL J. ELLIS, Stanford Linear Accelerator Center, 2575 Sand Hill Road, Menlo Park, CA 94025, USA; HERBERT J. BERNSTEIN, Department of Mathematics and Computer Science, Kramer Science Center, Dowling College, Idle Hour Blvd, Oakdale, NY 11769, USA.