Earth Observation
Mission CFI Software

EO_DATA_HANDLING
SOFTWARE USER MANUAL

Code:	EO-MA-DMS-GS-0007
Issue:	4.30
Date:	04/02/2026

	Name	Function	Signature
Prepared by:	Davide Aiello	Project Engineer	
Checked by:	Davide Aiello	Project Engineer	
Approved by:	Davide Aiello	Project Engineer	

DEIMOS Space S.L.U.
Ronda de Poniente, 19
Edificio Fiteni VI, Portal 2, 2ª Planta
28760 Tres Cantos (Madrid), SPAIN
Tel.: +34 91 806 34 50
Fax: +34 91 806 34 51
E-mail: deimos@deimos-space.com

© DEIMOS Space S.L.U.

All Rights Reserved. No part of this document may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior written permission of DEIMOS Space S.L.U. or ESA.

DOCUMENT INFORMATION

Contract Data		Classification	
Contract Number:	4000102614/1O/NL/FF/ef	Internal	
		Public	
Contract Issuer:	ESA / ESTEC	Industry	X
		Confidential	

External Distribution		
Name	Organisation	Copies

Electronic handling	
Word Processor:	LibreOffice 5.2.3.3
Archive Code:	P/SUM/DMS/01/026-044
Electronic file name:	eo-ma-dms-gs-007-10

DOCUMENT STATUS LOG

Issue	Change Description	Date	Approval
3.4	New document. Library first version. Issue number corresponds to CFI library issue	18/11/05	
3.5	• Maintenance release. • New features: - function xd_xml_validate	26/05/06	
3.6	• Maintenance release • New features: - Validator function and executable for XML files (xd_xml_validate and xml_validator)	24/11/06	
3.7	• Maintenance release • New features: - Function expcfi_check_libs - Library version for MAC OS X on Intel (32 and 64-bits)	13/07/07	
3.7.2	• Maintenance release • New features: - Reading and writing functions for TLE - New format for orbit files: reference frame added to the variable header.	31/07/08	
4.0	• Maintenance release • Reading function for the numerical propagator configuration file	16/01/09	
4.1	• Maintenance release • New section added: List of schema's versions • New features: - GETASSEv2 DEM supported - IERS B Bulletin format 2010 reading supported - Ground Station DB 1.4 file reading supported	07/05/10	
4.2	• Maintenance release • New features: - New format for the OSF to support curved MLST - New DEM configuration file	31/01/11	

4.3	• Maintenance release • New features: - Support for reading new IERS bulletins A and B - New functions to decimate orbit and attitude data: xd_orbit_file_decimate and xd_attitude_file_decimate	06/02/12	
4.4	•Maintenance release • New features: - New tags in DEM configuration for DEM cache	05/07/12	
4.5	•Maintenance release • New features: - New tags in DEM configuration for mini tiles and geoid compuation. - EarthCare filenames compliant with FFS 2.0.	01/03/13	
4.6	•Maintenance release • New features: - Support for new Attitude Definition File	03/10/13	
4.7	• Maintenance release • New features: - Reading support for SP3 files - Reading support for S3 DORIS files	03/28/14	
4.8	•Maintenance release •New features: - Added support for DEM GETASSE v3.0 - Added support for dataset GDEM v2 - New function to add stylesheet to files: xd_xslt_add	29/10/2014	
4.9	•Maintenance release •New features: - Support for Orbit Ephemeris Message files	23/04/2015	

4.10	• Maintenance release • New features: - Support for DEM ACE2 30 secs - New diagnostic function for orbit files with state vectors: xd_orbit_file_diagnostics - Change of interface in functions xd_read_oem and xd_read_sp3	29/10/2015	
4.11	• Maintenance release • New features: - Support for DEM ACE2 3 secs • Updated table 232: Added BIOMASS, SENTINEL_5 AND SAOCOM_CS Satellites	15/04/2016	
4.12	• Maintenance release • New features: - Added support for File Format Standard V3	03/11/2016	
4.13	• Maintenance release	05/04/2017	
4.14	• Maintenance release • New features: - Added support for Jason-CS Doris	16/11/2017	
4.15	• Maintenance release • New Features: - Refactored code	20/04/2017	
4.16	• Maintenance release	09/11/2018	

4.17	• Maintenance release	10/05/2019	

4.18	• Maintenance release
	• New features:
	  • Support for TanDEM-X 90m DEM
	  • Support for new IERS bulletin B format
		08/11/2019	

4.19	• Maintenance release
	• New features:
	  • Reference frame added for attitude files with rotaton angles
	  • Added support for ASTER GDEM v3
	  • Added support for XML Orbit Ephemeris Message Files
		29/05/2020	

4.20	• Maintenance release • New features: • Added support for Attitude Ephemeris Message Files (AEM files)	30/11/2020	
4.21	• Maintenance release	23/06/2021	
4.22	• Maintenance release • New features: • Enabled support for AEM with custom REF_FRAME used by EUM	22/12/2021	

4.23	• Maintenance release
	• New features:
	  • Define the SP3 identifier on the Satellite Configuration File
	  • Update the parameters of the ESA mission: CRISTAL
	  • Include support for new ESA mission: TRUTHS
		23/06/2022	

4.24	• Maintenance release	29/11/2022	
4.25	Maintenance release New features: • Orbit initialization with new Orbit Scenario files with ANX longitude drift parameters.	10/05/2023	
4.26	Maintenance release New features: • API support for longitude drift	31/10/2023	
4.27	Maintenance release • Use NORAD catalog number to identify TLE instead of name	07/06/2024	

4.28	Maintenance release	13/12/2024	
4.29	Maintenance release	09/04/2025	
4.30	Maintenance release New features: • Add support for SENTINEL1D, SENTINEL2D, SENTINEL6C, HARMONY-A/B, and ALTIUS • Update orbital parameters and SATCAT/NORAD ID for BIOMASS	04/02/2026	

TABLE OF CONTENTS

LIST OF TABLES

1. SCOPE

The EO_DATA_HANDLING Software User Manual provides:

- a detailed description of usage of the CFI functions included within the EO_DATA_HANDLING CFI software library.
- The format description of the Earth Observation Missions files as well as the available versions of those files.
- The format description (or reference to it) of other file types (e.g. TLE, SP3, IERS bulletins).

2. ACRONYMS, NOMENCLATURE AND TERMINOLOGY

2.1. Acronyms

ANX	Ascending Node Crossing
AOCS	Attitude and Orbit Control Subsystem
ASCII	American Standard Code for Information Interchange
BOM	Beginning Of Mission
CFI	Customer Furnished Item
EOM	End Of Mission
ESA	European Space Agency
ESTEC	European Space Technology and Research Centre
GPL	GNU Public License
GPS	Global Positioning System
IERS	International Earth Rotation Service
I/F	Interface
LS	Leap Second
OBT	On-board Binary Time
OSF	Orbit Scenario File
SRAR	Satellite Relative Actual Reference
SUM	Software User Manual
TAI	International Atomic Time
UTC	Coordinated Universal Time
UT1	Universal Time UT1
WGS[84]	World Geodetic System 1984

2.2. Nomenclature

CFI	A group of CFI functions, and related software and documentation. that will be distributed by ESA to the users as an independent unit
CFI function	A single function within a CFI that can be called by the user
Library	A software library containing all the CFI functions included within a CFI plus the supporting functions used by those CFI functions (transparently to the user)

2.3.Note on Terminology

In order to keep compatibility with legacy CFI libraries, the Earth Observation Mission CFI Software makes use of terms that are linked with missions already or soon in the operational phase like the Earth Explorers.

This may be reflected in the rest of the document when examples of Mission CFI Software usage are proposed or description of Mission Files is given.

3. APPLICABLE AND REFERENCE DOCUMENTS

3.1. Applicable Documents

No applicable documents

3.2. Reference Documents

[MCD]	Earth Observation Mission CFI Software. Mission Conventions Document. EO-MA- DMS-GS-0001.
[F_H_SUM]	Earth Observation Mission CFI Software. EO_FILE_HANDLING Software User Manual. EO-MA-DMS-GS-0008.
[LIB_SUM]	Earth Observation Mission CFI Software. EO_LIB Software User Manual. EO-MA-DMS-GS-0003.
[ORBIT_SUM]	Earth Observation Mission CFI Software. EO_ORBIT Software User Manual. EO-MA-DMS-GS-0004.
[POINT_SUM]	Earth Observation Mission CFI Software. EO_POINTING Software User Manual. EO-MA-DMS-GS-0005.
[GEN_SUM]	Earth Observation Mission CFI Software. General Software User Manual. EO-MA- DMS-GS-0002.
[IERS]	IERS Bulletins https://www.iers.org/IERS/EN/Publications/Bulletins/bulletins.html
[PDS_FMT]	Cryosat-2 Ground Segment Payload Data Segment L0 Product Specification Format CS-ID-ACS-GS-0119 (Please contact the CryoSat-2 project team to obtain a copy of this document)
[FFS1]	Earth Observation Ground Segment File Format Standard (version 1.4), http://eop-cfi.esa.int/Repo/PUBLIC/DOCUMENTATION/SYSTEM_SUPPORT_DOCS/obsolete/Format-Standard-1.4.pdf
[FFS2]	Earth Observation Ground Segment File Format Standard (version 2.0), http://eop-cfi.esa.int/Repo/PUBLIC/DOCUMENTATION/SYSTEM_SUPPORT_DOCS/PE-TN-ESA-GS-0001%20EO%20GS%20File%20Format%20Standard%202.0.pdf
[FFS3]	Earth Observation Ground Segment File Format Standard (version 3.0), http://eop-cfi.esa.int/Repo/PUBLIC/DOCUMENTATION/SYSTEM_SUPPORT_DOCS/PE-TN-ESA-GS-0001%20EO%20GS%20File%20Format%20Standard%203.0.pdf
[EO_SCH_HB]	Handbook for EO XML and Binary Schemas (version 1.7.1),

	http://eop-cfi.esa.int/Repo/PUBLIC/DOCUMENTATION/SYSTEM_SUPPORT_DOCS/PE-TN-ESA-GS-121%20%20Handbook%20for%20EO%20XML%20and%20Binary%20Schemas%201.7.1.pdf
[EO_ICD]	Earth Observation Mission Software File Format Specification , https://eop-cfi.esa.int/Repo/PUBLIC/DOCUMENTATION/SYSTEM_SUPPORT_DOCS/PE-ID-ESA-GS-584-EO_Mission_SW_File_Format_Specs_latest.pdf
[SP3]	Extended Standard Product 3 Orbit Format (SP3-c) http://igscb.jpl.nasa.gov/igscb/data/format/sp3c.txt
[OEM]	Orbit Ephemeris Message Format (OEM) http://public.ccsds.org/publications/archive/502x0b2c1.pdf
[OEM_XML]	XML SPECIFICATION FOR NAVIGATION DATA MESSAGES https://public.ccsds.org/Pubs/505x0b1.pdf
[TLE]	Two Line Element File http://celestrak.com
[PDGS_S3]	Sentinel-3 Core Payload Data Ground Segment (PDGS) Instrument Processing Facility (IPF) Implementation, Doc. Nr. S3IPF.PDS.001 (Please contact the Sentinel-3 project team to obtain a copy of this document)
[AEM]	Attitude Ephemeris Message Format (AEM) • [TXT] Attitude Data Messages, May 2008, CCSDS 504.0-B-1 (https://public.ccsds.org/Pubs/504x0b1c1.pdf) • [XML] XML Specification for Navigation Data Messages, December 2010, CCSDS 505.0-B-1 (https://public.ccsds.org/Pubs/505x0b1.pdf)

The latest applicable version of [MCD], [F_H_SUM], [LIB_SUM], [ORBIT_SUM], [POINT_SUM], [VISIB_SUM], [GEN_SUM] is v4.30 and can be found at: http://eop-cfi.esa.int/REPO/PUBLIC/DOCUMENTATION/CFI/EOCFI/BRANCH_4X/

4. INTRODUCTION

4.1.Functions Overview

This software library contains a set of functions for reading and writing Earth Observation Mission Files. The following CFI functions are included:

4.1.1. Reading routines

- **xd_read_att:** reads a generic attitude file.
- **xd_read_att_def:** reads a whole attitude definition file
- **xd_read_bulletin:** reads the time correlations from an IERS bulletin B (1980 and 2010 format).
- **xd_read_bulletin_2:** reads the time correlations from a IERS bulletins A and B (only 2010 format).
- **xd_read_dem:** provides the points of a DEM that are adjacent to a given point.
- **xd_read_dem_config_file:** reads a DEM configuration file.
- **xd_read_doris**: reads DORIS Navigator files for CRYOSAT and SENTINEL 3.
- **xd_read_doris_header:** reads the MPH and SPH data from a DORIS Navigator file for CRYOSAT.
- **xd_read_fhr:** reads the fixed header for an Earth Observation XML file.
- **xd_read_orbit_file**: reads orbit files consisting in a list of state vectors of the satellite in the orbit. The following files are supported: Predicted Orbit files, Restituted Orbit files and DORIS Preliminary files.
- **xd_read_osf**: reads Orbit Scenario files.
- **xd_read_precise_propag_file:** reads a data file used to configure the numerical propagator
- **xd_read_sdf**: reads swath definition files.
- **xd_read_sp3:** reads a Standard Product 3 C (SP3-C) File
- **xd_read_star:** reads the parameters of one star in a star database file.
- **xd_read_star_file:** reads a star database file.
- **xd_read_star_id:** reads the list of star id. from a star database file
- **xd_read_star_tracker:** reads an star traker file for CRYOSAT.
- **xd_read_star_tracker_conf_file:** reads an star tracker configuration file for CRYOSAT.
- **xd_read_station:** reads the parameters of one station in a station database file.
- **xd_read_station_file:** reads a station database file.
- **xd_read_station_id:** reads the list of station names from a station database file
- **xd_read_stf:** reads swath template files.
- **xd_read_stf_vhr:** reads the variable header for swath template files
- **xd_read_tle:** reads a TLE file

- **xd_read_zone:** reads the parameters of one zone in a zone database file.
- **xd_read_zone_file:** reads a zone database file.
- **xd_read_zone_id:** reads the list of zone names from a zone database file.
- **xd_read_oem:** reads CCSDS Orbit Ephemeris Message files.
- **xd_read_aem:** reads CCSDS Attitude Ephemeris Message files.

4.1.2. Writing routines

- **xd_write_att:** writes a generic attitude file.
- **xd_write_att_def:** writes a attitude definition file
- **xd_write_doris:** writes a DORIS Navigator file.
- **xd_write_orbit_file**: writes an orbit file using as input an structure with the data of the file
- **xd_write_osf**: writes an orbit scenario file using as input an structure with the data of the file
- **xd_write_stf**: writes a swath template file using as input the data structure containing the data for the swath.
- **xd_write_tle**: writes a TLE file using as input a data structure.

4.1.3. Functions to free memory

- **xd_free_orbit**: frees the memory allocated during the reading function **xd_read_orbit_file**.
- **xd_free_doris**: frees the memory allocated during the reading function **xd_read_doris**
- **xd_free_osf**: frees the memory allocated during the reading function **xd_read_osf**.
- **xd_free_sdf**: frees the memory allocated during the reading function **xd_read_sdf**.
- **xd_free_stf**: frees the memory allocated during the reading function **xd_read_stf**.
- **xd_free_stf_vhr**: frees the memory allocated during the reading function **xd_read_stf_vhr**.
- **xd_free_att**: frees the memory allocated during the reading function **xd_read_att**.
- **xd_free_star_tracker**: frees the memory allocated during the reading function
- xd_read_star_tracker.
- **xd_free_dem**: frees the memory allocated in the reading function **xd_read_dem**
- **xd_free_zone**: frees the memory allocated during the reading function **xd_read_zone**.
- **xd_free_zone_file**: rees the memory allocated during the reading function **xd_read_zone_file**.
- **xd_free_zone_id**: frees the memory allocated during the reading function **xd_read_zone_id**.
- **xd_free_station_file**: frees the memory allocated during the reading functionxd_read_station_file.
- **xd_free_station_id**: frees the memory allocated during the reading function **xd_read_station_id**.
- **xd_free_oem**: frees the memory allocated during the reading function **xd_read_oem**.
- **xd_free_aem**: frees the memory allocated during the reading function **xd_read_aem**.

4.1.4. Validation of XML files

- **xd_xml_validate**: validates an XML file using an XML schema as reference.
- **xd_select_schema**: it returns the most recent schema name supported for a given file type and mission

4.2.Reading and writing files

When reading files, the user should be aware that:

- Many of the structures used for reading files contain dynamic data that is allocated within the reading function. In these cases, the memory has to be freed when it is not going to be used any more by calling the suitable function.
- The reading functions for each of the file types, does not read the fixed header. The fixed header could be read independently using the CFI function **xd_read_fhr**.
- When reading the fixed header with **xd_read_fhr**, the schema name is not read (the "schema" element in the output structure **xd_fhr** will be set to "_NOSCHEMA_") . If required, the schema name and version should be read independently with the CFI functions in explorer_file_handling.

When writing files, the user should be aware that:

- The schema name and version can be written in the file in the following ways:
  - Setting the schema name in the "schema" element in the **xd_fhr** structure. When calling the **xd_write_*xxx*** function, the schema name and version will be written in the file. Note that if the schema name is set to "_NOSCHEMA_", the schema attributes will no be written in the file.
  - After writing the file, by calling the function **xf_set_schema** (in explorer_file_handling). Note that the CFI function **xd_select_schema** allows to get the default schema name with which the file to be written is compliant.

4.3. Memory usage

Note: due to the implementation of the third-party library handling XML files, large amount of memory may be needed by an application handling (reading or writing) a file with many entries. Therefore the user is recommended to perform handling of large files on computers equipped with adequate memory resources. If these resources are not available, the user has to properly configure virtual memory and take into account long execution times. In extreme cases, due to platform limitation or operating system settings, the operation may fail. In order to give an indication, a restituted orbit file covering a period of 10 days and a time interval between OSVs of 30 sec contains 28800 OSVs and its size on disk is about 14MB. The memory usage peak during the writing of such file is about 215MB.

4.4. DEM data

In order to use the DEM files (reading the files or using it with the the pointing functions) the user should do the following:

- The DEM has to be acquired by the user with the means provided by the DEM provider (FTP, Web service…)

- The DEM files has to be stored under a single directory chosen the by the user. This directory has to be written down on the DEM configuration file so the that EOCFI functions can find the DEM files when needed.

Example: For TamDEM-X, the files can be downloaded via FTP. The DEM is organised in folders according to the longitude and latitude covered by the files:

DEM_root:

|--N00/

|--E000/

| |---TDM1_DEM__30_N00E006.zip

| |---TDM1_DEM__30_N00E00.zip

|--/E010

| |----TDM1_DEM__30_N00E010.zip

| |----TDM1_DEM__30_N00E011.zip

| |----...

...

Every TDM1_DEM.xxxx.zip file contains a set of files reletad to the DEM of which we are interested in those containing the altitudes over the ellipsoid, for instance:

TDM1_DEM__30_N00E006.zip

|---- AUXFILES

|---- DEM → **TDM1_DEM__30_N00E006_DEM.tif**

|---- PREVIEW

In this case, all the TDM1_DEM__30_*hxxmyyy*_DEM.tif files should be moved to the directory chosen by the user to store the DEM.

5. LIBRARY INSTALLATION

For a detailed description of the installation of any CFI library, please refer to [GEN_SUM].

6. LIBRARY USAGE

The EO_DATA_HANDLING software library has the following dependencies:

- Other EOCFI libraries: EO_FILE_HANDLING (See [F_H_SUM]).
- Third party libraries:
  - POSIX thread library: libpthread.so (Note: this library is normally pre-installed in Linux and MacOS platforms. For Windows platforms, pthread.lib is included in the distribution package, with license LGPL);
  - GEOTIFF, TIFF, PROJ, LIBXML2 libraries (these libraries are included in the distribution package. Their usage terms and conditions are available in the file "TERMS_AND_CONDITIONS.TXT" which is part of the distribution package).

The following is required to compile and link a Software application that uses the EO_DATA_HANDLING software library functions (it is assumed that the required EOCFI and third-part libraries are located in directory *cfi_lib_dir* and the required header files are located in *cfi_include*, see [GEN_SUM] for installation procedures):

1) include the following header files in the source code:

   - explorer_data_handling.h (for a C application)

2) use the following compile and link options:

   Linux and MacOS platforms:

   -I*cfi_include_dir* -L*cfi_lib_dir* -lexplorer_data_handling

   -lexplorer_file_handling -lgeotiff -ltiff -lproj -lxml2 -lm -lc -lpthread

   Windows platforms:

   /I *"cfi_include_dir"* /libpath:*"cfi_lib_dir"* libexplorer_data_handling.lib

   libexplorer_file_handling.lib libgeotiff.lib libtiff.lib libproj.lib libxml2.lib pthread.lib Ws2_32.lib

All functions described in this document have a name starting with the prefix `xd_`

To avoid problems in linking a user application with the `EO_DATA_HANDLING` software library due to the existence of names multiple defined, the user application should avoid naming any global software item beginning with either the prefix `XD_` or `xd_`.

It is possible to call the following CFI functions from a user application.

Table 1: CFI functions included within EO_DATA_HANDLING library

Function Name	Enumeration value	Long
Main CFI Functions		

Function Name	Enumeration value	Long
xd_read_fhr	XD_READ_FHR_ID	0
xd_read_bulletin	XD_READ_BULLETIN_ID	1
xd_read_orbit_file	XD_READ_ORBIT_FILE_ID	2
xd_read_doris	XD_READ_DORIS_ID	3
xd_read_doris_header	XD_READ_DORIS_HEADER_ID	4
xd_read_osf	XD_READ_OSF_ID	5
xd_read_sdf	XD_READ_SDF_ID	6
xd_read_stf	XD_READ_STF_ID	7
xd_read_stf_vhr	XD_READ_STF_VHR_ID	8
xd_read_att	XD_READ_ATT	9
xd_read_star_tracker	XD_READ_STAR_TRACKER_ID	10
xd_read_str_conf_file	XD_READ_STR_CONF_FILE_ID	11
xd_read_dem_config_file	XD_READ_DEM_CONFIG_FILE_ID	12
xd_read_dem	XD_READ_DEM_ID	13
xd_read_star	XD_READ_STAR_ID	14
xd_read_star_file	XD_READ_STAR_FILE_ID	15
xd_read_star_id	XD_READ_STAR_ID_ID	16
xd_read_station	XD_READ_STATION_ID	17
xd_read_station_file	XD_READ_STATION_FILE_ID	18
xd_read_station_id	XD_READ_STATION_ID_ID	19
xd_read_zone	XD_READ_ZONE_ID	20
xd_read_zone_file	XD_READ_ZONE_FILE_ID	21
xd_read_zone_id	XD_READ_ZONE_ID_ID	22
xd_write_orbit_file	XD_WRITE_ORBIT_FILE_ID	23
xd_write_doris	XD_WRITE_DORIS_ID	24
xd_write_osf	XD_WRITE_OSF_ID	25
xd_write_stf	XD_WRITE_STF_ID	26
xd_write_att	XD_WRITE_ATT_ID	27
xd_xml_validate	XD_XML_VALIDATE_ID	28
xd_read_tle	XD_READ_TLE	29

Function Name	Enumeration value	Long
xd_write_tle	XD_WRITE_TLE	30
xd_read_precise_propag_file	XD_READ_PRECISE_PROPAG_FILE_ID	31
xd_orbit_file_decimate	XD_ORBIT_FILE_DECIMATE_ID	33
xd_attitude_file_decimate	XD_ATTITUDE_FILE_DECIMATE_ID	34
xd_read_att_def	XD_READ_ATT_DEF_ID	35
xd_write_att_def	XD_WRITE_ATT_DEF_ID	36
xd_read_sp3	XD_READ_SP3_ID	37
xd_xslt_add	XD_XSLT_ADD_ID	38
xd_read_oem	XD_READ_OEM_ID	39
xd_orbit_file_diagnostics	XD_ORBIT_FILE_DIAGNOSTICS_ID	40
xd_read_fov_constraints_file	XD_READ_FOV_CONSTRAINTS_FILE_ID	41
xd_set_file_format_standard_version	XD_SET_FILE_FORMAT_STANDARD_VERSION_ID	42
xd_detect_file_format_standard_version	XD_DETECT_FILE_FORMAT_STANDARD_VERSION_ID	43
xd_read_aem	XD_READ_AEM_ID	44
Error Handling Functions		
xd_verbose	not applicable	
xd_silent		
xd_get_code		
xd_get_msg		
xd_print_msg		

Notes about the table:

- To transform the extended status flag returned by a CFI function to either a list of error codes or a list of error messages, the enumeration value (or the corresponding long value) described in the table must be used
- The error handling functions have no enumerated values

Whenever available it is strongly recommended to use enumeration values rather than integer values.

6.1.Usage hints

Every CFI function has a different length of the Error Vector, used in the calling I/F examples of this SUM and defined at the beginning of the library header file. In order to provide the user with a single value that

could be used as Error Vector length for every function, a generic value has been defined (XD_ERR_VECTOR_MAX_LENGTH) as the maximum of all the Error Vector lengths. This value can therefore be safely used for every call of functions of this library.

6.2.General Enumerations

The aim of the current section is to present the enumeration values that can be used rather than integer parameters for some of the input parameters of the EO_DATA_HANDLING routines, as shown in the table below. The enumerations presented in [GEN_SUM] are also applicable.

Table 2: Enumerations within EO_DATA_HANDLING library

Input	Description	Enumeration value	Long
Boolean values	False value	XD_FALSE	0
	True value	XD_TRUE	1
Returned status code	Error	XD_ERR	-1
	Ok status	XD_OK	0
	Warning	XD_WARN	1
Time initialization	Select the whole file	XD_SEL_FILE	0
	Select a time range	XD_SEL_TIME	1
	Select an orbit range	XD_SEL_ORBIT	2
	Select the default value	XD_SEL_DEFAULT	3
Time reference	Undefined	XD_TIME_UNDEF	-1
	TAI	XD_TIME_TAI	0
	UTC	XD_TIME_UTC	1
	UT1	XD_TIME_UT1	2
	GPS	XD_TIME_GPS	3
Attitude data type	Quaternions	XD_ATT_QUATERNIONS	0
	Angles	XD_ATT_ANGLES	1
Ray tracing model		XD_NO_REF	0
		XD_STD_REF	1
		XD_USER_REF	2
		XD_PRED_REF	3
		XD_STD_REF_N	10
		XD_USER_REF_N	20
		XD_PRED_REF_N	30
		XD_US76_REF	300
		XD_TROPIC_REF	301
		XD_MID_SUM_REF	302
		XD_MID_WIN_REF	303
		XD_SUBAR_SUM_REF	304
		XD_SUBAR_WIN_REF	305
		XD_LUT_REF	400
		XD_US76_REF_N	3000
		XD_TROPIC_REF_N	3001
		XD_MID_SUM_REF_N	3002
		XD_MID_WIN_REF_N	3003
		XD_SUBAR_SUM_REF_N	3004
		XD_SUBAR_WIN_REF_N	3005

Input	Description	Enumeration value	Long
		XD_LUT_REF_N	4000
Swath Types		XD_OPEN_SWATH	0
		XD_CLOSED_SWATH	1
Swath Point types		XD_GEODETIC_SWATH_TYPE	0
		XD_INERTIAL_SWATH_TYPE	1
Swath geometry definition = algorithm		XD_SWATH_POINTING_GEOM	0
		XD_SWATH_DISTANCE_GEOM	1
		XD_SWATH_LIMB_GEOM	2
		XD_SWATH_INERTIAL_GEOM	3
		XD_SWATH_SUBSATELLITE_GEOM	4
		XD_SWATH_ASAR_GEOM	5
		XD_SWATH_ASAR_GEOM_RANGERATE_ALGO	6
		XD_SWATH_INCIDENCE_ANGLE_GEOM	7
Asar swath types		XD_NO_ASAR	0
		XD_NARROW_ASAR	1
		XD_WIDE_ASAR	2
Orbit file types	Orbit Scenario File	XD_REF_FILETYPE_OSF	0
	Orbit Event file used as an OSF	XD_REF_FILETYPE_OEF_OSF	1
	FOS Predicted Orbit File	XD_REF_FILETYPE_POF	2
	Orbit Event file used as a POF	XD_REF_FILETYPE_OEF_POF	3
	DORIS Navigator File	XD_REF_FILETYPE_DORIS_NAV	4
	FOS Restituted Orbit File	XD_REF_FILETYPE_ROF	5
	DORIS Preliminary Orbit File	XD_REF_FILETYPE_DORIS_PREM	6
	DORIS Precise Orbit File	XD_REF_FILETYPE_DORIS_PREC	7
Orbit modes and file types	Unknown	XD_UNKNOWN_TYPE	-1
	Detect automatically	XD_AUTO	0
	Orbit from orbital change info	XD_ORBIT_CHANGE	1
	Orbit from one state vector	XD_STATE_VECTOR	2
	Orbit Scenario File	XD_OSF_TYPE	3
	FOS Predicted Orbit File	XD_POF_TYPE	4
	FOS Restituted Orbit File DORIS Preliminary Orbit File DORIS Precise Orbit File	XD_ROF_TYPE	5
	DORIS Navigator File	XD_DORIS_TYPE	6
	Predicted orbit file plus DORIS Navigator file	XD_POF_N_DORIS_TYPE	7
	Orbit Event file used as an OSF	XD_OEF_OSF_TYPE	8
	Orbit Event file used as a POF	XD_OEF_POF_TYPE	9
	IERS Bulletin B file	XD_IERS_B_TYPE	10
	Two line elements file	XD_TLE_TYPE	11
	Swath Template file	XD_STF_TYPE	12
	DORIS Precise file	XD_DORISPREC_TYPE	13
	Doris Preliminary file	XD_DORISPREM_TYPE	14
	Attitude file	XD_ATT_TYPE	15
	Swath Control file	XD_SCF_TYPE	16
	Precise Propagation configuration file	XD_PRECISE_PROPAG_TYPE	17

Input	Description	Enumeration value	Long
	DEM Configuration file	XD_DEMCFG_TYPE	18
	Satellite Configuration file	XD_SATCFG_TYPE	19
	Ground Station Database file	XD_GND_DB_TYPE	20
	Swath Definition file	XD_SW_DEF_TYPE	21
	Zone Database file	XD_ZON_DB_TYPE	22
	Star Tracker file	XD_STR1ATT_TYPE	23
	IERS Bulletin A file	XD_IERS_A_TYPE	24
	IERS Bulletin B plus A	XD_IERS_B_AND_A_TYPE	25
	Attitude definition	XD_ATT_DEF_TYPE	26
	Generic list of state vectors	XD_USER_OSV_LIST_TYPE	27
	SP3 file	XD_SP3_TYPE	28
	OSF plus POF file	XD_OSF_POF_MODE	29
	OSF plus ROF file	XD_OSF_ROF_MODE	30
	OSF plus DORIS file	XD_OSF_DORIS_MODE	31
	OEM file	XD_OEM_TYPE	32
	OSF plus OEM file	XD_OSF_OEM_MODE	33
	AEM file	XD_AEM_TYPE	34
Coordinate systems	Barycentric Mean of 2000.0	XD_BAR_MEAN_2000	1
	Heliocentric Mean of 2000.0	XD_HEL_MEAN_2000	2
	Geocentric Mean of 2000.0	XD_GEO_MEAN_2000	3
	Mean of date	XD_MEAN_DATE	4
	True of date	XD_TRUE_DATE	5
	Pseudo Earth Fixed	XD_PSEUDO_EARTH_FIXED	6
	Earth Fixed	XD_EARTH_FIXED	7
	Launch Inertial Frame	XD_LIF	8
	Barycentric Mean of 1950.0	XD_BAR_MEAN_1950	9
	Galactic	XD_GALACTIC	10
	Satellite relative actual reference	XD_SAT_ACT_REF	11
	Quasi-Mean of Date	XD_QUASI_MEAN_DATE	12
	Pseudo-True of Date	XD_PSE_TRUE_DATE	13
	Topocentric	XD_TOPOCENTRIC	14
	Satellite reference	XD_SAT_REF	15
	Satellite relative reference	XD_SAT_REL_REF	16
Attitude reference frames	Orbital reference frame	XD_SAT_ORBITAL_REF	0
	Satellite nominal attitude frame	XD_SAT_NOMINAL_ATT	1
	Satellite attitude frame	XD_SAT_ATT	2
	Instrument attitude frame	XD_INSTR_ATT	3
Different models for DEM	ACE Model (deprecated)	XD_DEM_ACE_MODEL	0
	GETASSE 30 v1	XD_DEM_GETASSE30_V1	1
	GETASSE 30 v2	XD_DEM_GETASSE30_V2	2
	ACE2 9 seconds	XD_DEM_ACE2_9SEC	3
	GETASSE 30 v3	XD_DEM_GETASSE30_V3	4
	ASTER GDEM v2	XD_DEM_GDEM_V2	5
	ACE2 30 seconds	XD_DEM_ACE2_30SEC	6
	ACE2 3 seconds	XD_DEM_ACE2_3SEC	7
	ACE2 5 minutes	XD_DEM_ACE2_5MIN	8
	GENERIC	XD_DEM_GENERIC_RASTER	9
	TanDEM-X 90m	XD_DEM_TANDEMX_90	10

Input	Description	Enumeration value	Long
	ASTER GDEM v3	XD_DEM_GDEM_V3	11
Zone types	zone is not defined as an input and must be read from a file	XD_NOT_DEFINED	-1
	Point zone	XD_POINT	0
	Circular zone	XD_CIRCLE	1
	Segment zone	XD_SEGMENT	2
	Polygonal zone	XD_POLYGON	3
Projection types	Read projection from DB file	XD_READ_DB	0
	Use gnomonic projection	XD_GNOMONIC	1
	Use rectangular projection	XD_RECTANGULAR	2
Validation Status	Invalid file	XD_XML_INVALID	-1
	Valid file	XD_XML_VALID	0
Quality Index	Adjusted out of orbit manoeuvre period	XD_3_ADJUST_NOMI	1
	Adjusted during an orbit manoeuvre	XD_4_ADJUST_DMAN	2
	Interpolated during a data gap	XD_5_INTERP_DGAP	3
	Extrapolated from less than 1 day	XD_6_EXTRAP_LT1D	4
	Extrapolated from more than 1 day, but less than 2 days	XD_7_EXTRAP_1D2D	5
	Extrapolated from more than 2 days	XD_8_EXTRAP_GT2D	6
	Extrapolated after an orbit manoeuvre	XD_8_EXTRAP_AMAN	7
Draw modes for the SCF	SOLID	XD_SCF_DRAW_SOLID	0
	DASHED	XD_SCF_DRAW_DASHED	1
	DOTTED	XD_SCF_DRAW_DOTTED	2
	TIMELINE	XD_SCF_DRAW_TIMELINE	3
Fill modes for the SCF	SOLID	XD_SCF_FILL_SOLID	0
	HOLLOW	XD_SCF_FILL_HOLLOW	1
Event types for the SCF	No event description	XD_EVENT_TYPE_NONE	0
	Zone visibility event	XD_EVENT_TYPE_ZONE	1
	Station visibility event	XD_EVENT_TYPE_STATION	2
	Geodetic event	XD_EVENT_TYPE_GEO	3
Reference time values	TAI reference	XD_TIME_REF_OF_TAI	0
	UTC reference	XD_TIME_REF_OF_UTC	1
	UT1 reference	XD_TIME_REF_OF_UT1	2
DEM Data Source Types for GETASSE30 V1, V2 and V3	Data from ACE (land-ice/snow)	XD_DEM_GETASSE30_SOURCE_ACE	0
	Data from MSS (Sea)	XD_DEM_GETASSE30_SOURCE_MSS	1
	Data from EGM96 (Sea-Ice)	XD_DEM_GETASSE30_SOURCE_EGM96	2
	Data from SRTM30 (Land)	XD_DEM_GETASSE30_SOURCE_SRTM30	3
DEM Data Source Types for ACE2: 9secs , 30secs and 3 secs	Pure SRTM (above 60°N pure GLOBE data, below 60S pure ACE [original] data)	XD_DEM_ACE2_SOURCE_SRTM0	0
	SRTM voids filled by interpolation and/or altimeter data	XD_DEM_ACE2_SOURCE_SRTM1	1
	SRTM data warped using the ERS-1 Geodetic Mission	XD_DEM_ACE2_SOURCE_SRTM2	2
	SRTM data warped using EnviSat & ERS-2 data	XD_DEM_ACE2_SOURCE_SRTM3	3

Input	Description	Enumeration value	Long
	Mean lake level data derived from Altimetry	XD_DEM_ACE2_SOURCE_SRTM_LAKE	4
	GLOBE/ACE data warped using combined altimetry (only above 60°N)	XD_DEM_ACE2_SOURCE_SRTM_GLOBE	5
	Pure altimetry data (derived from ERS-1 Geodetic Mission, ERS-2 and EnviSat data using Delaunay Triangulation and Bilinear interpolation)	XD_DEM_ACE2_SOURCE_SRTM_ALT	6
DEM Data Source Types for GDEM v2	No source file; QA value contain the number of scene-based DEMs contributing to the final GDEM value for each 30m pixel (stack number)	XD_DEM_GDEM_SOURCE_NO_SOURCE_FILE	-1
	SRTM3 V3	XD_DEM_GDEM_SOURCE_SRTM3_V3	0
	SRTM3 V2	XD_DEM_GDEM_SOURCE_SRTM3_V2	1
	NED	XD_DEM_GDEM_SOURCE_NED	2
	CDED	XD_DEM_GDEM_SOURCE_CDED	3
	ALASKA DEM	XD_DEM_GDEM_SOURCE_ALASKA_DEM	4
IERS Bulletin type	Bulletin A	XD_BULLETIN_A	0
	Bulletin B	XD_BULLETIN_B	1
Data file type for xd_eocfi_file structure	Orbit file type (POF or ROF)	XD_ORBIT_FILE	0
	Orbit Scenario file	XD_OSF_FILE	1
	DORIS Navigator file	XD_DORIS_FILE	2
	IERS Bulletin file	XD_BULLETIN_FILE	3
	Generic list of state vectors	XD_USER_OSV_LIST	4
	SP3 file	XD_SP3_FILE	5
	OEM file	XD_OEM_FILE	6
DEM cache type	Computations performed without memory cache	XD_NO_CACHE	0
	Computations performed with preloadmemory cache	XD_PRELOAD_CACHE	1
	Computations performed with FIFO memory cache	XD_FIFO_CACHE	2
Attitude definition type enum	No model	XD_ATT_NONE_MODEL	0
	AOCS model	XD_ATT_DEF_AOCS_MODEL	1
	Parameter model	XD_ATT_PARAMETER_MODEL	2
	Harmonic model	XD_ATT_HARMONIC_MODEL	3
	File model	XD_ATT_FILE_MODEL	4
	Angle model	XD_ATT_ANGLE_MODEL	5
	Matrix model	XD_ATT_MATRIX_MODEL	6
	Quaternions plus angle model	XD_ATT_QUATERNION_ANGLE_MODEL	7
	Quaternions plus matrix model	XD_ATT_QUATERNION_MATRIX_MODEL	8
Attitude reference frame for definition	Satellite nominal attitude	XD_SAT_NOMINAL_ATT_DEF	0
	Satellite attitude	XD_SAT_ATT_DEF	1

Input	Description	Enumeration value	Long
	Instrument attitude	XD_INSTR_ATT_DEF	2
AOCS model	Geocentric pointing	XD_AOCS_GPM	0
	Local normal pointing	XD_AOCS_LNP	1
	Yaw steering + local normal pointing	XD_AOCS_YSM	2
	Zero-Doppler YSM	XD_AOCS_ZDOPPLER	3
Parameter model	Generic model	XD_MODEL_GENERIC	0
	ENVISAT model	XD_MODEL_ENVISAT	1
	CRYOSAT model	XD_MODEL_CRYOSAT	2
	ADM model	XD_MODEL_ADM	3
	SENTINEL1 model	XD_MODEL_SENTINEL1	4
	SENTINEL 2 model	XD_MODEL_SENTINEL2	5
	Geostationary model	XD_MODEL_GEO	6
Angle type	True Latitude (True of Date)	XD_ANGLE_TYPE_TRUE_LAT_TOD	0
	True Latitude (Earth Fixed)	XD_ANGLE_TYPE_TRUE_LAT_EF	1
SP3 file type	Only positions are provided in file. Velocities are computed by numerical derivation.	XD_SP3_POSITION_TYPE	0
	Positions and velocities are provided in the file.	XD_SP3_POSITION_PLUS_VELOCITY_TYPE	1
XD_DORIS_file_type_enum	DORIS file with Cryosat format	XD_DORIS_CRYOSAT_TYPE	0
	DORIS file with Sentinel-3 format	XD_DORIS_SENTINEL3_TYPE	1
	DORIS file with Jason-CS format	XD_DORIS_JASON_TYPE	2
	Unknown DORIS format	XD_DORIS_UNKNOWN_TYPE	3
SP3 file type descriptor	GPS satellites	XD_SP3_GPS	0
	Mixed: satellites from different systems are listed	XD_SP3_MIXED	1
	GLONASS satellites	XD_SP3_GLONASS	2
	Low Earth Orbit satellites	XD_SP3_LEO	3
	GALILEO satellites	XD_SP3_GALILEO	4
	COMPASS satellites	XD_SP3_COMPASS	5
	QZSS satellites	XD_SP3_QZSS	6
SP3 satellite descriptor	GPS satellite	XD_SAT_GPS	0
	GLONASS satellite	XD_SAT_GLONASS	1
	Low Earth Orbit satellite	XD_SAT_LEO	2
	GALILEO satellite	XD_SAT_GALILEO	3
	COMPASS satellite	XD_SAT_COMPASS	4
	QZSS satellite	XD_SAT_QZSS	5
SP3 Time system	GPS time system	XD_SP3_TIME_GPS	0
	GLONASS time system	XD_SP3_TIME_GLONASS	1
	GALILEO time system	XD_SP3_TIME_GALILEO	2
	TAI time system	XD_SP3_TIME_TAI	3
	UTC time system	XD_SP3_TIME_UTC	4
	QZSS time system	XD_SP3_TIME_QZSS	5
Time initialization mode	Read the whole file	XD_SEL_FILE	0
	Read only those OSVs that fits into the requested time interval	XD_SEL_TIME	1
	Read only those OSVs that fits into the requested orbit interval	XD_SEL_ORBIT	2

Input	Description	Enumeration value	Long
	Default behaviour (when applicable)	XD_SEL_DEFAULT	3
	Read only the header of the file. OSVs are not read.	XD_SEL_NONE	4
Extension type	Additional OSVs are loaded before the beginning and after the ending of the selected OSV interval. The number of additional OSVs is explicitly set by the user.	XD_EXTEND_NUM_OSV	0
	Additional OSVs are loaded before the beginning and after the ending of the selected OSV interval. The additional OSVs are contained in the time interval specified by the user (in seconds)	XD_EXTEND_TIME	1
FOV type	Starcraft constraints	XD_FOV_CONSTRAINTS_SC_LINK	0
	Celestial body constraints	XD_FOV_CONSTRAINTS_CELESTIAL_BODY	1
	Value not allowed	XD_FOV_CONTRAINTS_MAX	2
Earth Observation Ground Segment File Format Standard (FFS) version number	Default value. It can be used to re-set the mission dependent default FFS version.	XD_FFS_DEFAULT	0
	File Format Standard Version 1	XD_FFS_V1	1
	File Format Standard Version 2	XD_FFS_V2	2
	File Format Standard Version 3	XD_FFS_V3	3
DEM Cell position	DEM value given at the center of the cell	XD_DEM_CELL_CENTER	0
	DEM value given at the north-west corner of the cell	XD_DEM_CELL_NORTHWEST	1
	DEM value given at the north-east corner of the cell	XD_DEM_CELL_NORTHEAST	2
	DEM value given at the south-east corner of the cell	XD_DEM_CELL_SOUTHEAST	3
	DEM value given at the south-west corner of the cell	XD_DEM_CELL_SOUTHWEST	4
DEM data types	INT16	XD_DEM_DATA_TYPE_INT16	0
	INT32	XD_DEM_DATA_TYPE_INT32	1
	INT64	XD_DEM_DATA_TYPE_INT64	2
	UINT16	XD_DEM_DATA_TYPE_UINT16	3
	UINT32	XD_DEM_DATA_TYPE_UINT32	4
	UINT64	XD_DEM_DATA_TYPE_UINT64	5
	FLOAT32	XD_DEM_DATA_TYPE_FLOAT32	6
	FLOAT64	XD_DEM_DATA_TYPE_FLOAT64	7
	INT8	XD_DEM_DATA_TYPE_INT8	8
DEM data units	METER	XD_DEM_DATA_UNIT_METER	0
	KILOMETER	XD_DEM_DATA_UNIT_KILOMETER	1
DEM data reference	WGS84	XD_DEM_DATA_REFERENCE_WGS84	0
	EGM96	XD_DEM_DATA_REFERENCE_EGM96	1

The use of the previous enumeration values could be restricted by the particular usage within the different CFI functions. The actual range to be used is indicated within a dedicated reference named ***allowed range***. When there are not restrictions to be mentioned, the allowed range column is populated with the label ***complete***.

6.3.Data Structures

The aim of this section is to present the data structures that are used in the EO_DATA_HANDLING library. These structures are used as output/inputs in the reading/writing routines. The following table show the data structures with their names and the data that contains:

Table 3: EO_DATA_HANDLING Structures

Structure name	Description	Structure Data		
		Variable Name	C type	Description
xd_fhr	Fixed header data	file_name	char [XD_MAX_STR]	File name
		schema	char [XD_MAX_STR]	Schema file
		file_description	char [XD_MAX_STR]	File description
		mission	char [XD_MAX_STR]	Mission name
		file_class	char [XD_MAX_STR]	File class
		file_type	char [XD_MAX_STR]	File type
		version	long	File version
		eoffs_version	char[XD_MAX_STR]	File Format Standard
		val_start_date	char [32]	Validity start date
		val_stop_date	char [32]	Validity stop date
		system	char [XD_MAX_STR]	System name
		creator	char [XD_MAX_STR]	Creator name
		creator_version	char [XD_MAX_STR]	Creator version
		creation_date	char [32]	Creation date
xd_fileinfo	File info data for getting the default schema	sat_id	long	“Satellite ID” enumeration value (see [GEN_SUM])
		filetype	XD_File_types	File type (see enumeration in Table 2)
xd_bulb_table	Data for one entry read from a IERS bulletin	day	double	MJ200 UTC Time
		ut1_utc	double	Difference between UT1 and UTC
		ut1_tai	double	Difference between UT1and TAI

		Structure Data		
xd_iers_bulletin_b	Data for time corre lations read from a IERS bulletin	table1	xd_bulb_table[100]	First table data in the IERS bulletin
		table2	xd_bulb_table[100]	ble Difference between UT1and TAI
xd_eocfi_file_union	Union containin any of the following data structures	orbit_file	xd_orbit_file	Data from an orbit file (POF or ROF)
		osf_file	xd_osf_file	Data from an Orbit Scenario File
		doris_file	xd_doris_file	Data from a DORIS Navigator File
		bulletin_file	xd_iers_bulletin_file	Data from an IERS bulletin file (A or B)
		sp3_file	xd_sp3_file	Data from SP3 file
		oem_file	xd_oem_file	Data for OEM file
xd_eocfi_file	Data from an EOCFI file (Orbit file, OSF, DORIS Navigator or IERS file)	file_type	long	File type (according to XD_data_file_type_enum)
		eocfi_file	xd_eocfi_file_union	File data
xd_eocfi_file_set	Set of EOCFI files	num_files	long	Number of structures with the data from the files
		eocfi_file_array	xd_eocfi_file*	Array with the data structures
xd_polar_motion_params	Polar motion parameters read from IERS bulletins	x	double	x-axis is in the direction of the IERS Reference Meridian (IRM),
		y	double	y-axis is in the direction 90 degrees West longitude
xd_iers_bulletin_b_rec	Data for one entry read from a IERS bulletin B	day	double	MJD200 UTC time
		ut1_utc	double	Difference between UT1 and UTC
		ut1_tai	double	Difference between UT1 and TAI
		polar_motion_params	xd_polar_motion_params	Polar motion parameters

		Structure Data		
xd_iers_bulletin_a_rec	Data for one entry read from a IERS bulletin A	day	double	MJD200 UTC time
		ut1_utc	double	Difference between UT1 and UTC
		ut1_tai	double	Difference between UT1 and TAI
		polar_motion_params	xd_polar_motion_params	Polar motion parameters
xd_polar_motion_formula	Polar motion prediction formula paramters	ax	double	x parameter formula: constant term
		bx	double	x parameter formula: cos(A) coefficient
		cx	double	x parameter formula: sin(A) coefficient
		dx	double	x parameter formula: cos(C) coefficient
		ex	double	x parameter formula: sin(C) coefficient
		ay	double	y parameter formula: constant term
		by	double	y parameter formula: cos(A) coefficient
		cy	double	y arameter formula: sin(A) coefficient
		dy	double	y parameter formula: cos(C) coefficient
		ey	double	y parameter formula: sin(C) coefficient formula:
		A_ref	double	Reference day for A parameter formula
		A_div	double	Divisor for A parameter formula
		C_ref	double	Reference day for C parameter formula
		C_div	double	Divisor for C parameter formula
xd_time_correlation_formula	It contains the parameters for the UT1-UTC prediction formula	a	double	Constant parameter in formula
		b	double	Linear parameter in formula
		b_ref	double	Reference orbit in formula
xd_iers_bulletin_b_file	It contains values read from an IERS Bulletin B file	bulletin_id	char[XD_MAX_STR]	Bulletin date and issue
		num_final_table	long	Number of record in final table
		num_preliminary_table	long	Number of records in preliminary table
		final_table	xd_iers_bulletin_b_rec*	Memory allocated with number of elements num_final_table

		Structure Data		
		preliminary_table	xd_iers_bulletin_b_rec*	Memory allocated with number of elements num_preliminary_table
xd_iers_bulletin_a_file	It contains values read from an IERS Bulletin A file	bulletin_id	char[XD_MAX_STR]	Bulletin date and issue
		num_rec_pred_table	long	Number of record in Prediction table
		prediction_table	xd_iers_bulletin_a_rec*	Memory allocated with number of elements num_rec_pred_table
		polar_motion_formula	xd_polar_motion_formula	Parameters read for Polar motion formula
		time_correlation_formula	xd_time_correlation_formula	Parameters read for time correlation formula
xd_iers_bulletin_file_union (union C type)	It contains the values read from Bulletin A or Bulletin B (only one of them)	iers_bulletin_a_fila	xd_iers_bulletin_a_file	Bulletin A data
		iers_bulletin_b_file	xd_iers_bulletin_b_file	Bulletin B data
xd_iers_bulletin_file	Bulletin type and Bulletin data	bulletin_type	long	It can take the following values: –XD_BULLETIN_A –XD_BULLETIN_B –
		iers_bulletin_file	xd_iers_bulletin_file_union	Bulletin data union
xd_time_rec	It contains the time correlations for a given time	tai_time	double	TAI time
		ut1_time	double	UT1 time
		utc_time	double	UTC time
		tai_utc	double	Difference between TAI and
		tai_ut1	double	Difference between TAI and
		tai_gps	double	Difference between TAI and GPS time
xd_osv_rec	It contains a satellite state vector for a given time	tai_time	double	TAI time for the state vector
		utc_time	double	UTC time for the state
		ut1_time	double	UT1 time for the state
		abs_orbit	double	Absolute orbit
		ref_frame	long	Reference frame

		Structure Data		
		time_ref_of	long	Reference time to be considered as base. This value is related to Time_Reference tag in orbit file. This parameter takes the values given by enumeration *Reference time values* (see Table 2). For more details on this
		pos	double[3]	Position vector (x, y, z components)
		vel	double[3]	Velocity vector (x, y, z components)
		quality	double	Quality index for DORIS Preliminary and DORIS Precise Orbit files, this value corresponds with the enumeration “Quality Index” (See Table 2)
xd_orbit_file	Structure for storing the data read from an orbit file	num_rec	long	Number of records
		osv_rec	xd_osv_rec*	Array with the state vectors
xd_doris_file	Structure for storing the data read from a DORIS Navigator file	file_type	Long	DORIS File type (XD_DORIS_file_type_enum)
		num_rec	long	Number of records in
		osv_rec	xd_osv_rec*	State vectors array (EF)
		num_rec_j2	long	Number of records in osv_rec_j2
		osv_rec_j2	xd_osv_rec*	State vectors array
		leap_time	double	Leap time
		leap_sign	int	Leap time sign
		abs_orbit	long	First absolute orbit
		rel_orbit	long	First relative orbit number
xd_doris_mph_sph	Structure for the main and specific product headers	filename	char [XD_MAX_STR]	The description for these fields can be found in [PDS_FMT]
		sensing_start	char [30]	
		sensing_stop	char [30]	
		abs_orbit	long	
		delta_ut1	long	
		rel_orbit	long	
		leap_utc	char [XD_MAX_STR]	
		leap_sign	int	

		Structure Data		
		leap_err	int	
		num_dsd	long	
		ds_offset	long	
		num_dsr	long	
		proc_stage_code	char [10]	
		ref_doc	char [24]	
		proc_time	char [31]	
		software_version	char [20]	
		phase	char [2]	
		cycle	long	
		state_vector_time	char [31]	
		x_position	double	
		y_position	double	
		z_position	double	
		x_velocity	double	
		y_velocity	double	
		z_velocity	double	
		state_vector_source	char [3]	
		ascii_utc_time_before_leap	double	
		product_err	char [2]	
		tot_size	long	
		num_data_sets	long	
		sph_descriptor	char [29]	
		sensing_start_tai	char [31]	
		abs_orbit_start	long	
		rel_time_asc_node_start	double	
		sensing_stop_tai	char [31]	
		abs_orbit_stop	long	
		rel_time_asc_node_stop	double	
		equator_cross_time	char [31]	
		equator_cross_long	long	
		ascending_flag	char [2]	
		start_lat	long	

		Structure Data		
		start_long	long	
		stop_lat	long	
		stop_long	long	
		num_isps	long	
		num_missing_isps	long	
		num_error_isps	long	
		num_discarded_isps	long	
		num_rs_isps	long	
		num_rs_corrections	long	
		dsr_size	long	
xd_osf_rec	It contains the data for an orbital change in an orbit scenario file	abs_orb	long	Absolute orbit number
		rel_orb	long	Relative orbit number
		cycle_days	long	Cycle length in days
		cycle_orbits	long	Number of orbits in a cycle
		mlst	double	Mean local solar time (in hours)
		mlst_drift	double	Mean local solar time drift (seconds per day)
		inclination	double	Orbit inclination
		drift_mode	long	Flag for choosing between inclination of drift model
		anx_tai	double	ANX TAI time
		anx_ut1	double	ANX UT1 time
		anx_utc	double	ANX UTC time
		anx_long	double	ANX longitude
		cycle	long	Cycle number
		phase	long	Phase number
		time_ref_of	long	Reference time to be considered as base. This value is related to Time_Reference tag in orbit file. For OSF, this value is always XD_TIME_REF_OF_UT1 (see enumeration *Reference time values* in Table 1).

		Structure Data		
		anx_longitude_drift	xd_anx_longitude_drift	ANX longitude drift parameters (offset and linear term)
xd_osf_file	Structure for storing the data read from an orbit scenario file	num_rec	long	Number of records
		osf_rec	xd_osf_rec*	Array of state vectors
xd_swath_geometry	It contains the swath geometry	geom_type	long	Geometry type
		az	double[3]	Azimuth points
		el	double[3]	Elevation points
		alt	double[3]	Altitude points
		distance	double[3]	Distance
		angle	double[3]	Incidence angle
xd_harmonic_data		num_terms	long[3]	Number of harmonics coefficient(pitch, roll and yaw)
		harmonic_type_pitch	long[XD_MAX_NUM_HARMONIC]	Harmonic type
		harmonic_type_roll	long[XD_MAX_NUM_HARMONIC]	Harmonic type
		harmonic_type_yaw	long[XD_MAX_NUM_HARMONIC]	Harmonic type
		harmonic_coef_pitch	double [XD_MAX_NUM_HA RMONIC]	Harmonic coefficient
		harmonic_coef_r oll	double [XD_MAX_NUM_HA RMONIC]	Harmonic coefficient
		harmonic_coef_yaw	double [XD_MAX_NUM_HA RMONIC]	Harmonic coefficient
xd_param_model_str		Model	long	Model type. It can take the enumeration values given in Parameter model enum (see table 2)
		param_num	long	Number of parameters
		model_param	double [XD_NUM_MODEL_ PARAM]	Model Parameters

Structure name	Description	Structure Data		
		Variable Name	C type	Description
xd_harmonic_model_str		angle_type	long	Angle type. It can take the enumeration values given by Angle type enum (see table 2)
		harmonics	xd_harmonic_dat	Harmonic data
		offsets	double [3]	Offsets
xd_file_model_str		num_files	long	Number of files
		files	char **	file list
		aux_file	char *	Auxiliary file. This value must be set to NULL or the empty string if it is not used
		time_ref	long	Time reference
		time0	double	Start time
		time1	double	Stop time
xd_angle_model_str		angles	double [3]	angles
		offsets	double [3]	offsets
xd_matrix_model_str	Matrix model	att_matrix	double [3][3]	Attitude matrix model
		offsets	double [3]	Offsets
xd_attitutude_model_str	Attitude model structure	attitude_model	long	Attitude model type
		data	Attitude union data	Attitude union. One of the attitude structures.
Attitude union data	One of the following attitude structures	AOCS	long	AOCS model
		param_mode	xd_param_model_st	Parameters model
		harmonic_mode	xd_harmonic_model	Harmonic model
		file_mode	xd_file_model_str	File model
		angle_mode	xd_angle_model_	Angle Model
		matrix_mode	xd_matrix_model_str	Matrix Model
xd_asar_geometry	ASAR geometry	asar_type	long	ASAR Swath types
		slant_range_left	double	Parameter for narrow and wide ASAR
		slant_range_righ t	double	Parameter only for wide ASAR
xd_sdf_rec	Swath Definition data	swath_descr	char [XD_MAX_STR]	Swath description
		swath_id	char [XD_MAX_STR]	Swath_id
		swath_type	long	Swath type (XD_Swath_type_enum)
		num_swath_rec	long	Number of swath records to write in a single OEF

		Structure Data		
		refr_mode	long	Refraction mode (XD_Target_ray_enum)
		freq	double	Frequency (Hz)
		num_points	long	Number of points in the instantaneous swath
		swath_geom	xd_swath_geometry*	Swath geometry
		asar_geom	xd_asar_geometry	ASAR parameters
		sat_nom_att	xd_attitude_model_str *	Attitude data for sat. nominal att
		sat_att	xd_attitude_model_str *	Attitude data for sat. attribute
		instr_att	xd_attitude_model_str *	Attitude data for instrument att
xd_sdf_file	Swath definition file data	num_rec	long	Number of swath records in a SDF
		sdf_rec	xd_sdf_rec *	Swath record data array
xd_stf_pt	Swath point definition structure	lon	double	Longitude or RA
		lat	double	Latitude or Dec
xd_stf_rec	Swath template record data	num_points	long	Number of points in the instantaneous swath
		stf_pt	xd_stf_pt*	Array with the points of the instantaneous swath
xd_stf_vhr	Swath template variable header data	stf_name	char *	swath template file name
		Reference_OSF	char*	Reference OSF file name used to generate the STF
		Reference_SDF	char*	Reference swath definition file used to generate the STF
		swath_type	XD_Swath_type_en	Swath type
		swath_point_type	XD_Swath_point_ty	Swath point type
		time_step	double	
		refr_mode	long	Refraction model

		Structure Data		
		freq	double	Frequency (Hz)
		num_points	long	Number of points in the instantaneous swath
		altitude	double*	Array with the values of the altitudes of the points
		geom_flag	long	true if the geometry of the orbit is defined.
		rep_cycl	long	repeat cycle
		cycle_length	long	cycle length
		mlst_drift	double	MLST drift
		abs_orbit	long	Absolut orbit
		orbit_start	long	Start orbit of validity range
		orbit_stop	long	Stop orbit of validity range
		pos	double [3]	ANX position vector
		vel	double [3]	ANX velocity vector
xd_stf_file	Swath template file data	num_rec	long	number of points in the swath
		vhr	xd_stf_vhr	variable header
		stf_rec	xd_stf_rec *	array with the points in the swath
xd_att_rec	Attitude record	time_ref	long	Time reference
		time	double	time (MJD2000)
		data	double [4]	Quaternions or angles. For angles, the fourth value is dummy
xd_att_file	Attitude file data	sat_ref	long	target reference frame
		source_ref	long	initial reference frame: the set of angles/quaternions in this structure define a rotation from this “source_ref” to the attitude
		data_type	long	angles or quaternions (see XD_Attitude_data_type_enum
		num_rec	long	number of records in the att_rec array

		Structure Data		
		max_gap	double	Maximum time gap between
		att_rec	xd_att_rec*	array with the angle/ quaternion records
xd_tracker_limits	star trackers limits data	max_penalty	double	Maximum penalty for the quaternions
		norm_thr	double	Threshold for the modulus of
		max_gap	double	Maximum time gap between two consequtive quaternions
xd_tracker_conf_file	star trackers cofiguration file data	aberr_correction	long	Aberration correction flag: -1 = Aberration correction with transposed matrix 0 = No aberration 1 = Aberration correction
		satellite	char [XD_MAX_STR]	Satellite name
		str_limit	xd_tracker_limits	Star tracker limits
		str_att_rot	double [3][3]	Satellite Attitude to star tracker frame rotation matrix
xd_star_tracker	Star tracker record	quaternion	float[4]	Quaternions
		time	double	MJ2000 in TAI
		status	unsigned char	quaternion status
xd_star_tracker_file	star tracker file data	str_id	long	Star tracker Id (1,2 or 3)
		num_rec	long	number of lines
		str_rec	xd_star_tracker*	array with the star tracker records
xd_dem_ace	DEM configuration data for ACE model (deprecated)	dir	char[XD_MAX_PATH]	Directory where the DEM files are stored
		res_X	double	Interval between points along X-axis
		res_Y	double	Interval between points along Y-axis
		res_unit	double	Conversion factor from x,y units to the res_X, res_Y units. For example, if res_X is given in seconds and X in degrees then res_unit=3600
		X_num_points	long	Number of points along X-axis (columns)
		Y_num_points	long	Number of points along Y-axis (files)

		Structure Data		
		x_range	double	longitude of the X-axis for one file (grid).
		x_range	double	longitude of the Y-axis for one file (grid).
		data_size	long	Size in bytes of the data stored in the files
		data_type	long	data type (int, long, float, double)
		north_alt	double[4]	Altitude at the North pole cell
		south_alt	double[4]	Altitude at the South pole
		offset_x	double	Distance from the middle of a cell to the vertical side.
		offset_y	double	Distance from the middle of a cell to the horizontal side.
xd_dem_mini_tiles	Mini-tile configuration parameters for maximum altitude DEM algorithm	file_name	char[XD_MAX_PATH]	Name of the maximum altitude file
		lon_size	double	Mini-tile longitude size [degrees]
		lat_size	double	Mini-tile latitude size [degrees]
xd_dem_user_params	User configuration parameters for DEM	directory	char[XD_MAX_PATH]	Directory where the DEM files are stored
		cache_type	long	Cache type (DEM cache type enumeration)
		cache_max_size	long	Cache maximum size (in MegaBytes)
		mini_tiles	xd_dem_mini_tiles	DEM mini-tile configuration for maximum altitude algorithm
		geoid_computation	long	Flag to indicate if geoid computation must be performed or not (see DEM geoid flag enum)
		geoid_num_harmonics	long	Number of harmonics to be used in geoid computation
xd_dem_metadata	DEM metadata	model	long	DEM Model
		n_rows	long	Total number of rows in the DEM
		n_cols	long	Total number of columns in the DEM
		cell_location	XD_Dem_cell_location_enum	Location of the given altitude position in the cell
xd_dem_raster	DEM raster configuration for generic DEM	model	long	DEM model
		data_type	long	Data type (XD_Dem_data_types_enum)

		Structure Data		
		data_unit	long	Data unit (XD_Dem_data_units_enum)
		rows	long	Number of rows
		columns	long	Number of columns
		data_resolution	long	Resolution value
		data_res_unit	long	Factor to convert resolution to degrees
		data_reference	long	Data reference (XD_Dem_data_references_enum)
		void_value	long	Value that identifies a void value
		flag_type	long	Flag data type (XD_Dem_data_types_enum)
xd_dem_config_file	DEM configuration data	model	long	DEM model
		dem_data	xd_dem_ace *	DEM ACE data (deprecated)
		dem_user_params	xd_dem_user_params	User configuration parameters
		dem_metadata	xd_dem_metadata	DEM extra information
		dem_raster	xd_dem_raster	Configuration for DEM generic raster
xd_dem_point	DEM file point	lon	double	longitude
		lat	double	latitude
		alt	double	altitude
xd_dem_file	DEM file	num_points_X	long	Number of points along the longitude
		num_points_Y	long	Number of points along the latitude
		point	xd_dem_point**	DEM points
xd_star_rec	Star data	flag	long	True if the star was found in the star database file.
		star_id	char [XD_MAX_STR]	Star ID
		par	double	Parallax of the star at JD2000
		mu_ra	double	RA's proper motion at JD2000
		mu_dec	double	DEC's proper motion at JD2000 (rad/century)
		rad_vel	double	Radial velocity of the star (km/
		star_ra	double	RA of the star at JD2000 (rads)

		Structure Data		
		star_dec	double	DEC of the star at JD2000 (rads)
xd_star_file	Structure containing all relevant information contained in the star's database file	num_rec	long	Number of stars
		star_rec	xd_star_rec *	Array with all the star data
xd_station_rec	Station record data	station_id	char [XD_MAX_STR]	Station ID
		descriptor	char [XD_MAX_STR]	Description of the station
		antenna	char [XD_MAX_STR]	Describes the frequency band
		purpose	char [XD_MAX_STR]	Purpose
		type	char [XD_MAX_STR]	Not used.
		num_mask_pt	long	Number of points to define the
		azimuth	double [XD_VERTICES]	Azimuth and elevation defining the antenna mask.
		elevation	double [XD_VERTICES]	
		station_long	double	Station longitude
		station_lat	double	Station latitude
		station_alt	double	Station altitude
		proj_long	double [XD_VERTICES]	longitude/latitude points for the station zone that are equivalent to the set of azimuth/elevation points. The longitude/latitude points are not read from the file but computed in xv_station_vis_time.
		proj_lat	double [XD_VERTICES]	
		points	long	Number of points in the azimuth/elevation and in
		long_max	double	Maximum longitude of the station zone
		lat_max	double	Maximum latitude of the station zone

		Structure Data		
		long_min	double	Minimum longitude of the station zone
		lat_min	double	Minimum latitude of the station
		mission_list	long	Number of spacecrafts defined for the station
		mission_name	char[XD_MISSIONS][XD_MAX_STR]	Names of the spacecrafts defined for the station
		mis_aos_el	double[XD_MISSIONS]	Elevations for acquisition of signal to defined spacecrafts
		mis_los_el	double[XD_MISSIONS]	Elevations for loss of signal to the defined spacecrafts
		mask_type	char[XD_MISSIONS][XD_MAX_STR]	Mask type for the spacecrafts defined in the station. Possible values: AOS_LOS_WITH_MASK AOS_LOS MASK_ONLY
xd_station_file		num_rec	long	Number of stations
		station_rec	xd_station_rec *	Array of station records
xd_zone_point	Longitude and latitude point	pt_long	double	Longitude
		pt_lat	double	Latitude
xd_zone_rec	Zone record data	zone_id	char [XD_MAX_STR]	Zone ID
		description	char [XD_MAX_STR]	Drescription of the zone
		surface	char [XD_MAX_STR]	Surface type
		creator	char [XD_MAX_STR]	Creator name
		zone_type	XD_Zone_type_enum	Zone type
		projection	long	Projection
		zone_diam	double	Zone diameter in meters. Only used when the ZONE is a POINT zone or a CIRCULAR zone.
		num_points	long	Number of ZONE points (last one, equal to the first one, included)
		zone_point	xd_zone_point *	Array of points of the zone
xd_zone_file	Zone file structure	num_rec	long	Number of zones
		zone_rec	xd_zone_rec *	Array of zone records
xd_scf_appear	Apearance data for	colour	long	Colour (hexadecimal value from 0x000000 to

		Structure Data		
	swath configuration files			0xFFFFFF)
		draw	long	Draw (see enumeration in Table 2)
		fill	long	Fill (see enumeration in Table 2)
		opacity	long	Opacity (0-100%)
xd_tle_rec	TLE record. It contains data for a TLE	norad_sat_cat	char[25]	Satellite name consistent with the NORAD SATCAT
		sat_number	long	NORAD Catalogue number
		classification	char	Classification: U=unclassified, S=secret
		int_des	char [9]	International Designator: (Last two digits of launch year) (Launch number of the year) (Piece of the
		time	double	reference time for the element set (UTC processing days MJ2000)
		n_1st	double	First Time Derivative of the Mean Motion
		n_2nd	double	Second Time Derivative of Mean Motion
		bstar	double	BSTAR drag term
		ephemeris_type	int	Ephemeris type
		index	int	Element number
		checksum1	int	Checksum for line 1
		i	double	inclination [Degrees]
		ra	double	Right Ascension of the Ascending Node [Degrees]
		e	double	Eccentricity
		w	double	Argument of Perigee [Degrees]
		m	double	Mean Anomaly [Degrees]
		n	double	Mean Motion [Revs per day]
		abs_orbit	long	Revolution number at epoch
		checksum2	int	Checksum for line 2
xd_tle_file	Structure to store the data from a TLE file	num_rec	long	Number of records (TLE)
		tle_rec	xd_tle_rec*	Array with of TLE records

		Structure Data		
xd_propag_pre cise_config	Parameters for precise propagation configuration	user_flag	long	Indicates if default (0) or user defined (1) values are used for some parameters.
		models_path	char[XD_MAX_PATH]	Path where files necessary for
		gravity_flag	long	Gravity perturbation used (1)
		thirdbody_flag	long	Third bodies (Sun and Moon) perturbation used (1) or not
		atmos_flag	long	Atmosphere perturbation used
		srp_flag	long	Solar radiation pressure perturbation used (1) or not
		step	double	Simulation step (seconds).
		grav_file	char[XD_MAX_PATH]	File with data of gravitational model.
		grav_degree	long	Degree used gravity model.
		grav_order	long	Order used in gravity
		sga_flag	long	ap, f107 and f107a parameters used (0) or data read from files sga_ap_file and sga_f107_file (1).
		sga_ap_file	char[XD_MAX_PATH]	File with Geomagnetic Activity index values.
		sga_f107_file	char[XD_MAX_PATH]	File with F10.7 Solar Activity index values.
		ap	double	Geomagnetic Activity Index (daily value).
		f107	double	F10.7 Index Solar Activity Index (daily value).
		f107a	double	F10.7 Index Solar Activity Index (value averaged over 3 months).
		sc_mass	double	S/C mass [kg].
		sc_drag_area	double	S/C effective drag area [m^2].
		sc_drag_coeff	double	S/C drag coefficient.
		sc_srp_area	double	S/C effective Solar Radiation Pressure area [m^2].

		Structure Data		
		sc_srp_coeff	double	S/C Solar Radiation Pressure coefficient.
xd_quaternion_plus_angle	Quaternions plus angles model	quat_def_file	char*	Name of the file (with full or relative path) where quaternions are stored
		angle_model	xd_angle_model_str	Angles value
xd_quaternion_plus_matrix	Quaternions plus matrix model	quat_def_file	char*	Name of the file (with full or relative path) where quaternions are stored
		matrix_model	xd_matrix_model_str	Rotation matrix
xd_osv_rec_sp3	Information corresponding to a satellite in SP3 file	type	long	Satellite type (GPS/GLONASS/LEO/GALILEO/COMPASS/GZSS). See SP3 satellite descriptor enumeration
		identifier	long	Identifier number for satellite
		id_string	char[4]	Satellite identifier as is found in SP3 file
		sat_accuracy	long	Satellite accuracy
		num_rec	long	Number of state vectors for satellite
		osv_rec	xd_osv_rec*	Array of state vectors corresponding to satellite
xd_sp3_file	SP3 file	type	long	position of position+velocity (see SP3 type enum)
		global_time_start	double	Gregorian initial time for all the file (MJD2000).
		num_rec	long	Number of epochs
		data_used	char[6]	Data used descriptor.
		coordinate_system	char[6]	Name of the coordinate system used as written in
		orbit_type	char[4]	Orbit type descriptor.
		agency	char[5]	Name of the agency that generated the file.
		gps_week	long	GPS week.
		seconds_of_week	double	Seconds of the week elapsed at the start of the

		Structure Data		
		epoch_interval_seconds	double	Epoch interval in seconds.
		julian_date_start	double	Modified Julian day start.
		fractional_day	double	Fractional part of the day (0.0 <= fractional < 1.0) at the start of the orbit.
		num_sat	long	Number of satellites.
		file_type_descriptor	long	File descriptor (See SP3 file descriptor enum)
		time_system_indicator	long	Time system (see SP3 time system enum).
		pos_vel_std_dev	double	Base number used for computing the standard deviations for the components of the satellite position and velocity (units: mm and 10**-4 mm/sec).
		clock_std_dev	double	Base number used for computing the standard deviations for the components of the satellite position and velocity (units: mm and 10**-4 mm/sec).
		comments	char*[4]	Comments in lines 19 to 22.
		delta_tai_gps	double	Difference in seconds between TAI and GPS times (TAI-GPS)
		osv_rec_sp3	xd_osv_rec_sp3*	Array with Information for every satellite in SP3 file (including state vectors). Each position in array corresponds to one satellite, in the order provided in the file
xd_oem_file	OEM file	ccsds_oem_vers	char[4]	Format version in the form of ‘x.y’

		Structure Data		
		comment_header	char[XD_MAX_STR]	Comments
		creation_date	char[24]	File creation date and time in UTC
		originator	char[XD_MAX_STR]	Creating agency or operator
		comment_metadata	char[XD_MAX_STR]	Comments
		object_name	char[XD_MAX_STR]	The name of the object for which the ephemeris is
		object_id	char[XD_MAX_STR]	Object identifier of the object for which the
		center_name	char[XD_MAX_STR]	Origin of reference frame
		ref_frame	char[XD_MAX_STR]	Name of the reference frame in which the ephemeris data are given
		ref_frame_epoch	char[24]	Epoch of reference frame
		time_system	char[3]	Time system used for metadata
		start_time	char[24]	Start of TOTAL time span covered by ephemeris data and covariance data immediately following this
		useable_start_time	char[24]	Optional start of USEABLE time span
		useable_stop_time	char[24]	Optional end of USEABLE time span
		stop_time	char[24]	End of TOTAL time span covered by ephemeris data and covariance data immediately following this
		interpolation	char[XD_MAX_STR]	This keyword may be used to specify the recommended interpolation method for ephemeris data
		interpolation_degree	char[1]	Recommended interpolation degree for ephemeris data in the immediately following set of ephemeris lines

		Structure Data		
		osv_rec	xd_osv_rec*	OSV records
xd_attitude_definition_data	Attitude definition data for Sat nom, sat and instrument	att_def_file_dir_path	char*	Directory where the Atittude DEF file is placed
		sat_nom_att	xd_attitude_definition_model_str *	Satellite nominal attitude initialization data
		sat_att	xd_attitude_definition_model_str *	Satellite attitude initialization data
		instr_att	xd_attitude_definition_model_str *	Instrument attitude initialization data
xd_attitude_definition_model_str	Attitude definition	attitude_model	long	Attitude model type (see attitude definition type enum in table 2)
		data	union { long AOCS; xd_param_model_str param_mode; xd_harmonic_model_str harmonic_mode; xd_file_model_str file_mode; xd_angle_model_str angle_mode; xd_matrix_model_str matrix_mode; xd_quaternion_plus_angle quaternion_angle_mode; xd_quaternion_plus_matrix quaternion_matrix_mode; }	Union with all the possible models of initialization AOCS can take the enumeration values given by AOCS enum (see table 2)
xd_orbit_file_diagnostics_settings	Diagnostics settings structure	gap_threashold	double	time to identify a gap [s]
		duplicated_osv_threashold	double	time to identify a duplicated OSV [s]
		time_step	double	expected time step [s]
		time_step_threshold	double	time step threshold, to identify non-equally spaced OSVs [s]

		Structure Data		
		time_ref	long	time system that will be used to fill time related fields in the report structure
xd_orbit_file_diagnostics_report	Diagnostics report structure	num_osv	long	number of OSVs which were checked
		total_time	double	total time covered by the file (i.e. from first to last OSV)
		time_first_osv	double	time of first OSV
		time_last_osv	double	time of last OSV
		time_ref	long	time system of time related fields in this structure
		time_start_gap	double *	list containing start time of GAPs
		time_stop_gap	double *	list containing stop time of GAPs
		index_gap	long *	list containing index of GAPs (the index represents the ID of OSV which is preceded by a GAP)
		num_gaps	long	number of identified GAPs
		time_going_back_osv	double *	list containing time of going back OSVs
		index_going_back_osv	long *	list containing index of going back OSVs
		num_going_back_osv	long	number of identified going back OSVs
		time_duplicated_osv	double *	list containing time of duplicated OSVs
		index_duplicated_osv	long *	list containing index of duplicated OSVs
		num_duplicated_osv	long	number of identified duplicated OSVs
		time_inconsistent_orbit_number	double *	list containing time of OSVs with inconsistent orbit number
		index_inconsistent_orbit_number	long *	list containing index of OSVs with inconsistent orbit number
		num_inconsistent_orbit_number	long	number of OSVs with inconsistent orbit number

		Structure Data		
		time_non_equally_spaced_osv	double *	list containing time of non equally spaced OSVs
		index_non_equally_spaced_osv	long *	list containing index of non equally spaced OSVs
		num_non_equally_spaced_osv	long	number of OSVs with time step different from expected (absolute value of difference from step and expected > threshold)
xd_osv_list_read_configuration	Configuration for reading OSV state vectors	time_mode	long	Time initialization mode: XD_Time_init_mode_enum
		time_ref	long	Time reference: XD_Time_ref_enum
		extend_type	long	Extension type: XD_Extend_type_enum
		time_start	double	Initialization time interval (only applicable if time_mode == XD_SEL_TIME)
		time_stop	double	Initialization time interval (only applicable if time_mode == XD_SEL_TIME)
		orbit_start	long	Initialization ORBIT interval (only applicable if time_mode == XD_SEL_ORBIT)
		orbit_stop	long	Initialization ORBIT interval (only applicable if time_mode == XD_SEL_ORBIT)
		extend_num_osv	long	Number of OSVs to be added to initialization interval (only applicable if extend_type == XD_EXTEND_NUM_OSV)
		extend_osv_sec	double	Size of interval whose OSVs must be added before/after input interval (only applicable if extend_type == XD_EXTEND_TIME)
xd_az_el_mask	Antenna mask	num_mask_pt	Long	Number of azimuth and elevation pairs defining the antenna mask
		status	long	Allow the user to enable/disable masks; The behaviour of the status field is described

		Structure Data		
				below for each type of mask: **Inclusive mask:** Status = XL_FALSE: no constraints (regardless of number of points) Status = XL_TRUE and number of points = 0 : no constraints **Exclusive mask:** Status = XL_FALSE: mask is ignored (regardless of number of points) Status = XL_TRUE and number of points = 0 : mask is ignored **Combining the two above:** Each mask define a polygon. Forbidden areas are: 1) the area OUTSIDE the inclusive polygon; 2) the area INSIDE the exclusive polygon;
		azimuth	double [XD_VERTICES]	Azimuth defining the antenna mask
		elevation	double [XD_VERTICES]	Elevation defining the antenna
xd_link_mask	Mask description	incl_mask	xd_az_el_mask	List of azimuth and elevation pairs in Instrument Frame defining an inclusive zone
		excl_mask	xd_az_el_mask	List of azimuth and elevation pairs in Instrument Frame defining an exclusive zone
xd_link_data	Link description	mask_data	xd_link_mask	List of azimuth and elevation pairs in Instrument Frame
		min_tg_height	double	Minimum tangent height
xd_fov_constraints_union	Fov constraints	sc_link_data	xd_link_data	Satellite link data
	Fov constraints	celestial_body_link_data	xd_link_data	Celestial Body link data
xd_fov_constraints_file	FOV file	type	long	FOV constraints type (FOV constraints enum)
		constraints	xd_fov_constraints_union	Constraints

		Structure Data		
xd_anx_longitude_drift	ANX longitude drift	Offset	Double	Longitude offset
		Linear_term	Double	Drift linear term
xd_mlst_nonlinear_drift	MLST nonlinear drift	linear_approx_validity	long	Number of orbits in which MLST linear approximation is valid
		quadratic_term	double	MLST quadratic term
		nof_harmonics	long	Number of harmonics
		mlst_harmonics	xd_mlst_harmonics*	Array of Harmonics allocated with number of elements nof_harmonics
xd_aem_metadata	Data related to the data stored in a segment of an AEM file	comment	char[XD_MAX_STR]	Comment
		object_name	char[XD_MAX_STR]	Spacecraft name of the object corresponding to the attitude data to be given.
		object_id	char[XD_MAX_STR]	Spacecraft identifier
		center_name	char[XD_MAX_STR]	Origin of reference frame,
		ref_frame_A	char[XD_MAX_STR]	The name of the reference frame specifying one frame of the transformation
		ref_frame_B	char[XD_MAX_STR]	Name of the reference frame specifying the second portion of the transformation
		attitude_dir	char[4]	Rotation direction of the attitude specifying from which frame the transformation is to: - A2B specifies a transformation from the REF_FRAME_A to the REF_FRAME_B - B2A specifies a transformation from the REF_FRAME_B to the REF_FRAME_A.
		time_system	char[4]	Time system used for attitude and maneuver data
		start_time	char[25]	Start of TOTAL time span covered by attitude ephemeris data immediately following the metadata block.

		Structure Data		
		useable_start_time	char[25]	Optional start of USEABLE time span covered by attitude ephemeris data immediately following the metadata block
		stop_time	char[25]	End of TOTAL time span covered by the attitude ephemeris data immediately following the metadata block.
		useable_stop_time	char[25]	Optional end of USEABLE time span covered by attitude ephemeris data immediately following the metadata block
		attitude_type	char[XD_MAX_STR]	The format of the data lines in the message: QUATERNION QUATERNION/DERIVATIVE QUATERNION/RATE EULER_ANGLE EULER_ANGLE/RATE SPIN SPIN/NUTATION
		quaternion_type	char[XD_MAX_STR]	The placement of the scalar portion of the quaternion: FIRST LAST
		euler_rot_seq	char[4]	The rotation sequence of the Euler angles that rotate from REF_FRAME_A to REF_FRAME_B
		rate_frame	char[15]	The frame of reference in which Euler rates are specified
		interpolation	char[XD_MAX_STR]	Recommended interpolation method for attitude ephemeris data
		interpolation_degree	char[2]	Recommended interpolation degree for attitude ephemeris data
xd_aem_att_re	Attitude record.	data_type	long	Data type:

		Structure Data		
c	Contains for a given time a set of: - angles - quaternions - spin/nutation			XD_ATT_QUATERNIONS XD_ATT_ANGLES -1 for SPIN/NNUTACION data
		time_ref	long	Time reference for the time of the current record
		time	double	Time for the current record.
		data	double[8]	Atttitude data as it appears in the AEM file
xd_aem_attitude	Set of attitude records contained in an AEM segment	att_rec	xd_aem_att*	Array of attitude records
		num_rec	long	Number of attitude records
xd_aem_segment	Data contained in an AEM segment	metadata	xd_aem_metadata	Segment metadata
		attitude	xd_aem_attitude	Attitude data
xd_aem_file	Data from an AEM file	ccsds_aem_vers	char[5]	Format version
		comment_header	char[XD_MAX_STR]	Comments in header
		creation_date	char[25]	File creation date
		originator	char[XD_MAX_STR]	Creating agency
		segment	xd_aem_segment*	Array of segments
		num_segment	long	Number of segments in file

7. CFI FUNCTIONS DESCRIPTION

The following sections describe each CFI function. The calling interfaces are described for C.

Input and output parameters of each CFI function are described in tables, where C programming language syntax is used to specify:

- Parameter types (e.g. long, double)
- Array sizes of N elements (e.g. param[N])
- Array element M (e.g. [M])

7.1.xd_read_fhr

7.1.1. Overview

The **xd_read_fhr** CFI function reads the fixed header for Earth ExplorerObservation XML files.

7.1.2. Calling interface

The calling interface of the **xd_read_fhr** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 char *file_name;
 xd_fhr fhr;
 long ierr[XD_NUM_ERR_READ_FHR];
 status = xd_read_fhr(file_name, &fhr, ierr);
}
```

7.1.3. Input parameters

The **xd_read_fhr** CFI function has the following input parameters:

Table 4: Input parameters of xd_read_fhr function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	file name	-	-

7.1.4. Output parameters

The output parameters of the **xd_read_orbit_file** CFI function are:

Table 5: Output parameters of xd_read_fhr function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xd_read_fhr	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
Fixed header data	xd_fhr	-	Data structure containing the data read from the fixed header	-	-
ierr	long[]	-	Error vector	-	-

7.1.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_fhr** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EOXPLORER_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_fhr** function by calling the function of the EOXPLORER_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 6: Error messages of xd_read_fhr function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Could not open the file	No calculation performed	XD_CFI_READ_FHR_OPEN_FILE_ERR	0
ERR	Error reading the fixed header	No calculation performed	XD_CFI_READ_FHR_GET_FIXED_HEADER_ERR	1
ERR	Error closing the file	No calculation performed	XD_CFI_READ_FHR_CLOSE_FILE_ERR	2

7.2.xd_read_bulletin

7.2.1. Overview

The **xd_read_bulletin** CFI function reads IERS bulletin files and returns the data relevant for time correlations. Either version 1980 as version 2010 of the IERS bulletins can be read.

This function is deprecated, it is recommended to use xd_read_bulletin_2.

7.2.2. Calling interface

The calling interface of the **xd_read_bulletin** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 char *bulb_file;
 xd_iers_bulletin_b iers_data
 long ierr[XD_NUM_ERR_READ_BULLETIN];
 status = xd_read_bulletin (bulb_file, &iers_data, ierr);
}
```

7.2.3. Input parameters

The **xd_read_bulletin** CFI function has the following input parameters:

Table 7: Input parameters of xd_read_bulletin function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
bulb_file	char*	-	File name	-	-

7.2.4. Output parameters

The output parameters of the **xd_read_bulletin** CFI function are:

Table 8: Output parameters of xd_read_bulletin function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xd_read_bulletin	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-

IERS bulletin data	xd_iers_bulletin_b	-	Data structure containing the data read from the file	-	-
ierr	long[]	-	Error vector	-	-

7.2.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_bulletin** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_bulletin** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 9: Error messages of xd_read_bulletin function

Error type	Error message	Cause and impact	Error code	Error No
ERR	File does not exist	No calculation performed	XD_CFI_READ_BULLETIN_FILE_ERR	0
ERR	Time table is empty or has wrong format	No calculation performed	XD_CFI_READ_BULLETIN_TABLE_ERR	1
ERR	File is not recognized	No calculation performed	XD_CFI_READ_BULLETIN_FILE_RECOG_ERR	2

7.3.xd_read_bulletin_2

7.3.1. Overview

The **xd_read_bulletin_2** CFI function reads IERS bulletin A and B files and returns the data relevant for time correlations and polar motion. Only version 2010 of the IERS bulletin B can be read.

7.3.2. Calling interface

The calling interface of the **xd_read_bulletin_2** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 char *bulb_file;
 xd_iers_bulletin_file iers_data
 long ierr[XD_NUM_ERR_READ_BULLETIN];
 status = xd_read_bulletin_2 (bulb_file, &iers_data, ierr);
}
```

7.3.3. Input parameters

The **xd_read_bulletin** CFI function has the following input parameters:

Table 10: Input parameters of xd_read_bulletin_2 function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
bulb_file	char*	-	File name	-	-

7.3.4. Output parameters

The output parameters of the **xd_read_bulletin** CFI function are:

Table 11: Output parameters of xd_read_bulletin_2 function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xd_read_bulletin_2	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-

IERS bulletin data	xd_iers_bulletin_file	-	Data structure containing the data read from the file	-	-
ierr	long[]	-	Error vector	-	-

7.3.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_bulletin_2** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_bulletin_2** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 12: Error messages of xd_read_bulletin_2 function

Error type	Error message	Cause and impact	Error code	Error No
ERR	File does not exist	No calculation performed	XD_CFI_READ_BULLETIN_FILE_ERR	0
ERR	Time table is empty or has wrong format	No calculation performed	XD_CFI_READ_BULLETIN_TABLE_ERR	1
ERR	File is not recognized	No calculation performed	XD_CFI_READ_BULLETIN_FILE_RECOG_ERR	2

7.4. xd_free_bulletin

7.4.1. Overview

The **xd_free_bulletin** CFI function frees the memory allocated during the reading function xd_read_bulletin_2.

7.4.2. Calling interface

The calling interface of the **xd_free_bulletin** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_iers_bulletin_file bulletin_data;
 xd_free_bulletin (&bulletin_data);
}
```

7.4.3. Input parameters

The **xd_free_bulletin** CFI function has the following input parameters:

Table 13: Input parameters of xd_free_bulletin function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
bulletin_data	xd_iers_bulletin_file	-	Bulletin file data structure	-	-

7.4.4. Output parameters

This function does not return any value nor parameters.

7.5.xd_read_orbit_file

7.5.1. Overview

The **xd_read_orbit_file** CFI function reads orbit files for Earth Observation Missions. The files have to be written in XML and consist on a list of state vectors of the satellite along the orbit.

A warning is raised if at least one of the following conditions is detected:

- OSV with time going back
- OSV with repeated time
- gap (that is, the separation between one OSV and the following one is more than 330 seconds)
- inconsistency in orbit number (that is, the orbit number should not decrease between one OSV and the following one)

7.5.2. Calling interface

The calling interface of the **xd_read_orbit_file** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 char *file_name;
 long read_fro_flag, time_orbit_flag, time_ref, reading_osv_flag;
 double start_range, stop_range;
 xd_orbit_file orbit_data
 long ierr[XD_NUM_ERR_READ_ORBIT_FILE];
 status = xd_read_orbit_file (file_name, &read_fro_flag,
                              &time_orbit_flag, &time_ref,
                              &start_range, &stop_range,
                              &reading_osv_flag,
                              &orbit_data, ierr);
}
```

7.5.3. Input parameters

The **xd_read_orbit_file** CFI function has the following input parameters:

Table 14: Input parameters of xd_read_orbit_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	Orbit file name	-	-

read_fro_flag	long*	-	flag to indicate if the input file is: • a predicted orbit file • a restituted orbit file or a DORIS Preliminary file		• XD_TRUE for ROF and DORIS files • XD_FALSE for POF files
time_orbit_flag	long*	-	Flag for selecting the time range of the initialisation. Select either: • XD_SEL_FILE: for reading the whole file • XD_SEL_ORBIT: for reading the interval given by the start_range and the stop range parameters in orbits • XD_SEL_TIME: for reading the interval given by the start_range and the stop range parameters in days	-	All
time_ref	long*	-	Time reference if time_orbit_flag is XD_SEL_TIME. Dummy otherwise.	-	-
reading_osv_flag	long*	-	flag to indicate if the state vectors data have to be read.	-	• XD_TRUE for reading the state vector data • XD_FALSE for reading just the times and orbit numbers
start_range	double*	-	Start orbit or day	orbits or days	-
stop_range	double*	-	Stop orbit or day	orbits or days	-

It is possible to use enumeration values rather than integer values for some of the input arguments:

- Time model ID: time_model. See [GEN_SUM].
- Time reference ID: time_ref. See [GEN_SUM].
- Time range initialisation flag: time_orbit_flag. See current document, section 6.2

7.5.4. Output parameters

The output parameters of the **xd_read_orbit_file** CFI function are:

Table 15: Output parameters of xd_read_orbit_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range

xd_read_orbit_file	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
orbit_data	xd_orbit_file	-	Data structure containing the data read from the file	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *orbit_data* structure contains pointers to memory allocated dinamically. In order to avoid memory leaks, the user will have to free that memory when the data structure is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_orbit_file**

7.5.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_orbit_file** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_orbit_file** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 16: Error messages of xd_read_orbit_file function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Error in reading file	No calculation performed	XD_CFI_READ_ORBIT_FILE_READ_ERR	0
ERR	Variable header not found	No calculation performed	XD_CFI_READ_ORBIT_FILE_VHR_NOT_FOUND_ERR	1
ERR	Error in getting the first element inside the input range	No calculation performed	XD_CFI_READ_ORBIT_FILE_INPUT_RANGE_ERR	2
ERR	Error allocating memory	No calculation performed	XD_CFI_READ_ORBIT_FILE_MEMORY_ERR	3

ERR	Internal Error # 1	No calculation performed	XD_CFI_READ_ORBIT_FILE_INTERNAL_1_ERR	4
ERR	Error while reading data	No calculation performed	XD_CFI_READ_ORBIT_FILE_DATA_READ_ERR	5
ERR	Gap found after OSV no. %li	No calculation performed	XD_CFI_READ_ORBIT_FILE_GAP_ERR	6
WARN	Ref_Frame tag is missing. Earth Fixed assumed.	File read	XD_CFI_READ_ORBIT_FILE_REF_CS_WARN	7
WARN	Time_Reference tag is missing. Input time_ref parameter assumed.	File read	XD_CFI_READ_ORBIT_FILE_DEFAULT_TIME_REF_OF_WARN	8
WARN	Repeated OSVs found	File read	XD_CFI_READ_ORBIT_FILE_REPEATED_OSV_WARN	9
WARN	Gap found between OSV	File read	XD_CFI_READ_ORBIT_FILE_GAP_WARN	10
WARN	Going back OSVs found	File read	XD_CFI_READ_ORBIT_FILE_TIME_GOING_BACK_WARN	11
WARN	Inconsistency in orbit number found	File read	XD_CFI_READ_ORBIT_FILE_ORBIT_NUMBER_WARN	12
WARN	Time gap between OSVs larger than 300s detected (may lead to insufficient accuracy in case of highly eccentric MEO)	File read	XD_CFI_READ_ORBIT_FILE_MEO_GAP_WARN	13

7.6.xd_free_orbit_file

7.6.1. Overview

The **xd_free_orbit_file** CFI function frees the memory allocated during the reading function xd_read_orbit_file.

7.6.2. Calling interface

The calling interface of the **xd_free_orbit_file** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_orbit_file orbit_data xd_free_orbit_file (&orbit_data);
}
```

7.6.3. Input parameters

The **xd_free_orbit_file** CFI function has the following input parameters:

Table 17: Input parameters of xd_free_orbit_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_data	xd_orbit_file	-	Orbit data structure	-	-

7.6.4. Output parameters

This function does not return any value nor parameters.

7.7.xd_read_doris

7.7.1. Overview

The **xd_read_doris** CFI function reads DORIS Navigator files for Cryosat, Sentinel 3 and Jason CS (the function detects automatically the type of file).

This function considers only packets of type NAV_T (ITRF), NAV_I (J2000) and NAV_G (Geodetic) -- detected based on their API. If the input file contains any other packet (i.e. packets with other APIDs), the reading procedure will fail with an error.

The description of S3 DORIS can be found in CNES doc CO-SP-D0-EA-16222-CN (note: it is an internal CNES document). Note: Jason CS DORIS follows a similar format.

The following items must be considered:

- Available navigation data is extracted from packets of types NAV_T (ITRF) and NAV_I (J2000) -- n.b. the information in NAV_I (J2000) is not actually used for orbit initialization. From packets of type NAV_G (Geodetic) only the absolute orbit number is extracted. If packets of this type are not available, the orbit number is set to 1 at the first OSV and increased are each ANR -- n.b the Cryosat DORIS files never contain NAV_G (Geodetic) packets.
- During reading operation, the following issues are taken into account:
  1) A packet is discarded and a warning is raised with the packet number if at least one of the following conditions is detected:
     - CRC error (only for Sentinel 3);
     - quality field = 0xFFFFFFFF (packet not valid);
     - OSV time going back or repeated.
  2) It is assumed that, within the file, packets with same APID are sorted by sequence counter and the sequence counter is increasing by 1. If it is not increased by one a warning is raised with the packet id where the difference was found.
  3) If a gap is found in the file (that is, the separation between one OSV and the following one is more than 1.5 times the nominal rate of the DORIS files, which is 10 seconds), a warning is raised with the packet id where the gap was found.
  4) Apart from packets discarded due to conditions listed in 1), all OSVs contained in the packets will be loaded in the output data structure, regardless of any other non-nominal condition (as the ones described in 2) and 3)).

7.7.2. Calling interface

The calling interface of the **xd_read_doris** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *doris_file;
 long time_mode, interpol_flag;
```

```
    double time0, time1;
    xd_doris_file doris_data
    long ierr[XD_NUM_ERR_READ_DORIS];

    status = xd_read_doris(doris_file, &time_mode,
                           &time0, &time1,
                           &interpol_flag,
                           &doris_data, ierr);
}
```

7.7.3. Input parameters

The **xd_read_doris** CFI function has the following input parameters:

Table 18: Input parameters of xd_read_doris function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
doris_file	char*	-	DORIS Navigator file name	-	-
time_mode	long	-	Flag for reading the whole file or just the requested time window	-	• XD_SEL_FILE • XD_SEL_TIME
time0	double	-	Start time for the requested time window (if XD_SEL_TIME selected)	days in UTC	-
time1	double	-	Stop time for the requested time window (if XD_SEL_TIME selected)	days in UTC	-
interpol_flag	long	-	Flag to indicate if the read data are used for interpolation purposes. In that case 4 extra state vectors are read out of the requested time window	-	• XD_TRUE for interpol data • XD_FALSE otherwise

It is possible to use enumeration values rather than integer values for some of the input arguments:

- Time model ID: time_mode. See [GEN_SUM].

7.7.4. Output parameters

The output parameters of the **xd_read_doris** CFI function are:

Table 19: Output parameters of xd_read_doris function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range

xd_read_doris	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
doris_data	xd_doris_file	-	DORIS data	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *doris_data* structure contains pointers to memory allocated dynamically. In order to avoid memory leaks, the user will have to free that memory when the data structure is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_doris**.

7.7.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_doris** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_doris** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 20: Error messages of xd_read_doris function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Error in memory assignation	No calculation performed	XD_CFI_READ_DORIS_ERROR_IN_MEMORY_ASIG_ERR	0
ERR	Wrong input parameter value: \"time_mode\"	No calculation performed	XD_CFI_READ_DORIS_WRONG_TIME_MODE_ERR	1
ERR	Wrong time on input (start time after stop time)	No calculation performed	XD_CFI_READ_DORIS_WRONG_TIME_1_ERR	2
ERR	Wrong time on input (out of limits)	No calculation performed	XD_RCFI_EAD_DORIS_WRONG_TIME_2_ERR	3
ERR	DORIS level 0 filename not supplied	No calculation performed	XD_CFI_READ_DORIS_NO_FILENAME_ERR	4
ERR	DORIS Level 0 file cannot be open	No calculation performed	XD_CFI_READ_DORIS_CANNOT_OPEN_ERR	5
ERR	Could not find keyword: %s	No calculation performed	XD_CFI_READ_DORIS_FINDKW_ERROR_ERR	6
ERR	Error reading DORIS data for keyword: %s	No calculation performed	XD_CFI_READ_DORIS_READ_ERR	7

ERR	Error reading DORIS binary data	No calculation performed	XD_CFI_READ_DORIS_READ_BIN_ERR	8
ERR	Error changing time from ascii to processing	No calculation performed	XD_CFI_READ_DORIS_ASCII_TO_PROCESSING_ERR	9
ERR	Gap found reading DORIS level0 data	No calculation performed	XD_CFI_READ_DORIS_GAP_IN_FILE_ERR	10
ERR	DORIS file does not cover user required time in terval	No calculation performed	XD_CFI_READ_DORIS_DOES_NOT_COVER_TIME_INTERVAL_ERR	11
ERR	DORIS Packages could not be identified	No calculation performed	XD_CFI_READ_DORIS_NO_SYNC_WORD_ERR	12
WARN	No time reference specified in DORIS file. Assuming TAI	File read.	XD_CFI_READ_DORIS_DEFAULT_TIME_REF_OF_WARN	13
WARN	No Orbit Number specified in DORIS file. Assuming orbit=1 for the 1st OSV	File read.	XD_CFI_READ_DORIS_DEFAULT_ORBIT_WARN	14
WARN	Packet %ld has wrong CRC. Discarded	File read.	XD_CFI_READ_DORIS_WRONG_CRC_WARN	15
WARN	Packet %ld is invalid (bad quality). Discarded	File read.	XD_CFI_READ_DORIS_BAD_QUALITY_PACKAGE_WARN	16
WARN	Some OSVs closer than one microsecond have been discarded	File read.	XD_CFI_READ_DORIS_OSV_TOO_CLOSE_WARN	17
WARN	Gap found reading DORIS level0 data before packet %ld	File read.	XD_CFI_READ_DORIS_GAP_IN_FILE_WARN	18
ERR	Error checking if keyword exists	No calculation performed	XD_CFI_READ_DORIS_KW_EXISTS_ERR	19
ERR	Input file recognized neither as Cryosat nor Sentinel 3 DORIS	No calculation performed	XD_CFI_READ_DORIS_TYPE_NOT_RECOGNIZED_ERR	20
WARN	Maximum number of CRC warnings achieved. No more will be reported	File read.	XD_CFI_READ_DORIS_MAX_NUM_CRC_WARN	21
WARN	Maximum number of bad quality warnings reached. No more will be reported	File read.	XD_CFI_READ_DORIS_MAX_NUM_BAD_QUALITY_WARN	22
WARN	Packet %ld has a non consecutive sequence number	File read.	XD_CFI_READ_DORIS_SEQ_COUNTER_WARN	23

WARN	Packet %ld contains Orbit State vector repeated or going back in time. Discarded	File read.	XD_CFI_READ_DORIS_OSV_REPEATED_WARN	24
WARN	Packet %ld does not follow the nominal rate for DORIS OSVs	File read.	XD_CFI_READ_DORIS_NOMINAL_RATE_GAP_WARN	25
ERR	Could not guess DORIS type	No calculation performed	XD_CFI_READ_DORIS_UNKNOWN_DORIS_TYPE_ERR	26
WARN	GEO Packet %ld sequence number does not match previous EF and J2000 Packet seq number	File read.	XD_CFI_READ_DORIS_SEQ_MISMATCH_WARN	27

7.8.xd_free_doris

7.8.1. Overview

The **xd_free_doris** CFI function frees the memory allocated during the reading function **xd_read_doris**.

7.8.2. Calling interface

The calling interface of the **xd_free_doris** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_doris_file doris_data xd_free_doris (&doris_data);
}
```

7.8.3. Input parameters

The **xd_free_doris** CFI function has the following input parameters:

Table 21: Input parameters of xd_free_doris function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
doris_data	xd_doris_file	-	DORIS data structure	-	-

7.8.4. Output parameters

This function does not return any value nor parameters.

7.9.xd_read_doris_header

7.9.1. Overview

The **xd_read_doris_header** CFI function reads the Main Product Header (MPH) and the Specific Product Header (SPH) from DORIS Navigator files for Cryosat.

7.9.2. Calling interface

The calling interface of the **xd_read_doris_header** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *doris_file;
 xd_doris_mph_sph doris_hdr;
 long ierr[XD_NUM_ERR_READ_DORIS_HEADER];

 status = xd_read_doris_header(doris_file, &doris_hdr, ierr);
}
```

7.9.3. Input parameters

The **xd_read_doris_header** CFI function has the following input parameters:

Table 22: Input parameters of xd_read_doris_header function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
doris_file	char*	-	DORIS file name	-	-

It is possible to use enumeration values rather than integer values for some of the input arguments:

- Time model ID: time_mode. See [GEN_SUM].

7.9.4. Output parameters

The output parameters of the **xd_read_doris_header** CFI function are:

Table 23: Output parameters of xd_read_doris_header function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range

xd_read_doris_header	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
doris_data	xd_doris_mph_sph	-	doris header structure	-	-
ierr	long []	.	Error vector	-	-

7.9.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_doris_header** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (WARN) or an error (ERR), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_doris_header** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 24: Error messages of xd_read_doris function

Error type	Error message	Cause and impact	Error code	Error No
ERR	DORIS level 0 filename not supplied	No calculation performed	XD_CFI_READ_DORIS_HEADER_NO_FILENAME_ERR	0
ERR	DORIS Level 0 file cannot be open	No calculation performed	XD_CFI_READ_DORIS_HEADER_CANNOT_OPEN_ERR	1
ERR	Could not find keyword: %s	No calculation performed	XD_CFI_READ_DORIS_HEADER_FINDKW_ERROR_ERR	2
ERR	Error reading DORIS data for keyword: %s	No calculation performed	XD_CFI_READ_DORIS_HEADER_READ_ERR	3

7.10.xd_read_osf

7.10.1. Overview

The **xd_read_osf** CFI function reads Orbit Scenario files for Earth Observation Missions. The files have to be written in XML and consist on a list of orbital changes of the satellite along the orbit.

This function can also be used for reading the list of orbital changes within Orbit Event files.

7.10.2. Calling interface

The calling interface of the **xd_read_osf** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *file_name;
 xd_osf_file osf_data;
 long ierr[XD_NUM_ERR_READ_OSF];

 status = xd_read_osf (file_name, &osf_data, ierr);
}
```

7.10.3. Input parameters

The **xd_read_osf** CFI function has the following input parameters:

Table 25: Input parameters of xd_read_osf function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	Orbit Scenario file name	-	-

7.10.4. Output parameters

The output parameters of the **xd_read_osf** CFI function are:

Table 26: Output parameters of xd_read_osf function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range

xd_read_osf	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
osf_data	xd_osf_file	-	Structure with the OSF data	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *osf_data* structure contains pointers to memory allocated dynamically. In order to avoid memory leaks, the user will have to free that memory when the data structure is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_osf**.

7.10.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_osf** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_osf** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 27: Error messages of xd_read_osf function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Error initializing the file parser	No calculation performed	XD_CFI_READ_XML_OSF_INIT_PARSER_ERR	0
ERR	Error finding the data block keyword	No calculation performed	XD_CFI_READ_XML_OSF_XML_DATA_BLOCK_ERR	1
ERR	Error reading the data block attribute	No calculation performed	XD_CFI_READ_XML_OSF_XML_ATTRIBUTE_ERR	2
ERR	"Error reading the xml attribute"	No calculation performed	XD_CFI_READ_XML_OSF_XML_TYPE_ERR	3
ERR	Error reading XML element: %s	No calculation performed	XD_CFI_READ_XML_OSF_READ_PARAM_ERR	4
ERR	Error the size of the list (negative)	No calculation performed	XD_CFI_READ_XML_OSF_XML_DATA_BLOCK_SIZE_ERR	5

ERR	Error allocating memory	No calculation performed	XD_CFI_READ_XML_OSF_MEMORY_ERR	6
ERR	Variable header not found	No calculation performed	XD_CFI_READ_XML_OSF_VHR_NOT_FOUND_ERR	7
ERR	Incorrect value of Time_Reference. OSF time reference must be UT1	No calculation performed	XD_CFI_READ_XML_OSF_TIME_REF_OF_ERR	8
WARN	No time reference specified in orbit scenario file. Assuming UT1	Calculation performed	XD_CFI_READ_XML_OSF_TIME_REF_OF_WARN	9

7.11. xd_free_osf

7.11.1. Overview

The **xd_free_osf** CFI function frees the memory allocated during the reading function **xd_read_osf**.

7.11.2. Calling interface

The calling interface of the **xd_free_osf** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_osf_file osf_data xd_free_osf (&osf_data);
}
```

7.11.3. Input parameters

The **xd_free_osf** CFI function has the following input parameters:

Table 28: Input parameters of xd_free_osf function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
osf_data	xd_osf_file	-	DORIS data structure	-	-

7.11.4. Output parameters

This function does not return any value nor parameters.

7.12. xd_read_sdf

7.12.1. Overview

The **xd_read_sdf** CFI function reads Swath Definition files for Earth Observation Missions. For compatibility, it is possible to read files with old format.

7.12.2. Calling interface

The calling interface of the **xd_read_sdf** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status; xd_sdf_file sdf_data; char *file_name;
 long ierr[XD_NUM_ERR_READ_SDF];

 status = xd_read_sdf (file_name, &sdf_data, ierr);
}
```

7.12.3. Input parameters

The **xd_read_sdf** CFI function has the following input parameters:

Table 29: Input parameters of xd_read_sdf function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	Swath Definition file name	-	-

7.12.4. Output parameters

The output parameters of the **xd_read_sdf** CFI function are:

Table 30: Output parameters of xd_read_sdf function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xd_read_sdf	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
sdf_data	xd_sdf_file	-	Swath Definition data structure	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *sdf_data* structure contains pointers to memory allocated dinamically. In order to avoid memory leaks, the user will have to free that memory when the data structure is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_sdf**.

7.12.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_sdf** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_sdf** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 31: Error messages of xd_read_sdf function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Error opening Swath Definition file	No calculation performed	XD_CFI_READ_SDF_OPEN_FILE_ERR	0
ERR	Error allocating memory	No calculation performed	XD_CFI_READ_SDF_MEMORY_ERR	1
ERR	Error reading swath record %d	No calculation performed	XD_CFI_READ_SDF_REC_READ_ERR	2
ERR	Could not get file version	No calculation performed	XD_CFI_READ_SDF_VERSION_ERR	3

7.13.xd_free_sdf

7.13.1. Overview

The **xd_free_sdf** CFI function frees the memory allocated during the reading function **xd_read_sdf**.

7.13.2. Calling interface

The calling interface of the **xd_free_sdf** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_sdf_file sdf_data xd_free_sdf (&sdf_data);
}
```

7.13.3. Input parameters

The **xd_free_sdf** CFI function has the following input parameters:

Table 32: Input parameters of xd_free_sdf function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sdf_data	xd_sdf_file	-	SDF data structure	-	-

7.13.4. Output parameters

This function does not return any value nor parameters.

7.14. xd_read_stf

7.14.1. Overview

The **xd_read_stf** CFI function reads Swath Template Files for Earth Observation Missions. For compatibility, it is possible to read files with old format.

7.14.2. Calling interface

The calling interface of the **xd_read_stf** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *file_name;
 xd_stf_file stf_data;
 long ierr[XD_NUM_ERR_READ_STF];

 status = xd_read_stf (file_name, &stf_data, ierr);
}
```

7.14.3. Input parameters

The **xd_read_stf** CFI function has the following input parameters:

Table 33: Input parameters of xd_read_stf function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	Swath Template file name	-	-

7.14.4. Output parameters

The output parameters of the **xd_read_stf** CFI function are:

Table 34: Output parameters of xd_read_stf function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range

xd_read_stf	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
stf_data	xd_stf_file	-	Swath template file data structure	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *stf_data* structure contains pointers to memory allocated dynamically. In order to avoid memory leaks, the user will have to free that memory when the data structure is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_stf**.

7.14.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_stf** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (WARN) or an error (ERR), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_stf** function by callingthe function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 35: Error messages of xd_read_stf function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Error initializing parser to read the file	No calculation performed	XD_CFI_READ_STF_INIT_PARSER_ERR	0
ERR	Error reading the variable header	No calculation performed	XD_READ_STF_VHR_ERR	1
ERR	Error reading element: %s"	No calculation performed	XD_CFI_READ_STF_PARAM_READ_ERR	2
ERR	Could not find data block.	No calculation performed	XD_CFI_READ_STF_DATA_BLOCK_ERR	3
ERR	Could not read Data_Block attribute.	No calculation performed	XD_CFI_READ_STF_ATTRIBUTE_ERR	4
ERR	Data block is not XML type.	No calculation performed	XD_CFI_READ_STF_XML_TYPE_ERR	5
ERR	Negative number of swath coordinates	No calculation performed	XD_CFI_READ_STF_DATA_BLOCK_SIZE_ERR	6

ERR	Error allocating memory	No calculation performed	XD_CFI_READ_STF_MEMORY_ERR	7
ERR	Error reading swath record #%d	No calculation performed	XD_CFI_READ_STF_REC_READ_ERR	8
ERR	Error in STF, latitude/Dec out of range for swath record #%ld	No calculation performed	XD_CFI_READ_STF_WRONG_LAT_ERR	9
ERR	Error in STF, longitude/RA out of range for swath record #%ld	No calculation performed	XD_CFI_READ_STF_WRONG_LONG_ERR	10

7.15.xd_free_stf

7.15.1. Overview

The **xd_free_stf** CFI function frees the memory allocated during the reading function **xd_read_stf**.

7.15.2. Calling interface

The calling interface of the **xd_free_stf** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_stf_file stf_data xd_free_stf (&stf_data);
}
```

7.15.3. Input parameters

The **xd_free_stf** CFI function has the following input parameters:

Table 36: Input parameters of xd_free_stf function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
stf_data	xd_stf_file	-	STF data structure	-	-

7.15.4. Output parameters

This function does not return any value nor parameters.

7.16.xd_read_stf_vhr

7.16.1. Overview

The **xd_read_stf_vhr** CFI function reads the variable header in Swath Template File for Earth Observation Missions.

7.16.2. Calling interface

The calling interface of the **xd_read_stf_vhr** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *file_name;
 xd_stf_vhr vhr_data;
 long ierr[XD_NUM_ERR_READ_STF_VHR];

 status = xd_read_stf_vhr (file_name, &vhr_data, ierr);
}
```

7.16.3. Input parameters

The **xd_read_stf_vhr** CFI function has the following input parameters:

Table 37: Input parameters of xd_read_stf_vhr function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*^	-	Swath Template file name	-	-

7.16.4. Output parameters

The output parameters of the **xd_read_stf_vhr** CFI function are:

Table 38: Output parameters of xd_read_stf_vhr function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range

xd_read_stf_vhr	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
vhr_data	xd_stf_vhr	-	Data structure for the Swath template variable header	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *vhr_data* structure contains pointers to memory allocated dynamically. In or- der to avoid memory leaks, the user will have to free that memory when the data structure is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_stf_vhr**.

7.16.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_stf_vhr** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (WARN) or an error (ERR), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_stf_vhr** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 39: Error messages of xd_read_stf_vhr function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Error initializing parser to read the file	No calculation performed	XD_CFI_READ_STF_VHR_INIT_PARSER_ERR	0
ERR	Could not find variable header	No calculation performed	XD_CFI_READ_STF_VHR_VARIABLE_HEADER_ERR	1
ERR	Error within the reading func- tion	No calculation performed	XD_CFI_READ_STF_VHR_INTERNAL_1_ERR	2
ERR	Error reading element: %s	No calculation performed	XD_CFI_READ_STF_VHR_PARAM_READ_ERR	3
ERR	Incorrect swath type	No calculation performed	XD_CFI_READ_STF_VHR_SWATH_TYPE_ERR	4
ERR	Incorrect swath point type	No calculation performed	XD_CFI_READ_STF_VHR_SWATH_POINT_TYPE_ERR	5

ERR	Error reading "Orbit_State_Vector"	No calculation performed	XD_CFI_READ_STF_VHR_ORBIT_PARAMS_ERR	6
ERR	Error reading "Orbit_Geometry"	No calculation performed	XD_CFI_READ_STF_VHR_GEOM_PARAMS_ERR	7
ERR	Error reading altitude	No calculation performed	XD_CFI_READ_STF_VHR_ALTITUDE_READ_ERR	8
ERR	Error allocating memory	No calculation performed	XD_CFI_READ_STF_VHR_MEMORY_ERR	9

7.17. xd_free_stf_vhr

7.17.1. Overview

The **xd_free_stf_vhr** CFI function frees the memory allocated during the reading function xd_read_stf_vhr.

7.17.2. Calling interface

The calling interface of the **xd_free_stf_vhr** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_stf_vhr stf_vhr;
 xd_free_stf_vhr (&stf_vhr);
}
```

7.17.3. Input parameters

The **xd_free_stf_vhr** CFI function has the following input parameters:

Table 40: Input parameters of xd_free_stf_vhr function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
stf_vhr	xd_stf_vhr	-	STF variable header data structure	-	-

7.17.4. Output parameters

This function does not return any value nor parameters.

7.18.xd_read_att

7.18.1. Overview

The xd_read_att CFI function reads attitude generic files. This files have to be written in XML and consits on a list of attitude angles or quaternions.

7.18.2. Calling interface

The calling interface of the **xd_read_att** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status; xd_att_file att_data; char *file_name;
 long ierr[XD_NUM_ERR_READ_ATT];

 status = xd_read_att (file_name, att_data, ierr);
}
```

7.18.3. Input parameters

The **xd_read_att** CFI function has the following input parameters:

Table 41: Input parameters of xd_read_att function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	Attitude file name	-	-

7.18.4. Output parameters

The output parameters of the **xd_read_** CFI function are:

Table 42: Output parameters of xd_read_att function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowd Range

xd_read_att	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
att_data	xd_att_file	-	Attitude data structure	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *att_data* structure contains pointers to memory allocated dynamically. In order to avoid memory leaks, the user will have to free that memory when the data structure is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_att**.

7.18.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_att** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (WARN) or an error (ERR), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_att** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 43: Error messages of xd_read_att function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Error initializing parser to read the file	No calculation performed	XD_CFI_READ_ATT_INIT_PARSER_ERR	0
ERR	Error reading element: %s	No calculation performed	XD_CFI_READ_ATT_READ_PARAM_ERR	1
ERR	Wrong file type	No calculation performed	XD_CFI_READ_ATT_WRONG_FILE_TYPE_ERR	2
ERR	Error navigating through the file	No calculation performed	XD_CFI_READ_XML_ATT_NAVIGATION_ERR	3
ERR	Wrong attitude data type. Only "Quaternions" and "Attitude_Angles_Data" allowed	No calculation performed	XD_CFI_READ_ATT_WRONG_DATA_TYPE_ERR	4

ERR	Inconsistent values for <Attitude_Data_Type> and the list of attitude data	No calculation performed	XD_CFI_READ_ATT_INCONSISTENT_DATA_TYPE_ERR	5
ERR	Wrong number of records in the list	No calculation performed	XD_CFI_READ_ATT_XML_DATA_BLOCK_SIZE_ERR	6
ERR	Wrong parameter in “Reference_Frame” or in "Inertial_Ref_Frame"	No calculation performed	XD_CFI_READ_ATT_WRONG_REF_FRAME_ERR	7
ERR	Error reading attitude data list	No calculation performed	XD_CFI_READ_ATT_READ_LIST_ERR	8
ERR	Error converting ascii date to processing	No calculation performed	XD_CFI_READ_ATT_TIME_CONV_ERR	9
ERR	Error allocating memory	No calculation performed	XD_CFI_READ_ATT_MEMORY_ERR	10
ERR	Could not close the file	No calculation performed	XD_CFI_READ_ATT_CLEANUP_PARSER_ERR	11
ERR	Wrong time reference for element n. %d. All time references should be equal	No calculation performed	XD_CFI_READ_ATT_WRONG_TIME_REF_ERR	12
ERR	Quaternion modulus out of limits. Check list element n.%d	No calculation performed	XD_CFI_READ_ATT_WRONG_QUATERNION_ERR	13
ERR	Angle out of limits. Check list element n. %d	No calculation performed	XD_CFI_READ_ATT_WRONG_ANGLE_ERR	14
ERR	Maximum Gap value must be positive	No calculation performed	XD_CFI_READ_ATT_MAX_GAP_ERR	15
WARN	Obsolete tag found: %s	Calculation performed	XD_CFI_READ_ATT_OBSOLETE_TAG_WARN	16

7.19.xd_free_att

7.19.1. Overview

The **xd_free_att** CFI function frees the memory allocated during the reading function **xd_read_att**.

7.19.2. Calling interface

The calling interface of the **xd_free_att** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_att_file att_data;
 xd_free_att (&att_data);
}
```

7.19.3. Input parameters

The **xd_free_att** CFI function has the following input parameters:

Table 44: Input parameters of xd_free_att function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
att_data	xd_att_file	-	Attitude data structure	-	-

7.19.4. Output parameters

This function does not return any value nor parameters.

7.20.xd_read_star_tracker

7.20.1. Overview

The **xd_read_star_tracker** CFI function reads a list of star tracker files for Cryosat.

7.20.2. Calling interface

The calling interface of the **xd_read_star_tracker** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 long n_files, time_init_mode;
 char **file_list;
     double time0, time1;
     xd_tracker_limits str_limit;
     xd_star_tracker_file str_data;
 long ierr[XD_NUM_ERR_READ_STAR_TRACKER];

 status = xd_read_star_tracker (&n_files, file_list,
                                &time_init_mode, &time0, &time1,
                                &str_limit,
                                &str_data, ierr);
}
```

7.20.3. Input parameters

The **xd_read_star_tracker** CFI function has the following input parameters:

Table 45: Input parameters of xd_read_star_tracker function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
n_files	long	-	Number of input files	-	> 0
file_list	char **	-	List of star tracker files	-	-
time_init_mode	long	-	Flag for reading the whole file or just the requested time window		• XD_SEL_FILE or • XD_SEL_TIME
time0	double	-	Start time for the requested time window	-	days (TAI)
time1	double	-	Stop time for the requested time window	-	days (TAI)

str_limit	xd_str_limits	-	data structure containing the limits for the quaternion validation	-	-

It is possible to use enumeration values rather than integer values for some of the input arguments:

- Time range initialisation flag: time_init_mode. See current document, section 6.2

7.20.4. Output parameters

The output parameters of the **xd_read_star_tracker** CFI function are:

Table 46: Output parameters of xd_read_star_tracker function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xd_read_star_tracker	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
str_data	xd_star_tracker_file	-	Star tracker data structure	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *str_data* structure contains pointers to memory allocated dynamically. In order to avoid memory leaks, the user will have to free that memory when the data structure is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_star_tracker**.

7.20.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_star_tracker** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_star_tracker** function by calling the functionof the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 47: Error messages of xd_read_star_tracker function

Error type	Error message	Cause and impact	Error code	Error No

ERR	Could not open input file	No calculation performed	XD_CFI_READ_STR_TRACKER_OPEN_FILE_ERR	0
ERR	Could not read input file	No calculation performed	XD_CFI_READ_STR_TRACKER_READ_FILE_ERR	1
ERR	Memory allocation error	No calculation performed	XD_CFI_READ_STR_TRACKER_MEMORY_FILE_ERR	2
ERR	Gap between quaternions above maximum allowed value after time %f	No calculation performed	XD_CFI_READ_STR_TRACKER_GAP_ERR	3
ERR	No enough valid quaternions to cover the requested interval	No calculation performed	XD_CFI_READ_STR_TRACKER_NO_ENOUGH_DATA_ERR	4

7.21.xd_free_star_tracker

7.21.1. Overview

The **xd_free_star_tracker** CFI function frees the memory allocated during the reading function xd_read_star_tracker.

7.21.2. Calling interface

The calling interface of the **xd_free_star_tracker** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_star_tracker_file str_data;
 xd_free_star_tracker (&str_data);
}
```

7.21.3. Input parameters

The **xd_free_star_tracker** CFI function has the following input parameters:

Table 48: Input parameters of xd_free_star_tracker function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
str_data	xd_star_tracker_file	-	Star tracker data structure	-	-

7.21.4. Output parameters

This function does not return any value nor parameters.

7.22.xd_read_star_tracker_conf_file

7.22.1. Overview

The **xd_read_star_tracker_conf_file** CFI function reads an star tracker configuration file for Cryosat. The files have to be written in XML.

7.22.2. Calling interface

The calling interface of the **xd_read_star_tracker_conf_file** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status, star_tracker_id;
 char *file_name;
 xd_tracker_conf_file conf_data;
 long ierr[XD_NUM_ERR_READ_STAR_TRACKER_CONF_FILE];

 status = xd_read_star_tracker_conf_file (file_name,
                                          &star_tracker_id,
                                          &conf_data, ierr);
}
```

7.22.3. Input parameters

The **xd_read_star_tracker_conf_file** CFI function has the following input parameters:

Table 49: Input parameters of xd_read_star_tracker_conf_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	Star Tracker configuration file name	-	-
star_tracker_id	long	-	Star tracker number for which the configuration data is to be read	-	1, 2 or 3

7.22.4. Output parameters

The output parameters of the **xd_read_star_tracker_conf_file** CFI function are:

Table 50: Output parameters of xd_read_star_tracker_conf_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range

xd_read_star_tracker_conf_file	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
conf_data	xd_tracker_conf_file	-	Star tracker configuration data structure with	-	-
ierr	long[]	-	Error vector	-	-

7.22.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_star_tracker_conf_file** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (WARN) or an error (ERR), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_star_tracker_conf_file** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 51: Error messages of xd_read_star_tracker_conf_file function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Wrong input file	No calculation performed	XD_CFI_READ_STR_CONF_FILE_READ_FILE_ERR	0

7.23. xd_read_dem

7.23.1. Overview

The **xd_read_dem** CFI function reads a DEM file providing the table with the altitudes for each point of the grid of the DEM file.

7.23.2. Calling interface

The calling interface of the **xd_read_dem** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *dem_name;
 xd_dem_config_file dem_conf_data;
 xd_dem_file dem_data;
 long ierr[XD_NUM_ERR_READ_DEM];

 status = xd_read_dem (dem_name, &dem_conf_data,
                       &dem_data, ierr);
}
```

7.23.3. Input parameters

The **xd_read_dem** CFI function has the following input parameters:

Table 52: Input parameters of xd_read_dem function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
dem_name	char*	-	DEM file name (do not include the path)	-	-
dem_conf_data	xd_dem_config_file	-	DEM configuration data structure. This data are read from a configuration file with **xd_read_dem_config_file**	-	-

It is possible to use enumeration values rather than integer values for some of the input arguments:

- Time model ID: time_model. See [GEN_SUM].
- Time reference ID: time_ref. See [GEN_SUM].
- Time range initialisation flag: time_init_mode. See current document, section 6.2

7.23.4. Output parameters

The output parameters of the **xd_read_dem** CFI function are:

Table 53: Output parameters of xd_read_dem function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xd_read_dem	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
dem_data	xd_dem_file	-	DEM data structure	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *dem_data* structure contains pointers to memory allocated dynamically. In or- der to avoid memory leaks, the user will have to free that memory when the data structure is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_dem**.

7.23.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_dem** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_dem** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 54: Error messages of xd_read_dem function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Memory allocation error	No calculation performed	XD_CFI_READ_DEM_MEMORY_ERR	0
ERR	Incorrect input DEM configu- ration file. In case of using a Generic Raster DEM, this error message is used also to indicate problems in 'dem_raster_configuration.xml'	No calculation performed	XD_CFI_READ_DEM_NO_CONFIG_FILE_ERR	1
ERR	Wrong input file name	No calculation performed	XD_CFI_READ_DEM_WRONG_FILENAME_ERR	2
ERR	Could not open the DEM file	No calculation performed	XD_CFI_READ_DEM_OPEN_FILE_ERR	3

ERR	Could not read the DEM file	No calculation performed	XD_CFI_READ_DEM_READ_FILE_ERR	4
ERR	Unknown DEM model	No calculation performed	XD_READ_DEM_UNKNOWN_MODEL_ERR	5

7.24.xd_free_dem

7.24.1. Overview

The **xd_free_dem** CFI function frees the memory allocated in the reading function **xd_read_dem**.

7.24.2. Calling interface

The calling interface of the **xd_free_dem** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_dem_file dem_data;
 xd_free_dem (&dem_data);
}
```

7.24.3. Input parameters

The **xd_free_dem** CFI function has the following input parameters:

Table 55: Input parameters of xd_free_dem function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
dem_data	xd_dem_file	-	DEM data structure	-	-

7.24.4. Output parameters

This function does not return any value nor parameters.

7.25.xd_read_dem_config_file

7.25.1. Overview

The **xd_read_dem_config_file** CFI function reads DEM configuration parameters. Note that the DEM version (1 or 2) is automatically detected (See [MCD] for further details about the DEM models).

7.25.2. Calling interface

The calling interface of the **xd_read_dem_config_file** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *file_name;
 xd_dem_config_file dem_config_data;
 long ierr[XD_NUM_ERR_READ_DEM_CONFIG];

 status = xd_read_dem_config_file (file_name,
                                   &dem_config_data,
                                   ierr);
}
```

7.25.3. Input parameters

The **xd_read_dem_config_file** CFI function has the following input parameters:

Table 56: Input parameters of xd_read_dem_config_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	DEM configuration file name	-	-

7.25.4. Output parameters

The output parameters of the **xd_read_dem_config_file** CFI function are:

Table 57: Output parameters of xd_read_dem_config_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range

xd_read_dem_config_file	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
dem_config_data	xd_dem_config_file	-	DEM configuration data structure	-	-
ierr	long[]	-	Error vector	-	-

7.25.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_dem_config_file** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (WARN) or an error (ERR), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_dem_config_file** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 58: Error messages of xd_read_dem_config_file function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Could not open the configuration file	No calculation performed	XD_CFI_READ_DEM_CONFIG_FILE_OPEN_ERR	0
ERR	Could not read the configuration file. In case of using a Generic Raster DEM, this error message is used also to indicate problems in 'dem_raster_configuration.xml'.	No calculation performed	XD_CFI_READ_DEM_CONFIG_FILE_READ_ERR	1
ERR	Could not open the model tag	No calculation performed	XD_CFI_READ_DEM_CONFIG_FILE_READ_MODEL_ERR	2
ERR	Memory allocation error	No calculation performed	XD_CFI_READ_DEM_CONFIG_FILE_MEMORY_ERR	3
WARN	Could not open a ACE Pole file	Calculation performed. Default value is taken.	XD_CFI_READ_DEM_CONFIG_FILE_OPEN_DEM_FILE_WARN	4
ERR	Could not read a ACE file	No calculation performed	XD_CFI_READ_DEM_CONFIG_FILE_READ_DEM_FILE_ERR	5

WARN	Input DEM configuration file version is deprecated	Calculation performed	XD_CFI_READ_DEM_CONFIG_FILE_DEPRECATED_WARN	6
WARN	DEM Cache Type not supplied, assuming FIFO_CACHE with maximum size of 2 GB	Calculation performed	XD_CFI_READ_DEM_CONFIG_FILE_CACHE_TYPE_WARN	7

7.26. xd_read_zone

7.26.1. Overview

The **xd_read_zone** CFI function reads a specific zone from a zone database file for Earth Observation Missions.

7.26.2. Calling interface

The calling interface of the **xd_read_zone** CFI function is the following (input parameters are underlined)

```
#include <explorer_data_handling.h>
{
 long status; char *zone_id; char *file_name;
 xd_zone_rec zone_rec;
 long ierr[XD_NUM_ERR_READ_ZONE];

 status = xd_read_zone (file_name, &zone_id, &zone_rec, ierr);
}
```

7.26.3. Input parameters

The **xd_read_zone** CFI function has the following input parameters:

Table 59: Input parameters of xd_read_zone function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	Zone database file name	-	-
zone_id	char*	-	Zone Id to be read	-	-

7.26.4. Output parameters

The output parameters of the **xd_read_zone** CFI function are:

Table 60: Output parameters of xd_read_zone function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range

xd_read_zone	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
zone_rec	xd_zone_rec	-	Zone Data structure	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *zone_rec* structure contains pointers to memory allocated dinamically. In order to avoid memory leaks, the user will have to free that memory when the data structure is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_zone**.

7.26.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_zone** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (WARN) or an error (ERR), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_zone** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 61: Error messages of xd_read_zone function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Zone File not found	No calculation performed	XD_CFI_READ_ZONE_INIT_PARSER_ERR	0
ERR	Data Block not found	No calculation performed	XD_CFI_READ_ZONE_DATA_BLOCK_ERR	1
ERR	Data Block attribute not read	No calculation performed	XD_CFI_READ_ZONE_DATA_BLOCK_ATTRIBUTE_ERR	2
ERR	Data Block not of XML type	No calculation performed	XD_CFI_READ_ZONE_XML_TYPE_ERR	3
ERR	List_of_Zones not found.	No calculation performed	XD_CFI_READ_ZONE_LIST_ZONES_READ_ERR	4
ERR	List_of_Zones attribute not read.	No calculation performed	XD_CFI_READ_ZONE_LIST_ZONES_SIZE_ERR	5
ERR	Internal error returned	No calculation performed	XD_CFI_READ_ZONE_INTERNAL_1_ERR	6

ERR	Zone_ID cannot be read.	No calculation performed	XD_CFI_READ_ZONE_ZONE_ID_READ_ERR	7
ERR	Zone_ID not found.	No calculation performed	XD_CFI_READ_ZONE_ZONE_ID_NOT_FOUND_ERR	8
ERR	Error reading zone record	No calculation performed	XD_CFI_READ_ZONE_RECORD_READ_ERR	9

7.27.xd_free_zone

7.27.1. Overview

The **xd_free_zone** CFI function frees the memory allocated during the reading function **xd_read_zone**.

7.27.2. Calling interface

The calling interface of the **xd_free_zone** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_zone_rec zone_data;
 xd_free_zone (&zone_data);
}
```

7.27.3. Input parameters

The **xd_free_zone** CFI function has the following input parameters:

Table 62: Input parameters of xd_free_zone function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
zone_data	xd_zone_rec	-	Zone record data structure	-	-

7.27.4. Output parameters

This function does not return any value nor parameters.

7.28. xd_read_zone_file

7.28.1. Overview

The **xd_read_zone_file** CFI function reads a zone database file for Earth Observation Missions.

7.28.2. Calling interface

The calling interface of the **xd_read_zone_file** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *file_name;
 xd_zone_file zone_data;
 long ierr[XD_NUM_ERR_READ_ZONE_FILE];

 status = xd_read_zone_file (file_name, &zone_data, ierr);
}
```

7.28.3. Input parameters

The **xd_read_zone_file** CFI function has the following input parameters:

Table 63: Input parameters of xd_read_zone_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	Zone database file name	-	-

7.28.4. Output parameters

The output parameters of the **xd_read_zone_file** CFI function are:

Table 64: Output parameters of xd_read_zone_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range

xd_read_zone_file	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
xd_zone_file	zone_data	-	Structure containing the data for all the zones read from the file	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *zone_data* structure contains pointers to memory allocated dynamically. In or- der to avoid memory leaks, the user will have to free that memory when the data structure is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_zone_file**.

7.28.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_zone_file** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (WARN) or an error (ERR), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_zone_file** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 65: Error messages of xd_read_zone_file function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Zone File not found.	No calculation performed	XD_CFI_READ_ZONE_FILE_INIT_PARSER_ERR	0
ERR	Data Block not found	No calculation performed	XD_CFI_READ_ZONE_FILE_DATA_BLOCK_ERR	1
ERR	Data Block attribute not read.	No calculation performed	XD_CFI_READ_ZONE_FILE_DATA_BLOCK_ATTRIBUTE_ERR	2
ERR	Data Block not of XML type.	No calculation performed	XD_CFI_READ_ZONE_FILE_XML_TYPE_ERR	3
ERR	List_of_Zones not found.	No calculation performed	XD_CFI_READ_ZONE_FILE_LIST_ZONES_READ_ERR	4
ERR	List_of_Zones attribute not read	No calculation performed	XD_CFI_READ_ZONE_FILE_LIST_ZONES_SIZE_ERR	5

ERR	Error allocating memory	No calculation performed	XD_CFI_READ_ZONE_FILE_MEM_ERR	6
ERR	Error reading zone record number %d	No calculation performed	XD_CFI_READ_ZONE_FILE_RECORD_READ_ERR	7

7.29. xd_free_zone_file

7.29.1. Overview

The **xd_free_zone_file** CFI function frees the memory allocated during the reading function xd_read_zone_file.

7.29.2. Calling interface

The calling interface of the **xd_free_zone_file** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_zone_file zone_data;
 xd_free_zone_file (&zone_data);
}
```

7.29.3. Input parameters

The **xd_free_zone_file** CFI function has the following input parameters:

Table 66: Input parameters of xd_free_zone_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
zone_data	xd_zone_file	-	Zone file data structure	-	-

7.29.4. Output parameters

This function does not return any value nor parameters.

7.30.xd_read_zone_id

7.30.1. Overview

The **xd_read_zone_id** CFI function reads the list of zone names (Id) in a zone database file for Earth Observation Missions.

7.30.2. Calling interface

The calling interface of the **xd_read_zone_id** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status, num_zones;
 char *file_name;
 char **zone_ids
 long ierr[XD_NUM_ERR_READ_ZONE_ID];

 status = xd_read_zone_id (file_name,
                           &num_zones, &zoned_ids,
                           ierr);
}
```

7.30.3. Input parameters

The **xd_read_zone_id** CFI function has the following input parameters:

Table 67: Input parameters of xd_read_zone_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*^	-	Zone database file name	-	-

7.30.4. Output parameters

The output parameters of the **xd_read_zone_id** CFI function are:

Table 68: Output parameters of xd_read_zone_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range

xd_read_zone_id	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
num_zones	long	-	Number of zones in the input file	-	-
zone_ids	char**	-	List fo zone names in the file	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *zone_ids* is a double pointer to memory allocated dinamically. In order to avoid memory leaks, the user will have to free that memory when the data is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_zone_id**.

7.30.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_zone_id** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (WARN) or an error (ERR), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_zone_id** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 69: Error messages of xd_read_zone_id function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Zone File not found.	No calculation performed	XD_CFI_READ_ZONE_ID_INIT_PARSER_ERR	0
ERR	Data Block not found	No calculation performed	XD_CFI_READ_ZONE_ID_DATA_BLOCK_ERR	1
ERR	List_of_Zones not found.	No calculation performed	XD_CFI_READ_ZONE_ID_LIST_ZONES_READ_ERR	2
ERR	List_of_Zones attribute not read.	No calculation performed	XD_CFI_READ_ZONE_ID_LIST_ZONES_SIZE_ERR	3
ERR	Error allocating memory	No calculation performed	XD_CFI_READ_ZONEI_D_MEMORY_ERR	4
ERR	Could not find the Zone_Id tag	No calculation performed	XD_CFI_READ_ZONE_ID_READ_ZONE_ERR	5

7.31. xd_free_zone_id

7.31.1. Overview

The **xd_free_zone_id** CFI function frees the memory allocated during the reading function xd_read_zone_id.

7.31.2. Calling interface

The calling interface of the **xd_free_zone_id** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 char** zone_ids;
 xd_free_zone_id (&zone_ids);
}
```

7.31.3. Input parameters

The **xd_free_zone_id** CFI function has the following input parameters:

Table 70: Input parameters of xd_free_zone_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
zone_ids	char**	-	Zone Id. list	-	-

7.31.4. Output parameters

This function does not return any value nor parameters.

7.32.xd_read_station

7.32.1. Overview

The **xd_read_station** CFI function reads the data of a station from a station database file.

7.32.2. Calling interface

The calling interface of the **xd_read_station** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *file_name, station_id;
 xd_station_rec station_rec;
 long ierr[XD_NUM_ERR_READ_STATION];

 status = xd_read_station (file_name, station_id,
                           &station_rec, ierr);
}
```

7.32.3. Input parameters

The **xd_read_station** CFI function has the following input parameters:

Table 71: Input parameters of xd_read_station function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	Station database file name	-	-
station_id	char*	-	Station name (Id)	-	-

7.32.4. Output parameters

The output parameters of the **xd_read_station** CFI function are:

Table 72: Output parameters of xd_read_station function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range

xd_read_station	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
station_rec	xd_station_rec	-	Station record data	-	-
ierr	long[]	-	Error vector	-	-

7.32.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_station** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_station** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 73: Error messages of xd_read_station function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Ground Station DB File not found.	No calculation performed	XD_CFI_READ_STATION_INIT_PARSER_ERR	0
ERR	Data Block not found.	No calculation performed	XD_CFI_READ_STATION_DATA_BLOCK_ERR	1
ERR	Data Block attribute not read.	No calculation performed	XD_CFI_READ_STATION_DATA_BLOCK_ATTRIBUTE_ERR	2
ERR	Data Block not of XML type.	No calculation performed	XD_CFI_READ_STATION_XML_TYPE_ERR	3
ERR	List_of_Ground_Stations not found	No calculation performed	XD_CFI_READ_STATION_LIST_GS_READ_ERR	4
ERR	Number of ground stations negative.	No calculation performed	XD_CFI_READ_STATION_LIST_GS_SIZE_ERR	5
ERR	Internal error returned.	No calculation performed	XD_CFI_READ_STATION_INTERNAL_1_ERR	6
ERR	Cannot read Station_Id.	No calculation performed	XD_CFI_READ_STATION_STATION_ID_READ_ERR	7
ERR	Station id not found.	No calculation performed	XD_CFI_READ_STATION_STATION_ID_NOT_FOUND_ERR	8

ERR	Error reading station record	No calculation performed	XD_CFI_READ_STATION_REC_READ_ERR	9

7.33.xd_read_station_file

7.33.1. Overview

The **xd_read_station_file** CFI function reads a whole station file for Earth Observation Missions.

7.33.2. Calling interface

The calling interface of the **xd_read_station_file** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *file_name;
 xd_station_file station_data;
 long ierr[XD_NUM_ERR_READ_];

 status = xd_read_station_file (file_name,
                                &station_data, ierr);
}
```

7.33.3. Input parameters

The **xd_read_station_file** CFI function has the following input parameters:

Table 74: Input parameters of xd_read_station_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	Station database file name	-	-

7.33.4. Output parameters

The output parameters of the **xd_read_station_file** CFI function are:

Table 75: Output parameters of xd_read_station_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xd_read_station_file	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
station_data	xd_station_file	-	Station file data structure	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *station_data* structure contains pointers to memory allocated dinamically. In order to avoid memory leaks, the user will have to free that memory when the data structure is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_station_file**.

7.33.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_station_file** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_station_file** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 76: Error messages of xd_read_station_file function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Ground Station DB File not found.	No calculation performed	XD_CFI_READ_STATION_FILE_INIT_PARSER_ERR	0
ERR	Data Block not found.	No calculation performed	XD_CFI_READ_STATION_FILE_DATA_BLOCK_ERR	1
ERR	Data Block attribute not read.	No calculation performed	XD_CFI_READ_STATION_FILE_DATA_BLOCK_ATTRIBUTE_ERR	2
ERR	Data Block not of XML type.	No calculation performed	XD_CFI_READ_STATION_FILE_XML_TYPE_ERR	3
ERR	List_of_Ground_Stations not found.	No calculation performed	XD_CFI_READ_STATION_FILE_LIST_GS_READ_ERR	4

ERR	Number of ground stations negative.	No calculation performed	XD_CFI_READ_STATION_FILE_LIST_GS_SIZE_ERR	5
ERR	Error allocating memory	No calculation performed	XD_CFI_READ_STATION_FILE_MEM_ERR	6
ERR	Error reading station record number %d	No calculation performed	XD_CFI_READ_STATION_FILE_REC_READ_ERR	7

7.34.xd_free_station_file

7.34.1. Overview

The **xd_free_station_file** CFI function frees the memory allocated during the reading function xd_read_station_file.

7.34.2. Calling interface

The calling interface of the **xd_free_station_file** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_station_file station_data;
 xd_free_station_file (&station_data);
}
```

7.34.3. Input parameters

The **xd_free_station_file** CFI function has the following input parameters:

Table 77: Input parameters of xd_free_station_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
station_data	xd_station_file	-	Station file data structure	-	-

7.34.4. Output parameters

This function does not return any value nor parameters.

7.35. xd_read_station_id

7.35.1. Overview

The **xd_read_station_id** CFI function reads the list of station names (Id) contained in a station database file.

7.35.2. Calling interface

The calling interface of the **xd_read_station_id** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status, num_stations;
 char *file_name;
 char **station_list;
 long ierr[XD_NUM_ERR_READ_STATION_ID];

 status = xd_read_station_id (file_name, &num_stations,
                              &station_list, ierr);
}
```

7.35.3. Input parameters

The **xd_read_station_id** CFI function has the following input parameters:

Table 78: Input parameters of xd_read_station_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	Station database file name	-	-

7.35.4. Output parameters

The output parameters of the **xd_read_station_id** CFI function are:

Table 79: Output parameters of xd_read_station_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range

xd_read_station_id	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
num_stations	long	-	Number of stations	-	-
station_list	char**	.	Station list name	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *station_list* is a double pointer to memory allocated dinamically. In order to avoid memory leaks, the user will have to free that memory when the data is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_station_id**.

7.35.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_station_id** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (WARN) or an error (ERR), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_station_id** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 80: Error messages of xd_read_station_id function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Ground Station DB File not found.	No calculation performed	XD_CFI_READ_STATION_ID_INIT_PARSER_ERR	0
ERR	Data Block not found.	No calculation performed	XD_CFI_READ_STATION_ID_DATA_BLOCK_ERR	1
ERR	List_of_Ground_Stations not found.	No calculation performed	XD_CFI_READ_STATION_ID_LIST_GS_READ_ERR	2
ERR	Number of ground stations negative.	No calculation performed	XD_CFI_READ_STATION_ID_LIST_GS_SIZE_ERR	3
ERR	Error allocating memory	No calculation performed	XD_CFI_READ_STATION_ID_MEM_ERR	4
ERR	Error reading station Id.	No calculation performed	XD_CFI_READ_STATION_ID_READ_ID_ERR	5

7.36.xd_free_station_id

7.36.1. Overview

The **xd_free_station_id** CFI function frees the memory allocated during the reading function xd_read_station_id.

7.36.2. Calling interface

The calling interface of the **xd_free_station_id** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 char **station_ids;
 xd_free_station_id (&station_ids);
}
```

7.36.3. Input parameters

The **xd_free_station_id** CFI function has the following input parameters:

Table 81: Input parameters of xd_free_station_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
station_ids	char **	-	Station Id list	-	-

7.36.4. Output parameters

This function does not return any value nor parameters.

7.37. xd_read_star

7.37.1. Overview

The **xd_read_star** CFI function reads the data for a star from a star database file.

7.37.2. Calling interface

The calling interface of the **xd_read_star** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *file_name, star_id;
 xd_star_rec star_data;
 long ierr[XD_NUM_ERR_READ_STAR];

 status = xd_read_star (file_name, star_id, &star_data, ierr);
}
```

7.37.3. Input parameters

The **xd_read_star** CFI function has the following input parameters:

Table 82: Input parameters of xd_read_star function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	Star database file name	-	-
star_id	char*	-	Star name (Id) to be read	-	-

7.37.4. Output parameters

The output parameters of the **xd_read_star** CFI function are:

Table 83: Output parameters of xd_read_star function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range

xd_read_star	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
star_data	xd_star_rec	-	Star data structure	-	-
ierr	long[]	-	Error vector	-	-

7.37.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_star** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_star** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 84: Error messages of xd_read_star function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Star database file not found: %s	No calculation performed	XD_CFI_READ_STAR_FILE_NOT_FOUND_ERR	0
ERR	star id. %s not found in the star database file	No calculation performed	XD_CFI_READ_STAR_STAR_NOT_FOUND_ERR	1

7.38. xd_read_star_file

7.38.1. Overview

The **xd_read_star_file** CFI function reads a star database file for Earth Observation Missions.

7.38.2. Calling interface

The calling interface of the **xd_read_star_file** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *file_name;
 xd_star_file star_data;
 long ierr[XD_NUM_ERR_READ_STAR_FILE];

 status = xd_read_star_file (file_name, &star_data, ierr);
}
```

7.38.3. Input parameters

The **xd_read_star_file** CFI function has the following input parameters:

Table 85: Input parameters of xd_read_star_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	Star database file name (full path)	-	-

7.38.4. Output parameters

The output parameters of the **xd_read_star_file** CFI function are:

Table 86: Output parameters of xd_read_star_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range

xd_read_star_file	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
star_data	xd_star_file	-	Star file structure	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *star_data* structure contains pointers to memory allocated dinamically. In order to avoid memory leaks, the user will have to free that memory when the data structure is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_star_file**.

7.38.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_star_file** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_star_file** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 87: Error messages of xd_read_star_file function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Could not open the Star database file: %s	No calculation performed	XD_CFI_READ_STAR_FILE_FILE_NOT_FOUND_ERR	0
ERR	Error allocating memory	No calculation performed	XD_CFI_READ_STAR_FILE_MEMORY_ERR	1
ERR	No stars found in file	No calculation performed	XD_CFI_READ_STAR_FILE_NO_STARS_ERR	2

7.39. xd_free_star_file

7.39.1. Overview

The **xd_free_star_file** CFI function frees the memory allocated during the reading function **xd_read_star_file**.

7.39.2. Calling interface

The calling interface of the **xd_free_star_file** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_star_file star_data;
 xd_free_star_file (&star_data);
}
```

7.39.3. Input parameters

The **xd_free_star_file** CFI function has the following input parameters:

Table 88: Input parameters of xd_free_star_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
star_data	xd_star_file	-	Star data that has been read with xd_read_star_file	-	-

7.39.4. Output parameters

This function does not return any value nor parameters.

7.40.xd_read_star_id

7.40.1. Overview

The **xd_read_star_id** CFI function reads the list of star names from star database files.

7.40.2. Calling interface

The calling interface of the **xd_read_star_id** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *file_name;
 char **star_list;
 long num_stars;
 long ierr[XD_NUM_ERR_READ_STAR_ID];

 status = xd_read_star_id (file_name, &num_stars,
                           &star_list, ierr);
}
```

7.40.3. Input parameters

The **xd_read_star_id** CFI function has the following input parameters:

Table 89: Input parameters of xd_read_star_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	Star database file	-	-

7.40.4. Output parameters

The output parameters of the **xd_read_star_id** CFI function are:

Table 90: Output parameters of xd_read_star_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range

xd_read_star_id	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
num_stars	long	-	Number of stars in the file	-	> 0
star_list	char**	-	Array of star names	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *star_list* is a double pointer to memory allocated dynamically. In order to avoid memory leaks, the user will have to free that memory when the data is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_star_id**.

7.40.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_star_id** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_star_id** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 91: Error messages of xd_read_star_id function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Could not open the Star data-base file: %s	No calculation performed	XD_CFI_READ_STAR_ID_FILE_NOT_FOUND_ERR	0
ERR	Error allocating memory	No calculation performed	XD_CFI_READ_STAR_ID_MEMORY_ERR	1
ERR	No stars found in file	No calculation performed	XD_CFI_READ_STAR_ID_NO_STARS_ERR	2

7.41. xd_free_star_id

7.41.1. Overview

The **xd_free_star_id** CFI function frees the memory allocated during the reading function **xd_read_star_id**.

7.41.2. Calling interface

The calling interface of the **xd_free_star_id** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long num_stars;
 char** star_list;
 xd_free_star_id (&num_stars, &star_list);
}
```

7.41.3. Input parameters

The **xd_free_star_id** CFI function has the following input parameters:

Table 92: Input parameters of xd_free_star_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
num_stars	long	-	Number of stars in the list	-	-
star_list	char**	-	List of stars that has been read with xd_read_star_id	-	-

7.41.4. Output parameters

This function does not return any value nor parameters.

7.42. xd_read_tle

7.42.1. Overview

The **xd_read_tle** CFI function read a TLE file.

7.42.2. Calling interface

The calling interface of the **xd_read_tle** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *file_name, satellite;
 xd_tle_file tle_data;
 long ierr[XD_NUM_ERR_READ_TLE];

 status = xd_read_tle(file_name, satellite, &tle_data, ierr);
}
```

7.42.3. Input parameters

The **xd_read_tle** CFI function has the following input parameters:

Table 93: Input parameters of xd_read_tle function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	File name for the orbit file.	-	-
satellite	char*	-	Satellite name as it appears in line 0 for a TLE or NORAD catalogue number as it appears in the first 5 positions of line 1. If it is an empty string (“”) or NULL, all the TLE are read, other way only the TLE for this satellite are read.	-	-

7.42.4. Output parameters

The output parameters of the **xd_read_tle** CFI function are:

Table 94: Output parameters of xd_read_tle function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xd_read_tle	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
tle_data	xd_tle_file	-	Orbital state vectors data structure	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *tle_data* is a pointer to memory allocated dinamically. In order to avoid memory leaks, the user will have to free that memory when the data is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_tle**.

7.42.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_tle** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (WARN) or an error (ERR), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_tle** function by callingthe function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 95: Error messages of xd_read_tle function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Could not open the TLE file %s	File not read	XD_CFI_READ_TLE_FILE_NOT_FOUND_ERR	0
ERR	Wrong file format %s, line 0	File not read	XD_CFI_READ_TLE_WRONG_LINE0_ERR	1
ERR	Wrong file format %s, line 1	File not read	XD_CFI_READ_TLE_WRONG_LINE1_ERR	2
ERR	Wrong file format %s, line 2	File not read	XD_CFI_READ_TLE_WRONG_LINE2_ERR	3
ERR	Error allocating memory	File not read	XD_CFI_READ_TLE_MEM_ERR	4
ERR	Wrong file format %s. Satellite number in line 1 and 2 should be equal	File not read	XD_CFI_READ_TLE_WRONG_SAT_ERR	5

ERR	No TLE found in %s	No TLE read File not read	XD_CFI_READ_TLE_NO_LINES_ERR	6
WARN	Wrong file format %s, line 1. Wrong checksum value. TLE discarded	TLE skipped	XD_CFI_READ_TLE_WRONG_CHECKSUM1_WARN	7
WARN	Wrong file format %s, line 2. Wrong checksum value. TLE discarded	TLE skipped	XD_CFI_READ_TLE_WRONG_CHECKSUM2_WARN	8

7.43. xd_free_tle

7.43.1. Overview

The **xd_free_tle** CFI function frees the memory allocated during the reading function **xd_read_tle**.

7.43.2. Calling interface

The calling interface of the **xd_free_tle** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_tle_file tle_data;
 xd_free_tle (&tle_data);
}
```

7.43.3. Input parameters

The **xd_free_tle** CFI function has the following input parameters:

Table 96: Input parameters of xd_free_tle function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
tle_data	xd_tle_file	-	TLE data that has been read with xd_read_tle	-	-

7.43.4. Output parameters

This function does not return any value nor parameters.

7.44.xd_read_precise_propag_file

7.44.1. Overview

The **xd_read_precise_propag_file** CFI function read a configuration file for precise propagation.

7.44.2. Calling interface

The calling interface of the **xd_read_precise_propag_file** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *file_name;
 xd_propag_precise_config precise_conf;
 long ierr[XD_NUM_ERR_READ_PRECISE_PROPAG];

 status = xd_read_precise_propag_file(file_name,
                                      &precise_conf, ierr);
}
```

7.44.3. Input parameters

The **xd_read_precise_propag** CFI function has the following input parameters:

Table 97: Input parameters of xd_read_precise_propag function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	File name for the orbit file.	-	-

7.44.4. Output parameters

The output parameters of the **xd_read_precise_propag** CFI function are:

Table 98: Output parameters of xd_read_precise_propag function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range

xd_read_precise_propag	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
precise_conf	xd_propag_precise_config	-	Structure that will contain the precise configuration data for precise propagation.	-	-
ierr	long[]	-	Error vector	-	-

7.44.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_precise_propag** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (WARN) or an error (ERR), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_precise_propag** function bycalling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 99: Error messages of xd_read_precise_propag function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Could not open file	File not read	XD_CFI_READ_PRECISE_PROPAG_INIT_PARSER_ERR	0
ERR	Could not read parameter %s	File not read	XD_CFI_READ_PRECISE_PROPAG_READ_PARAM_ERR	1
ERR	Flag nor correct. Its value must be 0 or 1	File not read	XD_CFI_READ_PRECISE_PROPAG_WRONG_FLAG_ERR	2
ERR	Could not close the file	File not read	XD_CFI_READ_PRECISE_PROPAG_CLEANUP_PARSER_ERR	3
ERR	Could not write the fixed header	File not read	XD_CFI_WRITE_PRECISE_PROPAG_WRITE_FHR_ERR	4
WARN	Cannot write schema in the file		XD_CFI_WRITE_PRECISE_PROPAG_SET_SCHEMA_WARN	5

7.45.xd_read_att_def

7.45.1. Overview

The **xd_read_att_def** CFI function reads a whole attitude definition file.

The description of the output struct can be found in table 3.

The detailed description of the Attitude Definition File can be found in [FFS3].

7.45.2. Calling interface

The calling interface of the **xd_read_att_def** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *file_name;
 xd_attitude_definition_data att_data;
 long ierr[XD_NUM_ERR_READ_ATT_DEF];

 status = xd_read_att_def (file_name,
                           &att_data, ierr);
}
```

7.45.3. Input parameters

The **xd_read_att_def** CFI function has the following input parameters:

Table 100: Input parameters of xd_read_att_def function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	Attitude definition file name	-	-

7.45.4. Output parameters

The output parameters of the **xd_read_att_def** CFI function are:

Table 101: Output parameters of xd_read_att_def function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xd_read_att_def	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
att_data	xd_attitude_definition_data	-	Attitude definition data structure	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *att_data* structure contains pointers to memory allocated dinamically. In order to avoid memory leaks, the user will have to free that memory when the data structure is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_att_def**.

7.45.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_att_def** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_att_def** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 102: Error messages of xd_read_att_def function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Error opening file	No calculation performed	XD_CFI_READ_ATT_DEF_OPEN_FILE_ERR	0
ERR	Error allocating memory	No calculation performed	XD_CFI_READ_ATT_DEF_MEMORY_ERR	1
ERR	Error reading record	No calculation performed	XD_CFI_READ_ATT_DEF_REC_READ_ERR	2
WARN	Obsolete tag found: "Inertial_Ref_Frame"	Calculation performed	XD_CFI_READ_ATT_DEF_OBSOLETE_TAG_WARN	3

7.46.xd_free_att_def

7.46.1. Overview

The **xd_free_att_def** CFI function frees the memory allocated during the reading function xd_read_att_def.

7.46.2. Calling interface

The calling interface of the **xd_free_att_def** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_attitude_definition_data att_data;
 xd_free_att_def (&att_data);
}
```

7.46.3. Input parameters

The **xd_free_att_def** CFI function has the following input parameters:

Table 103: Input parameters of xd_free_att_def function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
att_data	xd_attitude_definition_data	-	Attitude definition file data structure	-	-

7.46.4. Output parameters

This function does not return any value nor parameters.

7.47.xd_read_sp3

7.47.1. Overview

The **xd_read_sp3** CFI function reads a Standard Product 3 C (SP3-C) File.

The description of the output struct (xd_sp3_file) can be found in table 3.

The detailed description of the SP3 file can be found in [SP3].

The following items must be considered when reading a SP3 file:

1) SP3 file does not provide information about the orbit number.
2) The xd_read_sp3 function extracts file common information and only Orbit State Vectors for satellites (see output struct xd_sp3_file).
3) The following time conversions are performed, depending on the SP3 file time system:

   - If time system is GPS (identifier GPS), GALILEO (identifier GAL) or QZSS (identifier GZS), the times are converted to TAI, taking into account that TAI time is equal to GPS/GALILEO/QZSS time plus 19 seconds. Since no time correlation is provided, TAI-UTC and UT1-UTC differences are set to zero.

   - If time system is GLONASS (identifier GLO), the times are converted to UTC, taking into account that UTC time is equal to GLONASS time minus 3 hours. Since no time correlation is provided, TAI-UTC and UT1-UTC differences are set to zero.

   - If time system is TAI (identifier TAI) or UTC (identifier UTC), the times are taken as they are in the corresponding time reference system. Since no time correlation is provided, TAI-UTC and UT1-UTC differences are set to zero.
4) The Orbit State Vectors are recorded in output struct following the satellite order found in SP3 file. For example, if the identifiers of the satellites are G01G02G04, the corresponding OSVs information are (taking into account that this information is stored in the field osv_rec_sp3 of xd_sp3_file):

   - For G01: osv_rec_sp3[0]

   - For G02: osv_rec_sp3[1]

   - For G04: osv_rec_sp3[2]

   Note that the position in array corresponds to position in satellite list, not in the satellite identifier number.
5) A warning is raised if at least one of the following conditions is detected:
   - OSV with time going back
   - OSV with repeated time

7.47.2. Calling interface

The calling interface of the **xd_read_sp3** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
```

```
{
 long status;
 char *file_name;
 xd_sp3_file sp3_data;
 xd_osv_list_read_configuration read_config;

 long ierr[XD_NUM_ERR_READ_SP3];

 status = xd_read_sp3 (file_name,
                       &read_config,
                       &sp3_data, ierr);
}
```

7.47.3. Input parameters

The **xd_read_sp3** CFI function has the following input parameters:

Table 104: Input parameters of xd_read_sp3 function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	SP3 file name	-	-
read_config	xd_osv_list_read_configuration*	-	Configuration for reading OSV state vectors	-	-

7.47.4. Output parameters

The output parameters of the **xd_read_sp3** CFI function are:

Table 105: Output parameters of xd_read_sp3 function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range

xd_read_sp3	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
Sp3_data	xd_sp3_file	-	SP3 file structure	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *sp3_data* structure contains pointers to memory allocated dinamically. In order to avoid memory leaks, the user will have to free that memory when the data structure is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_sp3**.

7.47.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_sp3** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_sp3** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 106: Error messages of xd_read_sp3 function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Error opening file %s	No calculation performed	XD_CFI_READ_SP3_OPEN_FILE_ERR	0
ERR	Error reading line number %ld	No calculation performed	XD_CFI_READ_SP3_READ_LINE_ERR	1
ERR	Wrong file version number: %s	No calculation performed	XD_CFI_READ_SP3_WRONG_FILE_VERSION_ERR	2
ERR	Wrong file type found: %s	No calculation performed	XD_CFI_READ_SP3_WRONG_FILE_TYPE_ERR	3
ERR	Error getting processing time	No calculation performed	XD_CFI_READ_SP3_GET_PROC_TIME_ERR	4
ERR	Wrong sat identifier in string: %s	No calculation performed	XD_CFI_READ_SP3_SAT_ID_ERR	5
ERR	Error allocating memory	No calculation performed	XD_CFI_READ_SP3_MEMORY_ERR	6

ERR	Wrong number of satellite identifiers found	No calculation performed	XD_CFI_READ_SP3_NUM_SAT_ID_ERR	7
ERR	Wrong accuracy in line: %ld	No calculation performed	XD_CFI_READ_SP3_SAT_ACCURACY_ERR	8
ERR	Wrong time system: %s	No calculation performed	XD_CFI_READ_SP3_TIME_SYSTEM_ERR	9
ERR	Wrong file descriptor: %s	No calculation performed	XD_CFI_READ_SP3_TYPE_DESCRIPTOR_ERR	10
ERR	Wrong reading configuration	No calculation performed	XD_READ_SP3_READ_CONFIG_ERR	11
WARN	Time going back for epoch no. %ld	File read	XD_CFI_READ_SP3_TIME_GOING_BACK_WARN	12
WARN	Repeated OSV found for epoch no. %ld	File read	XD_CFI_READ_SP3_REPEATED_OSV_WARN	13
ERR	Error fitting the OSV array to the requested time interval	No calculation performed	XD_READ_SP3_FITING_OSV_ARRAY_TO_REQUESTED_TIME_ERR	14
WARN	Configuration time reference is different from file time system	File read	XD_READ_SP3_CONFIG_TIME_REF_WARN	15

7.48.xd_free_sp3

7.48.1. Overview

The **xd_free_sp3** CFI function frees the memory allocated during the reading function **xd_read_sp3**.

7.48.2. Calling interface

The calling interface of the **xd_free_sp3** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_sp3_file sp3_data;
 xd_free_sp3 (&sp3_data);
}
```

7.48.3. Input parameters

The **xd_free_sp3** CFI function has the following input parameters:

Table 107: Input parameters of xd_free_sp3 function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sp3_data	xd_sp3_file	-	SP3 file structure	-	-

7.48.4. Output parameters

This function does not return any value nor parameters.

7.49.xd_read_fov_constraints_file

7.49.1. Overview

The **xd_read_fov_constraints_file** CFI function reads a Field Of View configuration file

The detailed description of the FOV Configuration file can be found in section [FFS3].

7.49.2. Calling interface

The calling interface of the **xd_read_fov_constraints_file** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *file_name;
 xd_fov_constraints_file fov_data;

 long ierr[XD_NUM_ERR_READ_FOV];

 status = xd_read_fov_constraints_file (file_name,
                        &fov_data, ierr);
}
```

7.49.3. Input parameters

The **xd_read_fov_constraints_file** CFI function has the following input parameters:

Table 108: Input parameters of xd_read_fov_constraints_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	FOV constraints file name	-	-

7.49.4. Output parameters

The output parameters of the **xd_read_fov_constraints_file** CFI function are:

Table 109: Output parameters of xd_read_fov_constraints_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xd_read_fov_constraints_file	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
fov_data	xd_fov_constraints_file	-	FOV Constraints file structure	-	-
ierr	long[]	-	Error vector	-	-

7.49.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_fov_constraints_file** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (WARN) or an error (ERR), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_fov_constraints_file** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 110: Error messages of xd_read_sp3 function

Error type	*Error message*	*Cause and impact*	*Error code*	*Error*
ERR	*Error in reading*	*No calculation*	*XD_CFI_READ_FOV_REA*	*0*
ERR	*Error opening file: %s*	*No calculation*	*XD_CFI_READ_FOV_INIT*	*1*
ERR	*Error parameter*	*No calculation*	*XD_CFI_READ_FOV_PAR*	*2*
ERR	*Wrong attribute*	*No calculation*	*XD_CFI_READ_FOV_ATT*	*3*
ERR	*Error wrong value: %s*	*No calculation*	*XD_CFI_READ_FOV_WR*	*4*

7.50. xd_write_orbit_file

7.50.1. Overview

The **xd_write_orbit_file** CFI function writes an orbit file in XML format using the data structure provided by the user. The orbit file can be either:

- A Predicted orbit file
- A Restituted orbit file
- A DORIS Predicted file
- The Time_Reference and Ref_Frame fields in the variable header of the orbit file are filled according to the parameters time_ref_of and ref_frame in the OSV records. Therefore it is required that all OSVs contained in xd_orbit_file have the same time reference and reference frame.

7.50.2. Calling interface

The calling interface of the **xd_write_orbit_file** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *file_name;
 xd_fhr fhr;
 xd_orbit_file *osv_data;
 long ierr[XD_NUM_ERR_WRITE_ORBIT_FILE];

 status = xd_write_orbit_file(file_name, &fhr, &osv_data, ierr);
}
```

7.50.3. Input parameters

The **xd_write_orbit_file** CFI function has the following input parameters:

Table 111: Input parameters of xd_write_orbit_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	File name for the orbit file. If empty string (i.e, ""), then the file is written with the name in the fixed_header structure (fhr)	-	-
fhr	xd_fhr	-	Fixed header structure	-	-
xd_orbit_file	osv_data	-	Orbital state vectors data structure	-	-

7.50.4. Output parameters

The output parameters of the **xd_write_orbit_file** CFI function are:

Table 112: Output parameters of xd_write_orbit_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xd_write_orbit_file	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
ierr	long[]	-	Error vector	-	-

7.50.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_write_orbit_file** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_write_orbit_file** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 113: Error messages of xd_write_orbit_file function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Cannot create root element	No calculation performed	XD_CFI_WRITE_ORBIT_FILE_CREATE_TREE_ERR	0
ERR	Cannot create in-memory XML tree	No calculation performed	XD_CFI_WRITE_ORBIT_FILE_CREATE_ROOT_ERR	1
ERR	Cannot write the fixed header	No calculation performed	XD_CFI_WRITE_ORBIT_FILE_WRITE_FHR_ERR	2
ERR	Cannot add XML node to tree: %s	No calculation performed	XD_CFI_WRITE_ORBIT_FILE_CREATE_NODE_ERR	3
ERR	Cannot convert time from processing to external	No calculation performed	XD_CFI_WRITE_ORBIT_FILE_GET_ASCII_TIME_ERR	4
ERR	Cannot write XML file	No calculation performed	XD_CFI_WRITE_ORBIT_FILE_WRITE_ERR	5
ERR	Cannot go to the desired node	No calculation performed	XD_CFI_WRITE_ORBIT_FILE_GOTO_NODE_ERR	6

WARN	Cannot write schema in the file	File written to disk but without schema	XD_CFI_WRITE_ORBIT_FILE_SET_SCHEMA_WARN	7
ERR	All the orbit records must have the same reference frame	No calculation performed	XD_CFI_WRITE_ORBIT_FILE_REF_FRAME_ERR	8
ERR	All the orbit records must have the same time reference	No calculation performed	XD_CFI_WRITE_ORBIT_FILE_TIME_REF_OF_ERR	9

7.51. xd_write_osf

7.51.1. Overview

The **xd_write_osf** CFI function writes an Orbit Scenario file in XML format using the data provided by the user.

7.51.2. Calling interface

The calling interface of the **xd_write_osf** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *file_name;
 xd_fhr fhr;
 xd_osf_file osf_data;
 long ierr[XD_NUM_ERR_WRITE_OSF];

 status = xd_write_osf (file_name, &fhr, &osf_data, ierr);
}
```

7.51.3. Input parameters

The **xd_write_osf** CFI function has the following input parameters:

Table 114: Input parameters of xd_write_osf function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	File name for the orbit scenario file. If empty string (i.e, ""), then the file is written with the name in the fixed_header structure (fhr)	-	-
fhr	xd_fhr	-	Fixed header structure	-	-
xd_osf_file	osf_data	-	Orbital changes data structure	-	-

7.51.4. Output parameters

The output parameters of the **xd_write_osf** CFI function are:

Table 115: Output parameters of xd_write_osf function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xd_write_osf	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
ierr	long[]	-	Error vector	-	-

7.51.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_write_osf** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (WARN) or an error (ERR), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_write_osf** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 116: Error messages of xd_write_osf function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Cannot create in-memory XML tree	No calculation performed	XD_CFI_WRITE_OSF_CREATE_TREE_ERR	0
ERR	Cannot write the fixed header	No calculation performed	XD_CFI_WRITE_OSF_WRITE_FHR_ERR	1
ERR	Cannot create root element	No calculation performed	XD_CFI_WRITE_OSF_CREATE_ROOT_ERR	2
ERR	Cannot add XML node to tree	No calculation performed	XD_CFI_WRITE_OSF_CREATE_NODE_ERR	3
ERR	Cannot set XML node value	No calculation performed	XD_CFI_WRITE_OSF_SET_NODE_VALUE_ERR	4
ERR	Cannot convert time from processing to external	No calculation performed	XD_CFI_WRITE_OSF_TIME_TO_EXTERNAL_ERR	5
ERR	Cannot write XML file	No calculation performed	XD_CFI_WRITE_OSF_WRITE_ERR	6
WARN	Cannot write schema in the file	File written to disk but without schema	XD_CFI_WRITE_OSF_SET_SCHEMA_WARN	7
ERR	Time reference of orbital changes must be UT1	No calculation performed	XD_CFI_WRITE_OSF_TIME_REF_OF_ERR	8

7.52.xd_write_doris

7.52.1. Overview

The **xd_write_doris** CFI function writes a DORIS NAVIGATOR Product file for CRYOSAT, using the data provided by the user.

7.52.2. Calling interface

The calling interface of the **xd_write_doris** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *file_name;
 xd_doris_mph_sph fhr;
 xd_doris_file doris_data;
 long ierr[XD_NUM_ERR_WRITE_DORIS];

 status = xd_write_doris (file_name, &fhr, &doris_data, ierr);
}
```

7.52.3. Input parameters

The **xd_write_doris** CFI function has the following input parameters:

Table 117: Input parameters of xd_write_doris function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	DORIS file name	-	-
fhr	xd_doris_mph_sph	-	Main and Specific product headers	-	-
doris_data	xd_doris_file	-	DORIS data structure	-	-

7.52.4. Output parameters

The output parameters of the **xd_write_doris** CFI function are:

Table 118: Output parameters of xd_write_doris function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range

xd_write_doris	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
ierr	long[]	-	Error vector	-	-

7.52.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_write_doris** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_write_doris** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 119: Error messages of xd_write_doris function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Could not open the file %s for writing	No calculation performed	XD_CFI_WRITE_DORIS_OPEN_ERR	0
ERR	Error writing the fixed header	No calculation performed	XD_CFI_WRITE_DORIS_WRITE_FHR_ERR	1
ERR	Error writing the binary data	No calculation performed	XD_CFI_WRITE_DORIS_WRITE_BINARY_ERR	2

7.53.xd_write_stf

7.53.1. Overview

The **xd_write_stf** CFI function writes a swath template file XML format using the data provided by the user.

7.53.2. Calling interface

The calling interface of the **xd_write_stf** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *file_name;
 xd_fhr fhr;
 xd_stf_file stf_data;
 long ierr[XD_NUM_ERR_WRITE_STF];

 status = xd_write_stf (file_name, &fhr, &stf_data, ierr);
}
```

7.53.3. Input parameters

The **xd_write_stf** CFI function has the following input parameters:

Table 120: Input parameters of xd_write_stf function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	File name for the swath template file. If empty string (i.e, “”), then the file is written with the name in the fixed_header structure (fhr)	-	-
fhr	xd_fhr	-	Fixed header structure	-	-
xd_stf_file	stf_data	-	STF data structure	-	-

7.53.4. Output parameters

The output parameters of the **xd_write_stf** CFI function are:

Table 121: Output parameters of xd_write_stf function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xd_write_stf	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
ierr	long[]	-	Error vector	-	-

7.53.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_write_stf** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_write_stf** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 122: Error messages of xd_write_stf function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Cannot create XML tree.	No calculation performed	XD_CFI_WRITE_STF_CREATE_XML_ERR	0
ERR	Cannot create root node in the XML tree.	No calculation performed	XD_CFI_WRITE_STF_CREATE_ROOT_XML_ERR	1
ERR	Error writing fixed header.	No calculation performed	XD_CFI_WRITE_STF_XD_FHR_WRITE_ERR	2
ERR	Error while writing Swath Template File variable header.	No calculation performed	XD_CFI_WRITE_STF_XD_STF_VHR_WRITE_ERR	3
ERR	Cannot create the node %s	No calculation performed	XD_CFI_WRITE_STF_CREATE_NODE_ERR	4
ERR	Wrong swath_type	No calculation performed	XD_CFI_WRITE_STF_WRONG_SWATH_TYPE_ERR	5
ERR	Error while writing the swath record n.%d	No calculation performed	XD_CFI_WRITE_STF_WRITE_REC_ERR	6
ERR	Cannot write to disk the XML tree	No calculation performed	XD_CFI_WRITE_STF_WRITE_ERR	7
WARN	Cannot write schema in the file	File written to disk but without schema	XD_CFI_WRITE_STF_SET_SCHEMA_WARN	8

7.54.xd_write_att

7.54.1. Overview

The **xd_write_att** CFI function writes an attitude generic file in XML format using the data provided by the user.

Note about output format: the number of decimal digits written to file depends on the type of data:

•If angles are used, 6 decimal digits are written.

•If quaternions are used, 9 decimal digits are written.

It is done this way because having 9 decimal digits in quaternions reduces pointing error significantly .

7.54.2. Calling interface

The calling interface of the **xd_write_att** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *file_name;
 xd_fhr fhr;
 xd_att_file att_data;
 long ierr[XD_NUM_ERR_WRITE_ATT];

 status = xd_write_att (file_name, &fhr, &att_data, ierr);
}
```

7.54.3. Input parameters

The **xd_write_att** CFI function has the following input parameters:

Table 123: Input parameters of xd_write_att function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	File name for the attitude file. If empty string (i.e, ""), then the file is written with the name in the fixed_header structure (fhr)	-	-
fhr	xd_fhr	-	Fixed header structure	-	-
xd_att_file	att_data	-	Attitude data structure	-	-

7.54.4. Output parameters

The output parameters of the **xd_write_att** CFI function are:

Table 124: Output parameters of xd_write_att function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xd_write_att	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
ierr	long[]	-	Error vector	-	-

7.54.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_write_att** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_write_att** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 125: Error messages of xd_write_att function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Cannot create in-memory XML tree	No calculation performed	XD_CFI_WRITE_ATT_CREATE_TREE_ERR	0
ERR	Cannot create root element	No calculation performed	XD_CFI_WRITE_ATT_CREATE_ROOT_ERR	1
ERR	Cannot write the fixed header	No calculation performed	XD_CFI_WRITE_ATT_WRITE_FHR_ERR	2
ERR	Cannot add XML node to tree: %s	No calculation performed	XD_CFI_WRITE_ATT_CREATE_NODE_ERR	3
ERR	Cannot convert time from processing to external	No calculation performed	XD_CFI_WRITE_ATT_GET_ASCII_TIME_ERR	4
ERR	Cannot go to the desired node	No calculation performed	XD_CFI_WRITE_ATT_GOTO_NODE_ERR	5
ERR	Cannot write XML file	No calculation performed	XD_CFI_WRITE_ATT_WRITE_ERR	6

WARN	Cannot write schema in the file	File written to disk but without schema	XD_CFI_WRITE_ATT_SET_SCHEMA_WARN	7

7.55.xd_write_tle

7.55.1. Overview

The **xd_write_tle** CFI function writes a TLE file. The data to be written are in the input structure except for the checksum, that it is computed for every line.

7.55.2. Calling interface

The calling interface of the **xd_write_tle** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *file_name;
 xd_tle_file tle_data;
 long ierr[XD_NUM_ERR_WRITE_TLE]

 status = xd_write_tle (file_name, &tle_data, ierr);
}
```

7.55.3. Input parameters

The **xd_write_tle** CFI function has the following input parameters:

Table 126: Input parameters of xd_write_tle function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	File name for the TLE file.	-	-
xd_tle_file	tle_data	-	TLE data structure	-	-

7.55.4. Output parameters

The output parameters of the **xd_write_tle** CFI function are:

Table 127: Output parameters of xd_write_tle function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range

xd_write_tle	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
ierr	long[]	-	Error vector	-	-

7.55.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_write_tle** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_write_tle** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 128: Error messages of xd_write_tle function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Could not open the TLE file for writing: %s	No calculation performed	XD_WRITE_TLE_FILE_OPEN_ERR	0
ERR	Could not write the TLE file: %s	No calculation performed	XD_WRITE_TLE_WRITE_ERR	1

7.56.xd_write_att_def

7.56.1. Overview

The **xd_write_att_def** CFI function writes a Attitude Definition File.

The description of the input struct can be found in table 3.

The detailed description of the Attitude Definition File can be found in section [FFS3].

7.56.2. Calling interface

The calling interface of the **xd_write_att_def** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *file_name;
 xd_attitude_definition_data att_data;
 long ierr[XD_NUM_ERR_WRITE_ATT_DEF];

 status = xd_write_att_def (file_name, &fhr, &att_data, ierr);
}
```

7.56.3. Input parameters

The **xd_write_att_def** CFI function has the following input parameters:

Table 129: Input parameters of xd_write_att_def function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	File name.	-	-
fhr	xd_fhr	-	Fixed header		
att_data	xd_attitude_definition_data	-	Attitude definition data structure	-	-

7.56.4. Output parameters

The output parameters of the **xd_write_att_def** CFI function are:

Table 130: Output parameters of xd_write_att_def function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xd_write_att_def	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
ierr	long[]	-	Error vector	-	-

7.56.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_write_att_def** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_write_att_def** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 131: Error messages of xd_write_att_def function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Cannot create in-memory XML tree	No calculation performed	XD_CFI_WRITE_ATT_DEF_CREATE_TREE_ERR	0
ERR	Cannot create root element	No calculation performed	XD_CFI_WRITE_ATT_DEF_CREATE_ROOT_ERR	1
ERR	Cannot write the fixed header	No calculation performed	XD_CFI_WRITE_ATT_DEF_WRITE_FHR_ERR	2
ERR	Error writing in the file	No calculation performed	XD_CFI_WRITE_ATT_DEF_WRITE_ERR	3
WARN	Cannot write schema in the file	No calculation performed	XD_CFI_WRITE_ATT_DEF_SET_SCHEMA_WARN	4
ERR	Cannot add a child node	No calculation performed	XD_CFI_WRITE_ATT_DEF_ADD_CHILD_ERR	5
ERR	Cannot add an atribute	No calculation performed	XD_CFI_WRITE_ATT_DEF_ADD_ATTRIBUTE_ERR	6
ERR	Cannot set XML node	No calculation performed	XD_CFI_WRITE_ATT_DEF_SET_NODE_VALUE_ERR	7
ERR	Cannot get XML node value	No calculation performed	XD_CFI_WRITE_ATT_DEF_GET_NODE_VALUE_ERR	8

7.57.xd_xml_validate

7.57.1. Overview

The **xd_xml_validate** CFI function validates an XML file using its XML schema and checks the XML schema versioning.

The Earth Explorer XSD schemas are part of each EOCFI package but can be individually downloaded from here (latest and older versions). Note: due to the limitation in the third-party library libxml2, the remote schemas that present in their path 'https' instead of 'http' are not processed correctly. You need a local copy of the schemas to use this function.

7.57.2. Calling interface

The calling interface of the CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status, valid_status;
 char *filename, *schema, *logfile;
 long mode;
 long ierr[XD_NUM_ERR_XML_VALIDATE];

 status = xd_xml_validate (filename, &mode, schema, logfile,
                           &valid_status, ierr);
}
```

7.57.3. Input parameters

The **xd_xml_validate** CFI function has the following input parameters:

Table 132: Input parameters of xd_xml_validate function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
filename	char*	-	File name to validate	-	-
mode	long	-	Flag to select the schema to be used to validate the file. It can be either: • XD_DEFAULT_SCHEMA: use the schema that is in the root element of the XML file. or • XD_USER_SCHEMA: use the schema given in the *schema* parameter in the interface.	-	-

schema	char*	-	Schema file. The schema can be given as an absolute path or as a relative path from the file's directory (No the current directory)	-	-
logfile	char*	-	Log file (file path). It is used to store the messages returned by the validation process. The result of the validation can be seen at the end of the log in the following message: Validation result for "filename": [VALID]/[INVALID]	-	-

7.57.4. Output parameters

The output parameters of the **xd_xml_validate** CFI function are:

Table 133: Output parameters of xd_xml_validate function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xd_xml_validate	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
valid_status	long	-	The result of the validation: • XD_XML_INVALID (= -1) • XD_XML_VALID (= 0)	-	-
ierr	long[]	-	Error vector	-	-

7.57.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_xml_validate** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_xml_validate** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 134: Error messages of xd_xml_validate function

Error type	Error message	Cause and impact	Error code	Error No

ERR	Could not set schema within the XML file Severe errors in the file format.	The file is not well formed and cannot be opened because of severe errors. No calculation performed	XD_CFI_XML_VALIDATE_SET_SCHEMA_ERR	0
ERR	Could not open file: %s. Severe errors in the file format	The file is not well formed and cannot be opened because of severe errors. No calculation performed	XD_CFI_XML_VALIDATE_INIT_PARSER_ERR	1
ERR	Memory allocation error	No calculation performed	XD_CFI_XML_VALIDATE_MEMORY_ERR	2
ERR	No schema provided	No calculation performed	XD_CFI_XML_VALIDATE_NO_SCHEMA_ERR	3
ERR	Wrong input mode	No calculation performed	XD_CFI_XML_VALIDATE_WRONG_MODE_ERR	4
ERR	Could not open file: %s	No calculation performed	XD_CFI_XML_VALIDATE_OPEN_FILE_ERR	5
ERR	Could not lock other execution threads	No calculation performed	XD_CFI_XML_VALIDATE_LOCK_ERR	6
ERR	Could not copy input file to the current directory	No calculation performed	XD_CFI_XML_VALIDATE_COPY_FILE_ERR	7
ERR	Schema not found in root element	No calculation performed	XD_CFI_XML_VALIDATE_NO_SCHEMA_IN_FILE_ERR	8
ERR	Schema version differs from the version in the schema filename	No calculation performed	XD_CFI_XML_VALIDATE_INCONSITENT_SCHEMA_VERS_ERR	9
WARN	The XML file does not contain the schema version	Calculation performed	XD_CFI_XML_VALIDATE_NO_SCH_VERS_IN_FILE_WARN	10
WARN	Schema version not found	Calculation performed	XD_CFI_XML_VALIDATE_NO_VERS_IN_SCHEMA_WARN	11
WARN	Schema version in XML file is older than the schema version	Calculation performed	XD_CFI_XML_VALIDATE_LESS_SCHEMA_VERS_WARN	12
WARN	Schema version in XML file is newer than the schema version	Calculation performed	XD_CFI_XML_VALIDATE_GREATER_SCHEMA_VERS_WARN	13

7.57.6. Executable program

An XML file can also be validated using the executable program **xml_validate.** It can be called from a Unix shell as:

xml_validate -file filename

[**-sch** schema_filename] [**-log** log_filename]

[-help] [-v]

[-show]

Note that:

- Order of parameters does not matter.
- Bracketed parameters are not mandatory.
- [-v] option for Verbose mode (default is Silent).
- [-show] displays the inputs of the function and the results.
- The filename is validated using the schema_filename if it is provided. If not, the default schema is used (the one in the root element of the file).
- The validation log is stored in the log_filename. By default the standard output is used.

Example:

xml_validate **-file** ../../data/CRYOSAT_XML_OSF

-sch ../../../../schemas/public/CS_OPER_MPL_ORBSCT_01.00.XSD

-log log_file_exe **-show**

7.58.xd_select_schema

7.58.1. Overview

The **xd_select_schema** returns the most recent schema file name applicable for a given file type and mission.

7.58.2. Calling interface

The calling interface of the CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_fileinfo info;
     char        schema[XD_MAX_PATH];
     long        ierr[XD_NUM_ERR_SELECT_SCHEMA];

     status = xd_select_schema(&info, schema,
                               ierr);
}
```

7.58.3. Input parameters

The **xd_select_schema** CFI function has the following input parameters:

Table 135: Input parameters of xd_select_schema function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
info	xd_fileinfo	-	File info containing the mission and the file type (see Table 3)	-	-

7.58.4. Output parameters

The output parameters of the **xd_select_schema** CFI function are:

Table 136: Output parameters of xd_select_schema function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xd_select_schema	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
schema	char*	-	Schema name	-	-
ierr	long[]	-	Error vector	-	-

7.58.5. Warnings and errors

The current version of the **xd_select_schema** does not return any errors nor warnings.

7.59.xd_orbit_file_decimate

7.59.1. Overview

The **xd_orbit_file _decimate** adds capability to configure position interpolator according to user need (decimation).

The decimation is performed in the orbit file structure. This way user has two options using the output of this function:

1.to write a new orbit file and use this file to initialize the orbit id.

2.To initialize directly the orbit id with the new structure.

The fucntion works as follows:

•First and last state vectors in input list are copied to output list.

•Using the input decimation delta (D), and being t0 the time of the first state vector of the input list, the state vectors whose time is closer to time t=t0+k*D (k = 1, 2...n, t0<t<tn) are copied to output list.

7.59.2. Calling interface

The calling interface of the **xd_orbit_file_decimate** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_fhr fhr_in, fhr_out;
 xd_orbit_file osv_in, osv_out;
 double decimation_delta_time;
 long ierr[XD_NUM_ERR_ORBIT_FILE_DECIMATE];
 status = xd_orbit_file_decimate(&fhr_in, &osv_in,
                                 decimation_delta_time,
                                 &fhr_out, &osv_out,
                                 ierr);
}
```

7.59.3. Input parameters

The **xd_orbit_file_decimate** CFI function has the following input parameters:

Table 137: Input parameters of xd_orbit_file_decimate function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
fhr_in	xd_fhr	-	Data structure containing the fixed header data read from the input file	-	-

osv_in	xd_orbit_file	-	Data structure containing the data read from the input file	-	-
decimation_delta_time	double	-	Delta time used for decimation process.	seconds	>=0.

7.59.4. Output parameters

The output parameters of the **xd_orbit_file_decimate** CFI function are:

Table 138: Output parameters of xd_orbit_file_decimate function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xd_orbit_file_decimate	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
fhr_out	xd_fhr	-	Data structure containing the fixed header for output file		
osv_out	xd_orbit_file	-	Data structure containing the output file data	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *osv_out* structure contains pointers to memory allocated dinamically. In order to avoid memory leaks, the user will have to free that memory when the data structure is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_orbit_file**

7.59.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_orbit_file_decimate** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_orbit_file_decimate** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 139: Error messages of xd_orbit_file_decimate function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Error allocating memory	No calculation performed	XD_ORBIT_FILE_DECIMATE_MEM_ERR	0
WARN	The time difference between 2 consecutive OSVs is greater than twice input time decimation delta	No calculation performed	XD_ORBIT_FILE_DECIMATE_DELTA_WARN	1
ERR	Error computing validity interval	No calculation performed	XD_ORBIT_FILE_DECIMATE_VAL_TIME_ERR	2

7.60.xd_attitude_file_decimate

7.60.1. Overview

The **xd_attitude_file _decimate** adds capability to configure attitude interpolator according to user need (decimation).

The decimation is performed in the attitude file structure. This way user has two options using the output of this function:

1.to write a new attitude file and use this file to initialize the attitude id.

2.To initialize directly the attitude id with the new structure.

The fucntion works as follows:

•First and last attitude records in input list are copied to output list.

•Using the input decimation delta (D), and being t0 the time of the first attitude record of the input list, the attitude records whose time is closer to time t=t0+k*D (k = 1, 2...n, t0<t<tn) are copied to output list.

7.60.2. Calling interface

The calling interface of the **xd_attitude_file_decimate** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_fhr fhr_in, fhr_out;
 xd_att_file att_in, att_out;
 double decimation_delta_time;
 long ierr[XD_NUM_ERR_ATTITUDE_FILE_DECIMATE];
 status = xd_attitude_file_decimate(&fhr_in, &att_in,
                                    decimation_delta_time,
                                    &fhr_out, &att_out,
                                    ierr);
}
```

7.60.3. Input parameters

The **xd_attitude_file_decimate** CFI function has the following input parameters:

Table 140: Input parameters of xd_attitude_file_decimate function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
fhr_in	xd_fhr	-	Data structure containing the fixed header data read from the input file	-	-

att_in	xd_att_file	-	Data structure containing the data read from the input file	-	-
decimation_delta_time	double	-	Delta time used for decimation process.	seconds	>=0.

7.60.4. Output parameters

The output parameters of the **xd_attitude_file_decimate** CFI function are:

Table 141: Output parameters of xd_attitude_file_decimate function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xd_attitude_file_decimate	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
fhr_out	xd_fhr	-	Data structure containing the fixed header for output file		
att_out	xd_att_file	-	Data structure containing the output file data	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *osv_out* structure contains pointers to memory allocated dinamically. In order to avoid memory leaks, the user will have to free that memory when the data structure is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_att.**

7.60.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_attitude_file_decimate** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_attitude_file_decimate** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 142: Error messages of xd_attitude_file_decimate function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Error allocating memory	No calculation performed	XD_CFI_ATTITUDE_FILE_DECIMATE_MEM_ERR	0
WARN	The time difference between 2 consecutive records is greater than twice input time decimation delta	Calculation performed	XD_CFI_ATTITUDE_FILE_DECIMATE_DELTA_WARN	1
ERR	Error computing validity interval	No calculation performed	XD_CFI_ATTITUDE_FILE_DECIMATE_VAL_TIME_ERR	2
WARN	Attitude record reference not UTC. UTC Validity interval computed extending one minute or more (rounded to have exact number of minutes) every end of interval	Calculation performed	XD_CFI_ATTITUDE_FILE_DECIMATE_ATT_NOT_UTC_WARN	3

7.61.xd_xslt_add

7.61.1. Overview

The **xd_xslt_add function** adds to the input file the <xml-stylesheet> tag with reference to the default style sheet.

If the tag already exists it will be updated.

The default style sheet is determined by the **file type** and by the **attitude type** (in the case of attitude files). The correspondence can be found in the following table:

Note: examples of style sheets can be found in the distribution package, in the directory files/xslt.

File Type	Attitude Type	Default styles sheet
Reference Orbit Scenario File	N/A	OSF.xslt
Predicted Orbit File	N/A	OSV_list.xslt
Doris Navigator File	N/A	OSV_list.xslt
Restituted Orbit File	N/A	OSV_list.xslt
Doris Preliminary Orbit File	N/A	OSV_list.xslt
Doris Precise Orbit File	N/A	OSV_list.xslt
Attitude File	Quaternions	att_quaternions.xslt
Attitude File	Angles	att_angles.xslt

7.61.2. Calling interface

The calling interface of the **xd_xslt_add** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
       char fname_in[];
       long ierr[XD_NUM_ERR_XSLT_ADD];
       status = xd_xslt_add(filename, ierr);
}
```

7.61.3. Input parameters

The **xd_xslt_add** CFI function has the following input parameters:

Table 143: Input parameters of xd_xslt_add function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
fname_in	char*	-	The xml file that will be updated with xslt reference	-	-

7.61.4. Output parameters

The output parameters of the **xd_xslt_add** CFI function are:

Table 144: Output parameters of xd_xslt_add function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
ierr	long[]	-	Error vector	-	-

7.61.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_xslt_add** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_xslt_add** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 145: Error messages of xd_attitude_file_decimate function

Error type	Error message	Cause and impact	Error code	Error No
ERR	File type not supported	No calculation performed	XD_XSLT_ADD_WRONG_FILE_TYPE_ERR	0

ERR	Error during initialisation	No calculation performed	XD_XSLT_ADD_INIT_PARSER_ERR	1
ERR	Unable to save the XML document into disk	No calculation performed	XD_XSLT_ADD_SAVE_DOC_ERR	2
ERR	Error reading attitude file	No calculation performed	XD_CFI_XSLT_ADD_READ_ATT_ERR	3
ERR	Wrong attitude file type. Only quaternions or angles attitudes are allowed.	No calculation performed	XD_XSLT_ADD_WRONG_ATT_FILE_TYPE_ERR	4

7.62.xd_read_oem

7.62.1. Overview

The **xd_read_oem** CFI function reads Orbit Ephemeris Message files, in text and XML formats.

The following items must be considered:

A warning is raised if at least one of the following conditions is detected:

- time going back OSV
- repeated OSV

7.62.2. Calling interface

The calling interface of the **xd_read_oem** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *oem_file;
 xd_osv_list_read_configuration read_config;

 xd_oem_file oem_data
 long ierr[XD_NUM_ERR_READ_OEM];

 status = xd_read_oem(oem_file,
                      &read_config,
                      &oem_data, ierr);
}
```

7.62.3. Input parameters

The **xd_read_oem** CFI function has the following input parameters:

Table 146: Input parameters of xd_read_oem function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
oem_file	char*	-	OEM file name	-	-
read_config	xd_osv_list_read_configuration*		Configuration for reading OSV state vectors		

It is possible to use enumeration values rather than integer values for some of the input arguments:

- Time model ID: time_mode. See [GEN_SUM].

7.62.4. Output parameters

The output parameters of the **xd_read_oem** CFI function are:

Table 147: Output parameters of xd_read_oem function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
status	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
oem_data	xd_oem_file	-	OEM data	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *oem_data* structure contains pointers to memory allocated dynamically. In order to avoid memory leaks, the user will have to free that memory when the data structure is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_oem**.

7.62.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_oem** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_oem** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 148: Error messages of xd_read_oem function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Wrong input time mode	No calculation performed	XD_READ_OEM_TIME_MODE_ERR	0
ERR	Error opening file: 'file_name'	No calculation performed	XD_READ_OEM_OPEN_FILE_ERR	1

ERR	Error allocating memory	No calculation performed	XD_READ_OEM_MEMORY_ERR	2
ERR	Wrong reference frame: 'reference_frame'	No calculation performed	XD_READ_OEM_WRONG_REF_FRAME_ERR	3
ERR	Wrong time system: 'time_system'	No calculation performed	XD_READ_OEM_WRONG_TIME_SYSTEM_ERR	4
ERR	Error reading line number 'line_number'	No calculation performed	XD_READ_OEM_READ_LINE_ERR	5
ERR	Error getting processing time	No calculation performed	XD_READ_OEM_GET_PROC_TIME_ERR	6
WARN	Time going back at OSV no. %ld	File read	XD_READ_OEM_TIME_GOING_BACK_WARN	7
WARN	Repeated OSV found at OSV no. %ld	File read	XD_READ_OEM_REPEATED_OSV_WARN	8
ERR	Error fitting the OSV array to the requested time interval	No calculation performed	XD_READ_OEM_FITING_OSV_ARRAY_TO_REQUESTED_TIME_ERR	9
WARN	Configuration time reference is different from file time	File read	XD_READ_OEM_CONFIG_TIME_REF_WARN	10

7.63.xd_free_oem

7.63.1. Overview

The **xd_free_oem** CFI function frees the memory allocated during the reading function **xd_read_oem**.

7.63.2. Calling interface

The calling interface of the **xd_free_oem** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_oem_file oem_data xd_free_oem (&oem_data);
}
```

7.63.3. Input parameters

The **xd_free_oem** CFI function has the following input parameters:

Table 149: Input parameters of xd_free_oem function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
oem_data	xd_oem_file	-	OEM data structure	-	-

7.63.4. Output parameters

This function does not return any value nor parameters.

7.64.xd_orbit_file_diagnostics

7.64.1. Overview

The **xd_orbit_file_diagnostics** CFI function computes diagnostics data related to an orbit file. Such data can be analysed to detect problems in the file or identify fragments of the file to be discarded. The following information is returned:

- Size of the interval covered by the file.
- Times of first and last OSV.
- Number and interval of GAPs in the file.
- Number and indexes of duplicated OSVs, i.e. OSVs whose time is the same as the one of previous OSV; i.e. if *time_osv1* and *time_osv2* are the times of one OSV and the following one respectively, the duplicated OSVs fulfill the following condition:

  $|time\_osv2-time\_osv1| < diagnostics\_settings.duplicated\_osv\_threshold$
- being *diagnostics_settings* one input parameter to the function (check section **Error! Reference s ource not found.**).
- Number and indexes of the OSVs going back in time, i.e. OSVs whose time is in the past with respect to the previous one; i.e. the OSVs are not identified as duplicated OSVs and fulfill the following conditions:

  1) $time\_osv2-time\_osv1 < 0$.

  2) $|time\_osv2-time\_osv1| > diagnostics\_settings.duplicated\_osv\_threshold$
- Number and indexes of OSVs with inconsistent orbit number (i.e. OSVs whose number is not correlated with its neighbours OSVs).
- Number and indexes of OSVs with non-equally spaced OSVs (i.e. OSVs that are separated from its neighbours a different step from the one expected).

For DORIS files only EF OSVS are checked, because they are the ones used by orbit initialization.

7.64.2. Calling interface

The calling interface of the **xd_orbit_file_diagnostics** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 long status;
 char *input_file;
 xd_orbit_file_diagnostics_settings diagnostics_settings;
 xd_orbit_file_diagnostics_report diagnostics_report;
 xd_eocfi_file eocfi_file;
```

```
    long ierr[XD_NUM_ERR_ORBIT_FILE_DIAGNOSTICS];

    status = xd_orbit_file_diagnostics(input_file,
                                       &eocfi_file,
                                       &diagnostics_settings,
                                       &diagnostics_report,
                                       ierr);
}
```

7.64.3. Input parameters

The **xd_orbit_file_diagnostics** CFI function has the following input parameters:

Table 150: Input parameters of xd_orbit_file_diagnostics function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
input_file	char*	-	The file that will be checked for diagnostics. The file must have one of the following types: • orbit file • doris file • SP3 file • OEM file If the pointer value is NULL, then eocfi_file parameter is used	-	-
eocfi_file	xd_eocfi_file *	-	Data from an EOCFI file: • orbit file • doris file • SP3 file • OEM file that will be checked for diagnostics.	-	-
diagnostics_settings	xd_orbit_file_diagnostics_settings	-	Diagnostic settings structure	-	-

7.64.4. Output parameters

The output parameters of the **xd_orbit_file_diagnostics** CFI function are:

Table 151: Output parameters of xd_orbit_file_diagnostics function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
status	long	-	Function status flag: • = 0 No error • > 0 Warnings, results generated • < 0 Error, no results generated	-	-
diagnostics_report	xd_orbit_file_diagnostics_report	-	Diagnostics report structure	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *xd_orbit_file_diagnostics_report* structure contains pointers to memory allocated dynamically. In order to avoid memory leaks, the user will have to free that memory when the data structure is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_orbit_file_diagnostics_report**.

7.64.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_orbit_file_diagnostics** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_oem** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 152: Error messages of xd_orbit_file_diagnostics function

Error type	Error message	Cause and impact	Error code	Error No
ERR	XD_ORBIT_FILE_DIAGNOSTICS_DETECT_INPUT_ERR	No calculation performed	Error detecting input file	0
ERR	XD_ORBIT_FILE_DIAGNOSTICS_READ_ORBIT_FILE_ERR	No calculation performed	Error reading orbit file	1
ERR	XD_ORBIT_FILE_DIAGNOSTICS_READ_DORIS_ERR	No calculation performed	Error reading doris file	2

ERR	XD_ORBIT_FILE_DIAGNOSTICS_READ_OEM_ERR	No calculation performed	Error reading OEM file	3
ERR	XD_ORBIT_FILE_DIAGNOSTICS_READ_SP3_ERR	No calculation performed	Error reading SP3 file	4
ERR	XD_ORBIT_FILE_DIAGNOSTICS_COMPUTE_DIAGNOSTICS_ERR	No calculation performed	Error computing diagnostics	5
ERR	XD_ORBIT_FILE_DIAGNOSTICS_WRONG_FILE_TYPE_ERR	No calculation performed	Wrong input file type. Only orbit files, doris files, OEM files or SP3 files are supported	6
ERR	XD_ORBIT_FILE_DIAGNOSTICS_MEMORY_ERR	No calculation performed	Error allocating memory	7

7.65.xd_free_orbit_file_diagnostics_report

7.65.1. Overview

The **xd_free_orbit_file_diagnostics_report** CFI function frees the memory allocated by the function **xd_orbit_file_diagnostics**.

7.65.2. Calling interface

The calling interface of the **xd_free_orbit_file_diagnostics_report** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
  xd_orbit_file_diagnostics_report diagnostics_report;
  xd_free_orbit_file_diagnostics_report (&diagnostics_report);
}
```

7.65.3. Input parameters

The **xd_free_orbit_file_diagnostics_report** CFI function has the following input parameters:

Table 153: Input parameters of xd_free_orbit_file_diagnostics_report function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
diagnostics_report	xd_orbit_file_diagnostics_report	-	Diagnostics report structure	-	-

7.65.4. Output parameters

This function does not return any value nor parameters.

7.66.xd_set_file_format_standard_version

7.66.1. Overview

The **xd_set_file_format_standard_version** CFI function sets the version of the Earth Observation Ground Segment File Format Standard used by the EOCFI functions to generate, write and read files.

The version used by default is mission dependent, see section 8.2. Calling xd_set_file_format_standard_version overrides the version number for all missions.

Calling xd_set_file_format_standard_version with input eoffs=XD_FFS_DEFAULT re-sets the default mission dependent value.

7.66.2. Calling interface

The calling interface of the **xd_set_file_format_standard_version** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
  long status;
  long eoffs;
  long ierr[XD_NUM_ERR_SET_FILE_FORMAT_STANDARD_VERSION];

  status = xd_set_file_format_standard_version(eoffs,ierr);
}
```

7.66.3. Input parameters

The **xd_set_file_format_standard_version** CFI function has the following input parameters:

Table 154: Input parameters of xd_set_file_format_standard_version function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
eoffs	long	-	File Format Standard version	-	Allowed values: XD_FFS_DEFAULT XD_FFS_V1 XD_FFS_V2 XD_FFS_V3

Output parameters

The output parameters of the **xd_set_file_format_standard_version** CFI function are:

Table 155: Output parameters of xd_set_file_format_standard_version function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
ierr	long[]	-	Error vector	-	-

7.66.4. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_set_file_format_standard_version** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_set_file_format_standard_version** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 156: Error messages of xd_set_file_format_standard_version function

Error type	*Error message*	*Cause and impact*	*Error code*	*Error*
ERR	*Error in set eoffs version*	*No calculation performed*	*XD_CFI_SET_FILE_FORMAT_STANDARD_VERSION WRONG_INPUT_ERR*	*0*

7.67.xd_read_aem

7.67.1. Overview

The **xd_read_aem** CFI function read an AEM file and stores the read data in the output structure.

7.67.2. Calling interface

The calling interface of the **xd_read_aem** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
  long status;
  char *file_name;
  xd_aem_file aem_file;
  long ierr[XD_NUM_ERR_READ_AEM];

  status = xd_read_aem(file_name, &aem_file, &ierr);
}
```

7.67.3. Input parameters

The **xd_read_aem** CFI function has the following input parameters:

Table 157: Input parameters of xd_read_aem function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
file_name	char*	-	Input file name (path)	-	-

7.67.4. Output parameters

The output parameters of the **xd_read_aem** CFI function are:

Table 158: Output parameters of xd_read_aem function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
aem_file	xd_aem_file*	-	AEM data read	-	-
ierr	long[]	-	Error vector	-	-

Memory Management: The *aem_data* structure contains pointers to memory allocated dynamically. In order to avoid memory leaks, the user will have to free that memory when the data structure is not to be used any more. The memory can be freed by calling to the CFI function **xd_free_aem**.

7.67.5. Warnings and errors

Next table lists the possible error messages that can be returned by the **xd_read_aem** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_DATA_HANDLING software library **xd_get_msg** (see [GEN_SUM]).

This table also indicates the type of message returned, i.e. either a warning (`WARN`) or an error (`ERR`), the cause of such a message and the impact on the performed calculation.

The table is completed by the error code and value. These error codes can be obtained translating the error vector returned by the **xd_read_aem** function by calling the function of the EO_DATA_HANDLING software library **xd_get_code** (see [GEN_SUM])

Table 159: Error messages of xd_read_aem function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Error opening file: %s	No calculation performed	XD_CFI_READ_AEM_OPEN_FILE_ERR	0
ERR	Error allocating memory	No calculation performed	XD_CFI_READ_AEM_MEMORY_ERR	1
ERR	Error reading header	No calculation performed	XD_CFI_READ_AEM_HEADER_ERR	2
ERR	Error reading metadata from segment %ld	No calculation performed	XD_CFI_READ_AEM_WRONG_METADATA_ERR	3
ERR	Error reading ephemeris data from segment %ld	No calculation performed	XD_CFI_READ_AEM_WRONG_EPHEMERIS_ERR	4
ERR	Error reading XML tag %s	No calculation performed	XD_CFI_READ_AEM_READ_XML_TAG_ERR	5

7.68.xd_free_aem

7.68.1. Overview

The **xd_free_aem** CFI function frees the memory allocated during the reading function **xd_read_aem**.

7.68.2. Calling interface

The calling interface of the **xd_free_aem** CFI function is the following (input parameters are underlined):

```
#include <explorer_data_handling.h>
{
 xd_aem_file aem_data;
 xd_free_aem (&aem_data);
}
```

7.68.3. Input parameters

The **xd_free_aem** CFI function has the following input parameters:

Table 160: Input parameters of xd_free_aem function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
aem_data	xd_aem_file	-	AEM data structure	-	-

7.68.4. Output parameters

This function does not return any value nor parameters.

8. SUPPORTED FILE TYPES

8.1.Summary

Table 161 lists the file types that are supported by the EOCFI SW. The table indicates for each file type if the file can be read and/or written (RW column), if it is compliant with EO Ground Segment File Format Standard (FFS) [FFS3] and [FFS2] (FFS column).

Table 161: List of Earth Observation Ground Segment Files

File Type	Description	R/W	FFS	Format Described in
Predicted Orbit File	List of Orbit State Vectors (i.e. position and velocity at given times), one per orbit	RW	YES	[EO_ICD], section 3.1 (*)
Restituted Orbit File	List of Orbit State Vectors (i.e. position and velocity at given times)	RW	YES	[EO_ICD], section 3.1 (*)
Orbit Scenario File	Set of parameters describing an orbit, e.g. repeat cycle, cycle length, MLST	RW	YES	[EO_ICD], section 3.2
Satellite Configuration File	Set of parameters describing an orbit, e.g. keplerian elements	R	YES	[EO_ICD], section 3.3
Attitude Quaternion File	List of quaternions at given times	RW	YES	[EO_ICD], section 3.4
Attitude Roll Pitch Yaw File	List of roll pitch yaw angles at given times	RW	YES	[EO_ICD], section 3.5
Swath Definition File	Set of parameters defining an instrument swath	R	YES	[EO_ICD], section 3.6
Swath Template File	One or more lists of latitude, longitude points defining a swath footprint	RW	YES	[EO_ICD], section 3.7
Zone Database File	One or more lists of latitude, longitude points defining zones (e.g. polygons)	R	YES	[EO_ICD], section 3.8
Station Database File	One or more set of parameters defining Ground Stations	R	YES	[EO_ICD], section 3.9
Attitude Definition File	Set of data or models defining satellite attitude	RW	YES	[EO_ICD], section 3.10
Field of View Configuration File	Set of parameters (e.g. list of azimuth, elevation) defining a field of view	R	YES	[EO_ICD], section 3.11
DEM Configuration File	Set of parameters used for DEM configuration	R	YES	This document, section **Error! Reference source not found.**
Precise Propagator Configuration File	Set of parameters used for Precise Propagator configuration	R	YES	This document, section **Error! Reference source not found.**
TLE File	Two Line Element set encoding orbital parameters	RW	NO	This document, section **Error! Reference source not found.**
Extended Standard Product 3	File containing orbit information	R	NO	This document, section **Error! R**

Orbit File (SP3-c)	(e.g. list of Orbit State Vectors)			**eference source not found.**
Orbit Ephemeris Message File (OEM)	File containing orbit information (e.g. list of Orbit State Vectors)	R	NO	This document, section **Error! R eference source not found.**
IERS bulletins	Earth Orientation parameters	R	NO	This document, section **Error! R eference source not found.**
CryoSat-2 Orbit Event File	CryoSat-2 specific Orbit File (Orbital Change plus Orbit State Vectors)	RW	YES	This document, section **Error! R eference source not found.** (**)
CryoSat-2 DORIS Navigator File	CryoSat-2 Level-0 DORIS Navigator Data	RW	NO	This document, section **Error! R eference source not found.** (**)
Sentinel-3 DORIS Navigator File	Sentinel-3 Level-0 DORIS Navigator Data	R	NO	This document, section **Error! R eference source not found.** (***)
CryoSat-2 Star Tracker File	CryoSat-2 Level-0 Star Tracker Navigator Data	R	NO	This document, section **Error! R eference source not found.** (**)
CryoSat-2 Doris Preliminary/Precise File	List of Orbit State Vectors (i.e. position and velocity at given times)	RW	YES	[EO_ICD], section 3.1 (*)
CryoSat-2 Star tracker configuration File	Set of parameters for CryoSat-2 Star tracker configuration	RW	YES	This document, section **Error! R eference source not found.** (**)
Attitude Ephemeris Message File (AEM)	File containing attitude information (e.g. list of quaternions/rotation angles)	R	NO	This document, section 0

(*) The Data Block of these files have identical format, the only difference is the name of the validating schema.

(**) these formats are deprecated for any mission except for CryoSat-2 and is maintained for backward compatibility

(***) these formats are deprecated for any mission except for Sentinel-3 and is maintained for backward compatibility

8.2.File Format Version

For files compliant with FFS, a format version number is maintained to keep track of format modifications.

Each format version has an associated validating schema file. The format version is encoded in the schema file name (e.g. the validating schema for Orbit Scenario File Format version 3.1 is named EO_OPER_MPL_ORBSCT_0301.XSD). Validating schemas can be found in the EOCFI SW distribution package and at this URL: http://eop-cfi.esa.int/CFI/EE_CFI_SCHEMAS/

Table 162 provides, for each File Type and File Format Standard Version, the latest File Format Version and the relevant validating schema.

Table 162: Mapping between File Types, FFS Version, File Format Version and validating schemas

File Type	FFS Version	File Format Version	Validating schema

Predicted Orbit File	1.0	1.5	EO_OPER_MPL_ORBPRE_0105.XSD
	2.0	2.3	EO_OPER_MPL_ORBPRE_0203.XSD
	3.0	3.0	EO_OPER_MPL_ORBPRE_0300.XSD
Restituted Orbit File	1.0	1.5	EO_OPER_AUX_ORBRES_0105.XSD
	2.0	2.3	EO_OPER_AUX_ORBRES_0203.XSD
	3.0	3.0	EO_OPER_AUX_ORBRES_0300.XSD
CryoSat-2 DORIS Preliminary File	1.0	1.5	EO_OPER_MPL_ORBDOP_0105.XSD
	2.0	N/A	N/A
	3.0	N/A	N/A
CryoSat-2 DORIS Precise File	1.0	1.5	EO_OPER_MPL_ORBDOR_0105.XSD
	2.0	N/A	N/A
	3.0	N/A	N/A
CryoSat-2 Orbit Event File	1.0	1.6	EO_OPER_MPL_ORBREF_0106.XSD
	2.0	N/A	N/A
	3.0	N/A	N/A
Orbit Scenario File	1.0	1.5	EO_OPER_MPL_ORBSCT_0105.XSD
	2.0	2.4	EO_OPER_MPL_ORBSCT_0204.XSD
	3.0	3.1	EO_OPER_MPL_ORBSCT_0301.XSD
Satellite Configuration File	1.0	1.4	EO_OPER_INT_SATCFG_0104.XSD
	2.0	2.3	EO_OPER_INT_SATCFG_0203.XSD
	3.0	3.1	EO_OPER_INT_SATCFG_0301.XSD
Attitude Quaternion File Attitude Roll Pitch Yaw File	1.0	1.4	EO_OPER_INT_ATTREF_0104.XSD
	2.0	2.4	EO_OPER_INT_ATTREF_0204.XSD
	3.0	3.1	EO_OPER_INT_ATTREF_0301.XSD
Swath Definition File	1.0	2.5	EO_OPER_MPL_SW_DEF_0205.XSD
	2.0	3.5	EO_OPER_MPL_SW_DEF_0305.XSD
	3.0	4.2	EO_OPER_MPL_SW_DEF_0402.XSD
Swath Template File	1.0	2.3	EO_OPER_MPL_SWTREF_0203.XSD
	2.0	3.3	EO_OPER_MPL_ SWTREF _0303.XSD
	3.0	4.0	EO_OPER_MPL_ SWTREF _0400.XSD
Zone Database File	1.0	1.3	EO_OPER_MPL_ZON_DB_0103.XSD
	2.0	2.2	EO_OPER_MPL_ZON_DB_0202.XSD
	3.0	3.0	EO_OPER_MPL_ZON_DB_0300.XSD
Station Database File	1.0	1.5	EO_OPER_MPL_GND_DB_0105.XSD
	2.0	2.2	EO_OPER_MPL_GND_DB_0202.XSD
	3.0	3.0	EO_OPER_MPL_GND_DB_0300.XSD
Field of View Configuration File	1.0	1.0	EO_OPER_INT_FOVCFG_0100.XSD
	2.0	2.0	EO_OPER_INT_FOVCFG_0200.XSD
	3.0	3.0	EO_OPER_INT_FOVCFG_0300.XSD
DEM configuration File	1.0	1.12	EO_OPER_INT_DEMCFG_0112.XSD
	2.0	2.10	EO_OPER_INT_DEMCFG_0210.XSD
	3.0	3.4	EO_OPER_INT_DEMCFG_0304.XSD
Precise Propagator Configuration File	1.0	1.1	EO_OPER_INT_PPRCFG_0101.XSD
	2.0	2.2	EO_OPER_INT_PPRCFG_0202.XSD

	3.0	3.0	EO_OPER_INT_PPRCFG_0300.XSD
Attitude Definition File	1.0	1.4	EO_OPER_INT_ATTDEF_0104.XSD
	2.0	2.5	EO_OPER_INT_ATTDEF_0205.XSD
	3.0	3.2	EO_OPER_INT_ATTDEF_0302.XSD
CryoSat Star Tracker Configuration File	1.0	1.2	EO_OPER_INT_STRCFG_0102.XSD
	2.0	N/A	N/A
	3.0	N/A	N/A

Example files for each File Format Version are provided within the distribution package and at the following URL:

http://eop-cfi.esa.int/CFI/EE_CFI_SCHEMAS/example_files

Example files are listed in Table 163.

Table 163: List of example files

File Type	FFS Version	File Format Version	Validating schema
Predicted Orbit File	1.0	1.5	CS_TEST_MPL_ORBPRE_20100409T105737_20100410T015421_0007.EEF
	2.0	2.3	S1A_TEST_MPL_ORBPRE_20140404T183104_20140405T091945_0004.EOF
	3.0	3.0	MA1_TEST_MPL_ORBPRE_20210401T174620_20210402T085834_0001.EOF
Restituted Orbit File	1.0	1.5	CS_TEST_AUX_ORBRES_20100616T174826_20100616T194756_0007.EEF CS_TEST_AUX_ORBRES_20100616T175926_20100616T180826_0007.EEF
	2.0	2.3	S1A_TEST_AUX_ORBRES_20140611T104016_20140611T123846_0004.EOF S1A_TEST_AUX_ORBRES_20140611T105116_20140611T110016_0004.EOF
	3.0	3.0	MA1_TEST_AUX_ORBRES_20210610T050853_20210610T051753_0001.EOF MA1_TEST_AUX_ORBRES_20210610T045753_20210610T065853_0001.EOF
CryoSat-2 DORIS Preliminary File	1.0	1.5	CS_TEST_AUX_ORBDOP_20100616T174826_20100616T194756_0007.EEF
	2.0	N/A	N/A
	3.0	N/A	N/A
CryoSat-2 DORIS Precise File	1.0	1.5	CS_TEST_AUX_ORBDOR_20100616T174826_20100616T194756_0007.EEF
	2.0	N/A	N/A
	3.0	N/A	N/A
CryoSat-2 Orbit Event File	1.0	1.6	EO_OPER_MPL_ORBREF_0106.XSD
	2.0	N/A	N/A
	3.0	N/A	N/A

Orbit Scenario File	1.0	1.5	CS_TEST_MPL_ORBSCT_20100408T150159_99999999T999999_0006.EEF
	2.0	2.4	S1A_TEST_MPL_ORBSCT_20140403T224609_99999999T999999_0006.EOF
	3.0	3.1	MA1_TEST_MPL_ORBSCT_20210331T213001_99999999T999999_0001.EOF
Satellite Configuration File	1.0	1.4	CS_TEST_INT_SATCFG_00000000T000000_99999999T999999_0104.EEF
	2.0	2.3	S1A_TEST_INT_SATCFG_00000000T000000_99999999T999999_0203.EOF
	3.0	3.1	MA1_TEST_INT_SATCFG_00000000T000000_99999999T999999_0301.EOF
Attitude Quaternion File Attitude Roll Pitch Yaw File	1.0	1.4	CS_TEST_INT_ATTREF_20100616T174826_20100616T194756_0005.EEF
	2.0	2.4	S1A_TEST_INT_ATTREF_20140611T104016_20140611T123846_0005.EOF
	3.0	3.1	MA1_TEST_INT_ATTREF_20210610T045753_20210610T065853_0002.EOF
Swath Definition File	1.0	2.5	CS_TEST_MPL_SW_DEF_00000000T000000_99999999T999999_0006.EEF
	2.0	3.5	S2A_TEST_MPL_SW_DEF_00000000T000000_99999999T999999_0003.EOF
	3.0	4.2	MA1_TEST_MPL_SW_DEF_00000000T000000_99999999T999999_0002.EOF
Swath Template File	1.0	2.3	CS_TEST_MPL_SWTREF_00000000T000000_99999999T999999_0009.EEF
	2.0	3.3	S1A_TEST_MPL_SWTREF_00000000T000000_99999999T999999_0004.EOF
	3.0	4.0	MA1_TEST_MPL_SWTREF_00000000T000000_99999999T999999_0001.EOF
Zone Database File	1.0	1.3	CS_TEST_MPL_ZON_DB_00000000T000000_99999999T999999_0003.EEF
	2.0	2.2	S1A_TEST_MPL_ZON_DB_00000000T000000_99999999T999999_0002.EOF
	3.0	3.0	MA1_TEST_MPL_ZON_DB_00000000T000000_99999999T999999_0001.EOF
Station Database File	1.0	1.5	CS_TEST_MPL_GND_DB_00000000T000000_99999999T999999_0005.EEF
	2.0	2.2	S1A_TEST_MPL_GND_DB_00000000T000000_99999999T999999_0002.EOF
	3.0	3.0	MA1_TEST_MPL_GND_DB_00000000T000000_99999999T999999_0001.EOF
Field of View Configuration File	1.0	1.0	CS_TEST_INT_FOVCFG_00000000T000000_99999999T999999_0001.EEF
	2.0	2.0	S1A_TEST_INT_FOVCFG_00000000T000000_99999999T999999_0001.EOF
	3.0	3.0	MA1_TEST_INT_FOVCFG_00000000T000000_99999999T999999_0001.EOF

DEM configuration File	1.0	1.9	CS_TEST_INT_DEMCFG_00000000T000000_99999999T999999_0010.EEF
	2.0	2.7	S1A_TEST_INT_DEMCFG_00000000T000000_99999999T999999_0008.EOF
	3.0	3.4	MA1_TEST_INT_DEMCFG_00000000T000000_99999999T999999_0003.EOF MA1_TEST_INT_DEMCFG_00000000T000000_99999999T999999_0004.EOF
Precise Propagator Configuration File	1.0	1.1	CS_TEST_INT_PPRCFG_00000000T000000_99999999T999999_0002.EEF
	2.0	2.2	S1A_TEST_INT_PPRCFG_00000000T000000_99999999T999999_0002.EOF
	3.0	3.0	MA1_TEST_INT_PPRCFG_00000000T000000_99999999T999999_0001.EOF
Attitude Definition File	1.0	1.4	CS_TEST_INT_ATTDEF_00000000T000000_99999999T999999_0004.EEF
	2.0	2.5	S1A_TEST_INT_ATTDEF_00000000T000000_99999999T999999_0005.EOF
	3.0	3.2	MA1_TEST_INT_ATTDEF_00000000T000000_99999999T999999_0002.EOF
CryoSat Star Tracker Configuration File	1.0	1.2	CS_TEST_INT_STRCFG_20040101T000000_99999999T999999_0002.EEF
	2.0	N/A	N/A
	3.0	N/A	N/A
DEM Raster configuration file	3.0	3.1	dem_raster_configuration_0001.xml dem_raster_configuration_0002.xml

The EOCFI SW:

- Is able to read files of latest format versions listed in Table 2 regardless of the File Format Standard.
- Is able to read files of older format versions listed in Table 5 regardless of the File Format Standard.
- Writes files of format listed in Table 162 using the applicable File Format Standard Version. The applicable File Format Standard Version is mission dependent, the correspondence between missions and applicable File Format Standard Version is given in Table 164. The default can be overridden by using function **xd_set_file_format_standard_version** (section 7.66).

Table 164: Mapping between Missions and applicable FFS Version

Mission	Applicable FFS Version
ERS-1, ERS-2 ENVISAT Metop-A, Metop-B, Metop-C CryoSat-2 Aeolus Goce Smos Terrasar Swarm-A, Swarm-B, Swarm-C	1.0

Seosat	
Sentinel-1A, Sentinel-1B, Sentinel-1C, Sentinel-1D Sentinel-2A, Sentinel-2B, Sentinel-2C, Sentinel-2D Sentinel-3A, Sentinel-3B, Sentinel-3C Sentinel-5P EarthCARE	2.0
Sentinel-5 MetopSG-A1, MetopSG-A2, MetopSG-A3 MetopSG-B1, MetopSG-B2, MetopSG-B3 Biomass JasonCS-A, JasonCS-B Saocom-CS FLEX Sentinel-6A, Sentinel-6B, Sentinel-6C CIMR ROSEL CHIME CRISTAL CO2M LSTM FORUM TRUTHS, HARMONY-A, HARMONY-B ALTIUS	3.0

Table 165: List of older format versions and corresponding validating schemas

File Type	FFS Version	File Format Version	Validating schema
Predicted Orbit File	1.0	1.1 1.2 1.3 1.4	EO_OPER_MPL_ORBPRE_0101.XSD EO_OPER_MPL_ORBPRE_0102.XSD EO_OPER_MPL_ORBPRE_0103.XSD EO_OPER_MPL_ORBPRE_0104.XSD
	2.0	2.0 2.1 2.2	EO_OPER_MPL_ORBPRE_0200.XSD EO_OPER_MPL_ORBPRE_0201.XSD EO_OPER_MPL_ORBPRE_0202.XSD
	3.0	N/A	N/A
Restituted Orbit File	1.0	1.1 1.2 1.3 1.4	EO_OPER_AUX_ORBRES_0101.XSD EO_OPER_AUX_ORBRES_0102.XSD EO_OPER_AUX_ORBRES_0103.XSD EO_OPER_AUX_ORBRES_0104.XSD
	2.0	2.0 2.1 2.2	EO_OPER_AUX_ORBRES_0200.XSD EO_OPER_AUX_ORBRES_0201.XSD EO_OPER_AUX_ORBRES_0202.XSD
	3.0	N/A	N/A
CryoSat-2 DORIS Preliminary File	1.0	1.1 1.2 1.3 1.4	EO_OPER_AUX_ORBDOP_0101.XSD EO_OPER_AUX_ORBDOP_0102.XSD EO_OPER_AUX_ORBDOP_0103.XSD EO_OPER_AUX_ORBDOP_0104.XSD

	2.0	N/A	N/A
	3.0	N/A	N/A
CryoSat-2 DORIS Precise File	1.0	1.1 1.2 1.3 1.4	EO_OPER_AUX_ORBDOR_0101.XSD EO_OPER_AUX_ORBDOR_0102.XSD EO_OPER_AUX_ORBDOR_0103.XSD EO_OPER_AUX_ORBDOR_0104.XSD
	2.0	N/A	N/A
	3.0	N/A	N/A
CryoSat-2 Orbit Event File	1.0	1.1 1.2 1.3 1.4 1.5	EO_OPER_MPL_ORBREF_0101.XSD EO_OPER_MPL_ORBREF_0102.XSD EO_OPER_MPL_ORBREF_0103.XSD EO_OPER_MPL_ORBREF_0104.XSD EO_OPER_MPL_ORBREF_0105.XSD
	2.0	N/A	N/A
	3.0	N/A	N/A
Orbit Scenario File	1.0	1.1 1.2 1.3 1.4	EO_OPER_MPL_ORBSCT_0101.XSD EO_OPER_MPL_ORBSCT_0102.XSD EO_OPER_MPL_ORBSCT_0103.XSD EO_OPER_MPL_ORBSCT_0104.XSD
	2.0	2.0 2.1 2.2 2.3	EO_OPER_MPL_ORBSCT_0200.XSD EO_OPER_MPL_ORBSCT_0201.XSD EO_OPER_MPL_ORBSCT_0202.XSD EO_OPER_MPL_ORBSCT_0203.XSD
	3.0	N/A	N/A
Satellite Configuration File	1.0	1.2 1.3	EO_OPER_INT_SATCFG_0102.XSD EO_OPER_INT_SATCFG_0103.XSD
	2.0	2.0 2.1 2.2	EO_OPER_INT_SATCFG_0200.XSD EO_OPER_INT_SATCFG_0201.XSD EO_OPER_INT_SATCFG_0202.XSD
	3.0	3.0	EO_OPER_INT_SATCFG_0300.XSD
Attitude Quaternion File Attitude Roll Pitch Yaw File	1.0	1.1 1.2 1.3	EO_OPER_INT_ATTREF_0101.XSD EO_OPER_INT_ATTREF_0102.XSD EO_OPER_INT_ATTREF_0103.XSD
	2.0	2.0 2.1 2.2 2.3	EO_OPER_INT_ATTREF_0200.XSD EO_OPER_INT_ATTREF_0201.XSD EO_OPER_INT_ATTREF_0202.XSD EO_OPER_INT_ATTREF_0203.XSD
	3.0	3.0	EO_OPER_INT_ATTREF_0300.XSD
Swath Definition File	1.0	1.1 2.1 2.2 2.3 2.4 2.5	EO_OPER_MPL_SW_DEF_0101.XSD EO_OPER_MPL_SW_DEF_0201.XSD EO_OPER_MPL_SW_DEF_0202.XSD EO_OPER_MPL_SW_DEF_0203.XSD EO_OPER_MPL_SW_DEF_0204.XSD EO_OPER_MPL_SW_DEF_0205.XSD
	2.0	3.0 3.1 3.2 3.3 3.4	EO_OPER_MPL_SW_DEF_0300.XSD EO_OPER_MPL_SW_DEF_0301.XSD EO_OPER_MPL_SW_DEF_0302.XSD EO_OPER_MPL_SW_DEF_0303.XSD EO_OPER_MPL_SW_DEF_0304.XSD

		3.5	EO_OPER_MPL_SW_DEF_0305.XSD
	3.0	4.0	EO_OPER_MPL_SW_DEF_0400.XSD
		4.1	EO_OPER_MPL_SW_DEF_0401.XSD
		4.2	EO_OPER_MPL_SW_DEF_0402.XSD
Swath Template File	1.0	1.1	EO_OPER_MPL_SWTREF_0101.XSD
		2.0	EO_OPER_MPL_SWTREF_0200.XSD
		2.1	EO_OPER_MPL_SWTREF_0201.XSD
		2.2	EO_OPER_MPL_SWTREF_0202.XSD
	2.0	3.0	EO_OPER_MPL_SWTREF_0300.XSD
		3.1	EO_OPER_MPL_SWTREF_0301.XSD
		3.2	EO_OPER_MPL_SWTREF_0302.XSD
	3.0	N/A	N/A
Zone Database File	1.0	1.1	EO_OPER_MPL_ZON_DB_0101.XSD
		1.2	EO_OPER_MPL_ZON_DB_0102.XSD
	2.0	2.0	EO_OPER_MPL_ZON_DB_0200.XSD
		2.1	EO_OPER_MPL_ZON_DB_0201.XSD
	3.0	N/A	N/A
Station Database File	1.0	1.1	EO_OPER_MPL_GND_DB_0101.XSD
		1.2	EO_OPER_MPL_GND_DB_0102.XSD
		1.3	EO_OPER_MPL_GND_DB_0103.XSD
		1.4	EO_OPER_MPL_GND_DB_0104.XSD
		1.5	EO_OPER_MPL_GND_DB_0105.XSD
	2.0	2.0	EO_OPER_MPL_GND_DB_0200.XSD
		2.1	EO_OPER_MPL_GND_DB_0201.XSD
	3.0	N/A	N/A
Field of View Configuration File	1.0	1.0	N/A
	2.0	2.0	N/A
	3.0	3.0	N/A
DEM configuration File	1.0	1.1	EO_OPER_INT_DEMCFG_0101.XSD
		1.2	EO_OPER_INT_DEMCFG_0102.XSD
		1.3	EO_OPER_INT_DEMCFG_0103.XSD
		1.4	EO_OPER_INT_DEMCFG_0104.XSD
		1.5	EO_OPER_INT_DEMCFG_0105.XSD
		1.6	EO_OPER_INT_DEMCFG_0106.XSD
		1.7	EO_OPER_INT_DEMCFG_0107.XSD
		1.8	EO_OPER_INT_DEMCFG_0108.XSD
		1.9	EO_OPER_INT_DEMCFG_0109.XSD
		1.10	EO_OPER_INT_DEMCFG_0110.XSD
		1.11	EO_OPER_INT_DEMCFG_0111.XSD
	2.0	2.0	EO_OPER_INT_DEMCFG_0200.XSD
		2.1	EO_OPER_INT_DEMCFG_0201.XSD
		2.2	EO_OPER_INT_DEMCFG_0202.XSD
		2.3	EO_OPER_INT_DEMCFG_0203.XSD
		2.4	EO_OPER_INT_DEMCFG_0204.XSD
		2.5	EO_OPER_INT_DEMCFG_0205.XSD
		2.6	EO_OPER_INT_DEMCFG_0206.XSD
		2.7	EO_OPER_INT_DEMCFG_0207.XSD
		2.8	EO_OPER_INT_DEMCFG_0208.XSD
		2.9	EO_OPER_INT_DEMCFG_0209.XSD
	3.0	3.0	EO_OPER_INT_DEMCFG_0300.XSD

		3.1 3.2 3.3	EO_OPER_INT_DEMCFG_0301.XSD EO_OPER_INT_DEMCFG_0302.XSD EO_OPER_INT_DEMCFG_0303.XSD
Precise Propagator Configuration File	1.0	1.0	EO_OPER_INT_PPRCFG_0100.XSD
	2.0	2.0 2.1	EO_OPER_INT_PPRCFG_0200.XSD EO_OPER_INT_PPRCFG_0201.XSD
	3.0	N/A	N/A
Attitude Definition File	1.0	1.1 1.2 1.3	EO_OPER_INT_ATTDEF_0101.XSD EO_OPER_INT_ATTDEF_0102.XSD EO_OPER_INT_ATTDEF_0103.XSD
	2.0	2.0 2.1 2.2 2.3 2.4	EO_OPER_INT_ATTDEF_0200.XSD EO_OPER_INT_ATTDEF_0201.XSD EO_OPER_INT_ATTDEF_0202.XSD EO_OPER_INT_ATTDEF_0203.XSD EO_OPER_INT_ATTDEF_0204.XSD
	3.0	3.0 3.1	EO_OPER_INT_ATTDEF_0300.XSD EO_OPER_INT_ATTDEF_0301.XSD

8.3. File Format Specification

This section provides the description of file formats that are not specified in [EO_ICD].

For files compliant with File Format Standard, the specification includes:

- the content of the Variable Header;
- the content of the Data Block;
- the reference to the validating schema for FFS v3.0 (shortly named “Schema Reference“), of FFS v1.4 for files only applicable to CryoSat.

8.3.1. DEM Configuration File

8.3.1.1. Variable Header

The Variable Header is empty for this file type.

8.3.1.2. Data Block

The Data Block content is a sequence of XML elements described in Table 166.

Table 166: Data Block content

XML Tag name	Type	Attributes	C Format	Description
DEM	Structure (see Table 167)	-	-	Structure containing the DEM model.

Table 167: DEM structure

XML Tag name	Type	Attributes	C Format	Description
DEM_User_Parameters	Structure (see Table 168)	-	-	Structure containing the User parameters
DEM_Metadata	Structure (see Table 169)	-	-	Structure containing the DEM Metadata.

Table 168: DEM_User_Parameters structure

XML Tag name	Type	Attributes	C Format	Description
Directory	string	-	%s	Directory where all DEM files are located. It can be an absolute or relative path. All files shall be located in the same directory. About supported DEM types, see [MCD], section 8.2.5. If the tag is empty, the DEM files are looked for in the same directory where the DEM configuration file is located. If a relative path is used, then it will be considered as relative to the current working directory.
Cache_Type (optional)	string	-	%s	Type of cache used for DEM computations. Possible values: NO_CACHE PRELOAD_CACHE FIFO_CACHE
Cache_Max_Size (optional)	integer	size=”MB”	%d	Maximum size of memory cache
MiniTiles_Configuration (optional)	Structure (see Table 170)	-	-	Mini tile configuration for DEM maximum altitude algorithm. Optional parameter. If not provided, mini tiles will not be used.
Geoid_Computation (optional)	string	-	%s	Flag to indicate if geoid correction must be performed or not in DEM computations. Possible values: • Enabled • Disabled Optional parameter. If not provided, the geoid computation is enabled.
Geoid_Nof_Harmonics (optional)	integer	-	%d	Number of harmonics to be used in geoid correction computation. Optional parameter. If not provided the default number of parameters (=30) will be used.

Table 169: DEM_Metadata structure

XML Tag name	Type	Attributes	C Format	Description
Dataset_Model	String	-	%s	Supported dataset models (see [MCD], section 8.2.5). DEM model: • ACE2_3SEC • ACE2_30SEC • ACE2_9SEC • ACE2_5MIN • GDEM_V2 • GENERIC (*) • GETASSE30_V1 • GETASSE30_V2 • GETASSE30_V3 • TANDEMX90 • GDEM_V3
Description	String	-	%s	DEM description

(*) When GENERIC type is used, the file describing the raster (DEM Generic Raster Configuration File) must be also generated. The description of this file can be found in [EO_ICD].

Table 170: Mini tile configuration

XML Tag name	Type	Attributes	C Format	Description
Filename	String	-	%s	Filename or path of the maximum altitude binary file.
Lon_Size	Real	unit="deg"	%d	Longitude size of mini tiles
Lat_Size	Real	unit="deg"	%d	Latitude size of mini tiles

Example:

```
<Data_Block type="xml">
    <DEM>
      <DEM_User_Parameters>
        <Directory>../../data/ACE2_9SEC</Directory>
        <Cache_Type>FIFO_CACHE</Cache_Type>
        <Cache_Max_Size size="MB">2048</Cache_Max_Size>
      </DEM_User_Parameters>
      <DEM_Metadata>
        <Dataset_Model>ACE2_9SEC</Dataset_Model>
        <Description></Description>
      </DEM_Metadata>
    </DEM>
  </Data_Block>
```

8.3.1.3. Schema Reference

An example of validating XML schema for this file type is located at:

http://eop-cfi.esa.int/CFI/EE_CFI_SCHEMAS/EO_OPER_INT_DEMCFG_0304.XSD

This schema is compliant to [EO_SCH_HB] and includes format and range checks to ensure compliance to this specification and to the File Format Standard [FFS3]. The schema file is named according to section 6.1.1 in [EO_SCH_HB] and is applicable to files named MMM_**OPER_INT_DEMCFG**_<instance_id>.EOF.

The following is the content of the Earth_Observation_File required to reference the above schema.

```
<Earth_Observation_File xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://eop-cfi.esa.int/CFI http://eop-cfi.esa.int/CFI/EE_CFI_SCHEMAS/EO_OPER_INT_DEMCFG_0303.XSD" schemaVersion="3.3" xmlns="http://eop-cfi.esa.int/CFI">
```

8.3.2. Precise Propagator Configuration File

8.3.2.1. Variable Header

The Variable Header is empty for this file type.

8.3.2.2. Data Block

The Data Block content is a sequence of XML elements described in Table 171.

Table 171: Data Block content

XML Tag name	Type	Attributes	C Format	Description
Models_Path	string	-	%s	Path where files neccesary for models are looked for.
Gravity_Flag	long integer	-	%ld	Gravity perturbation used (1) or not (0).
Thirdbody_Flag	long integer	-	%ld	Third bodies (Sun and Moon) perturbation used (1) or not (0).
Atmosphere_Flag	long integer	-	%ld	Atmosphere perturbation used (1) or not (0).
Srp_Flag	long integer	-	%ld	Solar radiation pressure perturbation used (1) or not (0).
Time_Step	real	unit=”s”	%lf	Simulation step.
Gravity_File	string	-	%s	File with data of gravitational model.
Gravity_Degree	long integer	-	%ld	Degree used gravity model.
Gravity_Order	long integer	-	%ld	Order used in gravity model.
Sga_Flag	long integer	-	%ld	Parameters used (0) or data read from file (1).
Sga_Ap_File	string	-	%s	File with Geomagnetic Activity index values.
Sga_F107_File	string	-	%s	File with F10.7 Solar Activity index values
AP	real	-	%lf	Geomagnetic Activity Index (daily value).

F107	real	-	%lf	F10.7 Index Solar Activy Index (daily value).
F107A	real	-	%lf	F10.7 Index Solar Activy Index (value averaged over 3 months).
SC_Mass	real	unit="kg"	%lf	S/C mass.
SC_Drag_Area	real	unit="m2"	%lf	S/C effective drag area.
SC_Drag_Coef	real	-	%lf	S/C drag coefficient.
SC_Srp_Area	real	unit="m2"	%lf	S/C effective SRP area.
SC_Srp_Coef	real	-	%lf	S/C SRP coefficient.

Example:

```
<Data_Block type="xml">
    <Models_Path>/models_full_path/models</Models_Path>
    <Gravity_Flag>1</Gravity_Flag>
    <Thirdbody_Flag>1</Thirdbody_Flag>
    <Atmosphere_Flag>1</Atmosphere_Flag>
    <Srp_Flag>1</Srp_Flag>
    <Time_Step unit="s">100.000000</Time_Step>
    <Gravity_File>gravity_file.grv</Gravity_File>
    <Gravity_Degree>9</Gravity_Degree>
    <Gravity_Order>8</Gravity_Order>
    <Sga_Flag>1</Sga_Flag>
    <Sga_Ap_File>ap_file.sga</Sga_Ap_File>
    <Sga_F107_File>f107_file.sga</Sga_F107_File>
    <AP>100.000000</AP>
    <F107>30.000000</F107>
    <F107A>29.000000</F107A>
    <SC_Mass unit="kg">2000.000000</SC_Mass>
    <SC_Drag_Area unit="m2">4.000000</SC_Drag_Area>
    <SC_Drag_Coef>2.000000</SC_Drag_Coef>
    <SC_Srp_Area unit="m2">3.000000</SC_Srp_Area>
    <SC_Srp_Coef>1.000000</SC_Srp_Coef>
</Data_Block>
```

8.3.2.3. Schema Reference

An example of validating XML schema for this file type is located at:

http://eop-cfi.esa.int/CFI/EE_CFI_SCHEMAS/EO_OPER_INT_PPRCFG_0300.XSD

This schema is compliant to [EO_SCH_HB] and includes format and range checks to ensure compliance to this specification and to the File Format Standard [FFS3]. The schema file is named according to section 6.1.1 in [EO_SCH_HB] and is applicable to files named MMM_**OPER_INT_PPRCFG**_<instance_id>.EOF.

The following is the content of the Earth_Observation_File required to reference the above schema.

```
<Earth_Observation_File xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://eop-cfi.esa.int/CFI http://eop-cfi.esa.int/CFI/EE_CFI_SCHEMAS/EO_OPER_INT_PPRCFG_0300.XSD" schemaVersion="3.0" xmlns="http://eop-cfi.esa.int/CFI">
```

8.3.3. TLE File

The format of the TLE files are described in [TLE].

A few TLE items (Name, Designator, Catalog Number) are part of the NORAD Satellite Catalog (SATCAT) and are assigned by NORAD after satellite launch.

The EOCFI SW uses, for each pre-defined satellite ID, a set of default SATCAT items as defined in table Table: NORAD Identifiers for satellites.

For non pre-defined satellite IDs (i.e. "Default" Satellites, see section 7.2 of [GEN_SUM], the NORAD SATCAT items can be set directly via the satellite configuration file, see [EO_ICD]).

The user can change such default values by using function xl_set_tle_sat_data (see section 7.48 of [LIB_SUM]).

Table 172: NORAD Identifiers for satellites

Satellite ID	NORAD Satellite Number	NORAD Satellite Name	NORAD International Designator
XD_SAT_ERS1	21574	ERS1	“91050A "
XD_SAT_ERS2	23560	ERS2	"95021A "
XD_SAT_ENVISAT	27386	ENVISAT	"02009A "
XD_SAT_METOP1	29499	METOP-A	"06044A "
XD_SAT_METOP2	38771	METOP-B	"12049A "
XD_SAT_METOP3	00000	METOP-C	"00000 "
XD_SAT_CRYOSAT	36508	CRYOSAT 2	"10013A "
XD_SAT_ADM	43600	AEOLUS	"18066A "
XD_SAT_GOCE	34602	GOCE	"09013A "
XD_SAT_SMOS	36036	SMOS	"09059A "
XD_SAT_TERRASAR	00000	TERRASAR	"00000 "
XD_SAT_EARTHCARE	00000	EARTHCARE	"00000 "
XD_SAT_SWARM_A	39452	SWARM A	"13067B "
XD_SAT_SWARM_B	39451	SWARM B	"13067A "
XD_SAT_SWARM_C	39453	SWARM C	"13067C "
XD_SAT_SENTINEL_1A	39634	SENTINEL-1A	"14016A "
XD_SAT_SENTINEL_1B	41456	SENTINEL-1B	"16025A "
XD_SAT_SENTINEL_2	00000	SENTINEL2	"00000 "
XD_SAT_SENTINEL_3	00000	SENTINEL3	"00000 "
XD_SAT_SENTINEL_1C	00000	SENTINEL1C	"00000 "
XD_SAT_SENTINEL_2A	40697	SENTINEL-2A	"15028A "
XD_SAT_SENTINEL_2B	42063	SENTINEL2B	"17013A "
XD_SAT_SENTINEL_2C	00000	SENTINEL2C	"00000 "
XD_SAT_SENTINEL_3A	41335	SENTINEL-3A	"16011A "
XD_SAT_SENTINEL_3B	43437	SENTINEL-3B	"18039A "
XD_SAT_SENTINEL_3C	00000	SENTINEL3C	"00000 "

XD_SAT_JASON_CSA	00000	JASONCSA	"00000 "
XD_SAT_JASON_CSB	00000	JASONCSB	"00000 "
XD_SAT_METOP_SG_A1	00000	METOPSGA1	"00000 "
XD_SAT_METOP_SG_A2	00000	METOPSGA2	"00000 "
XD_SAT_METOP_SG_A3	00000	METOPSGA3	"00000 "
XD_SAT_METOP_SG_B1	00000	METOPSGB1	"00000 "
XD_SAT_METOP_SG_B2	00000	METOPSGB2	"00000 "
XD_SAT_METOP_SG_B3	00000	METOPSGB3	"00000 "
XD_SAT_SENTINEL_5P	42969	SENTINEL-5P	"17064A "
XD_SAT_BIOMASS	00000	BIOMASS	"00000 "
XD_SAT_SENTINEL_5	00000	SENTINEL_5	"00000 "
XD_SAT_SAOCOM_CS	00000	SAOCOM_CS	"00000 "
XD_SAT_FLEX	00000	FLEX	“00000 ”
XD_SAT_SEOSAT	00000	SEOSAT	"00000 "
XD_SAT_SENTINEL_6A	00000	UNKNOWN	"00000 "
XD_SAT_SENTINEL_6B	00000	UNKNOWN	"00000 "
XD_SAT_CIMR	00000	UNKNOWN	"00000 "
XD_SAT_ROSEL	00000	UNKNOWN	"00000 "
XD_SAT_CHIME	00000	CHIME	"00000 "
XD_SAT_CRISTAL	00000	CRISTAL	"00000 "
XD_SAT_CO2M	00000	CO2M	"00000 "
XD_SAT_LSTM	00000	LSTM	"00000 "
XD_SAT_FORUM	00000	UNKNOWN	"00000 "
XD_SAT_TRUTHS	00000	TRUTHS	"00000 "
XD_SAT_SENTINEL_1D	66315	SENTINEL1D	"25251A "
XD_SAT_SENTINEL_2D	00000	SENTINEL2D	"00000 "
XD_SAT_SENTINEL_6C	00000	SENTINEL6C	"00000 "
XD_SAT_HARMONY_A	00000	HARMONY-A	"00000 "
XD_SAT_HARMONY_B	00000	HARMONY-B	"00000 "
XD_SAT_ALTIUS	00000	ALTIUS	"00000 "
XD_SAT_GENERIC	00000	GENERIC	"00000 "

8.3.4. Extended Standard Product 3 Orbit File (SP3-c)

The format of the SP3 files is described in [SP3].

The SP3 files use an identifier (SP3 Id) to identify the satellite for which the SP3 data is given.

The EOCFI SW uses, for each pre-defined satellite ID, a default SP3 Id, so that CFI functions can extract the data from the SP3 files for a given satellite ID.

The user can change such default values by using the function xl_set_sp3_sat_data (see [LIB_SUM]).

The following table shows the relation between the CFI satellite Id and the SP3 Id:

Table 173: SP3 Identifiers for satellites

Satellite ID	SP3 Id

XD_SAT_DEFAULT	A00
XD_SAT_DEFAULT*x (x=1 to 9)*	A00
XD_SAT_ERS1	L31
XD_SAT_ERS2	L32
XD_SAT_ENVISAT	L33
XD_SAT_METOP1	A00
XD_SAT_METOP2	L41
XD_SAT_METOP3	L42
XD_SAT_CRYOSAT	L12
XD_SAT_ADM	A00
XD_SAT_GOCE	L15
XD_SAT_SMOS	A00
XD_SAT_TERRASAR	L13
XD_SAT_EARTHCARE	A00
XD_SAT_SWARM_A	L47
XD_SAT_SWARM_B	L48
XD_SAT_SWARM_C	L49
XD_SAT_SENTINEL_1A	L70
XD_SAT_SENTINEL_1B	L71
XD_SAT_SENTINEL_2	A00
XD_SAT_SENTINEL_3	A00
XD_SAT_SENTINEL_1C	A00
XD_SAT_SENTINEL_2A	L72
XD_SAT_SENTINEL_2B	L73
XD_SAT_SENTINEL_2C	A00
XD_SAT_SENTINEL_3A	L74
XD_SAT_SENTINEL_3B	L75
XD_SAT_SENTINEL_3C	A00
XD_SAT_JASON_CSA	L08
XD_SAT_JASON_CSB	L27
XD_SAT_METOP_SG_A1	L88
XD_SAT_METOP_SG_A2	A00
XD_SAT_METOP_SG_A3	A00
XD_SAT_METOP_SG_B1	L89
XD_SAT_METOP_SG_B2	A00
XD_SAT_METOP_SG_B3	A00
XD_SAT_SENTINEL_5P	A00
XD_SAT_BIOMASS	A00
XD_SAT_SENTINEL_5	A00
XD_SAT_SAOCOM_CS	A00
XD_SAT_FLEX	A00
XD_SAT_SEOSAT	A00
XD_SAT_SENTINEL_6A	L40
XD_SAT_SENTINEL_6B	A00

XD_SAT_CIMR	A00
XD_SAT_ROSEL	A00
XD_SAT_CHIME	A00
XD_SAT_CRISTAL	A00
XD_SAT_CO2M	A00
XD_SAT_LSTM	A00
XD_SAT_FORUM	A00
XD_SAT_TRUTHS	A00
XD_SAT_SENTINEL_1D	A00
XD_SAT_SENTINEL_2D	A00
XD_SAT_SENTINEL_6C	A00
XD_SAT_HARMONY_A	A00
XD_SAT_HARMONY_B	A00
XD_SAT_ALTIUS	A00
XD_SAT_GENERIC	A00
XD_SAT_GENERIC_GEO	A00
XD_SAT_MTG	MTG
XD_SAT_GENERIC_MEO	A00

8.3.5. Orbit Ephemeris Message File (OEM)

The format of the OEM files is described in [OEM] (text format) and [OEM_XML] (XML format).

Table 174 shows the mapping between the OEM file and the CFI structure xd_oem_file. The fields that are **not** read by the function **xd_read_oem** are marked with N/A.

Table 174: List of OEM fields read by EOCFI

OEM File Section	OEM File Field	xd_oem_file field	Notes
OEM Header	CCSDS_OEM_VERS	ccsds_oem_vers	
	COMMENT	comment_header	
	CREATION_DATE	creation_date	
	ORIGINATOR	originator	
OEM Metadata	META_START	N/A	
	COMMENT	comment_metadata	
	OBJECT_NAME	object_name	
	OBJECT_ID	object_id	
	CENTER_NAME	center_name	
	REF_FRAME	ref_frame	Only the following reference frames are supported by CFI: • TOD • EME2000 • ICRF • ITRF-93

			• ITRF-97 • ITRF2000 • ITRFxxxx The table 175 shows the mapping between OEM reference frames and EOCFI refrence frames.
	REF_FRAME_EPOCH	N/A	
	TIME_SYSTEM	time_system	Only the following time systems are supported by CFI: • UTC • TAI • GPS • UT1
	START_TIME	start_time	
	USEABLE_START_TIME	useable_start_time	Optional field: in case it is not present inside the input file the value is set equal to an empty string.
	USEABLE_STOP_TIME	useable_stop_time	Optional field: in case it is not present inside the input file the value is set equal to an empty string.
	STOP_TIME	stop_time	
	INTERPOLATION	N/A	
	INTERPOLATION_DEGREE	N/A	
	META_STOP	N/A	
EPHEMERIS DATA LINES	Epoch	osv_rec[num_rec].tai_time osv_rec[num_rec].utc_time osv_rec[num_rec].ut1_time	***num_rec*** represents the index in the array *osv_rec* from the structure *xd_oem_file* delta(ut1 – utc) and delta(tai – utc) are equal to 0. In the OEM file, the position and velocity are expressed in kilometers. Before they are stored in *osc_rec* structure
	X	osv_rec[num_rec].pos[0]	
	Y	osv_rec[num_rec].pos[1]	
	Z	osv_rec[num_rec].pos[2]	
	X_DOT	osv_rec[num_rec].vel[0]	
	Y_DOT	osv_rec[num_rec].vel[1]	
	Z_DOT	osv_rec[num_rec].vel[2]	

			they are transformed in meters.
	X_DDOT	N/A	
	Y_DDOT	N/A	
	Z_DDOT	N/A	
COVARIANCE MATRIX LINES	COVARIANCE MATRIX LINES	N/A	

Table 175: Correspondence between OEM reference frames and EOCFI reference frames

OEM File value	CFI value
TOD	XD_TRUE_DATE
EME2000	XD_GEO_MEAN_2000
ICRF	XD_BAR_MEAN_2000
ITRF-93	XD_EARTH_FIXED
ITRF-97	
ITRF2000	
ITRFxxxx	

8.3.6. IERS Bulletins

The EOCFI SW is able to read IERS Bulletins A, B, B (IAU1980), B (IAU2000), as described in [IERS].

8.3.7. CryoSat-2 Orbit Event File

8.3.7.1. Variable Header

The Variable Header has the same format as for Orbit State Vector Files (see reference [EO_ICD]).

8.3.7.2. Data Block

The Data Block content is a sequence of XML elements described in Table 176.

Table 176: Data Block content

XML Tag name	Type	Attributes	C Format	Description
List_of_Orbit_Changes	List of <Orbit_Change> Structures (See Table 166)	count="n" where n is the number of ele- ments in the list	-	List of Orbital Changes. This list has the same format as for Orbit Scenario Files (see reference [EO_ICD].
List_of_OSVs (optional)	List of <OSV> Structures (See Table 163)	count="n" where n is the number of	-	List of Orbit State Vectors This list has the same format as for Orbit State Vector Files (see reference [EO_ICD]).

		ele- ments in the list		

Example:

```
<Data_Block type="xml">
    <List_of_Orbit_Changes count="2">
        <Orbit_Change>
            <Orbit>
                <Absolute_Orbit>1</Absolute_Orbit>
                <Relative_Orbit>25</Relative_Orbit>
                <Cycle_Number>1</Cycle_Number>
                <Phase_Number>1</Phase_Number>
            </Orbit>
            <Cycle>
                <Repeat_Cycle unit="day">2</Repeat_Cycle>
                <Cycle_Length unit="orbit">29</Cycle_Length>
                <ANX_Longitude unit="deg">130.000000</ANX_Longitude>
                <MLST>21:00:00.000000</MLST>
                <MLST_Drift unit="s/day">-179.045927</MLST_Drift>
            </Cycle>
            <Time_of_ANX>
                <TAI>TAI=2002-03-01T21:00:52.365827</TAI>
                <UTC>UTC=2002-03-01T21:01:27.365827</UTC>
                <UT1>UT1=2002-03-01T21:01:27.665827</UT1>
            </Time_of_ANX>
        </Orbit_Change>
        <Orbit_Change>
            <Orbit>
                <Absolute_Orbit>30</Absolute_Orbit>
                <Relative_Orbit>1864</Relative_Orbit>
                <Cycle_Number>2</Cycle_Number>
                <Phase_Number>1</Phase_Number>
            </Orbit>
            <Cycle>
                <Repeat_Cycle unit="day">369</Repeat_Cycle>
                <Cycle_Length unit="orbit">5344</Cycle_Length>
                <ANX_Longitude unit="deg">129.998600</ANX_Longitude>
                <MLST>20:54:02.999999</MLST>
                <MLST_Drift unit="s/day">-179.208551</MLST_Drift>
            </Cycle>
            <Time_of_ANX>
                <TAI>TAI=2002-03-03T20:46:50.497469</TAI>
                <UTC>UTC=2002-03-03T20:47:25.497469</UTC>
                <UT1>UT1=2002-03-03T20:47:25.797469</UT1>
            </Time_of_ANX>
```

```
        </Orbit_Change>
    </List_of_Orbit_Changes>
        <List_of_OSVs count="2">
        <OSV>
            <TAI>TAI=2002-03-03T08:08:41.244734</TAI>
            <UTC>UTC=2002-03-03T08:09:16.244734</UTC>
            <UT1>UT1=2002-03-03T08:09:16.544734</UT1>
            <Absolute_Orbit>+00013</Absolute_Orbit>
            <X unit="m">-6937171.769</X>
            <Y unit="m">-1483270.979</Y>
            <Z unit="m">+0000000.000</Z>
            <VX unit="m/s">-0152.952889</VX>
            <VY unit="m/s">+0761.962112</VY>
            <VZ unit="m/s">+7493.050200</VZ>
            <Quality>000000.000000</Quality>
        </OSV>
        <OSV>
            <TAI>TAI=2002-03-03T09:47:47.517429</TAI>
            <UTC>UTC=2002-03-03T09:48:22.517429</UTC>
            <UT1>UT1=2002-03-03T09:48:22.817429</UT1>
            <Absolute_Orbit>+00014</Absolute_Orbit>
            <X unit="m">-6918815.899</X>
            <Y unit="m">+1566662.540</Y>
            <Z unit="m">+0000000.000</Z>
            <VX unit="m/s">+0181.123304</VX>
            <VY unit="m/s">+0755.761334</VY>
            <VZ unit="m/s">+7493.050200</VZ>
            <Quality>000000.000000</Quality>
        </OSV>
    </List_of_OSVs>
</Data_Block>
```

8.3.7.3. Schema Reference

An example of validating XML schema for this file type is located at:

http://eop-cfi.esa.int/CFI/EE_CFI_SCHEMAS/EO_OPER_MPL_ORBREF_0106.XSD

This schema is compliant to [EO_SCH_HB] and includes format and range checks to ensure compliance to this specification and to the File Format Standard [FFS1]. The schema file is named according to section 6.1.1 in [EO_SCH_HB] and is applicable to files named MM_**OPER_MPL_ORBREF**_<instance_id>.EOF.

The following is the content of the Earth_Explorer_File required to reference the above schema.

```
<Earth_Explorer_File xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://eop-
cfi.esa.int/CFI http://eop-cfi.esa.int/CFI/EE_CFI_SCHEMAS/EO_OPER_MPL_ORBREF_0106.XSD" schemaVersion="1.6"
xmlns="http://eop-cfi.esa.int/CFI">
```

8.3.8. CryoSat-2 DORIS Navigator File

The format of the Cryosat-2 DORIS Navigator files is described in [PDS_FMT].

8.3.9. Sentinel-3 DORIS Navigator File

The format of the Sentinel-3 DORIS Navigator files is described in [PDGS_S3].

8.3.10. CryoSat-2 Star Tracker File

A Star tracker file consists in a couple of files: the CryoSat standard header file and the data block file. They are compliant with [PDS_FMT].

8.3.11. CryoSat-2 Star Tracker Configuration File

8.3.11.1. Variable Header

The Variable Header is empty for this file type.

8.3.11.2. Data Block

The Data Block content is a sequence of XML elements described in Table 177.

Table 177: Data Block content

XML Tag name	Type	Attributes	C Format	Description
Satellite_Name	string	-	%s	Satellite Name
Mispointing	Structure (See Table 178)	-	-	Set of rotation angles needed for mispointing computation

Table 178: Mispointing

XML Tag name	Type	Attributes	C Format	Description
Aberration_Correction	string	-	%s	Aberration correction flag. Possible values are: • Yes: for applying the aberration- correction. • No: for not applying the aberration correction. • Reverse: for applying the aberra- tion correction with the transposed matrix.
Star_Trackers_Limits	Structure (See Table 179)	-	-	Limits for the validity fo the quaternions
Star_Trackers_Priority	Structure (See	-	-	Star trackers priority

	Table 180)			
List_of_Star_Trackers	Structure (See Table 181)	count="n"	-	List of rotation angles from the antenna bench to the star trackers frame
Satellite_Mechanical_To_Antenna_Bench	Structure (See Table 182)	-	-	Rotation angles from the satellite mechanical to the antenna bench frame
Satellite_Control_To_Satellite_Mechanical	Structure (See Table 183)	-	-	Rotation angles from the satellite control to the satellite mechanical frame
Satellite_Attitude_To_Satellite_Control	Structure (See Table 183)	-	-	Rotation angles from the satellite control to the satellite attitude frame

Table 179: Star Tracker limits

XML Tag name	Type	Attributes	C Format	Description
Max_Penalty	integer	-	%d	Maximum penalty for the quaternions
Quaternion_Norm_Threshold	real	-	%f	Threshold for the modulus of the quaternion
Max_Time_Gap	real	unit="s"	%f	Maximum time gap between two consequtive quaternions

Table 180: Star_Tracker_Priority

XML Tag name	Type	Attributes	C Format	Description
File_Type_1	string	-	%s	
File_Type_2	string	-	%s	
File_Type_3	string	-	%s	

Table 181: List_of_Star_Trackers

XML Tag name	Type	Attributes	C Format	Description
Star_Tracker	Structure (See Table 190)	-	-	Antenna bench to Star tracker rotation angles

Table 182: Launch angles

XML Tag name	Type	Attributes	C Format	Description
Pre_Launch_Angles	Structure (See Table 191)	-	-	pre-launch angles
Post_Launch_Misalign	Structure	-	-	post-launch angles

ment	(See Table 191)			

Table 183: Rotation_Angles

XML Tag name	Type	Attributes	C Format	Description
X_Rotation	real	unit="deg"	%f	Rotation around the X-axis
Y_Rotation	real	unit="deg"	%f	Rotation around the Y-axis
Z_Rotation	real	unit="deg"	%f	Rotation around the Z-axis

Example:

```
<Data_Block type="xml">
    <Satellite_Name>CryoSat</Satellite_Name>
    <Mispointing>
        <Aberration_Correction>Yes</Aberration_Correction>
        <Star_Trackers_Limits>
            <Max_Penalty>5</Max_Penalty>
            <Quaternion_Norm_Threshold>0.000001</Quaternion_Norm_Threshold>
            <Max_Time_Gap unit="s">600</Max_Time_Gap>
        </Star_Trackers_Limits>
        <Star_Trackers_Priority>
            <File_Type_1>STR1ATT_0_</File_Type_1>
            <File_Type_2>STR2ATT_0_</File_Type_2>
            <File_Type_3>STR3ATT_0_</File_Type_3>
        </Star_Trackers_Priority>
        <!-- Antenna Bench To Star Tracker rotation angles -->
        <List_of_Star_Trackers count="3">
            <Star_Tracker>
                <Pre_Launch_Angles>
                    <X_Rotation unit="deg">0.000</X_Rotation>
                    <Y_Rotation unit="deg">0.000</Y_Rotation>
                    <Z_Rotation unit="deg">0.000</Z_Rotation>
                </Pre_Launch_Angles>
                <Post_Launch_Misalignment>
                    <X_Rotation unit="deg">0.000</X_Rotation>
                    <Y_Rotation unit="deg">0.000</Y_Rotation>
                    <Z_Rotation unit="deg">0.000</Z_Rotation>
                </Post_Launch_Misalignment>
            </Star_Tracker>
            <Star_Tracker>
                <Pre_Launch_Angles>
                    <X_Rotation unit="deg">65.000</X_Rotation>
                    <Y_Rotation unit="deg">0.000</Y_Rotation>
```

```
                <Z_Rotation unit="deg">0.000</Z_Rotation>
            </Pre_Launch_Angles>
            <Post_Launch_Misalignment>
                <X_Rotation unit="deg">0.000</X_Rotation>
                <Y_Rotation unit="deg">0.000</Y_Rotation>
                <Z_Rotation unit="deg">0.000</Z_Rotation>
            </Post_Launch_Misalignment>
        </Star_Tracker>
        <Star_Tracker>
            <Pre_Launch_Angles>
                <X_Rotation unit="deg">295.000</X_Rotation>
                <Y_Rotation unit="deg">0.000</Y_Rotation>
                <Z_Rotation unit="deg">0.000</Z_Rotation>
            </Pre_Launch_Angles>
            <Post_Launch_Misalignment>
                <X_Rotation unit="deg">0.000</X_Rotation>
                <Y_Rotation unit="deg">0.000</Y_Rotation>
                <Z_Rotation unit="deg">0.000</Z_Rotation>
            </Post_Launch_Misalignment>
        </Star_Tracker>
    </List_of_Star_Trackers>
    <!-- End Antenna Bench To Star Tracker rotation angles -->
    <Satellite_Mechanical_To_Antenna_Bench>
        <Pre_Launch_Angles>
            <X_Rotation unit="deg">0.000</X_Rotation>
            <Y_Rotation unit="deg">354.000</Y_Rotation>
            <Z_Rotation unit="deg">0.000</Z_Rotation>
        </Pre_Launch_Angles>
        <Post_Launch_Misalignment>
            <X_Rotation unit="deg">0.000</X_Rotation>
            <Y_Rotation unit="deg">0.000</Y_Rotation>
            <Z_Rotation unit="deg">0.000</Z_Rotation>
        </Post_Launch_Misalignment>
    </Satellite_Mechanical_To_Antenna_Bench>
    <Satellite_Control_To_Satellite_Mechanical>
        <X_Rotation unit="deg">0.000</X_Rotation>
        <Y_Rotation unit="deg">6.000</Y_Rotation>
        <Z_Rotation unit="deg">0.000</Z_Rotation>
    </Satellite_Control_To_Satellite_Mechanical>
    <Satellite_Attitude_To_Satellite_Control>
        <X_Rotation unit="deg">0.000</X_Rotation>
        <Y_Rotation unit="deg">0.000</Y_Rotation>
        <Z_Rotation unit="deg">270.000</Z_Rotation>
    </Satellite_Attitude_To_Satellite_Control>
</Mispointing>
[...]
```

```
</Data_Block>
```

8.3.11.3. Schema Reference

An example of validating XML schema for this file type is located at:

http://eop-cfi.esa.int/CFI/EE_CFI_SCHEMAS/EO_OPER_INT_STRCFG_0102.XSD

This schema is compliant to [EO_SCH_HB] and includes format and range checks to ensure compliance to this specification and to the File Format Standard [FFS1]. The schema file is named according to section 6.1.1 in [EO_SCH_HB] and is applicable to files named MM_**OPER_MPL_STRCFG**_<instance_id>.EOF.

The following is the content of the Earth_Explorer_File required to reference the above schema.

```
<Earth_Explorer_File xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://eop-cfi.esa.int/CFI http://eop-cfi.esa.int/CFI/EE_CFI_SCHEMAS/EO_OPER_INT_STRCFG_0102.XSD" schemaVersion="1.2" xmlns="http://eop-cfi.esa.int/CFI">
```

Attitude Ephemeris Message File (AEM)

The format of the AEM files is described in [AEM].

Table 184 shows the mapping between the AEM file and the CFI structure xd_aem_file. Some fields are **not** read by the function **xd_read_aem** are marked with *N/A* (xd_aem_file field column). Some fields are read by the reading function but only for the user own purposes and they are not used by any other CFI functions, in this case it is marked in the “Notes” column as *not needed by EOCFI*

Table 184: List of AEM fields read by EOCFI

AEM File Section	AEM File Field	xd_aem_file field	Notes
AEM Header	CCSDS_AEM_VERS	ccsds_aem_vers	
	COMMENT	comment_header	
	CREATION_DATE	creation_date	
	ORIGINATOR	originator	
AEM Metadata	META_START	N/A	It marks the start of the ephemeris metadata for a given segment. In the following descriptions for the AEM Metada, “*i*” represents the index of the segment within the AEM file.
	COMMENT	segment[i].metadata.comment_metadata	
	OBJECT_NAME	segment[i].metadata.object_name	
	OBJECT_ID	segment[i].metadata.object_id	
	CENTER_NAME	segment[i].metadata.center_name	Not needed by EOCFI

	REF_FRAME_A	segment[i].metadata.ref_frame_A	The table 185 shows the mapping between OEM reference frames supported by the EOCFI and the EOCFI refrence frames.
	REF_FRAME_B	segment[i].metadata.ref_frame_B	The table 185 shows the mapping between OEM reference frames supported by the EOCFI and the EOCFI refrence frames.
	ATTITUDE_DIR	segment[i].metadata.attitude_dir	
	TIME_SYSTEM	segment[i].metadata.time_system	Only the following time systems are supported by CFI: • UTC • TAI • GPS • UT1 Other possibles values can be read but cannot be hadled by the EOCFI
	START_TIME	segment[i].metadata.start_time	Not needed by EOCFI
	USEABLE_START_TIME	segment[i].metadata.useable_start_time	Not needed by EOCFI
	USEABLE_STOP_TIME	segment[i].metadata.useable_stop_time	Not needed by EOCFI
	STOP_TIME	segment[i].metadata.stop_time	Not needed by EOCFI
	ATTITUDE_TYPE	segment[i].metadata.attitude_type	Possible values when reading: • QUATERNION • QUATERNION/DERIVATIVE • QUATERNION/RATE • EULER_ANGLE • EULER_ANGLE/RATE • SPIN • SPIN/NUTATION

			The EOCFI only supports the attitude data given as Euler angles or Quaternions (their derivative/rates are not used). “SPIN” data cannot b used in the EOCFI data.
	QUATERNION_TYPE	segment[i].metadata.quaternion_type	FIRST: The correspondece between the AEM quaternions and the EOCFI convention (see [MCD]) is the following: EOCFI AEM q1 q2 q2 q3 q3 q4 q4 q1 LAST: In this case it is assumed that quaternions are given in the same order as in the EOCFI convention (see [MCD])
	EULER_ROT_SEQ	segment[i].metadata.euler_rot_seq	EOCFI functions only support the sequence “213”
	RATE_FRAME	segment[i].metadata.rate_frame	Not needed by EOCFI
	INTERPOLATION_METHOD	segment[i].metadata.interpolation	Not needed by EOCFI
	INTERPOLATION_DEGREE	segment[i].metadata.interpolation_degree	Not needed by EOCFI
	META_STOP	N/A	It marks the stop of the ephemeris metadata for a given segment
EPHEMERIS DATA LINES	DATA_START	N/A	It marks the start of the ephemeris data for a given segment. In the following descriptions for the AEM Ephemeris, “*i*” represents the index of the segment within the AEM file and “*j*” the index of the of the

				array aem_rec in xd_aem_attitude.
	EPOCH		segment[i].attitude.att_rec.[j].time	
	ATTITUDE_TYPE = QUATERNION	QC, Q1, Q2, Q3	segment[i].attitude.att_rec.[j].data[0-3]	Data is stored in the same order as it is in the AEM file
	ATTITUDE_TYPE = QUATERNION/ DERIVATIVE	QC, Q1, Q2, Q3, QC_DOT, Q1_DOT, Q2_DOT, Q3_DOT	segment[i].attitude.att_rec.[j].data[0-7]	Data is stored in the same order as it is in the AEM file. Only quaternons data (segment[i].attitude.att_rec.[j].data[0-3]) is used
	ATTITUDE_TYPE = QUATERNION/ DERIVATIVE	QC, Q1, Q2, Q3, rotation1, rotation2, rotation 3	segment[i].attitude.att_rec.[j].data[0-6]	Data is stored in the same order as it is in the AEM file. Only quaternions data (segment[i].attitude.att_rec.[j].data[0-3]) are used in other EOCFI functions
	ATTITUDE_TYPE =EULER_ANGLE	rotation1, rotation2, rotation 3	segment[i].attitude.att_rec.[j].data[0-2]	
	ATTITUDE_TYPE =EULER_ANGLE/RATE	rotation1, rotation2, rotation 3, rotation1(rate), rotation2(rate), rotation 3 (rate)	segment[i].attitude.att_rec.[j].data[0-5]	Only angles data (segment[i].attitude.att_rec.[j].data[0-2]) are used in other EOCFI functions
	ATTITUDE_TYPE = SPIN	SPIN_ALPHA, SPIN_DELTA, SPIN_ANGLE, SPIN_ANGLE_VEL	segment[i].attitude.att_rec.[j].data[0-3]	Not used in EOCFI
	ATTITUDE_TYPE = SPIN/NUTATION	SPIN_ALPHA, SPIN_DELTA, SPIN_ANGLE, SPIN_ANGLE_VEL, NUTATION, NUTATION_PER, NUTATION_PHASE	segment[i].attitude.att_rec.[j].data[0-6]	Not used in EOCFI
	DATA_STOP		N/A	It marks the stop of the ephemeris data for a given segment

Table 185: Correspondence between AEM reference frames and EOCFI reference frames

OEM File value	CFI value
TOD	XD_TRUE_DATE
EME2000 GCRF	XD_GEO_MEAN_2000
ICRF	XD_BAR_MEAN_2000
ITRF-93	XD_EARTH_FIXED
ITRF-97	
ITRF2000	
ITRFxxxx	
ACTUATOR_xxx CSS_xxx DSS_xxx GYRO_xxx INSTRUMENT_xxx SC_BODY_xxx SENSOR_ xxx STARTRACKER_xxx TAM_xxx SATELLITE_RS	XD_SAT_REL_REF

NOTE: GCRF and SATELLITE_RS are custom referance frames for Sentinel 4.

9. RUNTIME PERFORMANCES

The library performance has been measured by dedicated test procedures run in 5 different platforms under the below specified machines:

OS ID	*Processor*	*OS*	*RAM*
LINUX64	Intel(R) Xeon(R) CPU E5-2609 v4 @ 1.70GHz (8 cores)	GNU LINUX 4.10.0-42-generic (Ubuntu 17.04)	64 GB
LINUX64_LEGACY	Intel(R) Xeon(R) CPU E5-2470 0 @ 2.30GHz (16 cores)	GNU LINUX 2.6.24-16-generic (Ubuntu 10.10)	16 GB
MACIN64	Intel Core i7 4 cores @2,6 GHz	MACOSX 10.12	16 GB
MACARM64	Apple M2 Max 12 cores (8 performance and 4 efficiency)	macOS 13.5.2	64 GB
WINDOWS64	Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz 3.19 GHz	Microsoft Windows 10 (or superior)	32 GB

The table below shows the time (in milliseconds - ms) each function takes to be run under each platform:

Function ID	WINDOWS64	LINUX64	LINUX64_LEGACY	MACIN64	MACARM64
xd_read_bulletin	1.64	1.6	2.1	0.9	0.7
xd_read_orbit_file * File with 864001 OSVs. Select whole file	2518	3410	3270	3050	980
xd_read_orbit_file * File with 864001 OSVs. Load a time range of 60 second	1367	2130	1510	2200	780
xd_read_orbit_file * File with 864001 OSVs. Load a time range covering the whole file	2946	3790	2680	3320	1080
xd_read_orbit_file * File with 864001 OSVs. Load one orbit	1295	2150	1490	2170	780
xd_read_orbit_file * File with 864001 OSVs. Load an orbit range covering the whole file	2607	3530	2540	3110	1030

xd_read_orbit_file * 3 OSVs	0.42	0.2	0.2	0.2	0
xd_read_fhr	0.34	0.2	0.1	0.1	0.1
xd_write_orbit_file * 3 OSVs written	0.9	0.4	0.3	0.3	0.1
xd_read_doris_header	0.17	0	0.1	0.1	0
xd_read_doris * 1171 elements	1.22	1	0.8	1.3	0.6
xd_write_doris * 1171 records written	7.1	4	1	5	1
xd_read_osf * 5 orbit changes	0.62	0.6	0.4	0.5	0.1
xd_write_osf * 5 orbit changes	1.18	0.5	0.4	0.5	0.2
xd_read_star_tracker_conf_file * 2000 records read	6.36	13.5	9.6	13.2	4.7
xd_read_star_tracker	3.7	3.8	2.7	5.7	2.3
xd_read_att * 5 Quaternions	0.48	0.1	0.1	0.1	0.1
xd_write_att * 5 Quaternions	0.825	0.31	0.24	0.31	0.06
xd_read_precise_propag_file	0.136	0.02	0.02	0.01	0.01
xd_free_dem_config_file	0.31	0.09	0.18	0.08	0.03
xd_read_dem	140	129	108	173	75
xd_read_sdf	0.94	0.3	0.2	0.3	0.1
xd_read_stf_vhr * 1200 records read	34.369999	85.4	58.8	90.2	29.2
xd_read_stf	48.009998	118.2	81.9	123.2	39.5
xd_write_stf * 1200 records written	36.400002	57	41	65	17
xd_read_zone	2.82	6.5	4.4	6.6	2.1
xd_read_zone_file * 41 zones, 888 Polygon_Pts	3.4	7.3	5	7.2	2.3
xd_read_zone_ids * 41 records read	3	6.4	4.4	6.5	2.2
xd_read_station	4.53	10.5	7.2	10.8	3.5
xd_read_station_file * 124 records read	6.6	12	8	12	4
xd_read_station_id * 124 records read	4.49	10.7	7.2	10.9	3.6
xd_read_star	0.63	0.5	0.7	0.9	0.4
xd_read_star_file * 1006 stars	42.900002	44	52	73	32
xd_read_star_id * 1006 stars	61.900002	60	31	64	27

xd_xml_validate	4.24	5	3.4	4	1.6
xd_xslt_add	1.84	0.8	0.6	0.6	0.2
xd_read_oem	114.459999	120.8	90.6	90.4	31.2
xd_orbit_file_diagnostics	4.7	7.4	5	6.6	2.2
xd_read_fov_constraints_file	0.4	0.5	0.5	0	0

Note that when the value “0.000000” is defined for a function in a certain platform, it means that its running time is lower than 1 nanosecond and so it can be considered as “0”.

10. LIBRARY PRECAUTIONS

The following precaution shall be taking into account when using EO_DATA_HANDLING library:

- None
-