J. Appl. Cryst. (1993). **26**, 474–479

CIF Applications. II. *CIFIO*: for CIF Input/Output in the *Xtal* System*

BY SYDNEY R. HALL

Crystallography Centre, University of Western Australia, Nedlands 6009, *Australia*

(*Received* 12 *December* 1992; *accepted* 15 *January* 1993)

Abstract

CIFIO is the primary CIF input and output program in the *Xtal* System and provides for global data exchange and machine-readible publication submission. *CIFIO* also creates and reads an *Xtal* archive file which is a CIF-like ASCII replica of an *Xtal* binary file. *CIFIO* is written in Ratmac and may be implemented, along with the other 94 calculations in the *Xtal*3.2 package, on any computer.

Introduction

The capabilities of *CIFIO* are unique amongst current CIF applications. It both creates and reads standard CIFs. That is, it is able to convert the internal *Xtal* binary data files into a CIF, and perform the reverse. The conversion table used to map *Xtal* data into CIF data exists as an external file which can be easily modified without changing the *CIFIO* software. In addition, *CIFIO* is able to create and read a CIF-like replica of an *Xtal* binary data file. This file, referred to as an *Xtal* archive file, provides a compact machine-independent and efficient method of transmitting data to other sites and machines where *Xtal* is used.

This paper is the second in a series describing CIF applications. The development of *CIFIO* started shortly after that of *QUASAR* (Hall & Sievers, 1993) and it was the first program to create an *ab initio* CIF and to convert CIF data into binary data used in a calculation. *CIFIO* was developed in 1989 and distributed with the *Xtal*3.0 System (Hall & Stewart, 1990). Since then it has undergone further generalization and forms the basis for the *Xtal*3.2 nucleus routines (Hall, Flack & Stewart, 1992), which read and write *CIFIO*-type archive files as part of the automatic backup feature for initializing and closing down a calculation sequence.

This paper is intended to provide detailed information on how to add software to a large package software for reading and writing CIF data. Of particular importance to developers is the approach of externalizing the information that maps the CIF data definitions onto the internal data representations. This is a powerful and flexible attribute for package software.

* This paper is one of a series of papers on CIF applications. Offprints are available from The Technical Editor, 5 Abbey Square, Chester CH1 2HU, England. See text of paper for availability of program(s) by email.

0021-8898/93/030474-06$06.00

Functions

CIFIO operates in four distinct modes. They are:

1. the creation of a CIF from internal data representation;
2. the conversion of a CIF into internal data;
3. the creation of an *Xtal*-specific STAR file from internal data;
4. conversion of an *Xtal*-specific STAR file into internal data;

The algorithms for modes 1 and 3 (or for modes 2 and 4) might be expected to be the same. This is definitely not the case. An important lesson from this development is that processing CIF data is very different to that of archiving non-global data. The differences in these approaches are summarized briefly here and discussed in more detail in the next section.

Modes 1 and 2 provide the interface between the external standard CIF data and the internal *Xtal* data. In many instances, there is no direct mapping between these two data representations. Data needed by *Xtal* may be missing from a CIF and *vice versa*. Also, the definition of the CIF data may vary from that of the *Xtal* internal representation. These are typical of the protocol problems which occur when communicating data between independent data systems and they necessitate an appropriate response in software design. In these modes, *CIFIO* makes no assumptions about the accessibility of specific data items (in either direction) and uses a general procedure to map data relationships across the CIF–*Xtal* boundary. The algorithms used to perform these two modes are described in the next section.

Modes 3 and 4 write and read a CIF-like archive file which is an exact ASCII representation of the *Xtal* binary data. In these modes, there is a direct mapping of data items between the binary file and the archive file and the algorithm is straightforward. It is worth emphasizing that, while the CIF created in mode 1 and the archive file created in mode 3 have a similar data structure, these files are not interchangeable, as the data names in an archive file do not conform to the CIF Core Dictionary definitions (Hall, Allen & Brown, 1991).

Algorithms

The algorithms used in the four *CIFIO* modes will now be described in detail.

Table 1. *An extract from the* `cifreq` *file showing non-loop data requests*

```
data_?

_audit_creation_date                    ?
_audit_creation_method                  ?
_audit_update_record                    ?

_chemical_compound_source               ?   #ActaC
_chemical_name_systematic               ?   #ActaC
_chemical_name_common                   ?   #ActaC
_chemical_formula_analytical            ?   #ActaC
_chemical_formula_moiety                ?   #ActaC
_chemical_formula_structural            ?   #ActaC
_chemical_formula_sum                   ?   #ActaC
_chemical_formula_weight                ?   #ActaC
_chemical_melting_point                 ?   #ActaC

_computing_data_collection              ?   #ActaC
_computing_cell_refinement              ?   #ActaC
_computing_data_reduction               ?   #ActaC
_computing_structure_solution           ?   #ActaC
_computing_structure_refinement         ?   #ActaC
_computing_molecular_graphics           ?   #ActaC
_computing_publication_material         ?   #ActaC

_cell_length_a                          ?   #ActaC
_cell_length_b                          ?   #ActaC
_cell_length_c                          ?   #ActaC
_cell_angle_alpha                       ?   #ActaC
_cell_angle_beta                        ?   #ActaC
_cell_angle_gamma                       ?   #ActaC
_cell_volume                            ?   #ActaC
_cell_formula_units_Z                   ?   #ActaC
_cell_measurement_temperature           ?   #ActaC
_cell_measurement_reflns_used           ?   #ActaC
_cell_measurement_theta_min             ?   #ActaC
_cell_measurement_theta_max             ?   #ActaC
_cell_special_details                       #ActaC
;
   ?
;

_symmetry_cell_setting                  ?   #ActaC
_symmetry_space_group_name_H-M          ?   #ActaC
_symmetry_space_group_name_Hall         ?   #ActaC
```

Table 2. *An extract from the* `cifreq` *file showing some looped data requests*

```
loop_
_atom_type_symbol                           #ActaC
_atom_type_description                      #ActaC
_atom_type_oxidation_number
_atom_type_number_in_cell
_atom_type_scat_dispersion_real             #ActaC
_atom_type_scat_dispersion_imag             #ActaC
_atom_type_scat_source                      #ActaC
      ?  ?  ?  ?  ?  ?  ?

loop_
_atom_site_label                            #ActaC
_atom_site_fract_x                          #ActaC
_atom_site_fract_y                          #ActaC
_atom_site_fract_z                          #ActaC
_atom_site_U_iso_or_equiv                   #ActaC
_atom_site_thermal_displace_type            #ActaC
_atom_site_calc_flag                        #ActaC
_atom_site_calc_attached_atom               #ActaC
      ?  ?  ?  ?  ?  ?  ?  ?

loop_
_atom_site_aniso_label                      #ActaC
_atom_site_aniso_U_11                       #ActaC
_atom_site_aniso_U_22                       #ActaC
_atom_site_aniso_U_33                       #ActaC
_atom_site_aniso_U_12                       #ActaC
_atom_site_aniso_U_13                       #ActaC
_atom_site_aniso_U_23                       #ActaC
      ?  ?  ?  ?  ?  ?  ?
```

Table 3. *An extract from the* `cifreq` *file showing data items which must be added manually*

Data items needed for submission to *Acta Crystallographica* C

```
_publ_requested_journal                 ?   #ActaC
_publ_requested_coeditor_name           ?   #ActaC

_publ_contact_author                        #ActaC
;
        ?  name
        ?  institute
        ?  department
        ?  address
        ?  city
        ?  country
;
_publ_contact_letter                        #ActaC
;
        ?  letter to co-editor
;
_publ_contact_author_phone              ?   #ActaC
_publ_contact_author_fax                ?   #ActaC
_publ_contact_author_email              ?   #ActaC

loop_
_publ_author_name                           #ActaC
_publ_author_address

   '?  surname, initials'
;       ?  institute
        ?  department
        ?  address
        ?  city
        ?  country
;

_publ_section_title                         #ActaC
;
   ?
;
_publ_section_abstract                      #ActaC
;
   ?
;
```

Mode 1: *write a CIF*

CIFIO creates a CIF containing data items specified by the user. The user enters a CIF request file which identifies the data items to be output to the CIF. A standard request file `cifreq` is supplied with the *Xtal* System (see Tables 1 and 2) for generating CIFs suitable for submission to *Acta Crystallographica*. It may, however, be tailored for any purpose and customized to include data that are not available in the *Xtal* System (*e.g.* the author's name and address as in Table 3). Note that the `cifreq` file is, in fact, a CIF in which all data items are flagged as 'missing' (*i.e.* '?').

Step 1. Read and store the data names from `cifreq`.

Step 2. Read the *Xtal*–CIF mapping file `cifmap` and store the links between the *Xtal* and CIF data. This file contains the information on the locality of a data item in the binary file and its relationship to the CIF data item. The format of the `cifmap` file, shown in Table 4, is as follows: (*a*) the data name; (*b*) the logical record identifier in the binary file; (*c*) the identification number within the logical record and, if the item is a number, the identification

Table 4. *An extract from the* `cifmap` *file, with the control keys*

Mapping of CIF data names to *Xtal* bdf data for use by the *Xtal*3.2 version of *CIFIO*. (Version October 1992.)

CIF data name	Xtal	Lrec	Idn1	Idn2	Nc	P/N	Lrec	Idn1
_audit_creation_date		hist	1_1		$34			
_audit_creation_method		hist	1_1		$35			
_audit_update_record								
_atom_site_label		atom	14		24	AD	atom	1
_atom_site_label_component_0		atom	14		$26			
_atom_site_label_component_1		atom	14		$27			
_atom_site_label_component_2		atom	14		$28			
_atom_site_label_component_3		atom	14		$29			
_atom_site_label_component_4		atom	14		$30			
_atom_site_label_component_5		atom	14		$31			
_atom_site_label_component_6		atom	14		$32			
_atom_site_type_symbol		atom	14		$26			
_atom_site_Wyckoff_symbol		atom						
_atom_site_description		atom						
_atom_site_disorder_group		atom						
_atom_site_calc_flag		atom	50		4			
_atom_site_calc_attached_atom		atom	51		24			
_atom_site_attached_hydrogens		atom	52					
_atom_site_thermal_displace_type		atom	23		$01			
_atom_site_refinement_flags		atom	55		8			
_atom_site_constraints		atom	57		40			
_atom_site_restraints		atom	58		40			
_atom_site_fract_x		atom	1(101)			AD	atom	2(atom 7)
_atom_site_fract_y		atom	2(102)			AD	atom	3(atom 8)
_atom_site_fract_z		atom	3(103)			AD	atom	4(atom 9)
_atom_site_Cartn_x		atom						
_atom_site_Cartn_y		atom						
_atom_site_Cartn_z		atom						
_atom_site_U_iso_or_equiv		atom	4(104)			AD	atom	5(atom 10)
_atom_site_occupancy		atom	11(111)			AD	atom	6(atom 11)
_atom_site_symmetry_multiplicity		atom	21					
_atom_site_chemical_conn_number		atom						
_atom_site_aniso_label		atom	14		$25	AD	uij	1(suij 1)
_atom_site_aniso_type_symbol		atom	14		$26			
_atom_site_aniso_U_11		atom	5(105)			AD	uij	2(suij 2)
_atom_site_aniso_U_22		atom	6(106)			AD	uij	3(suij 3)
_atom_site_aniso_U_33		atom	7(107)			AD	uij	4(suij 4)
_atom_site_aniso_U_12		atom	8(108)			AD	uij	5(suij 5)
_atom_site_aniso_U_13		atom	9(109)			AD	uij	6(suij 6)
_atom_site_aniso_U_23		atom	10(110)			AD	uij	7(suij 7)
_atom_sites_solution_primary								
_atom_sites_solution_secondary								
_atom_sites_solution_hydrogens								
_atom_type_symbol		scat	11		8	SX	celcon	1 formgn 1 *
_atom_type_analytical_mass_%		scat	98(99)			SX	celcon	3 *
_atom_type_description		scat						
_atom_type_number_in_cell		scat	1			SX	celcon	2
_atom_type_radius_bond		scat	5		D 2	SX	celcon	6
_atom_type_radius_contact		scat	6		D 2	SX	celcon	7

number of the standard deviation in parentheses; (*d*) the number of characters if the item is type CHARACTER; (*e*) the special conversion code '$<n>' if the item must be converted from the *Xtal* representation to the CIF representation.

Step 3. Read the binary file and transfer the requested data to the CIF output buffer. Numerical data are output in a format determined by either the standard deviation of the number (which is attached within parentheses) or the expected precision of the data item (see Table 4). Packed data items are output as hexadecimal strings starting with a 'Z'. Character strings are output as ASCII strings bounded by single or double quotes.

Mode 2: *read a CIF*

Converting CIF data to an *Xtal* binary file is more complicated than writing a CIF. In mode 1, missing data items are simply stored in the CIF as a '?'. The *Xtal* binary file has, however, a hierarchical data structure in which certain data items must be present. Furthermore, an *Xtal* file contains large amounts of 'derived' data which have been calculated from data primitives and stored for convenient access by other calculations (*e.g.* the metric tensor derived from the cell data). CIFs, on the other hand, contain mainly primitive data. Production of an *Xtal* file necessitates two separate mechanisms: the transfer of

Table 5. *An example Xtal command file generated by CIFIO to prepare derived data in mode* 2

```
CIFIO arci
STARTX upd
cell          8.53     8.53     20.37    90       90       120
cellsd        .01      .01      .01      .01      .01      .01
sgname        P_61_2___(0_0_-1)
celcon        s        6        $1       $1       $1       $1       $1       .319     .557
celcon        o        6        $1       $1       $1       $1       $1       .047     .032
celcon        c        12       $1       $1       $1       $1       $1       .017     .009
exper 1
ADDATM
atom   s      .202     .798     .91667   .02528   $1       0        0        0        0
uij    s      .0253    .0253    .0253    .0127    0        0
suij   s      0        0        0        0        0        0
atom   o      .498     .498     .66667   .0252    $1       0        0        0        0
atom   c1     .488     .096     .038     .0317    $1       0        0        0        0
ADDREF nobay
bdfin all
end
```

Table 6. *An extract from the start of an Xtal archive file showing some fixed data records*

```
data_p6122

_hist
;
STARTX 30-Jun92 18: 2: 3ADDREF 30-Jun92 18: 2: 4SORTRF 30-Jun92 18: 2: 4
ADDATM 30-Jun92 18: 2: 4
;

_labl
;
30-Jun92 18: 2: 3
   Test case from Larson -- dummy P6122 structure.
;

loop_
      _cell_pak_1
   .8530000+01    .8530000+01    .2037000+02    .0000000+00    .0000000+00    -.5000001+00
   .2500000+00    .2500000+00    .3333334+00

loop_
      _cell_pak_2
   .1000000-01    .1000000-01    .1000000-01    .1745329-03    .1745329-03    .1511499-03
   .2777778-04    .2777778-04    .2777778-04

loop_
      _cell_pak_3
   .1353694+00    .1353694+00    .4909180-01    .0000000+00    .0000000+00    .5000001-00
   .2500000+00    .2500000+00    .1666667+00

loop_
      _cell_pak_4
   .7387197+01    -.4265000+01    .0000000+00    .0000000+00    .8530000+01    .0000000+00
   .0000000+00    .0000000+00    .2037000+02

_symm_pak_1_1_lattice_type                          1
_symm_pak_1_2_centro_type                           1
_symm_pak_1_3_total_symops                          12
_symm_pak_1_4_basis_symops                          12
_symm_pak_1_5_equiv_symops                          2
_symm_pak_1_6_multiplicity                          1
_symm_pak_1_7_cedar_symops                          0
_symm_pak_1_8_moles/cell                            24
_symm_pak_1_9_cryst_system                          7
_symm_pak_1_10                                      3
_symm_pak_1_11                                      0
_symm_pak_1_12                                      0

_sgnm_pak_1                                         P_61_2___(0_0_-1)
```

data primitives from the CIF and then their expansion to provide the derived data.

Step 1. Copy the CIF to a direct-access file and store the location of each data name.

Step 2. Read the `cifmap` file and store links to the binary file as a list of data names in the list.

Step 3. Scan the data-name list for data primitives identified by the `cifmap` file (*e.g.* cell dimensions, space-group notation *etc.*). These are used to construct a series of *Xtal* line commands for later expansion of the binary file. Table 5 shows a generated command file.

Step 4. Scan the data-name list again and identify 'us-

Table 7. *An extract from an Xtal archive file showing some variable data records*

```
loop_
   _atom_0014          _atom_0001          _atom_0002          _atom_0003          _atom_0004          _atom_0011
   _atom_0101          _atom_0102          _atom_0103          _atom_0104          _atom_0111          _atom_0017
   _atom_0021          _atom_0022          _atom_0023          _atom_0005          _atom_0006          _atom_0007
   _atom_0008          _atom_0009          _atom_0010          _atom_0105          _atom_0106          _atom_0107
   _atom_0108          _atom_0109          _atom_0110
 'S'     .2020000     .7980000     .9166700     .0252778     1     0     0     0     0     0     1
    .5000000     1     2     .0253000     .0253000     .0253000     .0127000     0     0     0     0
    0     0     0     0
 'O'     .4980000     .4980000     .6666700     .0252000     1     0     0     0     0     0     1
    .5000000     2     1     0     0     0     0     0     0     0     0     0     0     0
 'Cl'     .4880000     .0960000     .0380000     .0317000     1     0     0     0     0     0     1
    1     3     1     0     0     0     0     0     0     0     0     0     0     0     0

loop_
   _cons_0001          _cons_0011          _cons_0002          _cons_0005          _cons_0012          _cons_0013
   _cons_0003          _cons_0004
 'S'     'S'     2          1          1          -1          1          1
 'S'     'S'     3          3          1          0           1          .9166700
 'S'     'S'     5          1          4          1           1          0
 'S'     'S'     9          1          8          1           1          0
 'O'     'O'     2          1          1          1           1          0
 'O'     'O'     3          3          1          0           1          .6666700
 'O'     'O'     5          1          4          1           1          0
 'O'     'O'     9          1          8          -1          1          0

  loop_
   _refl_0001          _refl_0002          _refl_0003          _refl_1600          _refl_1308          _refl_1304
   _refl_1305
'    0    0    6'          .1472754          Z000010c1          .4100000          1     39.40000     3.51000
'    0    0    12'         .2945508          Z000010c1          .9000000          1     48.44000     2.38000
'    0    1    0'          .0676847          Z00001043          .6200000          1     43.20000     5.27000
'    0    1    1'          .0719980          Z00001026          1.18000           1     57.36000     5.27000
'    0    1    2'          .0836135          Z00001026          1.62000           1     76.96000     4.83000
'    0    1    3'          .1000186          Z00001026          1.16000           1     53.04000     4.30000
'    0    1    4'          .1192528          Z00001026          .2300000          1     9.88000      2.62000
'    0    1    5'          .1401562          Z00001026          1.11000           1     44.88000     3.70000
'    0    1    6'          .1620841          Z00001026          .2000000          2     7.48000      1
'    0    1    7'          .1846721          Z00001026          1.01000           1     34.56000     2.98000
'    0    1    8'          .2077048          Z00001026          .4600000          1     14.40000     2.67000
'    0    1    9'          .2310494          Z00001026          .6300000          1     17.96000     2.50000
'    0    1    10'         .2546200          Z00001026          .7900000          1     20.40000     2.23000
'    0    1    11'         .2783592          Z00001026          .9500000          1     22.24000     2.21000
'    0    1    12'         .3022274          Z00001026          .3100000          2     6.68000      1
'    0    2    0'          .1353694          Z00001043          .6100000          1     35.64000     3.81000
'    0    2    1'          .1375768          Z00001026          .5300000          1     21.64000     3.37000
'    0    2    2'          .1439961          Z00001026          .9900000          1     39.36000     3.37000
'    0    2    3'          .1541018          Z00001026          1.41000           1     54.12000     3.38000
'    0    2    4'          .1672271          Z00001026          .9800000          1     36           3.05000
'    0    2    5'          .1827222          Z00001026          .3800000          1     13.28000     2.28000
'    0    2    6'          .2000373          Z00001026          1.86000           1     60.28000     2.88000
```

ing signals in the `cifmap` file) which data belong to which logical record in the binary file. This is done in the sequential order of the binary file. That is, all the scattering-factor data are identified and output, followed by all the experimental data and so on. The *Xtal* binary file produced by this step contains all of the primitive data from the CIF but, as yet, no derived data.

Step 5. Initiate calculations external to *CIFIO* which are used to expand the primitive data. This step is possible because the *Xtal* System nucleus can suspend the normal line input command stream and insert the command lines generated in step 3. These calculations expand the file generated in step 4 by including derived data. The normal input command stream is then resumed. This approach is important because it enables the expansion of the CIF data to be done using the standard checking software, which applies all of the symmetry, cell and atom-site constraint tests.

Mode 3: *write an archive file*

An *Xtal* binary file is subdivided into distinct logical records containing data of a similar type. For example, all of the cell data and derived cell data are grouped in the logical record `cell`, scattering factors and atom-type information are grouped into the logical record `scat`, atom-site data are in `atom`, reflection data are in `refl` and so on. There are two methods of storing data in a logical record. Data may be stored at *fixed* locations in the record or may be placed at *variable* locations, which are identified by indices in a special directory packet.

Binary data in fixed records may be easily converted into single ASCII 'tag and value' pairs or into simple loop structures, as shown in Table 6. Data in variable records are always converted into a single loop structure of data names derived from the logical record name and the directory index. In the `atom` directory, the data item

identified by the directory index 12 will be assigned the data name `_atom_0012`. Each packet of data names is preceded by a `loop_` command and is followed by the list of repeated data items (see Table 7).

Step 1. Read the binary file and identify logical records as containing either fixed or variable data.

Step 2. For fixed data, convert the individual data items, or sequences of data items (*e.g.* matrices), into customized data names and output these to the archive file. Binary numbers are converted to ASCII number representations. Packed integer data are converted from binary representations into ASCII hexadecimal strings preceded by a 'Z'. Character strings are converted from binary into ASCII characters bounded by single or double quotes.

Step 3. For variable data, generate loop structures based on the name of the logical record and the identification index (see description above) and output these to the archive file. The data conversions are the same as performed in step 2.

Mode 4: *read an archive file*

The algorithm for reading an archive file is the reverse of that in mode 3. The archive file is read sequentially and data items, identified by their data names, are transferred into binary fixed or variable logical records. Data are converted from numerical strings or character strings into binary representations. The generated binary file is an exact replica of that used to create the archive file in mode 3.

Distribution

The *CIFIO* program may be implemented only as part of the *Xtal* System. However, a copy of the files `cifmap` and `cifreq` may be obtained free of charge in several different ways. The simplest and fastest approach is to use anonymous FTP to *get* the file from the directory cif on the host 130.95.232.12. Alternatively, send an email containing the line `send cifmap` to sendcif@crystal.uwa.edu.au. As a last resort, airmail a floppy disk to the author stating the mode of copy required.

References

HALL, S. R., ALLEN, F. H. & BROWN, I. D. (1991). *Acta Cryst.* A**47**, 655–685.

HALL, S. R. & SIEVERS, R. (1993). *J. Appl. Cryst.* **26**, 469–473.

HALL, S. R., FLACK, H. D. & STEWART, J. M. (1992). Editors. *Xtal*3.2 *Users Manual*. Univs. of Western Australia, Australia, and Maryland, USA.

HALL, S. R. & STEWART, J. M. (1990). Editors. *Xtal*3.0 *Users Manual*. Univs. of Western Australia, Australia, and Maryland, USA.