Earth Observation
Mission CFI Software

EO_ORBIT
SOFTWARE USER MANUAL

Code:	EO-MA-DMS-GS-0004
Issue:	4.25
Date:	10/05/2023

	Name	Function	Signature
Prepared by:	EOCFI Development Team	Project Engineers	
Checked by:	Inês Estrela	Project Manager	
Approved by:	Inês Estrela	Project Manager	

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

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/01/026-011
Electronic file name:	eo-ma-dms-gs-004-21

DOCUMENT STATUS LOG

Issue	Change Description	Date	Approval
1.0	New document	08/11/01	
1.1	New xo_orbit_to_time, xo_time_to_orbit and _free_osf_records functions. xo_cart_extra removal	23/05/02	
1.2 Draft	Cosmetic changes Updated error handling	19/07/02	
2.0	Maintenance release.	29/11/02	
2.1	Maintenance release.	13/05/03	
2.2	New options for xo_propag_init_file and _interpol_init_file functions. Maintenance release.	30/09/03	
2.2.2	Option to use a simplified algorithm to initialise using _propag_init_def Absolute orbit and time since ANX calculated within _propag_extra and xo_interpol_extra functions. Nodal period calculated by _orbit_info_from_<source> functions Use of enumerations to size extra results arrays	26/04/04	
3.0	New initialisation strategy for orbit calculations, pagation and interpolation. New interfaces	21/07/04	
3.1	Maintenance release	13/10/04	
3.2	Maintenance release	15/11/04	
3.3	Maintenance release New features: Changes for dealing with the new library xplorer_data_handling Identifier accessors. Support for ENVISAT ASCII files removed	11/07/05	
3.4	Maintenance release New features: Orbit file generation functions moved to this ɒrary	18/11/05	

3.5	Maintenance release New features: time-orbit conversion executable Support for SWARM and EARTHCARE	26/05/06	
3.6	Maintenance release New features: xo_gen_oef xo_check_osf and xo_check_oef	24/11/06	
3.7	Maintenance release New features: Function expcfi_check_libs Library version for MAC OS X on Intel (32 nd 64-bits)	13/07/06	
3.7.2	Maintenance release New features: TLE data for orbit operations and ropagation New executable: gen_oef	31/07/08	
4.0	Maintenance release New features: Numerical propagator New interfaces for propagation/interpolation	19/01/09	
4.1	Maintenance release New features: me correlation compatibility check between time_id d orbit file data	07/05/10	
4.2	Maintenance release New features: upport for curved MLST in Orbit Scenario files	31/01/2011	
4.3	Maintenance release New features: ew initialization function for the OrbitID: _orbit_id_init	06/02/12	
4.4	intenance release New features: - Support for GEO orbits (including new functions _orbit_init_geo, xo_orbit_(get/set)_geo_orbit_info).)_position_on_orbit_to_time		

4.5	Maintenance release New features: - New function xo_orbit_data_filter - Acceleration vector is computed for TLE and neric propagators		
4.6	Maintenance release New features: - SDP4 TLE propagator - Executable to generate TLE files - Fitting method to compute TLE in xo_osv_to_tle ction.		
4.7	• Maintenance release. • New features: ◦ Spacecraft Midnight computed by xo_orbit_info. ◦ New function xo_orbit_id_change, xo_orbit_info_configure ◦ Support for SENTINEL-5P, Metop-SG, and Jason-CS satellites.	28/03/14	
4.8	• Maintenance release.	29/10/2014	
4.9	• Maintenance release. • New features: ◦ Support for Orbit Ephemeris Message files	23/04/2015	
4.10	• Maintenance release	29/10/2015	
4.11	• Maintenance release	15/04/2016	
4.12	• Maintenance release • New features: ◦ Function xo_osv_check	03/11/2016	

4.13	• Maintenance release	05/04/2017	
4.14	• Maintenance release	16/11/2017	
4.15	• Maintenance release	20/04/2018	
4.16	• Maintenance release • New features: • TLE support for Sentinel-5P, Sentinel-3B, Aeolus	09/11/2018	
4.17	• Maintenance release	10/05/2019	
4.18	• Maintenance release	08/11/2019	
4.19	• Maintenance release	29/05/2020	

4.20	• Maintenance release
	• New features:
	  • Added support for LEO satellites with SP3 files
	  • Added support for new missions: Sentinel 6, CIMR, ROSEL, CHIME, CRISTAL, CO2M, LSTM, FORUM
		30/11/2020	

4.21	• Maintenance release	23/06/2021	
4.22	• Maintenance release	22/12/2021	
4.23	• Maintenance release • New feature: - Added support for new mission: TRUTHS	23/06/2022	
4.24	• Maintenance release	23/06/2022	
4.25	• Maintenance release New features: - Orbit initialization with new Orbit Scenario files with ANX longitude drift parameters	01/05/2023	

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 SCOPE

The EO_ORBIT Software User Manual provides a detailed description of usage of the CFI functions included within the EO_ORBIT CFI software library.

2 ACRONYMS, NOMENCLATURE AND TERMINOLOGY

2.1 Acronyms

ANX	Ascending Node Crossing
AOCS	Attitude and Orbit Control Subsystem
CFI	Customer Furnished Item
EF	Earth Fixed reference frame
EOCFI	Earth Observation CFI
ESA	European Space Agency
ESTEC	European Space Technology and Research Centre
FOS	Flight Operations Segment
GS	Ground Station
OBT	On-board Binary Time
OEF	Orbit Event File
OSF	Orbit Scenario File
OSV	Orbit State Vector
POF	Predicted Orbit File
ROF	Restituted Orbit File
SSP	Sub-Satellite Point
SRAR	Satellite Relative Actual Reference
SUM	Software User Manual
TLE	Two Line Elements
TOD	True of Date reference frame
UTC	Universal Time Coordinated
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. Conventions Document. EO-MA-DMS-GS-0001.

[MSC] Earth Observation Mission CFI Software. Mission Specific Customizations Document. EO-MA- DMS-GS-0018.

[GEN_SUM] Earth Observation Mission CFI Software. General Software User Manual. EO-MA- DMS-GS-0002.

[F_H_SUM] Earth Observation Mission CFI Software. EO_FILE_HANDLING Software User Manual. EO-MA-DMS-GS-0008.

[D_H_SUM] Earth Observation Mission CFI Software. EO_DATA_HANDLING Software User Manual. EO-MA-DMS-GS-007.

[LIB_SUM] Earth Observation Mission CFI Software. EO_LIB Software User Manual. EO-MA-DMS-GS-003.

[FORMATS] Earth Explorer File Format Guidelines. CS-TN-ESA-GS-0148.

[EO_OPS] Earth Observation OPS Commanding.
Link to Technical note

The latest applicable version of [MCD], [MSC], [GEN_SUM], [F_H_SUM], [D_H_SUM], [LIB_SUM] is v4.25 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:

- CFI functions allowing accurate computation of orbit state vectors, either at ascending node or (by propagation) at any point in the orbit of any Earth Observation satellite.
- The orbit propagation may be performed based on different propagation models. The initial set of models supported are:
  - Mean Keplerian model
  - TLE model
  - Numerical model
- It includes an interpolator and orbit propagators.
- CFI functions required to compute the orbit scenario file, used for Earth Observation mission planning purposes, and several orbit files useful for testing purposes (Predicted Orbit File, Restituted Orbit File, DORIS Navigator Files).
- It contains:
  - a library of functions which can be called from a main executable program
  - a set of executable programs (1 for each function) with the exact same functionality as the functions

The following sections summarize the set of functions in this library:

4.1.1 Orbit Initialisation

Before doing any orbit calculation, the orbit should be initialized using one of the following functions:

- ***xo_orbit_init_def***: this software generates a cartesian state vector around the true ascending node crossings as a function of the date (processing time), the longitude of the ascending node, the satellite Repeat Cycle Length, the mean local solar time and either the drift in mean local solar time or the inclination.
- ***xo_orbit_cart_init, xo_orbit_cart_init_precise***: This software initializes the orbit using as input a cartesian orbit state vector. The “precise” function allows the introduction of data to propagate a state vector with a numeric propagator (see 4.1.2). Numerical propagator uses external files for the configuration of gravity model and F10.7 coefficient and Geomagnetic Activity index values. You can find some files that can be used in *files/models* directory of the Earth Observation CFI package, and following you can find some references for them:
  - Gravity model egm96:
  - http://cddisa.gsfc.nasa.gov/926/egm96/egm96.html
  - F10.7 coefficient and Geomagnetic Activity index: ECSS-E-10-04A Space Environment
  - (http://www.spacelab.dti.supsi.ch/Tecnica/ECSS-E-10-04ASpaceEnvironment1.pdf).
- ***xo_orbit_init_file, xo_orbit_init_file_precise***: This software initializes the orbit using a set of files containing the orbital information (state vectors, orbital geometry or TLE data). The “precise”

function allows the introduction of data to propagate an state vector with a numeric propagator (see 4.1.2). The following input file types are accepted:

- Flight Dynamics predicted ascending node state vectors.
- DORIS Navigator Data
- FOS Restituted Orbit Files
- DORIS Preliminary Orbit
- DORIS Precise Orbit
- Ascending node state vectors from the Orbit Scenario File
- TLE files (not for precise propagator)
- SP3 files (not for precise propagator)

- ***xo_orbit_id_init***: This software initializes the orbit using a data structure that contain a set of data read from files containing the orbital information (from Orbit files, DORIS navigator, Orbit Scenario files or SP3 files).
- ***xo_orbit_init_geo:*** This software initializes the orbit for a geostationary orbit using as input fixed longitude coordinates of the satellite.

In all cases a variable of the type xo_orbit_id (*Orbit ID.*) is returned. This variable is a CFI Identifier of the type described in [GEN_SUM]. This variable keeps internally the orbit information that will be used for further calculations. That orbit information can be retrieved by calling the following CFI functions:

CFI Function[1]	Orbit ID data	Condition to get the data
xo_orbit_init_status	Orbit ID initialisation status	Always
xo_orbit_get_sat_id	Satellite ID	The Orbit ID is initialised.
xo_orbit_get_mode	Mode used for the Orbit ID initialisation	The Orbit ID is initialised.
xo_orbit_get_osv	OSV stored in the Orbit ID	The Orbit ID has been initialised with state vectors.
xo_orbit_get_anx	ANX data stored in the Orbit ID	The Orbit ID has been initialised with state vectors that are not located at the ANX (Restituted orbit files, DORIS Navigator files...)
xo_orbit_get_osf_rec	Orbital Geometry data stored in the Orbit ID	The Orbit ID has been initialised with orbit geometry data.
xo_orbit_get_val_time	Validity time interval where the Orbit ID can be used except for **xo_osv_compute** and **xo_osv_compute_extra**	The Orbit ID is initialised.
xo_orbit_get_precise_propag_config	Configuration for the precise propagator	The Orbit ID has been initialised with **xo_orbit_init_file_precise** or

1 These functions are defined in the current SUM (section 7) or in [GEN_SUM].

		xo_orbit_cart_init_precise
xo_orbit_get_time_id	Time ID used for the Orbit ID initialisation	The Orbit ID is initialised.
xo_orbit_get_model_id	Model ID used for the Orbit ID initialisation	The Orbit ID is initialised.
xo_orbit_get_osv_compute_validity	Validity time interval where the Orbit ID can be used to call **xo_osv_compute** and **xo_osv_compute_extra**	The Orbit ID is initialised.
xo_orbit_get_propag_mode	Propagation model used when calling **xo_osv_compute**	Orbit Id is configured for propagation (see section 4.1.2)
xo_orbit_get_interpol_mode	Interpolation model used when calling **xo_osv_compute**	Orbit Id is configured for interpolation (see section 4.1.2)
xo_orbit_get_propag_config	Configuration data for the propagator	Orbit Id is configured for propagation (see section 4.1.2)
xo_orbit_get_interpol_config	Configuration data for the interpolator	Orbit Id is configured for interpolation (see section 4.1.2)

Finally, note that it is possible to create a copy of *Orbit ID* with **xo_orbit_id_clone** .

4.1.2 State Vector Computation (Propagation/Interpolation)

The software provides a set of functions to compute orbit state vectors at a given time:

- ***xo_osv_compute:*** This software computes the state vector at the requested time. The method used to compute that vector is transparent for the user and depends on the data type used for the orbit initialisation. Propagation is performed when the orbit_id is initialised with:

– One Orbit State Vector (xo_orbit_cart_init)

– Orbit Geometry (xo_orbit_init_def)

– Orbit Scenario File

– Predicted orbit file (plus an optional DORIS Navigator file)

– Orbit Event Files (Note: Orbit Event File is deprecated, only supported for CRYOSAT mission).

– TLE files

Interpolation is used in these other cases:

– DORIS Navigator Data

– FOS Restituted Orbit Files

– DORIS Preliminary Orbit

– DORIS Precise Orbit

– SP3 files

4.1.3 Ancillary Results Computation

xo_osv_compute_extra: This software returns ancillary results, i.e. mean and osculating Keplerian orbit state vectors, satellite osculating true latitude, latitude rate and latitude rate-rate, Sun zenith angle and many more.

4.1.4 Time/Orbit Transformation

xo_time_to_orbit: This software calculates the absolute orbit, number of seconds and number of microseconds since ascending node that corresponds to a given time in processing format.

xo_orbit_to_time: This software calculates the time, in processing format, that corresponds to a given absolute orbit, number of seconds and number of microseconds since ascending node.

4.1.5 Orbit Information Parameters

xo_orbit_rel_from_abs: This software calculates the relative orbit, the phase number giving as input an absolute orbit number.

xo_orbit_abs_from_rel: This software calculates the absolute orbit number giving as input a relative orbit number and its cycle number.

xo_orbit_abs_from_phase: This software calculates the absolute orbit number, the relative orbit, the phase number giving as input a phase number.

xo_orbit_info: This software calculates orbit related parameters providing as input the absolute orbit number.

4.1.6 File Generation

xo_gen_osf_create/xo_gen_osf_create_2: generates the orbit scenario file with user provided inputs

xo_gen_osf_append_orbit_change/xo_gen_osf_append_orbit_change_2: adds an orbit change to a previously generated OSF

xo_gen_osf_change_repeat_cycle/xo_gen_osf_change_repeat_cycle_2: adds an orbit change for a given target orbit to an existing OSF.

xo_gen_osf_add_drift_cycle: adds an orbit change for a requested orbit with a particular ascending node longitude and an orbit for the manoeuvre.

xo_gen_pof: generates a Predicted Orbit File from several different reference input files.

xo_gen_rof and ***xo_gen_rof_prototype***: generates a Restituted Orbit File from several different reference input files.

xo_gen_oef generates an orbit event file from an orbit scenario file and a predicted orbit file. Note: Orbit Event File is deprecated, only supported for CRYOSAT mission

xo_gen_dnf: generates a DORIS Navigator File from several different reference input files.

xo_gen_tle: generates a TLE file from a Predicted or Restituted Orbit file.

4.1.7 Clean-up Memory

xo_orbit_close: This software frees the memory allocated by the orbit initialization routines. It closes the xo_orbit_id, so that it cannot be used for further computations.

4.1.8 Check Orbit files

xo_check_osf: checks the continuity between the last orbit of an orbital change and the next orbit in an orbit scenario file.

xo_check_oef: checks the consistency between the list of the orbital changes and the list of orbit state vectors in an orbit event file. Note: Orbit Event File is deprecated, only supported for CRYOSAT mission

4.2 State Vector Computation Calling Sequence (Propagation/ Interpolation)

A complete propagation sequence consists of:

A call to any of the initialization routines for orbit, *xo_orbit_init_def*, *xo_orbit_init_def_2*, *xo_orbit_init_file[_precise]* or *xo_orbit_cart_init[_precise]*, *xo_orbit_id_init*, *xo_orbit_init_geo* to generate the internal data necessary for whatever calculation involving orbits.

An optional call to *xo_osv_compute_extra* to calculate any desired ancillary result related to the initializing state vector.

A call to the *xo_osv_compute* function to compute the orbit state vector at a requested time (Optionally, the user can check if the requested time is within the validity interval by calling the function *xo_orbit_get_osv_compute_validity*).

To obtain some ancillary results associated to the computed OSV, the user might call the *xo_osv_compute_extra* function.

At the end of a sequence is mandatory to call *xo_orbit_close* to free the memory allocated.

The possible propagation sequences of calls allowing to produce an orbit state vector are shown in Figure 1.

4.3 Time/Orbit Transformation and Orbit Information Parameters Calling Sequence

A complete time/orbit transformation and orbit information parameters sequence consists of:

A call to any of the initialization routines for orbit, *xo_orbit_init_def*, *xo_orbit_init_file[_precise]* or *xo_orbit_cart_init[_precise]*, *xo_orbit_id_init*, to generate the internal data necessary for whatever calculation involving orbits. Note that time to orbit transformations cannot be computed if the orbit was initialised with *xo_orbit_cart_init*.

A call to a time/orbit transformation or an orbit information parameters routine.

When no more *time/orbit transformations* and *orbit information parameters* routines are going to be used, call to *xo_orbit_close* to free the memory allocated.

The possible time/orbit transformation and orbit information parameters sequences of calls allowing to produce an orbit state vector are shown in Figure 1.

A detailed description of each function is provided in section 7. Please refer also to:

[MCD]for a detailed description of the time references and formats, reference frames, parameters and models used in this document.

[GEN_SUM] for a complete overview of the CFI, and in particular the detailed description of the *Id* concept and the error handling functions.

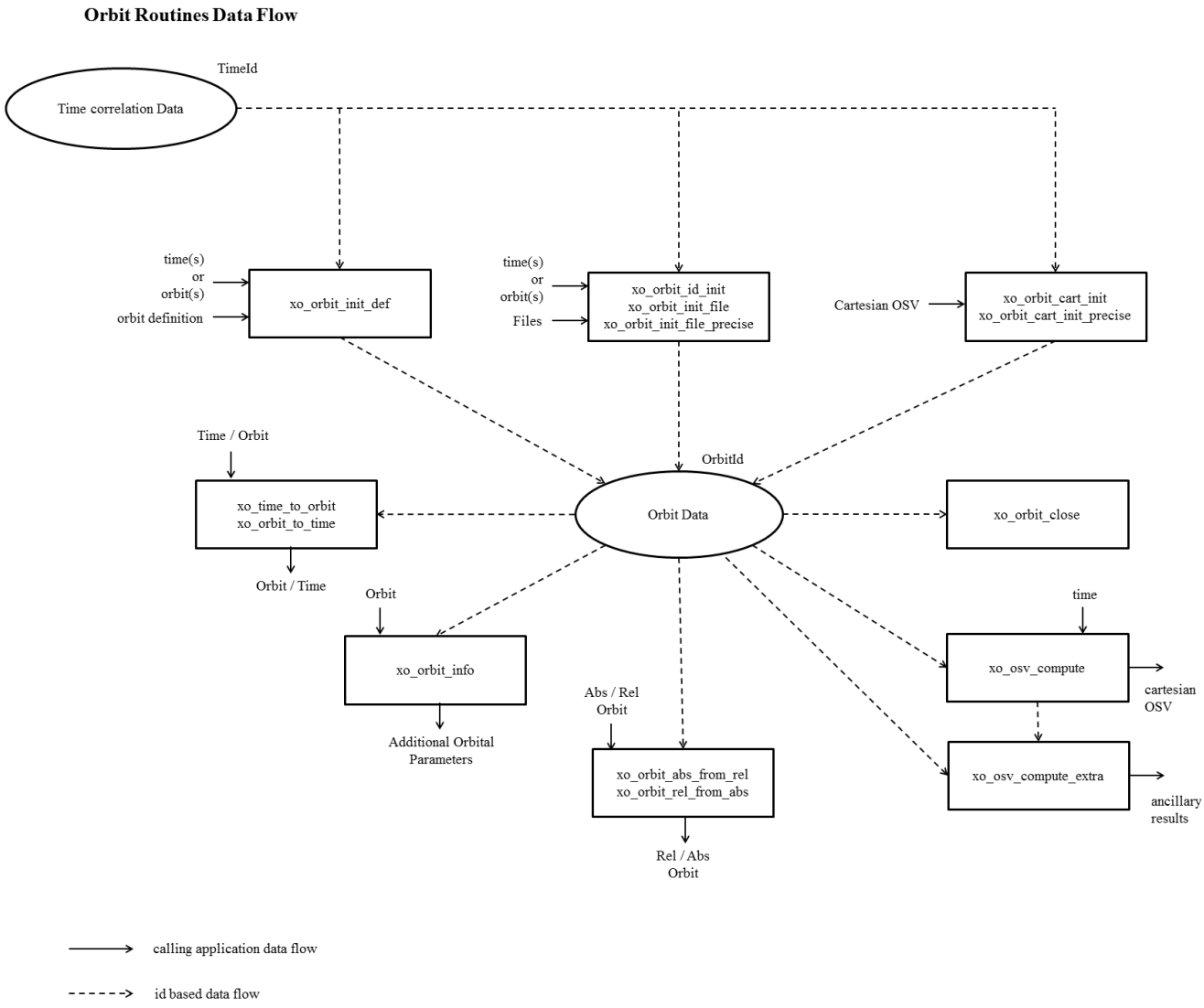


Figure 1: Orbit Calling Sequence

4.4 File Generation Calling Sequence

The calling sequence for the file generators consists of:

One call to a time initialization routine

One call to the generation routine providing the input parameters. For xo_gen_pof, xo_gen_rof, xo_gen_oef and xo_gen_dnf a reference orbit file has to be provided as well.

The following figure shows a schema of the calling sequence:

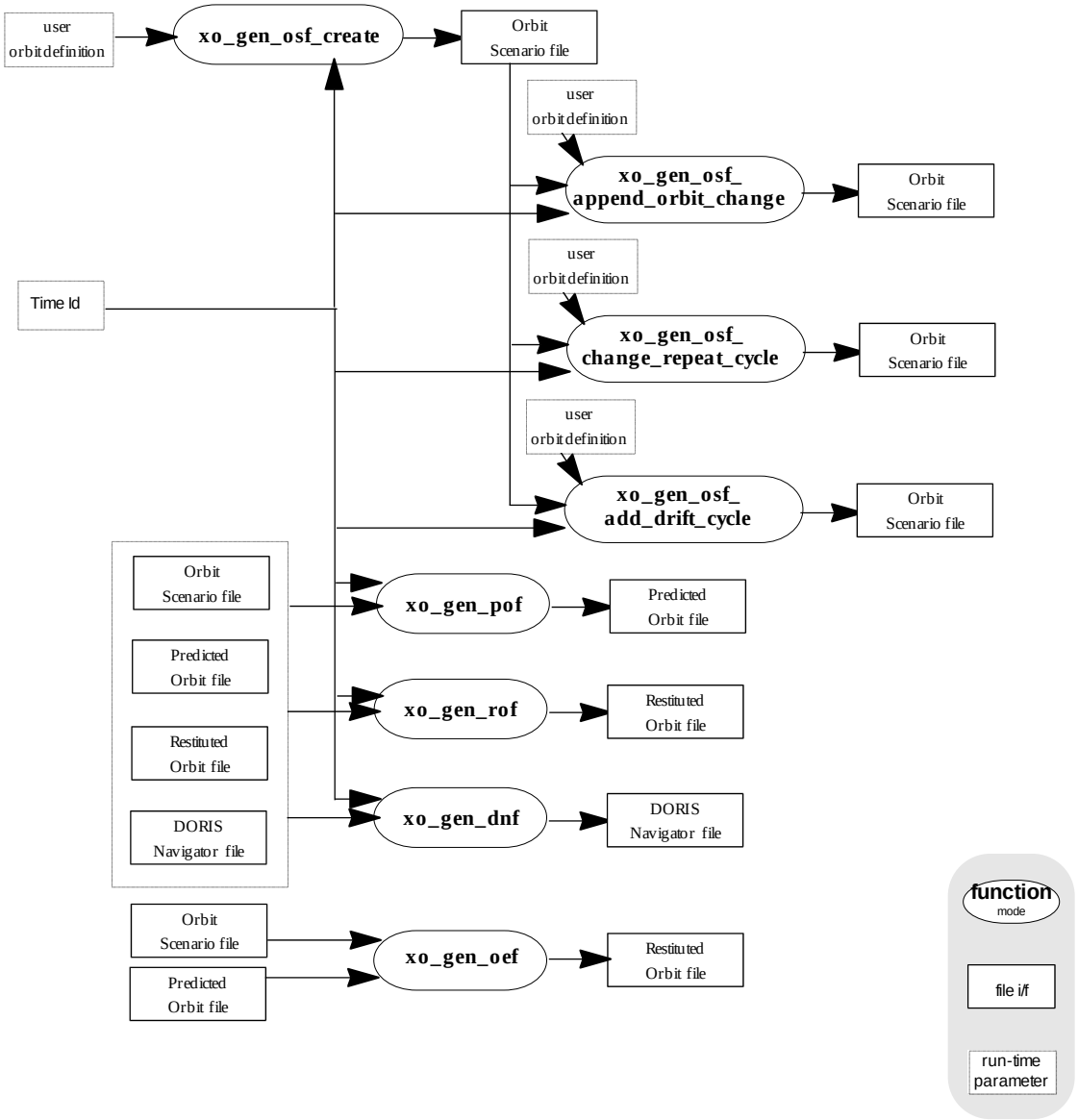


Figure 2: File Generation Calling Sequence

5 LIBRARY INSTALLATION

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

6 LIBRARY USAGE

The EO_ORBIT software library has the following dependencies:

- Other EOCFI libraries:
    - EO_FILE_HANDLING (See [F_H_SUM]).
    - EO_DATA_HANDLING (See [D_H_SUM]).
    - EO_LIB (See [LIB_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_ORBIT 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_orbit.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_orbit

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

   Windows platforms:

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

   libexplorer_lib.lib 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 xo_.

To avoid problems in linking a user application with the `EO_ORBIT` 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 XO_ or xo_.

This is summarized in Table 1.

Table 1: CFI functions included within EO_ORBIT library

Function Name	Enumeration value	long
Main CFI Functions		
xo_orbit_init_def	XO_ORBIT_INIT_DEF_ID	0
xo_orbit_cart_init	XO_ORBIT_CART_INIT_ID	1
xo_orbit_cart_init_precise	XO_ORBIT_CART_INIT_PRECISE_ID	2
xo_orbit_id_init	XO_ORBIT_ID_INIT_ID	3
xo_orbit_init_file	XO_ORBIT_INIT_FILE_ID	4
xo_orbit_init_file_precise	XO_ORBIT_INIT_FILE_PRECISE_ID	5
xo_orbit_close	XO_ORBIT_CLOSE_ID	6
xo_osv_compute	XO_OSV_COMPUTE_ID	7
xo_osv_compute_extra	XO_OSV_COMPUTE_EXTRA_ID	8
xo_orbit_to_time	XO_ORBIT_TO_TIME_ID	9
xo_time_to_orbit	XO_TIME_TO_ORBIT_ID	10
xo_orbit_abs_from_rel	XO_ORBIT_ABS_FROM_REL_ID	11
xo_orbit_rel_from_abs	XO_ORBIT_REL_FROM_ABS_ID	12
xo_orbit_abs_from_phase	XO_ORBIT_ABS_FROM_PHASE_ID	13
xo_orbit_info	XO_ORBIT_INFO_ID	14
xo_osv_to_tle	XO_OSV_TO_TLE_ID	15
xo_run_init	XO_RUN_INIT_ID	16
xo_gen_oef	XO_GEN_OEF_ID	17
xo_gen_osf_create	XO_GEN_OSF_CREATE_ID	18
xo_gen_osf_append_orbit_change	XO_GEN_OSF_APPEND_ORBIT_CHANGE_ID	19
xo_gen_osf_change_repeat_cycle	XO_GEN_OSF_CHANGE_REPEAT_CYCLE_ID	20
xo_gen_osf_add_drift_cycle	XO_GEN_OSF_ADD_DRIFT_CYCLE_ID	21
xo_gen_pof	XO_GEN_POF_ID	22
xo_gen_rof	XO_GEN_ROF_ID	23
xo_gen_rof_prototype	XO_GEN_ROF_PROTOTYPE_ID	24
xo_gen_dnf	XO_GEN_DNF_ID	25
xo_gen_tle	XO_GEN_TLE_ID	26

xo_check_osf	XO_CHECK_OSF_ID	27
xo_check_oef	XO_CHECK_OEF_ID	28
xo_position_on_orbit_to_time	XO_POSITION_ON_ORBIT_TO_TIME_ID	29
xo_orbit_data_filter	XO_ORBIT_DATA_FILTER_ID	30
xo_orbit_id_init_data_close	XO_ORBIT_ID_INIT_DATA_CLOSE_ID	31
xo_orbit_id_change	XO_ORBIT_ID_CHANGE_ID	32
xo_orbit_info_configure	XO_ORBIT_INFO_CONFIGURE_ID	33
xo_osv_check	XO_OSV_CHECK_ID	34
xo_orbit_id_check	XO_ORBIT_ID_CHECK_ID	35
Error Handling Functions		
xo_verbose	not applicable	
xo_silent		
xo_get_code		
xo_get_msg		
xo_print_msg		

Notes about the table:

- To transform the status vector returned by a CFI function to either a list of error codes or list of error messages, the enumeration value (or the corresponding integer value) described in the table must be used.
- The error handling functions have no enumerated value.
- Orbit Event File is deprecated, only supported for CRYOSAT mission

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 (XO_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_ORBIT routines, as shown in the table below. The enumerations presented in [GEN_SUM] are also applicable

Table 2: Some enumerations within EO_ORBIT library

Input	Description	Enumeration value	Long
Propagation model	Propagation not initialized	XO_PROPAG_MODEL_NOT_INITIALIZED	-1
	Mean Kepler elements model	XO_PROPAG_MODEL_MEAN_KEPL	0
	SPOT elements model (this model is not implemented)	XO_PROPAG_MODEL_SPOT	1
	TLE model	XO_PROPAG_MODEL_TLE	2
	Precise model (analytical propagator)	XO_PROPAG_MODEL_PRECISE	3
	Auto initialization mode	XO_PROPAG_MODEL_AUTO	10
	Double initialization mode	XO_PROPAG_MODEL_DOUBLE	100
Non Sun-synchronous orbit characterisation	MLST drift	XO_NOSUNSYNC_DRIFT	0
	Inclination	XO_NOSUNSYNC_INCLINATION	1
	MLST non-linear drift	XO_NOSUNSYNC_DRIFT_NONLINEAR	2
	Selection of simplified algorithm (additive value)	XO_NOSUNSYNC_USE_SIM_MODEL	10
Time inputs selection	Select the whole file	XO_SEL_FILE	0
	Time	XO_SEL_TIME	1
	Orbit	XO_SEL_ORBIT	2
	Default value	XO_SEL_DEFAULT	3
Interpolation model	Default	XO_INTERPOL_MODEL_DEFAULT	0
Orbit Init Model	Unknown mode	XO_ORBIT_INIT_UNKNOWN_MODE	-1
	Automatic detection of file	XO_ORBIT_INIT_AUTO	0
	Orbit Change mode	XO_ORBIT_INIT_ORBIT_CHANGE_MODE	1
	State Vector mode	XO_ORBIT_INIT_STATE_VECTOR_MODE	2
	Orbit Scenario File mode	XO_ORBIT_INIT_OSF_MODE	3
	Predicted Orbit File mode	XO_ORBIT_INIT_POF_MODE	4
	Restituted Orbit File mode	XO_ORBIT_INIT_ROF_MODE	5
	DORIS mode	XO_ORBIT_INIT_DORIS_MODE	6
	POF refined with DORIS mode	XO_ORBIT_INIT_POF_N_DORIS_MODE	7
	OSF part of the OEF mode	XO_ORBIT_INIT_OEF_OSF_MODE	8
	POF part of the OEF mode	XO_ORBIT_INIT_OEF_POF_MODE	9
	TLE file	XO_ORBIT_INIT_TLE_MODE	11
		XO_ORBIT_INIT_TLE_SGP4_MODE	36
		XO_ORBIT_INIT_TLE_SDP4_MODE	37
	SP3 file mode	XO_ORBIT_INIT_SP3_MODE	28
	OEM file mode	XO_ORBIT_INIT_OEM_MODE	32

	State Vector plus precise mode	XO_ORBIT_INIT_STATE_VECTOR_PRECISE_MODE	33
	Predicted Orbit File plus precise mode	XO_ORBIT_INIT_POF_PRECISE_MODE	34
	Restituted Orbit File plus precise mode	XO_ORBIT_INIT_ROF_PRECISE_MODE	35
	DORIS plus precise mode	XO_ORBIT_INIT_DORIS_PRECISE_MODE	36
	Orbit Event File plus precise mode	XO_ORBIT_INIT_OEF_POF_PRECISE_MODE	37
	POF and DORIS files plus precise mode	XO_ORBIT_INIT_POF_N_DORIS_PRECISE_MODE	38
	Geostationary satellite with fixed longitude and altitude	XO_ORBIT_INIT_GEO_LON_ALT_MODE	39
	TLE initialization with SGP4 propagator	XO_ORBIT_INIT_TLE_SGP4_MODE	40
	TLE initialization with SDP4 propagator	XO_ORBIT_INIT_TLE_SDP4_MODE	41
	Initialization with a generic list of state vectors	XO_ORBIT_INIT_USER_OSV_LIST_MODE	42
	Initialize with POF but update the state vector orbit numbers with the information of OSF	XO_ORBIT_INIT_POF_ORBNUM_ADJ_MODE	43
	Initialize with ROF but update the state vector orbit numbers with the information of OSF	XO_ORBIT_INIT_ROF_ORBNUM_ADJ_MODE	44
	Initialize with DORIS but update the state vector orbit numbers with the information of OSF	XO_ORBIT_INIT_DORIS_ORBNUM_ADJ_MODE	45
	Initialize with OEM but update the state vector orbit numbers with the information of OSF	XO_ORBIT_INIT_OEM_ORBNUM_ADJ_MODE	46
	Maximum value of enumeration	XO_ORBIT_INIT_MAX_VALUE	47
Phase increment	Do not increment phase number at next orbit change	XO_NO_PHASE_INCREMENT	0
	Do increment phase number at next orbit change	XO_PHASE_INCREMENT	1
Orbit change search direction	Search forward	XO_SEARCH_FORWARD	1
	Search backward	XO_SEARCH_BACKWARD	-1
File Type	Orbit Scenario File	XO_REF_FILETYPE_OSF	1
	OSF from an Orbit Event File	XO_REF_FILETYPE_OEF_OSF	2
	FOS Predicted Orbit File	XO_REF_FILETYPE_POF	3

	POF from an Orbit Event File	XO_REF_FILETYPE_OEF_POF	4
	DORIS Navigator File	XO_REF_FILETYPE_DORIS_NAV	5
	FOS Restituted Orbit File	XO_REF_FILETYPE_ROF	6
	DORIS Preliminary Orbit File	XO_REF_FILETYPE_DORIS_PREM	7
	DORIS Precise Orbit File	XO_REF_FILETYPE_DORIS_PREC	8
Precision for ROF and DORIS state vectors times	Default value, non-precise	XO_OSV_PRECISE_NO	1
	Precise location every integer minute	XO_OSV_PRECISE_MINUTE	2
	Precise location every ten seconds	XO_OSV_PRECISE_TEN_SECONDS	3
Number of parameters for Orbit file checking	Number of parameters to check in the functions for checking orbit files	XO_NUM_CHECK_PARAMS	6
TLE generation mode	The OSVs in the requested range are fitted to one TLE. For POF files, the set of OSVs to fit are generated with propagation. For ROF files, the set of OSVs are taken from the file.	XO_FIT_TLE	0
	The OSVs in the requested range are fitted to one TLE. The OSVs are taken from the file.	XO_FIT_TLE_LIST	1
	One TLE is generated for every OSV	XO_ONE_TLE_PER_OSV	2
Precise propagator user flag	Use predefined default values for some parameters	XO_DEFAULT_VALUES	0
	Use values introduced by use	XO_USER_VALUES	1
Precise propagator propagation contribution selection	Do not select contribution	XD_NOT_SELECT	0
	Select contribution	XD_SELECT	1
Precise propagator propagation SGA input data	Use SGA input parameter	XO_SGA_USE_PARAMETERS	0
	Read SGA values from files	XO_SGA_READ_VALUES_FROM_FILE	1
Reference time of records	TAI reference	XO_TIME_REF_OF_TAI	0
	UTC reference	XO_TIME_REF_OF_UTC	1
	UT1 reference	XO_TIME_REF_OF_UT1	2
Orbit id change enumeration	Update orbit numbers using OSF	XO_ORBIT_ID_CHANGE_OSF	0
	Update orbit number using input time+orbit	XO_ORBIT_ID_CHANGE_TIME_ORBIT	1

GEO input geodetic coordinates type	Only input longitude is taken into account for initialization	XO_GC_LONGITUDE_ONLY	0
Data filter type	Outliers filter	XO_FILTER_OUTLIERS	0
Data filter action	Remove the sample	XO_REMOVE	
Orbit info items	Spacecraft midnight (SMX)	XO_ORBIT_INFO_ITEM_SMX	0
Orbit info flag for activation/deactivation of computation of xo_orbit_info items	Deactivate computation	XO_DEACTIVATE_ITEM	0
	Activate computation	XO_ACTIVATE_ITEM	1
Orbit type: Absolute or relative (XO_Orbit_type_enum)	Absolute Orbit	XO_ORBIT_ABS	0
	Relative Orbit	XO_ORBIT_REL	1
Time type (XO_Time_type_enum)	Only UTC time info is provided	XO_UTC_TYPE	0
	Only orbit info is provided	XO_ORBIT_TYPE	1
	UTC time and orbit info is provided	XO_BOTH_TYPE	2

Note: Orbit Event File is deprecated, only supported for CRYOSAT mission

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 the current section is to present the data structures that are used in the EO_ORBIT library. The structures are currently used for the CFI Identifiers accessor functions. The following table show the structures with their names and the data that contain:

Table 3: EO_ORBIT structures

Structure name	Data		
	Variable Name	C type	Description
xo_osv_rec	tai_time	double	TAI time for the state vector
	utc_time	double	UTC time for the state vector
	ut1_time	double	UT1 time for the state vector
	abs_orbit	long	Absolute orbit
	ref_frame	long	Reference frame
	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 from [D_H_SUM])
	pos	double[3]	Position of the OSV (x, y, z) components
	vel	double[3]	Velocity of the OSV (x, y, z) components
	quality	double	Quality index for DORIS Preliminary and DORIS Precise Orbit files, this value corresponds to the enumeration *Quality Index (see Table 2 from* [D_H_SUM]*)*
xo_anx_extra_info	abs_orbit	long	Absolute orbit number
	tanx	double	ANX time (UT1)
	tnod	double	Nodal period of the orbit
xo_mission_info	abs_orbit	long	Absolute orbit number
	rel_orbit	long	Relative orbit number
	cycle_num	long	Cycle number
	phase_num	long	Phase number
xo_ref_orbit_info	drift_mode	long	Non Sun-synchronous orbit characterisation (see Table 2 for possible values)
	inclination	double	Orbit inclination
	rep_cycle	long	Repeat cycle (days)
	cycle_len	long	Cycle length (orbits)
	ANX_long	double	ANX longitude
	`anx_longitude_drift`	`xo_anx_longitude_drift`	ANX longitude drift
	mlst	double	MLST for the ANX
	mlst_drift	double	MLST drift
	mlst_nonlinear_drift	xo_mlst_nonlinear_drift	MLST non linear drift
`xo_anx_longitude_drift`	`offset`	double	ANX offset
	`linear_term`	double	ANX drift rate (deg/day)
xo_anx_info	anx_tai	double	TAI time for the ANX
	anx_utc	double	UTC time for the ANX
	anx_ut1	double	UT1 time for the ANX
	time_ref_of	long	Reference time of the ANX
	anx_pos	double[3]	Position vector
	anx_vel	double[3]	Velocity vector
	kepl	double[6]	Keplerian elements

	tnod	double	Nodal period
xo_osf_records	mission_info	xo_mission_info	Orbit numbers
	ref_orbit_info	xo_ref_orbit_info	Orbit Geometry data
	anx_info	xo_anx_info	ANX Data
xo_validity_time	time_ref	long	Time reference
	start	double	Validity star time
	stop	double	Validity stop time
xo_uni_propag	time_ref	long	Time reference in use
	val_time	xo_validity_time	validity propagation time range in UT1 time
	abs_orbit	long	Predicted Absolute orbit
	time_since_anx	double	Time since ANX
	time	double	Predicted time (UT1)
	pos	double[3]	Osculating position vector at pred. time (EF)
	vel	double[3]	Osculating velocity vector at pred. time (EF)
	acc	double[3]	Osculating acceleration vector at pred. time (EF)
	x	double[6];	Osculating keplerian elements at pred. time (TOD)
xo_propag_id_data	double_propag_flag	long	XL_TRUE if the using double propagation
	accu_mode	long	Flag to indicate if using high or low accuracy mode: 1 = low accuracy 2= high accuracy
	propag_osv	xo_uni_propag	Reference data for propagation
xo_interpol_id_data	time_ref	long	Time reference
	time	double	Time for the interpol reference state vector
	abs_orbit	long	Absolute orbit number
	time_since_anx	double	Time since ANX
	pos	double[3]	Position vector
	vel	double[3]	Velocity vector
	acc	double[3]	Acceleration vector
	kep	double[6]	Keplerian elements
	val_time	xo_validity_time	Interpolation validity time range
xo_propag_precise_	user_flag	long	Indicates if default (XO_DEFAULT_VALUES) or user-

config			defined (XO_USER_VALUES) values are used for some parameters.
	models_path	char[XD_MAX_STR]	Path where files necessary for models are looked for.
	gravity_flag	long	Gravity perturbation used (XO_SELECT) or not (XO_NOT_SELECT).
	thirdbody_flag	long	Third bodies (Sun and Moon) perturbation used (XO_SELECT) or not (XO_NOT_SELECT).
	atmos_flag	long	Atmosphere perturbation used (XO_SELECT) or not (XO_NOT_SELECT).
	srp_flag	long	Solar radiation pressure perturbation used (XO_SELECT) or not (XO_NOT_SELECT).
	step	double	Simulation step (seconds).
	grav_file	char[XD_MAX_STR]	File with data of gravitational model.
	grav_degree	long	Degree used gravity model.
	grav_order	long	Order used in gravity model.
	sga_flag	long	ap, f107 and f107a parameters used (XD_SGA_USE_PARAMETERS) or data read from files sga_ap_file and sga_f107_file (XD_SGA_READ_VALUES_FROM_FILE)
	sga_ap_file	char[XD_MAX_STR]	File with Geomagnetic Activity index values.
	sga_f107_file	char[XD_MAX_STR]	File with F10.7 Solar Activity index val ues.
	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 [m2].
	sc_drag_coeff	double	S/C drag coefficient.
	sc_srp_area	double	S/C effective Solar Radiation Pressure area [m2].
	sc_srp_coeff	double	S/C Solar Radiation Pressure coefficient
xo_mlst_nonlinear_	linear_approx_validity	long	Number of orbits in which MLST linear

drift			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
xo_orbit_id_init_data_union	file_set	xd_eocfi_file_set	Set of data structures with the data read from files
xo_orbit_id_init_data	data_type	long	Enumeration: only XO_FILE_DATA
	orbit_id_init_data	xo_orbit_id_init_data_union	Set of orbit data read from files
	change_data	xo_orbit_id_change_data	Values to be used to update the orbit numbers in the state vectors, if needed
xo_geo_geod_coord	gc_longitude	double	Geostationary geocentric longitude [deg]
	gd_latitude	double	Geostationary geodetic latitude [deg]
	gd_altitude	double	Geostationary geodetic altitude [m]
xo_geo_orbit_info	geod_coord	xo_geo_geod_coord	Geostationary geodetic corrdinates
xo_geo_orbit_init_data	init_type	long	GEO orbit info type (see XO_Geo_coord_enum)
	geo_orbit_info	xo_geo_orbit_info	Geostationary orbit information
xo_orbit_filter_cfg_union	outliers_cfg	xo_orbit_filter_outliers_cfg	Outliers filter configuration
xo_orbit_filter_settings	type	long	Filter type (Data filter type enumeration)
	filter_cfg	xo_orbit_filter_cfg_union	Filter configuration
xo_orbit_filter_report_union	outliers_report	xo_orbit_filter_outliers_report	Outliers filter report
xo_orbit_filter_report	type	long	Filter type (Data filter type enumeration)
	filter_report	xo_orbit_filter_report_union	Filter report
xo_orbit_filter_outliers_cfg	threshold_pos	double	Threshold in position [m]
	threshold_vel	double	Threshold in velocity [m/s]
	action	long	Action to be taken (Data filter action enumeration)
xo_orbit_filter_outliers_report	nof_OSV_in	long	Number of input state vectors
	nof_OSV_filtered	long	Number of filtered state vectors

	min_time_gap	double	Minimum time gap between state vectors [seconds]
	max_time_gap	double	Maximum time gap between state vectors [seconds]
	min_RMS_pos	double	Minimum RMS for position.
	max_RMS_pos	double	Maximum RMS for position.
	min_RMS_vel	double	Minimum RMS for velocity.
	max_RMS_vel	double	Maximum RMS for velocity.
xo_orbit_id_change_data	change_mode	long	See Orbit id change enum
	eocfi_file	xd_eocfi_file	File to be used as reference for orbit update
	change_time_ref	long	Time reference
	change_time	double	Time corresponding to change_orbit (in change_time_ref reference time)
	change_orbit	long	Orbit corresponding to change_time
xo_time	type	long	Time value. According to XV_Time_type_enum, the time can be given as: • A UTC time value • An orbit number plus the time elapsed since the ANX • Both values above. The struct values below are filled according to the type given in this field.
	utc_time	double	UTC time value
	orbit_type	long	Orbit type number or orbit_num (according to XO_Orbit_type_enum)
	orbit_num	long	Orbit number (absolute or relative)
	cycle	long	Cycle number (if orbit type is a relative orbit)
	sec	long	Seconds after ANX
	msec	long	Microseconds after ANX
xo_time_interval	tstart	xo_time	Time interval start
	tstop	xo_time	Time interval stop
xo_orbit_id_check_osv_report	num_osvs_outside_loose_tolerance	long	Number of state vectors outside of loose tolerance
	index_osvs_outside_lose_tolerance	long*	Array with indices of state vectors outside of loose tolerance
	num_osvs_outside_tig	long	Number of state vectors outside of tight

	ht_tolerance		tolerance
	index_osvs_outside_tight_tolerance	long*	Array with indices of state vectors ouside of tight tolerance
xo_orbit_id_check_report	file_diag_report	xd_orbit_file_diagnostics_report	Diagnostics data from xd_orbit_file_diagnostics
	osv_report	xo_orbit_id_check_osv_report	Diagnostics data from xo_osv_check

7 CFI FUNCTIONS DESCRIPTION

The following sections describe each CFI function.

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 xo_orbit_init_def

7.1.1 Overview

The **xo_orbit_init_def** routine generates a Cartesian orbit state vector around the true ascending node crossings. The result is stored and returned through the xo_orbit_id variable so that can fed other routines involving orbit calculations. The data generated by the **xo_orbit_init_def** function is based on:

- Date (processing time),
- Longitude of the ascending node,
- Satellite Repeat Cycle and Cycle Length
- Mean local solar time at ascending node
- Drift of mean local solar time or the inclination

The user should take into account that **xo_orbit_init_def** only retrieve and stores internal data for one orbit.

The validity start and stop times of the initialization (***val_time0*** and ***val_time1*** output parameters) represents the allowed time window for orbit calculations. If the **xo_orbit_init_def** function is called, this time window starts at the time of the first ANX and ends at 31/12/2099 23:59:59.

Before calling this function it is required to initialise the time correlations, using either **xl_time_ref_init** or **xl_time_ref_init_file** EO_LIB functions (see [LIB_SUM]).

In order to obtain results consistent with the ones obtained initializing the orbit id with an equivalent Orbit Scenario file, the drift_mode flag has to be set to drift_mode = XO_NOSUNSYNC_DRIFT + XO_NOSUNSYNC_USE_SIM_MODEL.

Warning: The algorithm used in this function is only valid for satellites with a finite valid range for the inclination and the semi-major axis of the orbit. In CRYOSAT, for example, as there are no minimum and maximum values defined of these two orbital elements, there are defined provisional ranges of the same size as the ones defined in ENVISAT until new requirements are defined. The nominal values have been taken from the [MCD]. There is not available any other nominal orbital element for any other satellite, so this routine is only valid (at this moment) for both CRYOSAT and ENVISAT.

A complete calling sequence of the orbit calculations procedure is presented in section 4.2.

Note: function xo_orbit_init_def is deprecated. It is recommended to use xo_orbit_init_def_2 instead.

7.1.2 Calling interface

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

```
#include <explorer_orbit.h>
{
  long sat_id, propag_model, time_ref, time_init_mode;
  xl_model_id model_id = {NULL};
  xl_time_id time_id = {NULL};
```

```
    xo_orbit_id orbit_id = {NULL};
    long drift_mode, irep, icyc;
    long orbit0, orbit;
    double time0, time, val_time0, val_time1;
    double ascmlst_drift, inclination, rlong, ascmlst;
    long status, ierr[XO_NUM_ERR_ORBIT_INIT_DEF];
    status = xo_orbit_init_def (&sat_id, &model_id, &time_id,
                                &time_ref, &time0, &orbit0,
                                &drift_mode,
                                &ascmlst_drift, &inclination,
                                &irep, &icyc, &rlong, &ascmlst,
                                &val_time0, &val_time1,
                                &orbit_id, ierr);
}
```

7.1.3 Input parameters

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

Table 4: Input parameters of xo_orbit_init_def function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
model_id	xl_model_id*	-	Model ID	-	-
time_id	xl_time_id*	-	Structure that contains the time correlations	-	-
time_ref	long*	-	Time reference ID	-	Complete
time0	double*	-	Reference time	Decimal days (Processing format)	[-18262.0,36524.0]
orbit0	long*	-	Absolute orbit number of the reference orbit	-	>= 0
drift_mode	long*	-	Flag to select between drift in mean local solar time and inclination as input characterization of the reference orbit. Note: When initializing a Sun-synchronous orbit, the selected drift mode must be	-	XO_NOSUNSYNC_DRIFT, XO_NOSUNSYNC_INCLINATION, XO_NOSUNSYNC_DRIFT + XO_NOSUNSYNC

			XO_NOSUNSYNC_DRIFT and the ascmlst_drift parameter must be set to zero. Note 2: Add *XO_NOSUNSYNC_USE_SIM_MODEL* to the drift mode to select the simplified model in the algorithm.		_USE_SIM_MODE L, XO_NOSUNSYNC_INCLINATION + XO_NOSUNSYNC_USE_SIM_MODE L
ascmlst_drift	double*	-	If *drift_mode* = *XO_NOSUNSYNC_DRIFT* Drift in mean local solar time of the reference orbit: · MLST[N+1]=MLST[N]+MLSTdrift See drift_mode entry in this table.	seconds/day	TBD
inclination	double*	-	If *drift_mode* = *XO_NOSUNSYNC_INCLINATION* Inclination of the reference orbit	deg	[0,180]
irep	long *	-	Repeat cycle of the reference orbit The actual repeat cycle is calculated as per definition included in [MCD]	days	> 0
icyc	long *	-	Cycle length of the reference orbit	orbits	> 0
rlong	double*	-	Geocentric longitude of the [Earth fixed] ascending node (Earth fixed CS)	deg	[0,360)
ascmlst	double*	-	Mean local solar time at ascending node	Decimal hours	[0, 24)

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

- Satellite ID: sat_id. See [GEN_SUM].
- Time reference ID: time_ref. See [GEN_SUM].
- Time initialisation mode: time_init_mode. See [GEN_SUM].
- Drift mode: drift_mode. Current document, section 6.2.

7.1.4 Output parameters

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

Table 5: Output parameters of xo_orbit_init_def function

C name	C type	Array	Description	Unit	Allowed Range

		Element	(Reference)	(Format)	
xo_orbit_init_def	long	-	Main status flag	-	-1, 0, +1
val_time0	double*	-	Validity start time of the initialization	Decimal days (Processing format)	[-18262.0,36524.0]
val_time1	double*	-	Validity stop time of the initialization	Decimal days (Processing format)	[-18262.0,36524.0]
orbit_id	xo_orbit _id*	-	Structure that contains the orbit initialization.	-	-
ierr[XO_NUM_ERR_ORBIT_INIT_DEF]	long	all	Status vector	-	-

7.1.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_orbit_init_def** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_orbit_init_def** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 6: Error messages of xo_orbit_init_def function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Wrong satellite flag	No calculation performed	XO_CFI_ORBIT_INIT_DEF_SAT_ERR	0
ERR	Geostationary satellite currently not supported for this function	No calculation performed	XO_CFI_ORBIT_INIT_DEF_GEO_SAT_ERR	1
ERR	Wrong input flag: %s	No calculation performed	XO_CFI_ORBIT_INIT_DEF_FLAG_ERR	2
ERR	Could not perform a time transformation	No calculation performed	XO_CFI_ORBIT_INIT_DEF_TIME_CHANGE_ERR	3
ERR	Input out of range: %s	No calculation performed	XO_CFI_ORBIT_INIT_DEF_INPUTS_ERR	4
ERR	An error occurred in the genstate routine	No calculation performed	XO_CFI_ORBIT_INIT_DEF_GENSTATE_ERR	5
ERR	Memory Error	No calculation performed	XO_CFI_ORBIT_INIT_DEF_MEMORY_ERR	6

ERR	Propagation cannot be initialised	No calculation performed	XO_CFI_ORBIT_INIT_DEF_ PROPAG_INIT_ERR	7

7.2 xo_orbit_init_def_2

7.2.1 Overview

The **xo_orbit_init_def_2** is equivalent to **xo_orbit_init_def** except for the fact that this one allows to introduce as inputs non linear terms of Mean Local Solar Time.

Note: the ANX longitude drift parameters in the input structure xo_ref_orbit_info are ignored in the orbit initialization. The orbit will be initialized as if the ANX longitude drif parameters are set to 0. If the user requires the initialization of the orbit using these parameters, an OSF with the drifting parameters has to be created and then use the CFI function **xo_orbit_init_file**.

7.2.2Calling interface

The calling interface of the **xo_orbit_init_def_2** CFI function is the following (input parameters are underlined):

```
#include <explorer_orbit.h>
{
  long sat_id, time_ref;
  xl_model_id model_id = {NULL};
  xl_time_id time_id = {NULL};
  xo_orbit_id orbit_id = {NULL};
  long orbit0;
  double time0;
  xo_ref_orbit_info ref_orbit_info;
  long status, ierr[XO_NUM_ERR_ORBIT_INIT_DEF];
  status = xo_orbit_init_def_2(&sat_id, &model_id, &time_id,
                               &time_ref, &time0, &orbit0,
                               &ref_orbit_info,
                               &val_time0, &val_time1,
                               &orbit_id, ierr);
}
```

7.2.3 Input parameters

The **xo_orbit_init_def_2** CFI function has the following input parameters:

Table 7: Input parameters of xo_orbit_init_def_2 function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
model_id	xl_mod	-	Model ID	-	-

	el_id*				
time_id	xl_time_id*	-	Structure that contains the time correlations	-	-
time_ref	long*	-	Time reference ID	-	Complete
time0	double*	-	Reference time	Decimal days (Processing format)	[-18262.0,36524.0]
orbit0	long*	-	Absolute orbit number of the reference orbit	-	>= 0
ref_orbit_info	xo_ref_orbit_info*	-	Struct with inputs for the function. The parameters are equivalent to the ones in xo_orbit_init_def (see table 4) but also MLST non linear terms can be introduced. Note: the xo_anx_longitude drift parameters are ignored (see also the in section 7.2.1)	-	-

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

- Satellite ID: sat_id. See [GEN_SUM].
- Time reference ID: time_ref. See [GEN_SUM].

7.2.4 Output parameters

The output parameters of the **xo_orbit_init_def_2** CFI function are:

Table 8: Output parameters of xo_orbit_init_def_2 function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_init_def_2	long	-	Main status flag	-	-1, 0, +1
val_time0	double*	-	Validity start time of the initialization	Decimal days (Processing format)	[-18262.0,36524.0]
val_time1	double*	-	Validity stop time of the initialization	Decimal days (Processing format)	[-18262.0,36524.0]
orbit_id	xo_orbit_id*	-	Structure that contains the orbit initialization.	-	-
ierr[XO_NUM_ERR_ORBIT_INIT_DEF]	long	all	Status vector	-	-

7.2.5 Warnings and errors

Warning and errors are the same as in function xo_orbit_init_def (see section 7.1.5).

7.3 xo_orbit_cart_init

7.3.1 Overview

This software initializes the orbit data using as input a Cartesian orbit state vector.

The validity start and stop times of the initialization (***val_time0*** and ***val_time1*** output parameters) represents the allowed time window for orbit calculations. If the **xo_orbit_cart_init** function is called, this time window starts at 01/01/1950 00:00:00 and ends at 31/12/2099 23:59:59.

Before calling this function it is required to initialise the time correlations, using either **xl_time_ref_init** or **xl_time_ref_init_file** EO_LIB functions (see [LIB_SUM]).

A complete calling sequence of the orbit calculations procedure is presented in section 4.2.

7.3.2 Calling interface

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

```
#include <explorer_orbit.h>
{
  xl_model_id model_id = {NULL};
  xl_time_id time_id = {NULL};
  xo_orbit_id orbit_id = {NULL};
  long sat_id, time_ref, abs_orbit;
  double time, pos[3], vel[3], val_time0, val_time1;
  long status, ierr[XO_NUM_ERR_ORBIT_CART_INIT];

  status = xo_orbit_cart_init(&sat_id, &model_id, &time_id,
                              &time_ref, &time,
                              pos, vel, &abs_orbit,
                               &val_time0, &val_time1,
                               &orbit_id, ierr);
}
```

7.3.3 Input parameters

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

Table 9: Input parameters of xo_orbit_cart_init function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
model_id	xl_model_id*	-	Model ID	-	-
time_id	xl_time_i	-	Structure that contains the time	-	-

	d*		correlations		
time_ref	long*	-	Time reference ID	-	Complete
time	double*	-	Reference time	Decimal days (Processing format)	[-18262.0,36524.0]
pos	double[3]	all	Initial osculating position vector (X, Y, Z) (EF reference frame)	m	-
vel	double[3]	all	Initial osculating velocity vector (X, Y, Z) (EF reference frame)	m/s	-
abs_orbit	long*	-	Orbit of the state vector	-	> 0

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

- Satellite ID: sat_id. See [GEN_SUM].
- Time reference ID: time_ref. See [GEN_SUM].

7.3.4 Output parameters

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

Table 10: Output parameters of xo_orbit_cart_init function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_cart_init	long	-	Main status flag	-	-1, 0, +1
val_time0	double*	-	Validity start time of the initialization	Decimal days (Processing format)	[-18262.0,36524.0]
val_time1	double*	-	Validity stop time of the initialization	Decimal days (Processing format)	[-18262.0,36524.0]
orbit_id	xo_orbit_id*	-	Structure that contains the orbit initialization.	-	-
ierr[XO_NUM_ERR_ORBIT_CART_INIT]	long	all	Status vector	-	-

7.3.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_orbit_cart_init** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_orbit_cart_init** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 11: Error messages of xo_orbit_cart_init function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Wrong Satellite Id.	No calculation performed	XO_CFI_ORBIT_CART_INIT_SAT_ERR	0
ERR	Geostationary satellite currently not supported for this function	No calculation performed	XO_CFI_ORBIT_CART_INIT_GEO_SAT_ERR	1
ERR	Wrong input flag	No calculation performed	XO_CFI_ORBIT_CART_INIT_FLAG_ERR	2
ERR	Input Time Id. is not initialized.	No calculation performed	XO_CFI_ORBIT_CART_INIT_TIME_STATUS_ERR	3
ERR	Orbit Id is already initialized.	No calculation performed	XO_CFI_ORBIT_CART_INIT_STATUS_ERR	4
ERR	Time conversion error.	No calculation performed	XO_CFI_ORBIT_CART_INIT_TIME_TRANSFORMING_ERR	5
ERR	Time out of limits.	No calculation performed	XO_CFI_ORBIT_CART_INIT_TIME_RANGE_ERR	6
ERR	Memory allocation error.	No calculation performed	XO_CFI_ORBIT_CART_INIT_MEMORY_ERR	7

7.4 xo_orbit_cart_init_precise

7.4.1 Overview

This software initializes the orbit data using as input a Cartesian orbit state vector for precise propagation (the state vectors will be computed with a numeric propagator).

The validity start and stop times of the initialization (***val_time0*** and ***val_time1*** output parameters) represents the allowed time window for orbit calculations. If the **xo_orbit_cart_init_precise** function is called, this time window starts at the time of the state vector and ends at 31/12/2099 23:59:59.

Before calling this function it is required to initialise the time correlations, using either **xl_time_ref_init** or **xl_time_ref_init_file** EO LIB functions (see [LIB_SUM]).

A complete calling sequence of the orbit calculations procedure is presented in section 4.2.

7.4.2 Calling interface

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

```
#include <explorer_orbit.h>
{
  xl_model_id model_id = {NULL};
  xl_time_id time_id = {NULL};
  xo_orbit_id orbit_id = {NULL};
  long sat_id, time_ref, abs_orbit;
  double time, pos[3], vel[3], val_time0, val_time1;
  xo_propag_precise_config precise_conf;
  long status, ierr[XO_NUM_ERR_ORBIT_CART_INIT];
  status = xo_orbit_cart_init_precise(&sat_id, &model_id,
                                      &time_id,
                                          &time_ref, &time,
                                           pos, vel, &abs_orbit,
                                      &precise_conf,
                                            &val_time0, &val_time1,
                                            &orbit_id, ierr);
}
```

7.4.3 Input parameters

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

Table 12: Input parameters of xo_orbit_cart_init_precise function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete

model_id	xl_model_id*	-	Model ID	-	-
time_id	xl_time_id*	-	Structure that contains the time correlations	-	-
time_ref	long*	-	Time reference ID	-	Complete
time	double*	-	Reference time	Decimal days (Processing format)	[-18262.0,36524.0]
pos	double[3]	all	Initial osculating position vector (X, Y, Z) (EF reference frame)	m	-
vel	double[3]	all	Initial osculating velocity vector (X, Y, Z) (EF reference frame)	m/s	-
abs_orbit	long*	-	Orbit of the state vector	-	> 0
precise_conf	xo_propag_precise_config*	-	Configuration parameters for precise propagator.	-	struct members with restrictions: - All flags: 0 or 1. - step: > 0. - grav_degree, grav_order > 0. - ap, f107, f107a: >= 0. - sc_mass: > 0. - sc_drag_area: > 0. - sc_drag_coeff: > 0. - sc_srp_area: > 0. - sc_srp_coeff: > 0.

In precise_conf parameter, at least user_flag, models_path and satellite values (sc_mass, sc_drag_area, sc_drag_coeff, sc_srp_area, sc_srp_coeff) must be provided. The other values must be provided just in case the user does not want to use default values (user_flag = XO_USER_VALUES). If default values are seleted (user_flag = XO_DEFAULT_VALUES), then the following values are used:

- gravity_flag = XO_SELECT;
- thirdbody_flag = XO_SELECT;
- atmos_flag = XO_SELECT;
- srp_flag = XO_SELECT;
- step = 10. [s];
- grav_file = egm96.grv;
- grav_degree = 10;
- grav_order = 10;
- sga_flag = XO_SGA_READ_VALUES_FROM_FILE (Use files, not constant values for AP and F107A);

- sga_ap_file = ap_esa_ecss_jan2000_mean.sga;
- sga_f107_file = f107_esa_ecss_jan2000_mean.sga;
- ap = 0.;
- f107 = 0.;
- f107a = 0.;

Some files that can be used or taken as example by the user are provided in the files/models directory of the EOCFI libraries. There are files for gravity model (egm96.grv), F10.7 index (f107_*.sga) and Geomagnetic activity index (ap_*.sga).

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

- Satellite ID: sat_id. See [GEN_SUM].
- Time reference ID: time_ref. See [GEN_SUM].

7.4.4 Output parameters

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

Table 13: Output parameters of xo_orbit_cart_init_precise function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_cart_init_precise	long	-	Main status flag	-	-1, 0, +1
val_time0	double*	-	Validity start time of the initialization	Decimal days (Processing format)	[-18262.0,36524.0]
val_time1	double*	-	Validity stop time of the initialization	Decimal days (Processing format)	[-18262.0,36524.0]
orbit_id	xo_orbit_id *	-	Structure that contains the orbit initialization.	-	-
ierr[XO_NUM_ERR_ORBIT_CART_INIT_PRECISE]	long	all	Status vector	-	-

7.4.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_orbit_cart_init_precise** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_orbit_cart_init_precise** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 14: Error messages of xo_orbit_cart_init_precise function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Error in call to function xo_orbit_cart_init.	No calculation performed	XO_CFI_ORBIT_CART_INIT_FLAG_ERR	0
ERR	Error initialising propagation.	No calculation performed	XO_CFI_ORBIT_CART_INIT_PRECISE_PROPAG_INIT_ERR	1
ERR	Time conversion error.	No calculation performed	XO_CFI_ORBIT_CART_INIT_PRECISE_TIME_CONVERSION_ERR	2
ERR	Error in precise propagator input parameters	No calculation performed	XO_CFI_ORBIT_CART_INIT_PRECISE_PRECISE_PARAMS_ERR	3

7.5 xo_orbit_init_file

7.5.1 Overview

The **xo_orbit_init_file** function is used for initializing the orbit calculations using one of these orbit files:

- One Orbit Scenario File providing orbital changes.
- TLE files. xo_orbit_init_file extracts and uses only those TLE entries with the default NORAD satellite designator correspondent to the input satellite Id. The correspondence table between satellite ids and default NORAD satellite designators is given in table 224 from the section 9.17, [D_H_SUM]. In case the input file is not compliant with this correspondence, the default designator can be changed by using function xl_set_tle_sat_data, section 7.48 [LIB_SUM].
- Files containing a list of state vectors (see also note below):
  - One or more FOS Predicted ascending node cartesian state vectors file. In case multiple files are used, the files should be time ordered and the gap between them (i.e. time difference between the last vector of nth file and the first vector of the nth+1 file) should be less than two orbital periods.
  - One FOS Predicted Orbit File plus a DORIS Navigator unconsolidated level-0 products file.
  - One or more Orbit Event files. Note: Orbit Event File is deprecated, only supported for CRYOSAT mission
  - One or more FOS Restituted orbit files.
  - One or more DORIS Navigator files. Three formats of DORIS Navigator files are supported and automatically detected: Cryosat, Sentinel 3 and Jason CS (see [D_H_SUM] for further details). Notice that when using DORIS Navigator files the orbit is initialized based only on the navigation data in the NAV_T (ITRF) packets, and the absolute orbit number provided in NAV_G (Geodetic) packets if available in the same input file. NAV_G (Geodetic) packets are expected only for Sentinel 3 and Jason CS DORIS Navigation files and, as referred in [D_H_SUM], in case NAV_G (Geodetic) packets are not available the orbit number will be set to 1 at the first OSV and increased at each ANX. To adjust the orbit number, use the function xd_orbit_id_change as described in section 7.60.
  - One or more DORIS Predicted files.
  - One or more DORIS Preliminary files.
  - One or more generic OSVS lists. In this case any file with a list of state vectors can be used, but the propagator is not initialized.
  - One or more SP3 files.
  - One or more OEM files. Of the fields read with the funcion xd_read_oem (see section 9.21 of [D_H_SUM]), the ones that are relevant for the orbit initialisation are: REF_FRAME, TIME_SYSTEM, Epoch, X, Y, Z, X_DOT, Y_DOT, Z_DOT.
- One Orbit Scenario file plus files containing a list of state vectors (i.e. FOS Predicted, FOS Restituted or DORIS files): in this case, the orbit id is initialized with state vectors, but the orbit number of every state vector is corrected to be consistent with the one expected from OSF. In this case, the OSF shall be the first item of the input files input array; the orbit file mode must be set explicitly to respectively XO_ORBIT_INIT_POF_ORBNUM_ADJ_MODE,

XO_ORBIT_INIT_ROF_ORBNUM_ADJ_MODE,
XO_ORBIT_INIT_DORIS_ORBNUM_ADJ_MODE,
XO_ORBIT_INIT_OEM_ORBNUM_ADJ_MODE.

NOTE: OSVs are considered restituted if the time interval between one OSV and the next does not exceed 1800sec. In this case following OSV computation will be done via interpolation. In any other case, propagation will be used.

The format of the above files is described in [FORMATS].

In order to read files, xo_orbit_init_file function internally uses Data Handling functions. Please refer to [D_H_SUM], in particular sections 4.2 and 4.3, for further details.

Before calling this function it is required to initialise the time correlations, using either **xl_time_ref_init, xl_time_ref_init_file** or **xl_time_id_init** EO LIB functions (see [LIB_SUM]).

The user can select the time interval to be used from the input file(s) using three different ways:

Table 15: User requested time range in xo_orbit_init_file

time_mode (see 7.33.4)	input parameter	requested start time (t_req_start)	requested stop time (t_req_stop)
XL_SEL_TIME	time0 / time1	time0	time1
XL_SEL_ORBIT	orbit0 / orbit1	tANX(orbit0)	tANX(orbit1)
XL_SEL_FILE	-	first state vector in the file(s)	last state vector in the file(s)

The validity start and stop times of the initialization (**val_time0** and **val_time1** output parameters) represents the allowed time window for orbit calculation. The following table shows the validity time interval for the different input files:

Table 16: Validity periods for xo_orbit_init_file

Input file type	val_time0	val_time1
Orbit file providing Orbit changes	ANX Time of the first orbital change	Infinity
Orbit files providing a list of orbital state vectors (Predicted Orbit Files, Restituted Orbit Files, DORIS files, SP3 files)	time of the first state vector	Time of the last state vector
TLE files	time of the first TLE	Time of the last TLE + 1day

A complete calling sequence of the orbit calculation procedure is presented in section 4.2.

7.5.1.1 Recommendations on Orbit Files Usage

7.5.1.1.1 Reference Frames

The main usage of the Orbit Library is to support geo-location. As a consequence an Earth-Fixed frame is the natural reference frame to use, end-to-end. On the other hand, accurate conversion between inertial and earth-fixed frames happens only when polar motion data is available, more precisely when the time_id has been previously initialized with a IERS Bulletin.

Therefore, when polar motion data is not available, the user is recommended to initialize the orbit id by providing Orbit files with Earth-Fixed frame data (orbit state vectors), and to compute geo-location information in Earth-Fixed. This provides accurate computations. For any other usage the user shall be aware of the consequences and accept small inaccuracies, In particular:

- initializing the orbit id with inertial orbit data, and computing inertial parameters, is also supported and is accurate;
- initializing the orbit id with earth-fixed orbit data, and computing inertial parameters (or vice-versa), leads to slightly inaccurate computations and should be avoided unless ignoring polar motion is acceptable.

7.5.1.1.2 Time correlations

EOCFI expects that the time correlations defined in the time_id are identical to those defined with the orbit data loaded from an orbit file. If there is a mismatch between the time correlations defined in the time_id and the orbit loaded from file, the time correlation of the orbit records is recomputed using the time_id as reference. When this happens a warning is issued by the library.

To avoid the warning case, the user is recommended to initialize the time_id and the orbit_id with the same orbit file. However, notice that some use cases benefit from using different files -- i.e. when using a IERS bulletin to initialize a time_id, thus providing the most accurate time correlations, and then loading an Orbit State vector file. In this case, the warning is issued and should be ignored.

The value defined in the "Time_Reference" field in the variable header of the orbit file is used as base time for such re-computation. If this value is not provided, the input parameter time_ref is used.

For example, if the orbit file is a POF and time_ref is UTC, the UTC times are left unchanged but the TAI and UT1 times are recomputed applying to the UTC times the time_id correlations.

For Orbit Scenario Files (OSF), the time reference is always UT1 and, in case the time correlation defined in the time_id differs from the one defined in the OSF, a warning is issued and, in computations using the orbit_id, the time correlation defined in the time_id will prevail. For example, in the

computation of a state vector at a given UTC time, the input UTC time is converted to UT1 using the time correlation in the time_id and not the one in the OSF. Therefore, the user must be aware that, using a time_id with a time correlation different from the one defined in the OSF, he is altering the

orbit model defined in the OSF and he has to expect results different from those computed by systems that are using the OSF with the correct time correlation (this is often the case of planning systems). In order to avoid this inconsistency, it is always strongly advised to initialize the time_id using the same OSF file.

7.5.1.1.3 Delta UTC-UT1 in Orbit Scenario Files

The Reference Orbit defined within the Orbit Scenario File is based on the assumption that the Delta UTC-UT1 is zero and does not change across orbital changes.

Introducing changes in Delta UTC-DUT1 across consecutive orbital changes (for example, by using different time correlations in two consecutive orbital changes) may introduce discontinuities in orbit computations at the transition from one orbital change to the other.

7.5.1.1.4 Computing orbit diagnostics

The function computes internally diagnostics of OSV records for OEM/SP3/Doris/orbit files to check the parts of the file that are usable for orbit initialization. The following table describes the types of errors that are checked and the behaviour of the function when a particular error is found:

Table 17: OSV diagnostics behavior at orbit initialization

Type of error	Behavior
Repeated OSV ABS [OSV_TIME(N) – OSV_TIME(N-1)] < 1 microsecond	Warning at 1st inconsistency Action: Discard OSV
Going back OSV OSV_TIME(N-1) – OSV_TIME(N) >= 1 microsecond	Warning at 1st inconsistency Action: discard OSV
GAP ROF (LEO satellites) and DORIS: OSV_TIME(N) – OSV_TIME(N-1) > 330 seconds ROF (MEO satellites): OSV_TIME(N) – OSV_TIME(N-1) > 1000 seconds OEM and SP3: OSV_TIME(N) – OSV_TIME(N-1) > 30 minutes	Warning at 1st gap found Action: stop loading OSVs at 1st gap found.
Orbit number not consistent: orbit number should not decrease.	one warning at first inconsistency. Action: adjust orbit number.

NOTE: For POF files the diagnostics are not computed.

7.5.1.1.5 Criteria for loading several files

When the orbit is initialized with more than one file (in the input parameter *input_files*), the user has to provide the files sorted from lower to higher precedence. This way the possible overlap between files is solved as follows:

input_files[0] is the file with lower precedence

input_files[n_files-1] is the file with higher precedence

the state vectors from files with lower precedence that are after the first state vector of a file with higher precedence are skipped.

Example: The following figure represents thee orbit files. Every vertical line represents the positon in time of a state vector within the file. The initilazation with these files is equivalent to initialize with a single “equivalent file” in the following way:

Overlapping files in orbit initialization

7.5.. alli ng inte rface

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

```
#include <explorer_orbit.h>
{
  xl_time_id time_id = {NULL};
  xl_model_id model_id = {NULL};
  xo_orbit_id orbit_id = {NULL};
  long sat_id, orbit_file_mode, n_files, time_mode;
  long time_ref, orbit0, orbit1;
  char **input_files;
  double time0, time1, val_time0, val_time1;
  long status, ierr[XO_NUM_ERR_ORBIT_INIT_FILE];

  status = xo_orbit_init_file (&sat_id, &model_id, &time_id,
                               &orbit_file_mode, &n_files,
                               input_files,
                               &time_mode, &time_ref,
                               &time0, &time1, &orbit0, &orbit1,
                                &val_time0, &val_time1,
                                &orbit_id, ierr);
}
```

7.5.3 Input parameters

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

Table 18: Input parameters of xo_orbit_init_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete (Geostationary satellites can only be used with TLE files) For SP3 files, see an explanation under the table.
model_id	xo_model_id*	-	Model ID	-	-
time_id	xo_time _id*	-	Structure that contains the time correlations	-	-
orbit_file_mode	long*	-	Flag that indicates the type of the input orbit file. There exists the possibility of detecting automatically the type of the files using the value XO_ORBIT_INIT_AUTO. The Orbit Event files are used as Orbit Scenario files if the AUTO mode is selected. In case they want to be used as Predicted orbit files, the option XO_ORBIT_INIT_OEF_POF_MODE should be chosen. If the AUTO mode is selected and the file type is a list of OSV (POF or ROF), the Software configures the orbit_id for propagation if the time interval between an OSV and the next is 1800 seconds (half an hour), for interpolation otherwise. The user can verify this configuration by calling xo_orbit_get_propag_mode. In case the type of initialization is XO_ORBIT_INIT_USER_OSV_LIST_MODE, the propagator/interpolator is not initialized, so any operation involving these computations will return error. In case the type of initialization is XO_ORBIT_INIT_(POF/ROF/DORIS/OEM)_ORBNUM_ADJ_MODE, the OSF shall be the first	-	XO_ORBIT_INIT_AUTO XO_ORBIT_INIT_OSF_ MODE XO_ORBIT_INIT_POF_ MODE XO_ORBIT_INIT_ROF_ MODE XO_ORBIT_INIT_DORIS _MODE XO_ORBIT_INIT_POF_N _DORIS_MODE XO_ORBIT_INIT_OEF_O SF_MODE XO_ORBIT_INIT_OEF_P OF_MODE XO_ORBIT_INIT_TLE_MODE XO_ORBIT_INIT_TLE_SGP4_MODE XO_ORBIT_INIT_TLE_SDP4_MODE XO_ORBIT_INIT_USER_OSV_LIST_MODE XO_ORBIT_INIT_SP3_MODE XO_ORBIT_INIT_POF_ORBNUM_ADJ MODE XO_ORBIT_INIT_ROF_ORBNUM_ADJ _MODE

			item of the input files input array; the orbit file mode must be set explicitly, AUTO mode is not supported for these types. A description of the propagation models is given in section 7.33.2		XO_ORBIT_INIT_DORIS_ORBNUM_ADJ_MODE XO_ORBIT_INIT_OEM_MODE XO_ORBIT_INIT_OEM_ORBNUM_ADJ_MODE
n_files	long	-	Number of input files	-	>=1
input_files	char**	-	Vector of orbit files. In case multiple files are used, they should be time ordered. If there is overlap between files, the newest data have precedence (see **Error! Reference source not found.Error! Reference source not found.Error! Reference source not found.Error! Reference source not found.Error! Reference source not found.**)	-	-
time_init_mode	long*	-	Flag for selecting the time range of the initialisation. For TLE files, the whole file is always selected (this flag and the parameters time0/time1, orbit0/orbit1 are dummies)	-	Select either: · XO_SEL_FILE · XO_SEL_ORBIT · XO_SEL_TIME For DORIS Navigator files and SP3 files, XO_SEL_ORBIT is not allowed
time_ref	long*	-	Time reference ID	-	Complete When using DORIS Navigator files and time_mode is XO_SEL_TIME, only XL_TIME_UTC is allowed.
time0	double*	-	Start time. See section 7.33.1. Used only if: · time_init_mode=XO_SEL_TIME; otherwise NULL can be passed	Decimal days (Processing format)	[-18262.0,36524.0]
time1	double*	-	Stop time. Used only if: · time_init_mode=XO_SEL_TIME; otherwise NULL can be passed	Decimal days (Processing format)	[-18262.0,36524.0]
orbit0	long*	-	Absolute orbit number of the start orbit. Used only if: · time_init_mode=XO_SEL_ORBIT;	-	-

			otherwise NULL can be passed		
orbit1	long*		Absolute orbit number of the stop orbit. Used only if: · time_init_mode=XO_SEL_ORBIT; otherwise NULL can be passed	-	-

Note: Orbit Event File is deprecated, only supported for CRYOSAT mission

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

- Satellite ID: sat_id. See [GEN_SUM].
- Orbit init mode: orbit_init_mode. Current document, section 6.2.
- Time mode: time_init_mode. See [GEN_SUM].
- Time reference ID: time_ref. See [GEN_SUM].

Note: For SP3 files, the SP3 Id has to be defined for the input sat_id (as defined in section "*Extended Standard Product 3 Orbit File (SP3-c)*" in [D_H_SUM]). The state vectors to be read are those with the SP3 Id defined for the input satellite Id. Note that the SP3 Id can be defined also by the user with the function xl_set_sp3_sat_data (See [LIB_SUM]).

7.5.4 Output parameters

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

Table 19: Output parameters of xo_orbit_init_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_init_file	long	-	Main status flag	-	-1, 0, +1
val_time0	double*	-	Validity start time of the initialization	Decimal days (Processing format)	See 7.33.1
val_time1	double*	-	Validity stop time of the initialization	Decimal days (Processing format)	See 7.33.1
orbit_id	xo_orbit_id*	-	Structure that contains the orbit initialization data	-	-
ierr[XO_NUM_ERR_ORBIT_INIT_FILE]	long	all	Status vector	-	-

7.5.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_orbit_init_file** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_orbit_init_file** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 20: Error messages of xo_orbit_init_file function

Error type	Error message	Cause and impact	Error Code	Error No
ERR	Wrong satellite flag.	No calculation performed	XO_CFI_ORBIT_INIT_FILE_SAT_ERR	0
ERR	Geostationary satellite currently not supported for this function.	No calculation performed	XO_CFI_ORBIT_INIT_FILE_GEO_SAT_ERR	1
ERR	Wrong input flag.	No calculation performed	XO_CFI_ORBIT_INIT_FILE_FLAG_ERR	2
ERR	The Time Id was not initialized.	No calculation performed	XO_CFI_ORBIT_INIT_FILE_TIME_STATUS_ERR	3
ERR	The Orbit Id is already initialized.	No calculation performed	XO_CFI_ORBIT_INIT_FILE_ORBIT_STATUS_ER R	4
ERR	Memory allocation error.	No calculation performed	XO_CFI_ORBIT_INIT_FILE_MEMORY_ERR	5
ERR	Could not detect input files.	No calculation performed	XO_CFI_ORBIT_INIT_FILE_INPUT_FILES_ERR	6
ERR	Error reading OSF.	No calculation performed	XO_CFI_ORBIT_INIT_FILE_WRONG_OSF_FILE_FORMAT_ERR	7
ERR	Wrong time on input.	No calculation performed	XO_CFI_ORBIT_INIT_FILE_TIME_INPUT_INCORR_ERR	8
ERR	Error while processing DORIS file.	No calculation performed	XO_CFI_ORBIT_INIT_FILE_DORIS_INIT_ERR	9
ERR	Time Conversion Error.	No calculation performed	XO_CFI_ORBIT_INIT_FILE_TIME_CONVERSION_ERR	10
ERR	Error reading input files.	No calculation performed	XO_CFI_ORBIT_INIT_FILE_READ_FILES_ERR	11
ERR	No data read within the input range.	No calculation performed	XO_CFI_ORBIT_INIT_FILE_NO_ENOUGH_DATA_ERR	12
ERR	Error while computing ANX data for the state vectors	No calculation performed	XO_CFI_ORBIT_INIT_FILE_INTERPOL_INIT_ANX_ERR	13
ERR	Error computing the orbit number for every state vector	No calculation performed	XO_CFI_ORBIT_INIT_FILE_CALC_ORBIT_ERR	14

ERR	Propagation cannot be initialised	No calculation performed	XO_CFI_ORBIT_INIT_FILE_PROPAG_INIT_ERR	15
ERR	Interpolation cannot be initialised	No calculation performed	XO_ORBIT_INIT_FILE_INTERPOL_INIT_ERR	16
WARN	Warnings while computing ANX data	Calculation performed.	XO_CFI_ORBIT_INIT_FILE_INTERPOL_INIT_ANX_WARN	17
WARN	Warnings during DORIS initialization	Calculation performed.	XO_CFI_ORBIT_INIT_FILE_DORIS_INIT_WARN	18
WARN	Warnings while reading the input file list	Calculation performed.	XO_CFI_ORBIT_INIT_FILE_READ_FILES_WARN	19
WARN	Input time correlations not compatible with input file(s) time correlations	Calculation performed.	XO_CFI_ORBIT_INIT_FILE_WRONG_TIME_CORRELATIONS_WARN	20
WARN	Overriding file time correlations	Calculation performed.	XO_CFI_ORBIT_INIT_FILE_FILE_TIME_CORR_OVERRIDE_WARN	21
ERR	Error performing time conversion with reference	No calculation performed	XO_CFI_ORBIT_INIT_FILE_TIME_CONVERSION_WITH_REF_ERR	22
WARN	There is a discontinuity between overlapped files. Found in discarded OSV %s.	Calculation performed A message informs the user that there is a discontinuity between the input files.	XO_CFI_ORBIT_INIT_FILE_DISCARDED_OSV_WARN	23
WARN	The TLE SGP4 propagator will be used for the geostationary orbit	Calculation performed. When using calling xo_osv_compute, the OSV will be propagated with the SGP4 algorithm.	XO_CFI_ORBIT_INIT_FILE_GEO_AND_TLE_WARN	24
ERR	Cannot use XO_ORBIT_INIT_TLE_MODE for orbital periods >200 and <250 min	No calculation performed	XO_CFI_ORBIT_INIT_FILE_TLE_AUTO_ERR	25
ERR	Error reading OSF File to change Orbit_Id.	No calculation performed	XO_CFI_ORBIT_INIT_FILE_READ_OSF_ERR	26
ERR	Error changing Orbit Id orbit numbers	No calculation performed	XO_CFI_ORBIT_INIT_FILE_ORBIT_ID_CHANGE_ERR	27
WARN	Input DORIS files are not consistent with the Satellite ID.	Calculation performed.	XO_CFI_ORBIT_INIT_FILE_DORIS_TYPE_WARN	32
WARN	No Orbit Number specified in DORIS file. Assuming orbit=1 for the 1st OSV	Calculation performed	XO_CFI_ORBIT_INIT_FILE_DEFAULT_ORBIT_WARN	33
ERR	Error analysing orbit diagnostics	No calculation performed	XO_CFI_ORBIT_INIT_FILE_ANALYZE_ORBIT_DIAGNOSTI	34

			CS_ERR	
WARN	Gaps detected. All the OSVs that follows the first gap are discarded.	Calculation performed	XO_CFI_ORBIT_INIT_FILE_GAPS_WARN	35
WARN	Time going back or repeated OSVs detected. Those OSVs (going bank or repeated) are discarded.	Calculation performed	XO_CFI_ORBIT_INIT_FILE_DUPLICATED_GOING_BACK_OSV_WARN	36
WARN	Inconsistent orbit number detected. The orbit number is recomputed.	Calculation performed	XO_CFI_ORBIT_INIT_FILE_INCONSISTENT_ORBIT_NUM_WARN	37
WARN	Time gap between OSVs larger than 300s detected (may lead to insufficient accuracy in case of highly eccentric MEO)	Calculation performed	XO_CFI_ORBIT_INIT_FILE_MEO_GAP_WARN	38

7.6 xo_orbit_init_file_precise

7.6.1 Overview

The **xo_orbit_init_file_precise** function is used for initializing the orbit calculations in the same way as **xo_orbit_inti_file**, but in this case precise propagation will be used in state vector propagation. One of these orbit files can be used:

- One or more FOS Predicted ascending node cartesian state vectors file. In case multiple files are used, the files should be time ordered and the gap between them (i.e. time difference between the last vector of nth file and the first vector of the nth+1 file) should be less than two orbital periods.
- One FOS Predicted Orbit File plus a DORIS Navigator unconsolidated level-0 products file.
- One or more FOS Restituted orbit files.
- One or more DORIS Navigator files. Three formats of DORIS Navigator files are supported and automatically detected: Cryosat, Sentinel 3 and Jason CS (see [D_H_SUM] for further details). Notice that when using DORIS Navigator files the orbit is initialized based only on the navigation data in the NAV_T (ITRF) packets, and the absolute orbit number provided in NAV_G (Geodetic) packets if available in the same input file. NAV_G (Geodetic) packets are expected only for Sentinel 3 and Jason CS DORIS Navigation files and, as referred in [D_H_SUM], in case NAV_G (Geodetic) packets are not available the orbit number will be set to 1 at the first OSV and increased at each ANX. To adjust the orbit number, use the function xd_orbit_id_change as described in section 7.60.
- One or more DORIS Predicted files.
- One or more DORIS Preliminary files.

The format of the above files is described in [FORMATS].

In order to read files, xo_orbit_init_file_precise function internally uses Data Handling functions. Please refer to [D_H_SUM], in particular sections 4.2 and 4.3, for further details.

Before calling this function it is required to initialise the time correlations, using either **xl_time_ref_init, xl_time_ref_init_file** or **xl_time_id_init** EO LIB functions (see [LIB_SUM]).

The user can select the time interval to be used from the input file(s) using three different ways:

Table 21: User requested time range in xo_orbit_init_file_precise

time_mode (see 7.33.4)	input parameter	requested start time (t_req_start)	requested stop time (t_req_stop)
XL_SEL_TIME	time0 / time1	time0	time1
XL_SEL_ORBIT	orbit0 / orbit1	tANX(orbit0)	tANX(orbit1)
XL_SEL_FILE	-	first state vector in the file(s)	last state vector in the file(s)

The validity start and stop times of the initialization (**val_time0** and **val_time1** output parameters) represents the allowed time window for orbit calculation. The following table shows the validity time interval for the different input files:

Table 22: Validity periods for xo_orbit_init_file_precise

Input file type	val_time0	val_time1

Any	time of the first state vector	Infinity

A complete calling sequence of the orbit calculation procedure is presented in section 4.2.

See Recommendations on Orbit Files Usage in 67

7.6.2 Calling interface

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

```
#include <explorer_orbit.h>
{
  xl_model_id time_id = {NULL};
  xl_time_id time_id = {NULL};
  xo_orbit_id orbit_id = {NULL};
  long sat_id, orbit_file_mode, n_files, time_mode;
  long time_ref, orbit0, orbit1;
  char **input_files;
  xo_propag_precise_config precise_conf;
  double time0, time1, val_time0, val_time1;
  long status, ierr[XO_NUM_ERR_ORBIT_INIT_FILE];

  status = xo_orbit_init_file_precise (&sat_id, &model_id,
                                       &time_id,
                                       &orbit_file_mode, &n_files,
                                       input_files,
                                       &time_mode, &time_ref,
                                       &time0, &time1, &orbit0, &orbit1,
                                       &precise_conf,
                                        &val_time0, &val_time1,
                                        &orbit_id, ierr);
}
```

7.6.3 Input parameters

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

Table 23: Input parameters of xo_orbit_init_file_precise function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
model_id	xo_model_id*	-	Model ID	-	-

time_id	xo_time_id*	-	Structure that contains the time correlations	-	-
orbit_file_mode	long*	-	Flag that indicates the type of the input orbit file. There exists the possibility of detecting automatically the type of the files using the value XO_ORBIT_INIT_AUTO.	-	XO_ORBIT_INIT_AUTO XO_ORBIT_INIT_POF_PRECISE_MODE XO_ORBIT_INIT_ROF_PRECISE_MODE XO_ORBIT_INIT_DORIS_PRECISE_MODE XO_ORBIT_INIT_OEF_POF_PRECISE_MODE XO_ORBIT_INIT_POF_N_DORIS_PRECISE_MO DE
n_files	long	-	Number of input files	-	>=1
input_files	char**	-	Vector of orbit files	-	-
time_init_mode	long*	-	Flag for selecting the time range of the initialisation. For TLE files, the whole file is always selected (this flag and the parameters time0/time1, orbit0/orbit1 are dummies)	-	Select either: · XO_SEL_FILE · XO_SEL_ORBIT · XO_SEL_TIME For DORIS Navigator files, XO_SEL_ORBIT is not allowed
time_ref	long*	-	Time reference ID	-	Complete When using DORIS Navigator files and time_mode is XO_SEL_TIME, only XL_TIME_UTC is allowed.
time0	double*	-	Start time. See section 7.33.1. Used only if: · time_init_mode=XO_SEL_TIME; otherwise NULL can be passed	Decimal days (Processing format)	[-18262.0,36524.0]
time1	double*	-	Stop time. Used only if: · time_init_mode=XO_SEL_TIME; otherwise NULL can be passed	Decimal days (Processing format)	[-18262.0,36524.0]

orbit0	long*	-	Absolute orbit number of the start orbit. Used only if: · time_init_mode=XO_SEL_ORBIT; otherwise NULL can be passed	-	-
orbit1	long*		Absolute orbit number of the stop orbit. Used only if: · time_init_mode=XO_SEL_ORBIT; otherwise NULL can be passed	-	-
precise_conf	xo_propag_precise_config*	-	Configuration parameters for precise propagator.	-	-

For precise_conf, the same rules than in xo_orbit_cart_init_precise apply.

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

- Satellite ID: sat_id. See [GEN_SUM].
- Orbit init mode: orbit_init_mode. Current document, section 6.2.
- Time mode: time_init_mode. See [GEN_SUM].
- Time reference ID: time_ref. See [GEN_SUM].

7.6.4 Output parameters

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

Table 24: Output parameters of xo_orbit_init_file_precise function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_init_file_precise	long	-	Main status flag	-	-1, 0, +1
val_time0	double*	-	Validity start time of the initialization	Decimal days (Processing format)	See 7.33.1
val_time1	double*	-	Validity stop time of the initialization	Decimal days (Processing format)	see 7.33.1
orbit_id	xo_orbit_id*	-	Structure that contains the orbit initialization data	-	-
ierr[XO_NUM_ERR_ORBIT_INIT_FILE_PRECISE]	long	all	Status vector	-	-

7.6.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_orbit_init_file_precise** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_orbit_init_file_precise** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 25: Error messages of xo_orbit_init_file_precise function

Error type	Error message	Cause and impact	Error Code	Error No
ERR	File type not allowed for precise propagator initialisation	No calculation performed	XO_CFI_ORBIT_INIT_FILE_PRECISE_NOT_ALLOWED_FILE_TYPE_ERR	0
ERR	Error initialising orbit.	No calculation performed	XO_CFI_ORBIT_INIT_FILE_PRECISE_INIT_FILE_ERR	1
ERR	Error initialising propagator.	No calculation performed	XO_CFI_ORBIT_INIT_FILE_PRECISE_PROPAG_INIT_ERR	2
ERR	Could not detect input files.	No calculation performed	XO_CFI_ORBIT_INIT_FILE_PRECISE_INPUT_FILES_ERR	3
ERR	Error in precise propagator input parameters	No calculation performed	XO_CFI_ORBIT_INIT_FILE_PRECISE_PRECISE_PARAMS_ERR	4

7.7 xo_orbit_id_init

7.7.1 Overview

The **xo_orbit_id_init** function is used for initializing the orbit calculations using a set of data structures that have been read for one of these orbit files:

- One or more orbit files (Predicted, Restituted Orbit Files or generic OSV list. In the last case, propagator is not initialized).
- One Orbit Scenario File providing orbital changes.
- One or more DORIS Navigator files. Three formats of DORIS Navigator files are supported and automatically detected: Cryosat, Sentinel 3 and Jason CS (see [D_H_SUM] for further details). Notice that when using DORIS Navigator files the orbit is initialized based only on the navigation data in the NAV_T (ITRF) packets, and the absolute orbit number provided in NAV_G (Geodetic) packets if available in the same input file. NAV_G (Geodetic) packets are expected only for Sentinel 3 and Jason CS DORIS Navigation files and, as referred in [D_H_SUM], in case NAV_G (Geodetic) packets are not available the orbit number will be set to 1 at the first OSV and increased at each ANX. To adjust the orbit number, use the function xd_orbit_id_change as described in section 7.60.
- One or more SP3 files
- TLE files. xo_orbit_id_init uses only those TLE entries with the default NORAD satellite designator correspondent to the input satellite Id. The correspondence table between satellite ids and default NORAD satellite designators is given in table 224 from the section 9.17, [D_H_SUM]. In case the input file is not compliant with this correspondence, the default designator can be changed by using function xl_set_tle_sat_data, section 7.48 [LIB_SUM].
- One or more OEM files. Of the fields read with the funcion xd_read_oem (see section 9.21 of [D_H_SUM]), the ones that are relevant for the orbit initialisation are: REF_FRAME, TIME_SYSTEM, Epoch, X, Y, Z, X_DOT, Y_DOT, Z_DOT.

Note: for Predicted, Restituted, DORIS files or OEM files, the *change_data* struct of *xo_orbit_id_init_data* can be used. The field *change_data* is used only if orbit init mode is set to XO_ORBIT_INIT_(POF/ROF/DORIS/OEM)_ORBNUM_ADJ_MODE and the corresponding POF/ROF/DORIS file is used as input. If these orbit modes are introduced, the orbit number of the state vectors will be updated with the information in *change_data* structure, as per function **xo_orbit_id_change** (see section 7.60).

This function provide an alternative to initialize the orbit_id that is completely equivalent to **xo_orbit_init_file** with the difference that this function does not need to read the input files, so that the runtime of the function is improved (See section 7.5 for details about the orbit_id initialization)

This function is specially useful when the input time_id is initialized with the same set of input files. In this case it is better to do the following calling sequence, so that input files are read once:

- Read input file (EO_DATA_HANDLING functions)
- Initialize time_id with **xl_time_id_init.**
- Initialize the orbit_id with **xo_orbit_id_init.**
- Clean the read data.

Before calling this function it is required to initialise the time correlations, using either **xl_time_ref_init, xl_time_ref_init_file** or **xl_time_id_init** EO LIB functions (see [LIB_SUM]).

The user can select the time interval to be used from the input file(s) using three different ways:

Table 26: User requested time range in xo_orbit_id_init

time_mode (see 7.33.4)	input parameter	requested start time (t_req_start)	requested stop time (t_req_stop)
XL_SEL_TIME	time0 / time1	time0	time1
XL_SEL_ORBIT	orbit0 / orbit1	tANX(orbit0)	tANX(orbit1)
XL_SEL_FILE	-	first state vector in the file(s)	last state vector in the file(s)

The validity start and stop times of the initialization (**val_time0** and **val_time1** output parameters) represents the allowed time window for orbit calculation. The following table shows the validity time interval for the different input files:

Table 27: Validity periods for xo_orbit_id_init

Input data type	val_time0	val_time1
Orbit data from Orbit Scenario	ANX Time of the first orbital change	Infinity
List of orbital state vectors	time of the first state vector	Time of the last state vector

A complete calling sequence of the orbit calculation procedure is presented in section 4.2.

See Recommendations on Orbit Files Usage in section 7.5.1.1

The function computes diagnostics of OSV records for OEM/SP3/Doris/orbit files. The behaviour at orbit initialization is defined in the table 17, see section 7.5.1.1.

7.7.2 Calling interface

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

```
#include <explorer_orbit.h>
{
  xl_time_id time_id = {NULL};
  xl_model_id model_id = {NULL};
  xo_orbit_id orbit_id = {NULL};
  long sat_id, orbit_file_mode, time_mode;
  long time_ref, orbit0, orbit1;
  xo_orbit_id_init_data orbit_data;
  double time0, time1, val_time0, val_time1;
  long status, ierr[XO_NUM_ERR_ORBIT_ID_INIT];

  status = xo_orbit_id_init (&sat_id, &model_id, &time_id,
                             &orbit_file_mode,
                             &orbit_data,
```

```
                                        &time_mode, &time_ref,
                                        &time0, &time1, &orbit0, &orbit1,
                                         &val_time0, &val_time1,
                                         &orbit_id, ierr);
    }
```

7.7.3 Input parameters

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

Table 28: Input parameters of xo_orbit_id_init function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete. For SP3 files, see a description on how to set sat_id value under the present table.
model_id	xo_mo del_id*	-	Model ID	-	-
time_id	xo_time _id*	-	Structure that contains the time correlations	-	-
orbit_file_ mode	long*	-	Flag that indicates the type of the input orbit file data. In case the type of initialization is XO_ORBIT_INIT_USER_OSV_LIST_MODE, the propagator/interpolator is not initialized, so any operation involving these computations will return error.	-	XO_ORBIT_INIT_AUTO XO_ORBIT_INIT_OSF_MODE XO_ORBIT_INIT_POF_MODE XO_ORBIT_INIT_ROF_MODE XO_ORBIT_INIT_DORIS_MODE XO_ORBIT_INIT_OEF_OSF_MODE XO_ORBIT_INIT_OEF_POF_MODE XO_ORBIT_INIT_USER_OSV_LIST_MODE XO_ORBIT_INIT_SP3_MODE XO_ORBIT_INIT_POF_ORBNUM_ADJ_MODE XO_ORBIT_INIT_ROF_ORBNUM_ADJ_MODE XO_ORBIT_INIT_DORIS_ORBNUM_ADJ_MODE XO_ORBIT_INIT_OEM_MODE XO_ORBIT_INIT_OEM_

					ORBNUM_ADJ_MODE
orbit_data	xo_orbit_id_init_data*	-	Vector of orbit files	-	-
time_init_mode	long*	-	Flag for selecting the time range of the initialisation.	-	Select either: · XO_SEL_FILE · XO_SEL_ORBIT · XO_SEL_TIME For DORIS Navigator files and SP3 files, XO_SEL_ORBIT is not allowed
time_ref	long*	-	Time reference ID	-	Complete When using DORIS Navigator files and time_mode is XO_SEL_TIME, only XL_TIME_UTC is allowed.
time0	double*	-	Start time. See section 7.33.1. Used only if: time_init_mode=XO_SEL_TIME; otherwise NULL can be passed	Decimal days (Processing format)	[-18262.0,36524.0]
time1	double*	-	Stop time. Used only if: time_init_mode=XO_SEL_TIME; otherwise NULL can be passed	Decimal days (Processing format)	[-18262.0,36524.0]
orbit0	long*	-	Absolute orbit number of the start orbit. Used only if: time_init_mode=XO_SEL_ORBIT; otherwise NULL can be passed	-	-
orbit1	long*		Absolute orbit number of the stop orbit. Used only if: time_init_mode=XO_SEL_ORBIT; otherwise NULL can be passed	-	-

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

- Satellite ID: sat_id. See [GEN_SUM].
- Orbit init mode: orbit_init_mode. Current document, section 6.2.
- Time mode: time_init_mode. See [GEN_SUM].
- Time reference ID: time_ref. See [GEN_SUM].

Note: For SP3 files, the SP3 Id has to be defined for the input sat_id (as defined in section "*Extended Standard Product 3 Orbit File (SP3-c)*" in [D_H_SUM]). The state vectors to be selected are those with the

SP3 Id defined for the input satellite Id. Note that the SP3 Id can be defined also by the user with the function xl_set_sp3_sat_data (See [LIB_SUM]).

7.7.4 Output parameters

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

Table 29: Output parameters of xo_orbit_id_init function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_id_init	long	-	Main status flag	-	-1, 0, +1
val_time0	double*	-	Validity start time of the initialization	Decimal days (Processing format)	See 7.33.1
val_time1	double*	-	Validity stop time of the initialization	Decimal days (Processing format)	See 7.33.1
orbit_id	xo_orbit_id*	-	Structure that contains the orbit initialization data	-	-
ierr[XO_NUM_ERR_ORBIT_ID_INIT]	long	all	Status vector	-	-

7.7.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_orbit_id_init** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_orbit_id_init** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 30: Error messages of xo_orbit_id_init function

Error type	Error message	Cause and impact	Error Code	Error No
ERR	Wrong satellite flag	No calculation performed	XO_CFI_ORBIT_ID_INIT_SAT_ERR	0
ERR	Geostationary satellite currently not supported for this function.	No calculation performed	XO_CFI_ORBIT_ID_INIT_GEO_SAT_ERR	1
ERR	Wrong input orbit file mode	No calculation performed	XO_CFI_ORBIT_ID_INIT_FLAG_ERR	2
ERR	The Time Id was not initialized	No calculation performed	XO_CFI_ORBIT_ID_INIT_TIME_STATUS_ERR	3

ERR	The Orbit Id is already initialized	No calculation performed	XO_CFI_ORBIT_ID_INIT_ORBIT_STATUS_ERR	4
ERR	Memory allocation error.	No calculation performed	XO_CFI_ORBIT_ID_INIT_MEMORY_ERR	5
ERR	No data in the input structures	No calculation performed	XO_CFI_ORBIT_ID_INIT_NO_DATA_ERR	6
ERR	Orbit model is not correct	No calculation performed	XO_CFI_ORBIT_ID_INIT_WRONG_ORBIT_MODEL_ERR	7
ERR	input data structure contains data for different file types	No calculation performed	XO_CFI_ORBIT_ID_INIT_INCONSISTENT_FILES_ERR	8
ERR	Error loading input xo_orbit_id_init_data structure	No calculation performed	XO_CFI_ORBIT_ID_INIT_LOAD_FILES_ERR	9
ERR	Wrong time on input.	No calculation performed	XO_CFI_ORBIT_ID_INIT_TIME_INPUT_INCORR_ERR	10
ERR	Error performing time conversion with reference.	No calculation performed	XO_CFI_ORBIT_ID_INIT_TIME_CONVERSION_WITH_REF_ERR	11
ERR	No data read within the input range	No calculation performed	XO_CFI_ORBIT_ID_INIT_NO_ENOUGH_DATA_ERR	12
ERR	Error while computing ANX data for the state vectors	No calculation performed	XO_CFI_ORBIT_ID_INIT_INTERPOL_INIT_ANX_ERR	13
ERR	Error computing the orbit number for every state vector	No calculation performed	XO_CFI_ORBIT_ID_INIT_CALC_ORBIT_ERR	14
ERR	Time Conversion Error.	No calculation performed	XO_CFI_ORBIT_ID_INIT_TIME_CONVERSION_ERR	15
ERR	Interpolation cannot be initialised	No calculation performed	XO_CFI_ORBIT_ID_INIT_INTERPOL_INIT_ERR	16
ERR	Propagation cannot be initialised	No calculation performed	XO_CFI_ORBIT_ID_INIT_PROPAG_INIT_ERR	17
WARN	There is a discontinuity between overlapped files. Found in discarded OSV %s.	Calculation performed. A message informs the user that there is a discontinuity between the input files.	XO_CFI_ORBIT_ID_INIT_DISCARDED_OSV_WARN	18

WARN	Only one OSF file is admitted for this initialisation mode	Calculation performed using data from the first OSF inserted in the init_data structure	XO_CFI_ORBIT_ID_INIT_ONLY_FIRST_OSF_WARN	19
WARN	Input time correlations not compatible with input file(s) time correlations.	Calculation performed.	XO_CFI_ORBIT_ID_INIT_WRONG_TIME_CORRELATIONS_WARN	20
WARN	Overriding file time correlations.	Calculation performed	XO_CFI_ORBIT_ID_INIT_FILE_TIME_CORR_OVERRIDE_WARN	21
WARN	Warnings while computing ANX data	No calculation performed	XO_CFI_ORBIT_ID_INIT_INTERPOL_INIT_ANX_WARN	22
ERR	Error changing Orbit Id orbit numbers.	No calculation performed	XO_CFI_ORBIT_ID_INIT_ORBIT_ID_CHANGE_ERR	23
WARN	No Orbit Number specified in DORIS file. Assuming orbit=1 for the 1st OSV	Calculation performed	XO_CFI_ORBIT_ID_INIT_DEFAULT_ORBIT_WARN	24
WARN	Input DORIS files are not consistent with the Satellite ID.	Calculation performed	XO_CFI_ORBIT_ID_INIT_DORIS_TYPE_WARN	25
ERR	Invalid file type. Orbit mode can not be automatically detected.	No calculation performed	XO_CFI_ORBIT_ID_INIT_INVALID_FILE_TYPE_ERR	26
ERR	Only one TLE file is admitted for this initialisation mode	No calculation performed	XO_CFI_ORBIT_ID_INIT_ONLY_FIRST_TLE_WARN	32
ERR	Cannot use XO_ORBIT_INIT_TLE_MODE for orbital periods >200 and <250 min	No calculation performed	XO_CFI_ORBIT_ID_INIT_TLE_AUTO_ERR	33
ERR	Error analysing orbit diagnostics	No calculation performed	XO_CFI_ORBIT_ID_INIT_ANALYZE_ORBIT_DIAGNOSTICS_ERR	34
WARN	Gaps detected. All the OSVs that follows the first gap are discarded.	Calculation performed	XO_CFI_ORBIT_ID_INIT_GAPS_WARN	35
WARN	Time going back or repeated OSVs detected. Those OSVs (going bank or repeated) are discarded.	Calculation performed	XO_CFI_ORBIT_ID_INIT_DUPLICATED_GOING_BACK_OSV_WARN	36
WARN	Inconsistent orbit number	Calculation performed	XO_CFI_ORBIT_ID_INIT	37

	detected. The orbit number is recomputed.		_INCONSISTENT_ORBIT _NUM_WARN	
WARN	Time gap between OSVs larger than 300s detected (may lead to insufficient accuracy in case of highly eccentric MEO)	Calculation performed	XO_CFI_ORBIT_ID_INIT _MEO_GAP_WARN	38

7.8 xo_orbit_init_geo

7.8.1 Overview

This software initializes the orbit data of a geostationary orbit using as input the geodetic coordinates. The only input geodetic coordinate taken into account in initialization is the geocentric longitude; the latitude is set to 0. degrees and the altitude is set to 35786 km independently of the inputs.

The validity start and stop times of the initialization (***val_time0*** and ***val_time1*** output parameters) represents the allowed time window for orbit calculations. If the **xo_orbit_init_geo** function is called, this time window starts at 01/01/1950 00:00:00 and ends at 31/12/2099 23:59:59.

Before calling this function it is required to initialise the time correlations, using either **xl_time_ref_init** or **xl_time_ref_init_file** EO_LIB functions (see [LIB_SUM]).

A complete calling sequence of the orbit calculations procedure is presented in section 4.2.

7.8.2 Calling interface

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

```
#include <explorer_orbit.h>
{
  xl_model_id model_id = {NULL};
  xl_time_id time_id = {NULL};
  xo_orbit_id orbit_id = {NULL};
  long sat_id;
  xo_geo_orbit_init_data geo_orbit_init_data;
  double val_time0, val_time1;
  long status, ierr[XO_NUM_ERR_ORBIT_INIT_GEO];

  status = xo_orbit_init_geo(&sat_id, &model_id, &time_id,
                             &geo_orbit_init_data,
                               &val_time0, &val_time1,
                               &orbit_id, ierr);
}
```

7.8.3 Input parameters

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

Table 31: Input parameters of xo_orbit_init_geo function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
model_id	xl_model	-	Model ID	-	-

	_id*				
time_id	xl_time_id*	-	Structure that contains the time correlations	-	-
geo_orbit_info	xo_geo_orbit_info*	-	Input geodetic coordinates	Longitude, latitude → degress Altitude → meters	0 ≤ lon < 360. latitude and altitude are dummy parameters. They are set internally to default values (see section 7.8.1)

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

- Satellite ID: sat_id. See [GEN_SUM].

7.8.4 Output parameters

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

Table 32: Output parameters of xo_orbit_init_geo function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_init_geo	long	-	Main status flag	-	-1, 0, +1
val_time0	double*	-	Validity start time of the initialization	Decimal days (Processing format)	[-18262.0,36524.0]
val_time1	double*	-	Validity stop time of the initialization	Decimal days (Processing format)	[-18262.0,36524.0]
orbit_id	xo_orbit_id*	-	Structure that contains the orbit initialization.	-	-
ierr[XO_NUM_ERR_ORBIT_INIT_GEO]	long	all	Status vector	-	-

7.8.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_orbit_init_geo** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_orbit_init_geo** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 33: Error messages of xo_orbit_init_geo function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Wrong Satellite Id.	No calculation performed	XO_CFI_ORBIT_INIT_GEO_SAT_ERR	0
ERR	Wrong longitude: out of range [0, 360.)	No calculation performed	XO_CFI_ORBIT_INIT_GEO_LON_ERR	1
ERR	Wrong geo orbit init type	No calculation performed	XO_CFI_ORBIT_INIT_GEO_TYPE_ERR	2
ERR	Input Time Id. is not initialized.	No calculation performed	XO_CFI_ORBIT_INIT_GEO_TIME_STATUS_ERR	3
ERR	Orbit Id is already initialized.	No calculation performed	XO_CFI_ORBIT_INIT_GEO_STATUS_ERR	4
ERR	Memory allocation error.	No calculation performed	XO_CFI_ORBIT_INIT_GEO_MEMORY_ERR	5
ERR	Error converting geodetic coordinates to cartesian coordinates	No calculation performed	XO_CFI_ORBIT_INIT_GEO_GEOD_TO_CART_ERR	6
ERR	Error initializing propagator	No calculation performed	XO_CFI_ORBIT_INIT_GEO_PROPAG_INIT_ERR	7

7.9 xo_orbit_close

7.9.1 Overview

The **xo_orbit_close** function is used to free the memory allocated by the other orbit initialization routines, and it must be called after using them.

A complete calling sequence of the propagation procedure is presented in section 4.2.

7.9.2 Calling interface

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

```
#include <explorer_orbit.h>
{
  xo_orbit_id orbit_id = {NULL};
  long ierr[XO_NUM_ERR_ORBIT_CLOSE]
  long status;

  status = xo_orbit_close (&orbit_id, ierr);
}
```

7.9.3 Input parameters

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

Table 34: Input parameters of xo_orbit_close function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id *	-	Structure that contains the orbit initialization	-	-

7.9.4 Output parameters

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

Table 35: Output parameters of xo_orbit_close function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
ierr[XO_NUM_ERR_ORBIT_CLOSE]	long	all	Status vector	-	-
xo_orbit_close	long	-	Main status flag	-	-1, 0, +1

7.9.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_orbit_close** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_orbit_close** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 36: Error messages of xo_orbit_close function

Error type	Error message	Cause and impact	Error Code	Error No
ERR	Could not close the Orbit Id.	The Orbit Id. was not closed.	XO_CFI_ORBIT_CLOSE_ WRONG_ID_ERR	0

7.10 xo_orbit_get_osv

7.10.1 Overview

The **xo_orbit_get_osv** CFI function returns a data structure containing the list of state vectors used for the initialisation of an orbit_id. This function only can be called if the orbit_id was initialized with orbital state vectors (i.e., with **xo_orbit_cart_init** or with **xo_orbit_init_file** and a file containing a list of state vectors such as predicted orbit file, a restituted orbit file...)

7.10.2 Calling interface

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

```
#include <explorer_orbit.h>
{
        xo_orbit_id orbit_id;
        long num_rec;
        xo_osv_rec* data;
        long status;
        status = xo_orbit_get_osv(&orbit_id, &num_rec, &data);
}
```

7.10.3 Input parameters

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

Table 37: Input parameters of xo_orbit_get_osv function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_id	xo_orbit_id*	-	Structure for orbit initiaization	-	-

7.10.4 Output parameters

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

Table 38: Output parameters of xo_orbit_get_osv function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_get_osv	long	-	Status flag	-	-
num_rec	long	-	Number of records in the data array	-	-
data	xo_osv_rec	all	Dinamic array with the state vectors	-	-

The data structure xo_osv_rec can be seen in Table 3.

Note: The output *data* array is a pointer, not a static array. The memory for this dynamic array is allocated within the CFI function. So the user will only have to declare that pointer but not to allocate memory for it. However, once the function has returned without error, the user will have the responsibility of freeing the memory when it is not being used any more. For freeing the memory just call to (in a C program):

```
free(data);
```

7.10.5 Warnings and errors

This function does not return any error/warning code. Only the status of the function indicates if the execution was correct or not.

The possible causes of error are:

- The orbit_id was not initialised.
- The orbit_id was initialised with orbital changes, instead of state vectors.

7.11 xo_orbit_set_osv

7.11.1 Overview

The **xo_orbit_set_osv** CFI function changes the list of state vectors used for the initialisation within an orbit_id. This function only can be called if the orbit_id was initialized with orbital state vectors (i.e., with **xo_orbit_cart_init** or with **xo_orbit_init_file** and a file containing a list of state vectors such as predicted orbit file, a restituted orbit file...)

7.11.2 Calling interface

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

```
#include <explorer_orbit.h>
{
        xo_orbit_id orbit_id;
        long num_rec;
        xo_osv_rec* data;
        long status;
        status =    xo_orbit_set_osv(&orbit_id, &num_rec, data);
}
```

7.11.3 Input parameters

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

Table 39: Input parameters of xo_orbit_set_osv function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure for orbit initialization (input / output parameter)	-	-
num_rec	long	-	Number of records in the data array	-	-
data	xo_osv_rec	all	Dynamic array with the state vectors	-	-

7.11.4 Output parameters

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

Table 40: Output parameters of xo_orbit_set_osv function

C name	C type	Array	Description	Unit	Allowed Range

		Element	(Reference)	(Format)	
xo_orbit_set_osv	long	-	Status flag	-	-
orbit_id	xo_orbit_id*	-	Structure for orbit initialization (input / output parameter)	-	-

7.11.5 Warnings and errors

This function does not return any error/warning code. Only the status of the function indicates if the execution was correct or not.

The possible causes of error are:

- The orbit_id was not initialised.
- The orbit_id was initialised with orbital changes, instead of state vectors.

7.12 xo_orbit_get_anx

7.12.1 Overview

When initialising an orbit_id with a list of state vectors that are not in the ANX (restituted orbit file, DORIS Navigator files, SP3 files), the information about the ANX of the orbits of those state vectors are stored in the orbit_id. The **xo_orbit_get_anx** CFI function allows to retrieve that information.

This function only can be called if the orbit_id was initialized with orbital state vectors coming from:

- Restituted orbit file
- DORIS Navigator file(s).
- SP3 files. Note: since SP3 files has no orbit information inside the file itself, the orbit numbers for ANX start at 0.

7.12.2 Calling interface

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

```
#include <explorer_orbit.h>
{
        xo_orbit_id orbit_id;
        long num_rec;
        xo_anx_extra_info* extra_info;
        long status;
        status = xo_orbit_get_anx(&orbit_id, &num_rec, &extra_info);
}
```

7.12.3 Input parameters

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

Table 41: Input parameters of xo_orbit_get_anx function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure for orbit initialization	-	-

7.12.4 Output parameters

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

Table 42: Output parameters of xo_orbit_get_anx function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_get_anx	long	-	Status flag	-	-
num_rec	long	-	Number of records in the data array	-	-
extra_info	xo_anx_extra_info	all	Dinamic array with the ANX information	-	-

The data structure xo_osv_rec can be seen in Table 3.

Note: The output *extra_info* array is a pointer, not a static array. The memory for this dynamic array is allocated within the CFI function. So the user will only have to declare that pointer but not to allocate memory for it. However, once the function has returned without error, the user will have the responsibility of freeing the memory when it is not being used any more. For freeing the memory just call to (in a C program):

```
free(extra_info);
```

7.12.5 Warnings and errors

This function does not return any error/warning code. Only the status of the function indicates if the execution was correct or not.

The possible causes of error are:

- The orbit_id was not initialised.
- The orbit_id was not initialised with the suitable file

7.13 xo_orbit_set_anx

7.13.1 Overview

The **xo_orbit_set_anx** CFI function changes the ANX info that is stored in an orbit_id when this orbit id was initialised with a restituted orbit file, a DORIS Navigator file or a SP3 file.

Three formats of DORIS Navigator files are supported and automatically detected: Cryosat, Sentinel 3 and Jason CS (see [D_H_SUM] for further details). Notice that when using DORIS Navigator files the orbit is initialized based only on the navigation data in the NAV_T (ITRF) packets, and the absolute orbit number provided in NAV_G (Geodetic) packets if available in the same input file. NAV_G (Geodetic) packets are expected only for Sentinel 3 and Jason CS DORIS Navigation files and, as referred in [D_H_SUM], in case NAV_G (Geodetic) packets are not available the orbit number will be set to 1 at the first OSV and increased at each ANX. To adjust the orbit number, use the function xd_orbit_id_change as described in section 7.60.

Note: since SP3 files has no orbit information inside the file itself, the orbit numbers for ANX start at 0.

7.13.2 Calling interface

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

```
#include <explorer_orbit.h>
{
        xo_orbit_id orbit_id;
        long num_rec;
        xo_anx_extra_info* extra_info;
        long status;
        status =    xo_orbit_set_anx(&orbit_id, &num_rec, extra_info);
}
```

7.13.3 Input parameters

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

Table 43: Input parameters of xo_orbit_set_anx function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure for orbit initialization (input / output parameter)	-	-
num_rec	long	-	Number of records in the data array	-	-
extra_info	xo_anx_extra_info	all	Dynamic array with the state vectors	-	-

7.13.4 Output parameters

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

Table 44: Output parameters of xo_orbit_set_anx function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_set_anx	long	-	Status flag	-	-
orbit_id	xo_orbit_id*	-	Structure for orbit initialization (input / output parameter)	-	-

7.13.5 Warnings and errors

This function does not return any error/warning code. Only the status of the function indicates if the execution was correct or not.

The possible causes of error are:

- The orbit_id was not initialised.
- The orbit_id was initialised with orbital changes, instead of state vectors.

7.14 xo_orbit_get_osf_rec

7.14.1 Overview

The **xo_orbit_get_osf_rec** CFI function returns a data structure containing the list of orbital changes used for the initialisation of an orbit_id. This function only can be called if the orbit_id was initialized with orbital changes(i.e., with **xo_orbit_init_def** or with **xo_orbit_init_file** and an orbit scenario file)

7.14.2 Calling interface

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

```
#include <explorer_orbit.h>
{
        xo_orbit_id orbit_id;
        long num_rec;
        xo_osf_records* data;
        long status;
        status =    xo_orbit_get_osf_rec(&orbit_id, &num_rec, &data);
}
```

7.14.3 Input parameters

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

Table 45: Input parameters of xo_orbit_get_osf_rec function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure for orbit initialization	-	-

7.14.4 Output parameters

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

Table 46: Output parameters of xo_orbit_get_osf_rec function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_get_osf_rec	long	-	Status flag	-	-
num_rec	long	-	Number of records in the data array	-	-
data	xo_osf_rec	all	Dinamic array with the orbital changes	-	-

The data structure xo_osf_rec can be seen in Table 3.

Note: The output *data* array is a pointer, not a static array. The memory for this dynamic array is allocated within the CFI function. So the user will only have to declare that pointer but not to allocate memory for it. However, once the function has returned without error, the user will have the responsibility of freeing the memory when it is not being used any more. For freeing the memory just call to (in a C program):

```
free(data);
```

7.14.5 Warnings and errors

This function does not return any error/warning code. Only the status of the function indicates if the execution was correct or not.

The possible causes of error are:

- The orbit_id was not initialised.
- The orbit_id was not initialised with orbital changes.

7.15 xo_orbit_set_osf_rec

7.15.1 Overview

The **xo_orbit_set_osf_rec** CFI function changes the list of orbital changes used for the initialisation within an orbit_id. This function only can be called if the orbit_id was initialized with **xo_orbit_init_def** or with **xo_orbit_init_file** and an orbit scenario file.

7.15.2 Calling interface

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

```
#include <explorer_orbit.h>
{
        xo_orbit_id orbit_id;
        long num_rec;
        xo_osv_rec* data;
        long status;
        status =    xo_orbit_set_osf_rec(&orbit_id, &num_rec, data);
}
```

7.15.3 Input parameters

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

Table 47: Input parameters of xo_orbit_set_osf_rec function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure for orbit initialization (input / output parameter)	-	-
num_rec	long	-	Number of records in the data array	-	-
data	xo_osf_rec	all	Dinamic array with the orbital changes	-	-

7.15.4 Output parameters

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

Table 48: Output parameters of xo_orbit_set_osf_rec function

C name	C type	Array	Description	Unit	Allowed Range

		Element	(Reference)	(Format)	
xo_orbit_set_osf_rec	long	-	Status flag	-	-
orbit_id	xo_orbit_id*	-	Structure for orbit initialization (input / output parameter)	-	-

7.15.5 Warnings and errors

This function does not return any error/warning code. Only the status of the function indicates if the execution was correct or not.

The possible causes of error are:

- The orbit_id was not initialised.
- The orbit_id was not initialised with orbital changes.

7.16 xo_orbit_get_val_time

7.16.1 Overview

The **xo_orbit_get_val_time** CFI function returns the validity period of an orbit_id.

7.16.2 Calling interface

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

```
#include <explorer_orbit.h>
{
        xo_orbit_id orbit_id;
        xo_validity_time val_time;
        long status;
        status =    xo_orbit_get_val_time(&orbit_id, &val_time);
}
```

7.16.3 Input parameters

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

Table 49: Input parameters of xo_orbit_get_val_time function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure for orbit initialization	-	-

7.16.4 Output parameters

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

Table 50: Output parameters of xo_orbit_get_val_time function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_get_val_time	long	-	Status flag	-	-
val_time	xo_validity_time	-	Validity Time structure	-	-

The data structure xo_validity_time can be seen in Table 3.

7.16.5 Warnings and errors

This function does not return any error/warning code. Only the status of the function indicates if the execution was correct or not.

The possible causes of error are:

- The orbit_id was not initialised.

7.17 xo_orbit_set_val_time

7.17.1 Overview

The **xo_orbit_set_val_time** CFI function changes the validity period of an orbit_id.

7.17.2 Calling interface

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

```
#include <explorer_orbit.h>
{
        xo_orbit_id orbit_id;
        xo_validity_time val_time;
        long status;
        status =    xo_orbit_set_val_time(&orbit_id, &val_time);
}
```

7.17.3 Input parameters

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

Table 51: Input parameters of xo_orbit_set_val_time function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure for orbit initialization (input / output parameter)	-	-
val_time	xo_validity_time	-	Validity Time structure	-	-

7.17.4 Output parameters

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

Table 52: Output parameters of xo_orbit_set_val_time function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_set_val_time	long	-	Status flag	-	-
orbit_id	xo_orbit_id*	-	Structure for orbit initialization (input / output parameter)	-	-

7.17.5 Warnings and errors

This function does not return any error/warning code. Only the status of the function indicates if the execution was correct or not.

The possible causes of error are:

- The orbit_id was not initialised.

7.18 xo_orbit_get_precise_propag_config

7.18.1 Overview

The **xo_orbit_get_precise_propag_config** CFI function returns the configuration structure of precise propagation.

7.18.2 Calling interface

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

```
#include <explorer_orbit.h>
{
        xo_orbit_id orbit_id;
        xo_propag_precise_config precise_conf;
        long status;
        status =    xo_orbit_get_precise_propag_config(&orbit_id,
                                                       &precise_conf);
}
```

7.18.3 Input parameters

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

Table 53: Input parameters of xo_orbit_get_precise_propag_config function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure for orbit initialization (input / output parameter)	-	-

7.18.4 Output parameters

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

Table 54: Output parameters of xo_orbit_get_precise_propag_config function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_get_precise_propag_config	long	-	Status flag	-	-
precise_conf	xo_propag_precise_config	-	Precise propagator configuration structure	-	-

7.18.5 Warnings and errors

This function does not return any error/warning code. Only the status of the function indicates if the execution was correct or not.

The possible causes of error are:

- The orbit_id was not initialised.

7.19 xo_orbit_set_precise_propag_config

7.19.1 Overview

The **xo_orbit_set_precise_propag_config** CFI function sets the configuration structure of precise propagation.

7.19.2 Calling interface

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

```
#include <explorer_orbit.h>
{
        xo_orbit_id orbit_id;
        xo_propag_precise_config precise_conf;
        long status;
        status =    xo_orbit_set_precise_propag_config(&orbit_id,
                                                       &precise_conf);
}
```

7.19.3 Input parameters

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

Table 55: Input parameters of xo_orbit_set_precse_propag_config function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure for orbit initial-ization (input / output parameter)	-	-
precise_conf	xo_propag_precise_config	-	Precise propagator configuration structure	-	-

7.19.4 Output parameters

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

Table 56: Output parameters of xo_orbit_set_precse_propag_config function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_set_precise_propag_config	long	-	Status flag	-	-
orbit_id	xo_orbit_id*	-	Structure for orbit initialization (input / output parameter)	-	-

7.19.5 Warnings and errors

This function does not return any error/warning code. Only the status of the function indicates if the execution was correct or not.

The possible causes of error are:

- The orbit_id was not initialised.

7.20 xo_orbit_get_time_id

7.20.1 Overview

The **xo_orbit_get_time_id** CFI function returns the time_id structure used for the orbit_id initialisation.

7.20.2 Calling interface

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

```
#include <explorer_orbit.h>
{
        xo_orbit_id orbit_id;
        xl_time_id time_id;
        time_id =   xo_orbit_get_time_id(&orbit_id);
}
```

7.20.3 Input parameters

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

Table 57: Input parameters of xo_orbit_get_time_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure for orbit initialization	-	-

7.20.4 Output parameters

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

Table 58: Output parameters of xo_orbit_get_time_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_get_time_id	xl_time_id	-	time id used for the orbit_id initialisation	-	-

7.20.5 Warnings and errors

This function does not return any error/warning code. In case of error, an empty time_id is returned (initialised with NULL)

The possible causes of error are:

- The orbit_id was not initialised

7.21 xo_orbit_get_model_id

7.21.1 Overview

The **xo_orbit_get_model_id** CFI function returns the model_id structure used for the orbit_id initialisation.

7.21.2 Calling interface

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

```
#include <explorer_orbit.h>
{
        xo_orbit_id orbit_id = {NULL};
        xl_model_id model_id = {NULL};
        model_id =  xo_orbit_get_model_id(&orbit_id);
}
```

7.21.3 Input parameters

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

Table 59: Input parameters of xo_orbit_get_model_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure for orbit initialization	-	-

7.21.4 Output parameters

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

Table 60: Output parameters of xo_orbit_get_model_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_get_model_id	xl_model_id	-	model id used for the orbit_id initialisation	-	-

7.21.5 Warnings and errors

This function does not return any error/warning code. In case of error, an empty time_id is returned (initialised with NULL)

The possible causes of error are:

- The orbit_id was not initialised.

7.22 xo_orbit_get_osv_compute_validity

7.22.1 Overview

The **xo_orbit_get_osv_compute_validity** CFI function returns the validity time interval where it is possible to compute an state vector using the CFI function **xo_osv_compute**. Out of this interval, the functions would return an error.

The validity interval for using **xo_osv_compute** depends on the type of data used for the orbit initialisation. In general, that interval will be different from the validity of the input orbit_id. More information about the validity interval for **xo_osv_compute** can be found in section 7.33.2.

7.22.2 Calling interface

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

```
#include <explorer_orbit.h>
{
        xo_orbit_id orbit_id;
        xo_validity_time val_time;
        time_id =   xo_orbit_get_osv_compute_validity(&orbit_id,
                                                      &val_time);
}
```

7.22.3 Input parameters

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

Table 61: Input parameters of xo_orbit_get_osv_compute_validity function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure for orbit initialization	-	-

7.22.4 Output parameters

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

Table 62: Output parameters of xo_orbit_get_osv_compute_validity function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_get_osv_compute_validity	xo_validity_time	-	validity time interval for the function **xo_osv_compute**	-	-

The data structure xo_validity_time can be seen in Table 3.

7.22.5 Warnings and errors

This function does not return any error/warning code. Only the status of the function indicates if the execution was correct or not.

The possible causes of error are:

- The orbit_id was not initialised.

7.23 xo_orbit_get_propag_mode

7.23.1 Overview

The **xo_orbit_get_propag_mode** CFI function returns the propagation mode that will be used to propagate the state vector when using the input orbit_id.

7.23.2 Calling interface

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

```
#include <explorer_orbit.h>
{
        xo_orbit_id orbit_id;
        long mode;
        mode = xo_orbit_get_propag_mode(&orbit_id);
}
```

7.23.3 Input parameters

The **xo_orbit_get_propag_mode** CFI function has the following input parameters:=

Table 63: Input parameters of xo_orbit_get_propag_mode function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure for orbit initialization	-	-

7.23.4 Output parameters

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

Table 64: Output parameters of xo_orbit_get_propag_mode function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_get_propag_mode	long	-	propagation mode. -1 if the orbit_id is not initialised for propagation.	-	-

7.23.5 Warnings and errors

This function does not return any error/warning code. If the orbit_id is not initialised or it is not initialised with propagation data, then the returned mode is -1.

7.24 xo_orbit_get_interpol_mode

7.24.1 Overview

The **xo_orbit_get_interpol_mode** CFI function returns the interpolation mode that will be used to interpolate the state vector when using the input orbit_id.

7.24.2 Calling interface

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

```
#include <explorer_orbit.h>
{
        xo_orbit_id orbit_id;
        long mode;
        mode = xo_orbit_get_interpol_mode(&orbit_id);
}
```

7.24.3 Input parameters

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

Table 65: Input parameters of xo_orbit_get_interpol_mode function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure for orbit initialization	-	-

7.24.4 Output parameters

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

Table 66: Output parameters of xo_orbit_get_interpol_mode function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_get_interpol_mode	long	-	propagation mode. -1 if the orbit_id is not initialised for interpolations.	-	-

7.24.5 Warnings and errors

This function does not return any error/warning code. If the orbit_id is not initialised or it is not initialised with interpolation data, then the returned mode is -1.

7.25 xo_orbit_get_propag_config

7.25.1 Overview

The **xo_orbit_get_propag_config** CFI function returns the propagation data that will be used to propagate the state vector when using the input orbit_id.

7.25.2 Calling interface

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

```
#include <explorer_orbit.h>
{
        xo_orbit_id orbit_id;
        long status;
        xo_propag_id_data propag_data;
        status =    xo_orbit_get_propag_config(&orbit_id,
                                               &propag_data);
}
```

7.25.3 Input parameters

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

Table 67: Input parameters of xo_orbit_get_propag_config function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure for orbit initialization	-	-

7.25.4 Output parameters

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

Table 68: Output parameters of xo_orbit_get_propag_config function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_get_propag_config	long	-	status	-	-
propag_data	xo_propag_id_data	-	Configuration data used to launch the propagation	-	-

The data structure xo_propag_id_data can be seen in Table 3.

7.25.5 Warnings and errors

This function does not return any error/warning code. If the orbit_id is not initialised or it is not initialised with propagation data, then the returned status is -1.

7.26 xo_orbit_get_interpol_config

7.26.1 Overview

The **xo_orbit_get_interpol_config** CFI function returns the propagation data that will be used to interpoloate the state vector when using the input orbit_id.

7.26.2 Calling interface

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

```
#include <explorer_orbit.h>
{
        xo_orbit_id orbit_id;
        long status;
        xo_interpol_id_data interpol_data;
        status =    xo_orbit_get_interpol_config(&orbit_id,
                                                 &interpol_data);
}
```

7.26.3 Input parameters

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

Table 69: Input parameters of xo_orbit_get_interpol_config function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure for orbit initialization	-	-

7.26.4 Output parameters

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

Table 70: Output parameters of xo_orbit_get_interpol_config function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_get_propag_config	long	-	status	-	-
interpol_data	xo_interpol_id_data	-	Configuration data used to launch the interpolation	-	-

The data structure xo_interpol_id_data can be seen in Table 3.

7.26.5 Warnings and errors

This function does not return any error/warning code. If the orbit_id is not initialised or it is not initialised with propagation data, then the returned status is -1.

7.27 xo_orbit_get_geo_orbit_info

7.27.1 Overview

The **xo_orbit_get_geo_orbit_info** CFI function returns the geostationary geodetic coordinates from orbit_id initialized with a geostationary satellite.

7.27.2 Calling interface

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

```
#include <explorer_orbit.h>
{
        xo_orbit_id orbit_id;
        long status;
        xo_geo_orbit_info geo_orbit_info;
        status =    xo_orbit_get_geo_orbit_info(&orbit_id,
                                                &geo_orbit_info);
}
```

7.27.3 Input parameters

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

Table 71: Input parameters of xo_orbit_get_geo_orbit_info function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure for orbit initialization	-	-

7.27.4 Output parameters

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

Table 72: Output parameters of xo_orbit_get_geo_orbit_info function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_get_geo_orbit_info	long	-	status	-	-
xo_geo_orbit_info	xo_geo_orbit_info	-	Geodetic coordinates of geostationary satellite	-	-

The data structure xo_geo_orbit_info can be seen in Table 3.

7.27.5 Warnings and errors

This function does not return any error/warning code. If the orbit_id is not initialized or it is not initialized with geostationary satellite, then the returned status is -1.

7.28 xo_orbit_set_geo_orbit_info

7.28.1 Overview

The **xo_orbit_set_geo_orbit_info** CFI function sets the geostationary geodetic coordinates for an orbit_id initialized with a geostationary satellite.

7.28.2 Calling interface

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

```
#include <explorer_orbit.h>
{
        xo_orbit_id orbit_id;
        long status;
        xo_geo_orbit_info geo_orbit_info;
        status =    xo_orbit_set_geo_orbit_info(&orbit_id,
                                                &geo_orbit_info);
}
```

7.28.3 Input parameters

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

Table 73: Input parameters of xo_orbit_set_geo_orbit_info function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure for orbit initialization	-	-
geo_orbit_info	xo_geo_orbit_info*	-	Input geodetic coordinates	Longitude, latitude → degress Altitude → meters	0 ≤ lon < 360. -90. ≤ lat ≤ 90. 0.≤ altitude

7.28.4 Output parameters

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

Table 74: Output parameters of xo_orbit_set_geo_orbit_info function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_set_geo_orbit_info	long	-	status	-	-

The data structure xo_geo_orbit_info can be seen in Table 3.

7.28.5 Warnings and errors

This function does not return any error/warning code. The returned status is -1 in the following cases:

- The orbit id is not initialized.
- The orbit id is not initialized with a geostationary satellite.
- The input parameters are not inside the allowed ranges.

7.29 xo_orbit_id_clone

7.29.1 Overview

The **xo_orbit_id_clone** CFI function copies the input orbit_id structure to the output one.

7.29.2 Calling interface

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

```
#include <explorer_orbit.h>
{
        xo_orbit_id orbit_id_in, xo_orbit_id_out;
        long status;
        status =    xo_orbit_id_clone(&orbit_id_in, &orbit_id_out);
}
```

7.29.3 Input parameters

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

Table 75: Input parameters of xo_orbit_id_clone function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id_in	xo_orbit_id*	-	Structure for orbit initialization	-	-

7.29.4 Output parameters

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

Table 76: Output parameters of xo_orbit_id_clone function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_id_clone	long	-	status	-	-
orbit_id_out	xo_orbit_id	-	Output orbit_id	-	-

7.29.5 Warnings and errors

This function does not return any error/warning code. Only the status of the function indicates if the execution was correct or not.

The possible causes of error are:

- The orbit_id was not initialised.

7.30 xo_run_init

7.30.1 Overview

The **xo_run_init** CFI function adds to the *run Id* the *orbit id.*

7.30.2 Calling interface

The calling interface of the **xo_run_init** CFI function is the following:

```
#include <explorer_orbit.h>
{
  long run_id;
  xo_orbit_id orbit_id = {NULL};
  long ierr[XO_NUM_ERR_RUN_INIT], status;
  status =    xo_run_init (&run_id, &orbit_id,
                                    ierr);
}
```

7.30.3 Input parameters

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

Table 77: Input parameters of xo_run_init function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
run_id	long *	-	Run ID	-	>=0
orbit_id	xo_orbit_id*	-	Structure that contains the orbit data	-	-

7.30.4 Output parameters

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

Table 78: Output parameters of xo_run_init function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_run_init	long	-	Status flag	-	-
run_id	long *	-	Run ID	-	>=0
ierr	long	-	Error vector	-	-

7.30.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_run_init** CFI function after translating the returned extended status flag into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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 extended status flag returned by the **xo_run_init** function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM])

Table 79: Error messages of xo_run_init function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Inputs Id no initialized or incompatible.	No calculation performed	XO_CFI_RUN_INIT_STATUS_ERR	0
ERR	Memory allocation error.	No calculation performed	XO_CFI_RUN_INIT_MEMORY_ERR	1
ERR	Input Ids incompatible with the run_id.	No calculation performed	XO_CFI_RUN_INIT_INCONSISTENCY_ERR	2

7.31 xo_run_get_ids

7.31.1 Overview

The **xo_run_get_ids** CFI function returns the *ids* being used..

7.31.2 Calling interface

The calling interface of the **xo_run_get_ids** CFI function is the following:

```
#include <explorer_orbit.h>
{
  long run_id;
  xo_orbit_id orbit_id = {NULL};
  xo_run_get_ids (&run_id,
                  &orbit_id);
}
```

7.31.3 Input parameters

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

Table 80: Input parameters of xo_run_get_ids function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
run_id	long *	-	Run ID	-	>=0

7.31.4 Output parameters

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

Table 81: Output parameters of xo_run_get_ids function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_run_get_ids	void	-	-	-	-
orbit_id	xo_orbit _id*	-	Structure that contains the orbit data	-	-

7.31.5 Warnings and errors

This function does not return any error/warning code.

7.32 xo_run_close

7.32.1 Overview

The **xo_run_close** CFI function cleans up any memory allocation performed by the initialization functions.

7.32.2 Calling interface

The calling interface of the **xo_run_close** CFI function is the following:

```
#include <explorer_orbit.h>
{
        long run_id;
   xo_run_close (&run_id);
}
```

7.32.3 Input parameters

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

Table 82: Input parameters of xo_run_close function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
run_id	long *	-	Run ID	-	>=0

7.32.4 Output parameters

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

Table 83: Output parameters of xo_run_close function

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

7.32.5 Warnings and errors

This function does not return any error/warning code.

7.33 xo_osv_compute

7.33.1 Overview

The **xo_osv_compute** function computes accurate state vectors for user requested times.

This function needs as input an orbit_id that contains the orbit initialisation data. The behavior of the function depends strongly in the way in which the orbit_id was initialised (see section 7.33.2).

During the computation, some results are stored in the input orbit_id. Those results can be used afterwards for other computations (in **xo_osv_compute** or **xo_osv_compute_extra**). This has to be taken into account when working with **multi-threads programs**. In those cases a different orbit_id has to be used in every thread to avoid problems. The orbit id can be initialised in every thread or copied (this is more efficient) to different orbit_id variables with **xo_orbit_id_clone**.

An OSV can be computed within a validity time range that also depends on input orbit_id variable. This validity time range can be retrieved with the CFI function **xo_orbit_get_osv_compute_validty** (see section 7.22).

For a general description of the initialization routines and how to use them in conjunction to the **xo_osv_compute** function, see section 4.1.2.

7.33.2 Computation methods (Propagation/interpolation)

The methods used to compute the state vector depend on the way in which the orbit was initialised. Note that the computation method is selected automatically by the routine. The Table 84 summarises the dependency of the computation method with the orbit initialisation. The first column indicates the orbit mode which is a parameter that can be get with the CFI function **xo_orbit_get_mode**. The methods in the second column are described in detail in the following subsections.

Table 84: OSV computation methods

Orbit initialisation mode	OSV computation method
XO_ORBIT_INIT_ORBIT_CHANGE_MODE Initialised with **xo_orbit_init_def**	Mean Keplerian Propagation Model with "AUTO" and "DOUBLE" modes (see section 7.33.2.1 and Table 85)
XO_ORBIT_INIT_OSF_MODE XO_ORBIT_INIT_OEF_OSF_MODE Initialised with **xo_orbit_init_file** with an OSF (or an OEF but reading the list of orbital changes in the data block of the file)	
XO_ORBIT_INIT_POF_MODE XO_ORBIT_INIT_OEF_POF_MODE Initialised with **xo_orbit_init_file** with a POF (or an OEF but reading the list of State Vectors in the data block of the file)	
XO_ORBIT_INIT_STATE_VECTOR_MODE Initialised with **xo_orbit_cart_init**	Mean Keplerian Propagation Model (see section 7.33.2.1 and Table 85)

XO_ORBIT_INIT_POF_N_DORIS_MODE: Initialised with **xo_orbit_init_file** with a POF and a DORIS file	
XO_ORBIT_INIT_TLE_MODE Initialised with **xo_orbit_init_file** with a TLE file	TLE propagation mode with “AUTO” mode (see section 7.33.2.1 and Table 85)
XO_ORBIT_INIT_TLE_SGP4_MODE Initialised with **xo_orbit_init_file** with a TLE file	TLE propagation mode with “SGP4” mode (see section 7.33.2.1 and Table 85)
XO_ORBIT_INIT_TLE_SDP4_MODE Initialised with **xo_orbit_init_file** with a TLE file	TLE propagation mode with “SDP4” mode (see section 7.33.2.1 and Table 85)
XO_ORBIT_INIT_POF_PRECISE_MODE XO_ORBIT_INIT_OEF_POF_PRECISE_MODE Initialised with **xo_orbit_init_file_precise** with a POF (or an OEF but reading the list of State Vectors in the data block of the file)	Numerical propagator with “AUTO” mode (see section 7.33.2.1 and Table 85)
XO_ORBIT_INIT_ROF_PRECISE_MODE Initialised with **xo_orbit_init_file_precise** with a ROF file	
XO_ORBIT_INIT_DORIS_PRECISE_MODE Initialised with **xo_orbit_init_file_precise** with a DORIS Navigator file	
XO_ORBIT_INIT_STATE_VECTOR_PRECISE_MODE Initialised with **xo_orbit_cart_init_precise**	Numerical propagator (see section 7.33.2.1 and Table 85)
XO_ORBIT_INIT_POF_N_DORIS_PRECISE_MODE Initialised with **xo_orbit_init_file_precise** with a POF and a DORIS Navigator file	
XO_ORBIT_INIT_ROF_MODE Initialised with **xo_orbit_init_file** with a ROF file	Interpolation (see section 7.33.2.2)
XO_ORBIT_INIT_DORIS_MODE Initialised with **xo_orbit_init_file** with a DORIS Navigator file	
XO_ORBIT_INIT_SP3_MODE Initialised with **xo_orbit_init_file** with a SP3 file	
XO_ORBIT_INIT_OEM_MODE Initialised with **xo_orbit_init_file** with OEM file	
XO_ORBIT_INIT_GEO_LON_ALT_MODE Initialized with **xo_orbit_init_geo**	No propagation is performed. The state vector corresponding to orbit initialization values is returned

Note: Orbit Event File is deprecated, only supported for CRYOSAT mission

7.33.2.1 Propagation methods

For the time being, the following propagation models are supported:

- *Mean Keplerian model*. It implies the use of a formulation for the time rates of change for the different mean Kepler elements as functions of a given initial set of mean Kepler elements. Using the above time rates of change, the mean orbital elements can be propagated forward or backward in time by extrapolating the individual time slopes of the superimposed secular and long-periodic perturbations functions. As the long periodic variations have typically periods on the order of months, a near-linear time slope for prediction intervals of many orbits is warranted.
- *TLE model*. This model propagates the state vector using the NORAD "two line elements" (TLE) and the SGP4 or SDP4 propagation algorithm. This theory was designed for near Earth Satellites (nodal period less than 225 minutes). The SGP4 theory uses an Earth gravitational field through zonal terms J2, J3 and J4 and a power density function for the atmospheric model (assuming a non-rotating spherical model).

  The algorithm to be used in the propagation is selected based on the input Orbit Init Mode:

  - XO_ORBIT_INIT_TLE_SGP4_MODE – Forces to use the SGP4 algorithm
  - XO_ORBIT_INIT_TLE_SDP4_MODE– Forces to use the SDP4 algorithm
  - XO_ORBIT_INIT_TLE_MODE – Chooses the algorithm depending or the orbital period: if less than 200minutes, the SGP4 is selected, if greater than 250 minutes, the SDP4 algorithm is chosen. If the orbital period is between 200 and 250 minutes the user cannot use this mode, it is mandatory to use one of the specific SDP4/SGP4 forcing modes, otherwise the CFI will return an error .XO_ORBIT_INIT_FILE_TLE_AUTO_ERR..
- *Numerical propagator*. This model consists on a numerical propagator that integrates the movement equations using a Runge-Kutta algorithm of 8th order. This propagator is expected to produce more precise results than the other models as it can be configured by the user (through the orbit initialisation function) to take into account the following perturbations:

- Non-spherical gravity: the model Earth Gravity Model 1996 is used. The file with the coefficients of the spherical harmonics must be provided by the user, besides the order and degree to be used in the calculations.
- Atmospheric drag: MSIS-E-90 atmospheric density model is used. The user must provide the Solar Geomagnetic Activity and F107 coefficient, either as input files or with constant values. The values can be obtained from ESA-ECSS or NASA documentation. The user must provide the drag effective area and drag coefficient of the satellite, besides the mass of the satellite.
- Solar radiation pressure: it is calculated using the solar radiation pressure effective area and solar radiation pressure coefficient of the satellite provided by the user.
- Third body perturbations: the perturbations produced by the Sun and the Moon gravities are calculated.

Apart from these models, in table there are two additional modes: "AUTO" and "DOUBLE". They refer to the way in which the seed (initial state vector used as reference to begin the propagation) is taken:

- AUTO mode: The seed is taken to be the closer ANX or OSV to the requested time. The propagation seed could change from one propagation to the following.
- DOUBLE mode: the two ANX covering the propagation time are used as seeds. When calling xo_osv_compute, the propagation is performed from each of the ANX and then a weighted average

$$\cos^2\left(\frac{\pi}{2}\cdot\frac{\Delta t}{T}\right)$$

is done. The weight function is

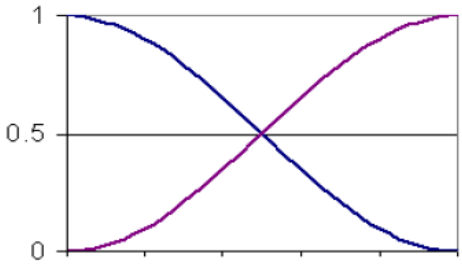
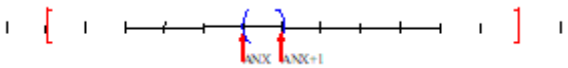
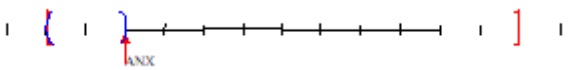
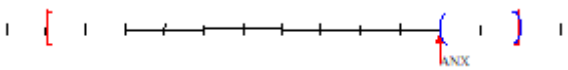


Figure 3: Weight Function for Double Propagation Model

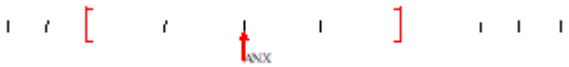
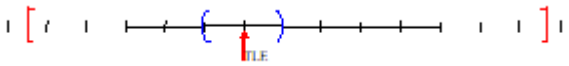
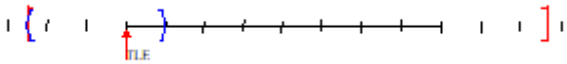
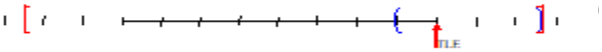
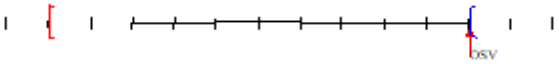
This propagation method removes any discontinuity that may arise when changing the state vector around the true ascending node crossing used as seed for the propagation.

Finally, the validity time interval for the propagation also depends on the propagation method. This validity interval can be computed with the CFI function **xo_orbit_get_osv_compute_validity** (section 7.22). The table summarises all the propagation cases, explaining the seed that is taken in every case and how the validity interval is taken. In the table, **t0** represent the input time for **xo_osv_compute**. In the fourth column, the horizontal solid line in the graphic represents the list of OSV or ANX that are stored in the orbit initialisation, t_start and t_stop are the first and last OSV(ANX) in that line. Red square brackets represent the validity period for propagation. When using the AUTO mode, the seed for the propagation changes if t0 jumps out of the region in blue brackets. The red arrow(s) represent the chosen seed(s) (OSV or ANX).

Table 85: Validity Time Intervals for Propagation

Propag model[2]	Requested time	Propagation seed	Validity time interval
Mean Keplerian + AUTO + DOUBLE Mode	t_start < t0 < t_stop	The two ANX that are before and after **t0**	[t_start - 2 orbits, t_stop + 2 orbits]; (ANX, ANX + 1orbit) 
	t0 < t_start	The first ANX in the orbit_id	[t_start - 2 orbits, t_stop + 2 orbits];  (ANX - 2 orbits, ANX)
	t0 > t_stop	The last ANX in the orbit_id	[t_start - 2 orbits, t_stop + 2 orbits];  (ANX , ANX + 2 orbits)

2 This column relates with the "OSV computation method" column in Table 84.

Mean Keplerian Mode	t_start - 2orbits < t0 < t_stop+2orbits Note that t_start = t_stop as there is only one OSV	The state vector used to initialise the orbit_id (there is only one)	[t_start - 2 orbits, t_stop + 2 orbits] 
TLE + AUTO Mode	t_start < t0 < t_stop	TLE that is before **t0**	[1st.TLE - 1day, Last TLE+ 1 day] (-1 TLE, +1 TLE) 
	t0 < t_start	First TLE	[1st.TLE - 1day, Last TLE+ 1 day] (TLE - 1day, + 1 TLE) 
	t0 > t_stop	Last TLE	
Numerical + AUTO Mode	t_start < t0 < t_stop	OSV that is before **t0**	[t_start, Infinity]; (OSV, Next OSV) 
	t0 < t_start	Propagation is not possible	
	t0 > t_stop	Last OSV	[t_start, Infinity]; (OSV, Infinity)

Note that the cases with “NO AUTO” mode are similar to those with “AUTO” mode with one OSV.

The precise propagator stores the result of the last propagation, so that the next propagation begins from that point if the time begins after the last propagation (it saves computation time). If the propagation is requested at a time that is before the time of the previous propagation time, then the original seed is taken.

7.33.2.2 Interpolation methods

The function **xo_osv_compute** computes the OSV using an interpolation algorithm when the orbit_id is initialised with **xo_orbit_init_file** plus ROF’s, DORIS Navigator files or SP3 files.

The interpolation method used is a Lagrange interpolation with 10 points.

The interpolation is highly accurate (1 mm. accuracy TBC) when it is performed between 4 OSV’s time intervals after start of file(s) and before end of file(s), but it degrades in the first 4 OSV's time intervals of the file and the last 4 OSV's time intervals of the file. The degradation depends strongly in the length of the

OSV's time intervals: it can be of the order of mm. for 1 second up to some cm. for 60 seconds. Figure 4 provides a graphical explanation.

The **xo_osv_compute** function allows to extrapolate, that is, compute results for at least 60 seconds before start of the input file(s) and after end of the input file. Anyway, extrapolation is not recommended. In this case, the extrapolation window is NOT included in the valid time interval. The following table shows some values for the degradation of the extrapolation:

Time out of validity interval [number of OSV time steps]	**Error in position [m] as a function of the OSV time step**			
	1 sec	**10 sec**	**30 sec**	**60 sec**
1	0.01	0.05	0.06	0.09
2	0.22	0.39	0.40	0.56
3	1.16	1.65	1.68	2.25
4	4.21	5.23	5.20	6.89

When the interpolation is in “degraded” mode, that is, when extrapolation is used, or when there is less than four orbit state vectors available in the input file before or after the requested time, **xo_osv_compute** function will issue different warnings messages indicating that a degraded interpolation or extrapolation is performed. If the requested time is out the allowed extrapolation range, the function will return an error message.

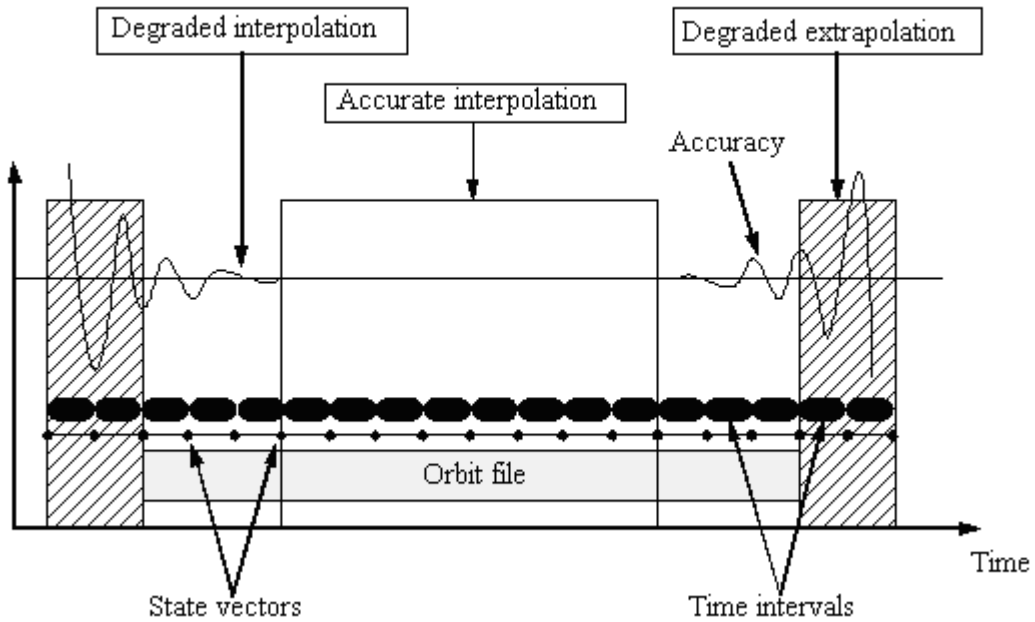


Figure 4: Performances of the interpolation algorithm.

7.33.3 Calling interface

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

```
#include <explorer_orbit.h>
{
  xo_orbit_id orbit_id = {NULL};
  long mode, time_ref;
  double time, pos_out[3], vel_out[3], acc_out[3];
  long status, ierr[XO_NUM_ERR_OSV_COMPUTE];

  status = xo_osv_compute (&orbit_id, &mode, &time_ref, &time,
                                   pos_out, vel_out, acc_out, ierr);

  /* Or, using the run_id */
  long run_id;

  status = xo_osv_compute_run (&run_id, &mode, &time_ref, &time,
                                       pos_out, vel_out, acc_out, ierr);
}
```

7.33.4 Input parameters

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

Table 86: Input parameters of xo_osv_compute function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure that contains the orbit data	-	-
mode	long *	-	Propagation model. Dummy input for current version.	-	-
time_ref	long*	-	Time reference ID	-	Complete
time	double*	-	Reference time	Decimal days (Processing format)	[-18262.0,36524.0]

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

- Time reference ID: time_ref. See [GEN_SUM].

7.33.5 Output parameters

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

Table 87: Output parameters of xo_osv_compute function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_osv_compute	long	-	Main status flag	-	-1, 0, +1
pos_out[3]	double	all	Osculating position vector at predicted time (Earth fixed CS)	m	-
vel_out[3]	double	all	Osculating velocity vector at predicted time (Earth fixed CS)	m/s	-
acc_out[3]	double	all	Osculating acceleration vector at predicted time (Earth fixed CS)	m/s^2	-
ierr[XO_NUM_ERR_PR OPAG]	long	all	Status vector	-	-

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

- Time reference ID: time_ref. See [GEN_SUM].

7.33.6 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_osv_compute** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_osv_compute** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 88: Error messages of xo_osv_compute function

Error type	Error message	Cause and impact	Error Code	Error No
ERR	Could not initialise the propagation	No calculation performed	XO_CFI_OSV_COMPUTE_PROPAG_INIT_ERR	0
ERR	The internal data were not	No calculation performed	XO_CFI_OSV_COMPUTE_NOT_INTERNAL_DATA_ER	1

	initialized		R	
ERR	Could not propagate the state vector	No calculation performed	XO_CFI_OSV_COMPUTE_PROPAG_ERR	2
ERR	Could not interpolate the state vector	No calculation performed	XO_CFI_OSV_COMPUTE_INTERPOL_ERR	3
ERR	Warnings during the propagation	No calculation performed	XO_CFI_OSV_COMPUTE_PROPAG_WARN	4
ERR	Warnings during the interpolation	No calculation performed	XO_CFI_OSV_COMPUTE_INTERPOL_WARN	5
ERR	Neither propagation nor interpolation can be performed with USER OSV LIST orbit initialization	No calculation performed	XO_CFI_OSV_COMPUTE_USER_OSV_LIST_ERR	6
WARN	Warnings during the propagation using a TLE File.	No calculation performed	XO_CFI_OSV_COMPUTE_TLE_OSV_GAP_WARN	7
WARN	Warnings during the acceleration computation	No calculation performed	XO_CFI_OSV_COMPUTE_INTERPOL_ACCELERATION_WARN	8

7.34 xo_osv_compute_extra

7.34.1 Overview

This software returns ancillary results derived from an orbit state vector obtained from the **xo_osv_compute** routine (stored within the *orbit Id*). This state vector depends on which is the last function called:

- when calling **xo_osv_compute_extra** after initialising the *orbit Id*, the selected state vector is:
  - the one that is selected as seed for the propagation.
  - the first OSV stored in the orbit_id if it is initialised for interpolation.
- when calling after **xo_osv_compute**, the Cartesian orbit state vector is the one predicted at the requested time in that routine.

A description of the ancillary results may be found in the section 7.34.5.

A complete calling sequence of the computation procedure is presented in section 4.2.

Note: model dependent para meters are not applicable to geostationary satellites and therefore are not computed.

7.34.2 Calling interface

The calling interface of the **xo_osv_compute_extra** CFI function is the following:

```
#include <explorer_orbit.h>
{
  xo_orbit_id orbit_id = {NULL};
  long extra_choice;
  double
model_out[XO_ORBIT_EXTRA_NUM_DEP_ELEMENTS],              extra_out[XO
_ORBIT_EXTRA_NUM_INDEP_ELEMENTS];
  long status, ierr[XO_NUM_ERR_OSV_COMPUTE_EXTRA];

  status = xo_osv_compute_extra (&orbit_id,
&extra_choice,                                  model_out,
extra_out, ierr);

  /* Or, using the run_id */
  long run_id;

  status = xo_osv_compute_run (&run_id,
&extra_choice,                                model_out,
extra_out, ierr);
}
```

7.34.3 Input parameters

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

Table 89: Input parameters of xo_osv_compute_extra

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure that contains the orbit initialisation data	-	-
extra_choice	long *	-	Flag to allow an ancillary results choice	-	[0, 4095]

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

- Flag to select ancillary results: extra_choice. See tables below:

Table 90: Enumeration values of extra_choice input flag

Model independant	Description	Long
XO_ORBIT_EXTRA_NO_RESULTS	No extra results	0
XO_ORBIT_EXTRA_GEOLOCATION	Geolocation results	1
XO_ORBIT_EXTRA_GEOLOCATION_D	Geolocation rate results	2
XO_ORBIT_EXTRA_GEOLOCATION_2D	Geolocation rate-rate results	4
XO_ORBIT_EXTRA_GEOLOCATION_EXTRA	Geolocation extra results	8
XO_ORBIT_EXTRA_EARTH_FIXED_D	Earth fixed velocity results	16
XO_ORBIT_EXTRA_EARTH_FIXED_2D	Earth fixed acceleration results	32
XO_ORBIT_EXTRA_SUN	Sun results	64
XO_ORBIT_EXTRA_MOON	Moon results	128
XO_ORBIT_EXTRA_OSCULATING_KEPLER	Osculating keplerian elements	256
XO_ORBIT_EXTRA_INERTIAL_AUX	Inertial auxiliary results	512
Model dependant (Mean Keplerian model)	**Description**	**Long**
XO_ORBIT_EXTRA_DEP_ANX_TIMING	ANX timing results	1024
XO_ORBIT_EXTRA_DEP_MEAN_KEPLER	Mean keplerian elements	2048

To calculate all results there is an extra enumeration value, defined as the addition of all the enumeration result values:

Enumeration value	Description	Long
XO_ORBIT_EXTRA_ALL_RESULTS	All results	4095

The elements calculated in each case are shown in section 7.34.5. It is possible to select the calculation of different sets of output parameters, or to make any combination of them by adding the results enumeration desired. In order to calculate some elements it might be necessary to calculate elements which have not been explicitly requested. The function identifies internally all the dependencies and those elements are also returned in the result vectors.

7.34.4 Output parameters

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

Table 91: Output parameters of xo_osv_compute_extra

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_osv_compute_extra	long	-	Main status flag	-	-1, 0, +1
model_out[XO_ORBIT_EXTRA_NUM_DEP_ELEMENTS]	double	all	Vector of model-dependent parameters	-	-
extra_out[XO_ORBIT_EXTRA_NUM_INDEP_ELEMENTS]	double	all	Vector of model-independent parameters. It depends upon extra-choice	-	-
ierr[XO_NUM_ERR_OSV_COMPUTE_EXTRA]	long	all	Status vector	-	-

7.34.5 Results vectors

The following tables describe the set of parameters are computed with **xo_osv_compute_extra**.

The model-dependent parameters vector (note that there is an enumeration associated to the elements of the results vectors) is in Table 92. These parameters depends on the way in which the input orbit_id was initialised (the orbit mode). So, in table a column “OSV compute model” indicates the models for which that parameter can be computed. To simplify, the models are indicated with the following correspondence:

Orbit initialisation mode	OSV compute model
XO_ORBIT_INIT_ORBIT_CHANGE_MODE XO_ORBIT_INIT_OSF_MODE XO_ORBIT_INIT_OEF_OSF_MODE XO_ORBIT_INIT_POF_MODE XO_ORBIT_INIT_OEF_POF_MODE XO_ORBIT_INIT_STATE_VECTOR_MODE XO_ORBIT_INIT_POF_N_DORIS_MODE	Mean Kepler
XO_ORBIT_INIT_TLE_MODE XO_ORBIT_INIT_TLE_SGP4_MODE	TLE

XO_ORBIT_INIT_TLE_SDP4_MODE	
XO_ORBIT_INIT_POF_PRECISE_MODE XO_ORBIT_INIT_OEF_POF_PRECISE_MODE XO_ORBIT_INIT_ROF_PRECISE_MODE XO_ORBIT_INIT_DORIS_PRECISE_MODE XO_ORBIT_INIT_STATE_VECTOR_PRECISE_MODE XO_ORBIT_INIT_POF_N_DORIS_PRECISE_MODE	Precise
XO_ORBIT_INIT_ROF_MODE XO_ORBIT_INIT_DORIS_MODE XO_ORBIT_INIT_SP3_MODE XO_ORBIT_INIT_OEM_MODE	Interpolation

Table 92: Ancillary results vector. Model-dependent parameters

Result parameter	Set	Description (Reference)	Unit (Format)	Allowed Range	OSV compute model
[0] XO_ORBIT_EXTRA_DEP_NODAL_PERIOD	ANX Timing a	Nodal period	s	>= 0	Mean Kepler Interpolation
[1] XO_ORBIT_EXTRA_DEP_UTC_CURRENT_ANX		Time of current ANX	decimal days (Processing format)	-	
[2] XO_ORBIT_EXTRA_DEP_ORBIT_NUMBER	Position in orbit[3]	Absolute Orbit Number		> 0	Mean Kepler TLE Interpolation Precise
[3] XO_ORBIT_EXTRA_DEP_SEC_SINCE_ANX		Time since ANX	s	>= 0 < Nodal Period	
[4:9] XO_ORBIT_EXTRA_DEP_MEAN_KEPL_A XO_ORBIT_EXTRA_DEP_MEAN_KEPL_E XO_ORBIT_EXTRA_DEP_MEAN_KEPL_I XO_ORBIT_EXTRA_DEP_MEAN_KEPL_RA XO_ORBIT_EXTRA_DEP_MEAN_KEPL_W XO_ORBIT_EXTRA_DEP_MEAN_KEPL_M	Mean Kepler b	Mean Kepler elements of the propagated OSV (True of Date)	-	-	Mean Kepler Precise

a. For TLE and Precise, these parameters can be computed by calling the CFI function xo_orbit_info.
b. For TLE and Interpolation, these parameters can be computed by calling the CFI function xo_orbit_info.

The model-independent parameters vector (note that there is an enumeration associated to the elements of the results vectors) is in Table 93:

Table 93: Ancillary results vector. Model-independent parameters

Result parameter (res element)	Set	Description (Reference)	Unit (Format)	Allowed Range

3 These parameters are calculated only when initialising with xo_orbit_init_file and xo_orbit_init_def

[0] XO_ORBIT_EXTRA_GEOC_LONG	Geolocation	Geocentric longitude of satellite and SSP (EF frame)	deg	>= 0 < 360
[1] XO_ORBIT_EXTRA_GEOD_LAT		Geodetic latitude of satellite and SSP (EF frame)	deg	>= -90 <= +90
[2] XO_ORBIT_EXTRA_GEOD_ALT		Geodetic altitude of the satellite (EF frame)	m	-
[3] XO_ORBIT_EXTRA_GEOC_LONG_D	Geolocation rate	Geocentric longitude rate of satellite and SSP (EF frame)	deg/s	-
[4] XO_ORBIT_EXTRA_GEOD_LAT_D		Geodetic latitude rate of satellite and SSP (EF frame)	deg/s	-
[5] XO_ORBIT_EXTRA_GEOD_ALT_D		Geodetic altitude rate of the satellite (EF frame)	m/s	-
[6] XO_ORBIT_EXTRA_GEOC_LONG_2D	Geolocation rate rate	Geocentric longitude rate-rate of satellite and SSP (EF frame)	deg/s2	-
[7] XO_ORBIT_EXTRA_GEOD_LAT_2D		Geodetic latitude rate-rate of satellite and SSP (EF frame)	deg/s2	-
[8] XO_ORBIT_EXTRA_GEOD_ALT_2D		Geodetic altitude rate-rate of the satel lite (EF frame)	m/s2	-
[9] XO_ORBIT_EXTRA_RAD_CUR_PARALLE L_MERIDIAN	Geolocation extra	Radius of curvature parallel to meridian at the SSP (EF frame)	m	>= 0
[10] XO_ORBIT_EXTRA_RAD_CUR_ORTHO_ MERIDIAN		Radius of curvature orthogonal to meridian at the SSP (EF frame)	m	>= 0
[11] XO_ORBIT_EXTRA_RAD_CUR_ALONG_G ROUNDTRACK		Radius of curvature along groundtrack at the SSP (EF frame)	m	>= 0
[12] XO_ORBIT_EXTRA_NORTH_VEL	Earth-fixed velocity	Northward component of the velocity relative to the Earth of the SSP (Topocentric frame)	m/s	-
[13] XO_ORBIT_EXTRA_EAST_VEL		Eastward component of the velocity relative to the Earth of the SSP (Topocentric frame)	m/s	-
[14] XO_ORBIT_EXTRA_MAG_VEL		Magnitude of the velocity relative to the Earth of the SSP (Topocentric frame)	m/s	>= 0
[15] XO_ORBIT_EXTRA_AZ_VEL		Azimuth of the velocity relative to the Earth of the SSP (Topocentric frame)	deg	>= 0 < 360
[16] XO_ORBIT_EXTRA_NORTH_ACC	Earth-fixed acceleration	Northward component of the acceleration relative to the Earth of the SSP (Topocentric frame)	m/s2	-

[17] XO_ORBIT_EXTRA_EAST_ACC		Eastward component of the acceleration relative to the Earth of the SSP (Topocentric frame)	m/s2	-
[18] XO_ORBIT_EXTRA_GROUNDTRACK_TA NG_ACC		Groundtrack tangential component of the acceleration relative to the Earth of the SSP (Topocentric frame)	m/s2	-
[19] XO_ORBIT_EXTRA_AZ_ACC		Azimuth of the acceleration relative to the Earth of the SSP (Topocentric frame)	deg	>= 0 < 360
[20] XO_ORBIT_EXTRA_SAT_ECLIPSE_F LAG	Sun	Satellite eclipse flag 0 = No 1 = Yes		0, 1
[21] XO_ORBIT_EXTRA_SZA		Sun Zenith Angle	deg	>= 0 < 180
[22] XO_ORBIT_EXTRA_MLST		Mean local solar time at the SSP	decimal hour	>= 0 < 24
[23] XO_ORBIT_EXTRA_TLST		True local solar time at the SSP	decimal hour	>= 0 < 24
[24] XO_ORBIT_EXTRA_TRUE_SUN_RA		True Sun's (centre) right ascension (TOD frame)	deg	>= 0 < 360
[25] XO_ORBIT_EXTRA_TRUE_SUN_DEC		True Sun's (centre) declination (TOD frame)	deg	>= -90 <= +90
[26] XO_ORBIT_EXTRA_TRUE_SUN_SEM I_DI AM		True Sun's semi-diameter	deg	>= 0
[27] XO_ORBIT_EXTRA_MOON_RA	Moon	Moon's (centre) right ascension (TOD frame)	deg	>= 0 < 360
[28] XO_ORBIT_EXTRA_MOON_DEC		Moon's (centre) declination (TOD frame)	deg	>= -90 <= +90
[29] XO_ORBIT_EXTRA_MOON_SEMI_DI AM		Moon's semi-diameter	deg	>= 0
[30] XO_ORBIT_EXTRA_MOON_AREA_LI T		Area of Moon lit by Sun		>= 0 <= 1
[31:36] XO_ORBIT_EXTRA_OSC_KEPL_A XO_ORBIT_EXTRA_OSC_KEPL_E XO_ORBIT_EXTRA_OSC_KEPL_I XO_ORBIT_EXTRA_OSC_KEPL_RA XO_ORBIT_EXTRA_OSC_KEPL_W	Osculating Kepler	Osculating Keplerian elements of the OSV (TOD frame)		

XO_ORBIT_EXTRA_OSC_KEPL_M				
[37] XO_ORBIT_EXTRA_ORBIT_RAD	Inertial Aux	Orbit radius (TOD frame)	m	>= 0
[38] XO_ORBIT_EXTRA_RADIAL_ORB_VEL		Radial orbit velocity component (TOD frame)	m/s	-
[39] XO_ORBIT_EXTRA_TRANS_ORB_VEL		Transversal orbit velocity component (TOD frame)	m/s	-
[40] XO_ORBIT_EXTRA_ORB_VEL_MAG		Orbit velocity magnitude (TOD frame)	m/s	>= 0
[41] XO_ORBIT_EXTRA_RA_SAT		Right ascension of the satellite (TOD frame)	deg	>= 0 < 360
[42] XO_ORBIT_EXTRA_DEC_SAT		Declination of the satellite (TOD frame)	deg	>= -90 <= +90
[43] XO_ORBIT_EXTRA_EARTH_ROTATION_ ANGLE		Earth rotation angle [H]	deg	>= 0 < 360
[44] XO_ORBIT_EXTRA_RA_SAT_D		Right ascension rate of the satellite (TOD frame)	deg/s	-
[45] XO_ORBIT_EXTRA_RA_SAT_2D		Right ascension rate-rate of the satellite (TOD frame)	deg/s2	-
[46] XO_ORBIT_EXTRA_OSC_TRUE_LAT		Satellite osculating true latitude (EF frame)	deg	>= 0 < 360
[47] XO_ORBIT_EXTRA_OSC_TRUE_LAT_D		Satellite osculating true latitude rate (EF frame)	deg/s	-
[48] XO_ORBIT_EXTRA_OSC_TRUE_LAT_2D		Satellite osculating true latitude rate-rate (EF frame)	deg/s2	-

7.34.6 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_osv_compute_extra** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_osv_compute_extra** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 94: Error messages of xo_osv_compute_extra function

Error type	Error message	Cause and impact	Error Code	Error No
ERR	Could not initialise the propagation	No calculation performed	XO_CFI_OSV_COMPUTE_EXTRA_PROPAG_INIT_ERR	0
ERR	The internal data were not initialized	No calculation performed	XO_CFI_OSV_COMPUTE_EXTRA_NOT_INTERNAL_D ATA_ERR	1
ERR	Could not propagate the state vector	No calculation performed	XO_CFI_OSV_COMPUTE_EXTRA_PROPAG_ERR	2
ERR	Could not interpolate the state vector	No calculation performed	XO_CFI_OSV_COMPUTE_EXTRA_INTERPOL_ERR	3

7.35 xo_orbit_to_time

7.35.1 Overview

The **xo_orbit_to_time** function converts an orbit-relative time into processing time.

7.35.2 Calling sequence of xo_orbit_to_time:

For C programs, the call to **xo_orbit_to_time** is (input parameters are underlined):

```
#include <explorer_orbit.h>
{
  xo_orbit_id   orbit_id = {NULL};
  long time_ref;
  long orbit, second, microsec;
  long status, ierr[XO_NUM_ERR_ORBIT_TO_TIME];
  double   time;

  status = xo_orbit_to_time (&orbit_id ,
                             &orbit, &second, &microsec,
                             &time_ref,
                             &time,    ierr);

  /* Or, using the run_id */
  long run_id;

  status = xo_orbit_to_time_run (&run_id   ,
                                 &orbit, &second, &microsec,
                                 &time_ref,
                                 &time,    ierr);
}
```

7.35.3 Input parameters

Table 95: Input parameters for xo_orbit_to_time

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure that contains the orbit data	-	-
orbit	long*		Absolute orbit number		> 0
second	long*		Seconds since ascending node	s	>= 0 <orbital period
microsec	long*		Micro seconds within second	µs	0 =<

					=< 999999
time_ref	long*		Time reference ID	-	Complete

7.35.4 Output parameters

Table 96: Output parameters for xo_orbit_to_time

C name	C type	Array Eleme nt	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_to_time	long		Main status flag		-1, 0, 1
time	double*		Resulting time	Dedimal days (processing format)	[-18262.0, +36519.0]
ierr[XO_NUM_ERR _ORBIT_TO_TIME]	long		Error status flags		

7.35.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_orbit_to_time** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_orbit_to_time** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 97: Error messages of xo_orbit_to_time function

Error type	Error message	Cause and impact	Error Code	Error No
ERR	Wrong input flag	Computation not performed	XO_CFI_ORBIT_TO_TIM E_FLAG_ERR	0
ERR	Input incorrect: negative orbit number	Computation not performed	XO_CFI_ORBIT_TO_TIM E_ORB_NUM_1ST_ERR	1
ERR	Orbit Id. is not initialised.	Computation not performed	XO_CFI_ORBIT_TO_TIM E_ORBIT_STATUS_ERR	2
ERR	Seconds and microseconds greater than nodal period	Computation not performed	XO_CFI_ORBIT_TO_TIM E_SEC_MICROSEC_ERR	3
ERR	Requested orbit less than the first orbital change	Computation not performed	XO_CFI_ORBIT_TO_TIM E_ORB_ERR	4
ERR	Input incorrect: negative number of seconds	Computation not performed	XO_CFI_ORBIT_TO_TIM E_SEC_ERR	5
ERR	Input incorrect: number of	Computation not	XO_CFI_ORBIT_TO_TIM	6

	microseconds out of range	performed	E_MICROSEC_ERR	
ERR	Error computing time.	Computation not performed	XO_CFI_ORBIT_TO_TIME_COMPUTE_ERR	7
ERR	Could not make a time transformation	Computation not performed	XO_CFI_ORBIT_TO_TIME_TIME_CHANGE_ERR	8
ERR	Geostationary satellite not allowed for this function.	Computation not performed	XO_CFI_ORBIT_TO_TIME_GEO_SAT_ERR	9
ERR	SP3 files do not have orbit information	Computation not performed	XO_CFI_ORBIT_TO_TIME_SP3_ERR	10

7.35.6 Executable Program

The conversion from orbit to time described before can be carried out by the **orbit_to_time** executable program as follows:

orbit_to_time **-sat** satellite_name
-file Orbit file
-tref time_ref
-orb orbit
-anx anx_time (seconds)
[**-v**]
[**-xl_v**]
[**-xo_v**]
[**-help**]
[**-show**]
{ (**-tai** TAI_time **-gps** GPS_time **-utc** UTC_time **-ut1** UT1_time) |
(**-tmod** time_model **-tfile** time_file **-trid** time_reference
{(**-tm0** time0 **-tm1** time1) | (**-orb0** orbit0 **-orb1** orbit1) }) **}**

Note that:

- Order of parameters does not matter.
- Bracketed parameters are not mandatory.
- Options between curly brackets and separated by a vertical bar are mutually exclusive.
- [**-xl_v**] option for EO_LIB Verbose mode.
- [**-xo_v**] option for EO_ORBIT Verbose mode.
- [**-v**] option for Verbose mode for all libraries (default is Silent).
- [**-show**] displays the inputs of the function and the results.

- Possible values for *satellite_name*: ERS1, ERS2, ENVISAT, METOP1 , METOP2, METOP3, CRYOSAT, ADM, GOCE, SMOS, TERRASAR, EARTHCARE, SWARM_A, SWARM_B, SWARM_C, SENTINEL_1A, SENTINEL_1B, SENTINEL_2, SENTINEL_3, SENTINEL_1C, SENTINEL_2A, SENTINEL_2B, SENTINEL_2C, SENTINEL_3A, SENTINEL_3B, SENTINEL_3C, SEOSAT, JASON_CSA, JASON_CSB, METOP_SG_A1, METOP_SG_A2, METOP_SG_A3, METOP_SG_B1, METOP_SG_B2, METOP_SG_B3, SENTINEL_5P, BIOMASS, SENTINEL_5, SAOCOM_CS, FLEX, SENTINEL_6A, SENTINEL_6B, CIMR, ROSE-L, CHIME, CRISTAL, CO2M, LSTM, FORUM, TRUTHS, GENERIC, GENERIC_GEO, MTG.
- Possible values for *time_model*: USER, NONE , IERS_B_PREDICTED, IERS_B_RESTITUTED, FOS_PREDICTED, FOS_RESTITUTED, DORIS_PRELIMINARY, DORIS_PRECISE, DORIS_NAVIGATOR, OSF
- Possible values for *time_ref* and *time_reference*: UNDEF, TAI, UTC, UT1, GPS.
- Data for initialising the time references are needed only when using an Orbit Scenario file. For other files the data is optional. In that case, if the iniialization parameters are not provided, the time correlations are initialised with the input orbit file.

The inputs needed for time initialiazation are provided in the last three lines of parameters. Note that only one set of parameters should be introduced:

- TAI, GPS, UTC and UT1 input times (as in xl_time_ref_init)
- A file with time reference data, the time mode, the time reference name and a time range (as in xl_time_ref_init_file)

- Example:

orbit_to_time *-**sat** CRYOSAT -**file** EARTH_EXPLORER_FPO -**tref** UTC*
*-**orb** 1001 -**anx** 0.0 -**show** -**v***

7.36 xo_time_to_orbit

7.36.1 Overview

The **xo_time_to_orbit** function converts an orbit-relative time into processing time.

7.36.2 Calling sequence

For C programs, the call to **xo_time_to_orbit** is (input parameters are underlined):

```
#include <explorer_orbit.h>
{
  xo_orbit_id   orbit_id = {NULL};
  long time_ref;
  long orbit, second, microsec;
  long status, ierr[XO_NUM_ERR_ORBIT_TO_TIME];
  double   time;

  status = xo_time_to_orbit ( &orbit_id,
                              &time_ref, &time,
                              &orbit, &second, &microsec,
                              ierr);

  /* Or, using the run_id */
  long run_id;

  status = xo_time_to_orbit_run ( &run_id,
                                  &time_ref, &time,
                                  &orbit, &second, &microsec,
                                  ierr);
}
```

7.36.3 Input parameters

Table 98: Input parameters for xo_time_to_orbit function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure that contains the orbit data	-	-
time_ref	long*		Time reference ID	-	Complete
time	double*		Requested time	Decimal days (processing format)	[-18262.0, +36519.0]

7.36.4 Output parameters

Table 99: Output parameters for xo_time_to_orbit

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_time_to_orbit	long		Main status flag		-1, 0, 1
orbit	long*		Absolute orbit number		> 0
second	long*		Seconds since ascending node	s	>= 0 <orbital period
microsec	long*		Micro seconds within second	µs	0 =< =< 999999
ierr[XO_NUM_ERR_TIME_TO_ORBIT]	long		Error status flags		

7.36.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_time_to_orbit** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_time_to_orbit** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 100: Error messages of xo_time_to_orbit function

Error type	Error message	Cause and impact	Error Code	Error No
ERR	Wrong input flag	Computation not performed	XO_CFI_TIME_TO_ORBIT_FLAG_ERR	0
ERR	Orbit Id. was not initialized.	Computation not performed	XO_CFI_TIME_TO_ORBIT_ORBIT_STATUS_ERR	1
ERR	Input incorrect: time out of range.	Computation not performed	XO_CFI_TIME_TO_ORBIT_TIME_ERR	2
ERR	Input time smaller than the first ANX time.	Computation not performed	XO_CFI_TIME_TO_ORBIT_BEFORE_RANGE_ERR	3
ERR	Could not compute the	Computation not performed	XO_CFI_TIME_TO_OR	4

	orbit number.		BIT_COMPUTE_ERR	
ERR	The current orbit initialization does not allow to compute the time.	Computation not performed	XO_CFI_TIME_TO_ORBIT_WRONG_ORBIT_MODE_ERR	5
WARN	Input time before first orbit.	Computation performed	XO_CFI_TIME_TO_ORBIT_TIME_BEFORE_RANGE_WARN	6
WARN	Input time after first orbit.	Computation performed	XO_CFI_TIME_TO_ORBIT_TIME_AFTER_RANGE_WARN	7
WARN	Orbit number computed with warnings.	Computation performed	XO_CFI_TIME_TO_ORBIT_COMPUTE_WAR N	8
ERR	Geostationary satellite not allowed for this function.	Computation not performed	XO_CFI_TIME_TO_ORBIT_GEO_SAT_ERR	9
ERR	SP3 files do not have orbit information	Computation not performed	XO_CFI_TIME_TO_ORBIT_SP3_ERR	10
WARN	The gap between the requested time/orbit and the nearest TLE differs more than one day	Computation performed	XO_CFI_TIME_TO_ORBIT_TLE_GAP_WARN	11

7.36.6 Executable Program

The conversion from time to orbit described before can be carried out by the **time_to_orbit** executable program as follows:

time_to_orbit **-sat** satellite_name
-file Orbit file
-tref time_ref
{**-time** time (days) | **-atime** time (CCSDSA format)}
[**-v**]
[**-xl_v**]
[**-xo_v**]
[**-help**]
[**-show**]
{ (**-tai** TAI_time **-gps** GPS_time **-utc** UTC_time **-ut1** UT1_time) |
(**-tmod** time_model **-tfile** time_file **-trid** time_reference
{(**-tm0** time0 **-tm1** time1) | (**-orb0** orbit0 **-orb1** orbit1) }) **}**

Note that:

- Order of parameters does not matter.
- Bracketed parameters are not mandatory.
- Options between curly brackets and separated by a vertical bar are mutually exclusive.
- [**-xl_v**] option for EO_LIB Verbose mode.
- [**-xo_v**] option for EO_ORBIT Verbose mode.
- [**-v**] option for Verbose mode for all libraries (default is Silent).
- [**-show**] displays the inputs of the function and the results.
- Possible values for *satellite_name*: ERS1, ERS2, ENVISAT, METOP1 , METOP2, METOP3, CRYOSAT, ADM, GOCE, SMOS, TERRASAR, EARTHCARE, SWARM_A, SWARM_B, SWARM_C, SENTINEL_1A, SENTINEL_1B, SENTINEL_2, SENTINEL_3, SENTINEL_1C, SENTINEL_2A, SENTINEL_2B, SENTINEL_2C, SENTINEL_3A, SENTINEL_3B, SENTINEL_3C, SEOSAT, JASON_CSA, JASON_CSB, METOP_SG_A1, METOP_SG_A2, METOP_SG_A3, METOP_SG_B1, METOP_SG_B2, METOP_SG_B3, SENTINEL_5P, BIOMASS, SENTINEL_5, SAOCOM_CS, FLEX, SENTINEL_6A, SENTINEL_6B, CIMR, ROSE-L, CHIME, CRISTAL, CO2M, LSTM, FORUM, TRUTHS, GENERIC, GENERIC_GEO, MTG.
- Possible values for *time_model*: USER, NONE , IERS_B_PREDICTED, IERS_B_RESTITUTED, FOS_PREDICTED, FOS_RESTITUTED, DORIS_PRELIMINARY, DORIS_PRECISE, DORIS_NAVIGATOR, OSF.
- Possible values for *time_ref* and *time_reference*: UNDEF, TAI, UTC, UT1, GPS.
- Data for initialising the time references are needed only when using an Orbit Scenario file. For other files the data are optional. In that case, if the iniialization parameters are not provided, the time correlations are initialised with the input orbit file

The inputs needed for time initialization are provided in the last three lines of parameters. Note that only one set of parameters should be introduced:

- TAI, GPS, UTC and UT1 input times (as in xl_time_ref_init)
- A file with time reference data, the time mode, the time reference name and a time range (as in xl_time_ref_init_file)

Example:

```
time_to_orbit -sat CRYOSAT -file EARTH_EXPLORER_FPO -tref UTC
              -time -2010.108657407 –show -v
```

7.37 xo_orbit_info

7.37.1 Overview

The **xo_orbit_info** function retrieves from the orbit initialisation, information related with a certain orbit (specified by means of absolute orbit number).

Note: the computation of the parameter Spacecraft Midnight (SMX) is disabled by default. To enable the computation of this parameter the function **xo_orbit_info_configure** must be used (see section 7.38).

7.37.2 Calling sequence

For C programs, the call to **xo_orbit_info** is (input parameters are underlined, some may be input or output depending on the calling mode):

```
#include <explorer_orbit.h>
{
  xo_orbit_id  orbit_id = {NULL};
  long         abs_orbit;
  long         ierr[XO_NUM_ERR_ORBIT_INFO], status;
  double       result_vector[XO_ORBIT_INFO_EXTRA_NUM_ELEMENTS];

  status = xo_orbit_info (&orbit_id,
                          &abs_orbit,
                          result_vector, ierr);

  /* Or, using the run_id */
  long run_id;

  status = xo_orbit_info_run (&run_id,
                              &abs_orbit,
                              result_vector, ierr);
}
```

7.37.3 Input parameters

Table 101: Input parameters for xo_orbit_info

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure that contains the orbit data.	-	-
abs_orbit	long *		Absolute orbit number		within orbit_id range

7.37.4 Output parameters

Table 102: Output parameters for xo_orbit_info

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_info	long		Main status flag,		-1, 0, 1
result_vector [XO_ORBIT_INFO_EXTRA_NUM_ELEMENTS]	double	XO_ORBIT_INFO_EXTRA_REPEAT_CYCLE	repeat_cycle[4]	days	>0
		XO_ORBIT_INFO_EXTRA_CYCLE_LENGTH	cycle_length	orbits	>0
		XO_ORBIT_INFO_EXTRA_MLST_DRIFT	MLST drift	s/day	
		XO_ORBIT_INFO_EXTRA_MLST	MLST[5]	hours	>0 <24
		XO_ORBIT_INFO_EXTRA_ANX_LONG	phasing	deg	>0 <360
		XO_ORBIT_INFO_EXTRA_UTC_ANX	UTC time at ascending node	days (processing format)	
		XO_ORBIT_INFO_EXTRA_POS_X XO_ORBIT_INFO_EXTRA_POS_Y XO_ORBIT_INFO_EXTRA_POS_Z	position at ANX	m	
		XO_ORBIT_INFO_EXTRA_VEL_X XO_ORBIT_INFO_EXTRA_VEL_Y XO_ORBIT_INFO_EXTRA_VEL_Z	velocity at ANX	m/s	
		XO_ORBIT_INFO_EXTRA_MEAN_KEPL_A XO_ORBIT_INFO_EXTRA_MEAN_KEPL_E XO_ORBIT_INFO_EXTRA_MEAN_KEPL_I XO_ORBIT_INFO_EXTRA_MEAN_KEPL_RA XO_ORBIT_INFO_EXTRA_MEAN_KEPL_W XO_ORBIT_INFO_EXTRA_MEAN_KEPL_M	mean keplerian elements at ANX		
		XO_ORBIT_INFO_EXTRA_OSC_KEPL_A XO_ORBIT_INFO_EXTRA_OSC_KEPL_E XO_ORBIT_INFO_EXTRA_OSC_KEPL_I XO_ORBIT_INFO_EXTRA_OSC_KEPL_RA XO_ORBIT_INFO_EXTRA_OSC_KEPL_W XO_ORBIT_INFO_EXTRA_OSC_KEPL_M	osculating keplerian elements at ANX		
		XO_ORBIT_INFO_EXTRA_NODAL_PERIOD	Nodal period	s	
		XO_ORBIT_INFO_EXTRA_UTC_SMX	UTC time of Spacecraft Midnight (see below the table for more information)	days (processing format)	
ierr[XO_ORBIT_I	long	all	Error status flags		

4 This parameter is only computed if the input orbit_id was computed either with an Orbit Scenario File using xo_orbit_init_file or with xo_orbit_init_def

5 This parameter is not computed if the input orbit_id was computed using a Restituted Orbit file or a DORIS file

NFO _FROM_ABS]					

Note: the Spacecraft Midnight (SMX) is the time just halfway the nadir day → night transition (the first transition of this type after ANX of the orbit) and the nadir night → day transition (the first transition of this type after the previous day → night transition). Such transitions are times at which the Sun Zenith Angle (SZA, angle satellite-nadir-sun) is 90 deg. In the day → night transition, the SZA is increasing (i.e. there is a transition from SZA<90 to SZA>90). In the night → day transition the SZA is decreasing (i.e. there is a transition from SZA>90 to SZA<90).

7.37.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_orbit_info** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_orbit_info** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 103: Error messages of xo_orbit_info function

Error type	Error message	Cause and impact	Error Code	Error No
ERR	Orbit Id. No initialised.	Computation not performed	XO_CFI_ORBIT_INFO _ORBIT_INIT_ERR	0
ERR	Orbit out of initialised limits.	Computation not performed	XO_CFI_ORBIT_INFO _OUT_OF_LIMITS_ER R	1
ERR	Could not compute extra results	Computation not performed	XO_CFI_ORBIT_INFO _RESULTS_ERR	2
ERR	Geostationary satellite not allowed for this function.	Computation not performed	XO_CFI_ORBIT_INFO_G EO_SAT_ERR	3
ERR	Input orbit mode not allowed	Computation not performed	XO_CFI_ORBIT_INFO_O RBIT_MODE_NOT_ALL OWED_ERR	4
ERR	Error computing Spacecraft Midnight	Computation not performed	XO_CFI_ORBIT_INFO_S MX_ERR	5
WARN	No Spacecraft Midnight point found for the orbit	Computation performed. This is true except for SMX parameter, which is set to 0.	XO_CFI_ORBIT_INFO_N O_SMX_WARN	6

7.38 xo_orbit_info_configure

7.38.1 Overview

The **xo_orbit_info_configure** function can be used to activate (or deactivate) the computation of several parameters in **Error! Reference source not found.** function. Some parameters computed by xo_orbit_info require a significant computation time, so their computation is disabled by default and must be activated with **xo_orbit_info_configure** function. The parameters that can be activated currently are the following ones:

- Spacecraft Midnight (SMX)

7.38.2 Calling sequence

For C programs, the call to **xo_orbit_info_configure** is (input parameters are underlined, some may be input or output depending on the calling mode):

```
#include <explorer_orbit.h>
{
  xo_orbit_id orbit_id = {NULL};
  long        item, flag;
  long        ierr[XO_NUM_ERR_ORBIT_INFO_CONFIGURE], status;

  status = xo_orbit_info_configure (&orbit_id,
                                    &item, &flag,
                                     ierr);
}
```

7.38.3 Input parameters

Table 104: Input parameters for xo_orbit_info_configure

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure that contains the orbit data.	-	-
Item	long *	-	Item to be activated/deactivated	-	Orbit items enumeration. (see Table 2)
flag	long *	-	Flag that indicates if the item must be activated or deactivated	-	Orbit flag enumeration (see Table 2)

7.38.4 Output parameters

Table 105: Output parameters for xo_orbit_info_configure

C name	C type	Array Element	Description	Unit	Allowed

			(Reference)	(Format)	Range
xo_orbit_info_configure	long		Main status flag,		-1, 0, 1
ierr[XO_ORBIT_INFO_CONFIGURE]	long	all	Error status flags		

7.38.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_orbit_info_configure** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_orbit_info_configure** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 106: Error messages of xo_orbit_info_configure function

Error type	Error message	Cause and impact	Error Code	Error No
ERR	Orbit Id not initialised.	Computation not performed	XO_ORBIT_INFO_CONFIGURE_ORBIT_INIT_STATUS_ERR	0
ERR	Wrong input option	Computation not performed	XO_ORBIT_INFO_CONFIGURE_WRONG_OPTION_ERR	1
ERR	Wrong input item	Computation not performed	XO_ORBIT_INFO_CONFIGURE_WRONG_ITEM_ERR	2

7.39 xo_orbit_rel_from_abs

7.39.1 Overview

The **xo_orbit_rel_from_abs** function retrieves from an Orbit Scenario File (previously initialised through the *orbit Id*) the relative orbit corresponding to a given absolute orbit number.

7.39.2 Calling sequence

For C programs, the call to **xo_orbit_rel_from_abs** is (input parameters are underlined, some may be input or output depending on the calling mode):

```
#include <explorer_orbit.h>
{
  xo_orbit_id    orbit_id = {NULL};
  long               abs_orbit, rel_orbit, cycle, phase;
  long           ierr[XO_NUM_ERR_ORBIT_REL_FROM_ABS], status;

  status = xo_orbit_rel_from_abs (&orbit_id,
                                  &abs_orbit,
                                  &rel_orbit, &cycle,
                                  &phase, ierr);

  /* Or, using the run_id */
  long run_id;
  status = xo_orbit_rel_from_abs_run (&run_id,
                                      &abs_orbit,
                                      &rel_orbit, &cycle,
                                      &phase, ierr);
}
```

7.39.3 Input parameters

Table 107: Input parameters for xo_orbit_rel_from_abs

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure that contains the orbit data	-	-
abs_orbit	long *		Absolute orbit number		within orbit_id range

7.39.4 Output parameters

Table 108: Output parameters for xo_orbit_rel_from_abs

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_rel_from_abs	long		Main status flag,		-1, 0, 1
rel_orbit	long *		Relative orbit number		
cycle	long *		Cycle number		
phase	long *		Phase number		
ierr[XO_ORBIT_REL_FROM_ABS]	long	all	Error status flags		

7.39.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_orbit_rel_from_abs** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_orbit_rel_from_abs** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 109: Error messages of xo_orbit_rel_from_abs function

Error type	Error message	Cause and impact	Error Code	Error No
ERR	Orbit Id. is not initialised.	Computation not performed	XO_CFI_ORBIT_REL_FROM_ABS_ORBIT_INIT_ERR	0
ERR	The relative orbit could not be computed with the current orbit initialization.	Computation not performed	XO_CFI_ORBIT_REL_FROM_ABS_ORBIT_WRONG_MODE_ERR	1
ERR	Wrong input orbit number	Computation not performed	XO_CFI_ORBIT_REL_FROM_ABS_WRONG_ORBIT	2

7.40 xo_orbit_abs_from_rel

7.40.1 Overview

The **xo_orbit_abs_from_rel** function retrieves from an Orbit Scenario File (previously initialised through the *orbit Id*) the absolute orbit corresponding to a given relative orbit number and cycle.

7.40.2 Calling sequence

For C programs, the call to **xo_orbit_abs_from_rel** is (input parameters are underlined, some may be input or output depending on the calling mode):

```
#include <explorer_orbit.h>
{
  xo_orbit_id  orbit_id = {NULL};
  long         abs_orbit, rel_orbit, cycle, phase;
  long         ierr[XO_NUM_ERR_ORBIT_ABS_FROM_REL], status;

  status = xo_orbit_abs_from_rel (&orbit_id,
                                  &rel_orbit, &cycle,
                                  &abs_orbit, &phase, ierr);

  /* Or, using the run_id */
  long run_id;

  status = xo_orbit_abs_from_rel_run (&run_id,
                                      &rel_orbit, &cycle,
                                      &abs_orbit, &phase, ierr);
}
```

7.40.3 Input parameters

Table 110: Input parameters for xo_orbit_abs_from_rel

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure that contains the orbit data	-	-
rel_orbit	long *		Relative orbit number		
cycle	long *		Cycle number		

7.40.4 Output parameters

Table 111: Output parameters for xo_orbit_abs_from_rel

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_abs_from_rel	long		Main status flag,		-1, 0, 1
abs_orbit	long *		Absolute orbit number		within orbit_id range
phase	long *		Phase number		
ierr[XO_ORBIT_ABS_FROM_REL]	long	all	Error status flags		

7.40.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_orbit_abs_from_rel** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_orbit_abs_from_rel** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 112: Error messages of xo_orbit_abs_from_rel function

Error type	Error message	Cause and impact	Error Code	Error No
ERR	Orbit Id. is not initialised.	Computation not performed	XO_CFI_ORBIT_ABS_FROM_REL_ORBIT_INIT_ERR	0
ERR	The orbit numbers could not be computed with the current orbit initialization.	Computation not performed	XO_CFI_ORBIT_ABS_FROM_REL_ORBIT_WRONG_MODE_ERR	1
ERR	Wrong input relative orbit and/or cycle.	Computation not performed	XO_CFI_ORBIT_ABS_FROM_REL_INPUT_PARAMETER_ERR	2

7.41 xo_orbit_abs_from_phase

7.41.1 Overview

The **xo_orbit_abs_from_phase** function retrieves from an Orbit Scenario File (previously initialised through the *orbit Id*) the absolute orbit corresponding to a given phase.

7.41.2 Calling sequence

For C programs, the call to **xo_orbit_abs_from_phase** is (input parameters are underlined, some may be input or output depending on the calling mode):

```
#include <explorer_orbit.h>
{
  xo_orbit_id  orbit_id = {NULL};
  long         abs_orbit, rel_orbit, cycle, phase;
  long         ierr[XO_NUM_ERR_ORBIT_ABS_FROM_REL], status;

  status = xo_orbit_abs_from_phase (&orbit_id,
                                    &phase,
                                    &abs_orbit,
                                    &rel_orbit, &cycle,
                                    ierr);

  /* Or, using the run_id */
  long run_id;
  status = xo_orbit_abs_from_phase_run (&run_id,
                                        &phase,
                                        &abs_orbit,
                                        &rel_orbit, &cycle,
                                        ierr);
}
```

7.41.3 Input parameters

Table 113: Input parameters for xo_orbit_abs_from_phase

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure that contains the orbit data	-	-
phase	long *		Phase number		

7.41.4 Output parameters

Table 114: Output parameters for xo_orbit_abs_from_phase

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_abs_from_phase	long		Main status flag,		-1, 0, 1
abs_orbit	long *		Absolute orbit number		within orbit_id range
rel_orbit	long *		Relative orbit number		
cycle	long *		Cycle number		
ierr[XO_ORBIT_ABS_FROM_PHASE]	long	all	Error status flags		

7.41.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_orbit_abs_from_phase** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_orbit_abs_from_phase** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 115: Error messages of xo_orbit_abs_from_phase function

Error type	Error message	Cause and impact	Error Code	Error No
ERR	Orbit Id. is not initialised.	Computation not performed	XO_CFI_ORBIT_ABS_FROM_PHASE_ORBIT_INIT_ERR	0
ERR	The orbit numbers could not be computed with the current orbit initialization.	Computation not performed	XO_CFI_ORBIT_ABS_FROM_PHASE_ORBIT_WRONG_MODE_ERR	1
ERR	Wrong input phase number.	Computation not performed	XO_CFI_ORBIT_ABS_FROM_PHASE_INPUT_PARAMETER_ERR	2

7.42 xo_osv_to_tle

7.42.1 Overview

The **xo_osv_to_tle** function generates a TLE by fitting the set of orbit state vectors stored in the orbit_id. This set of OSVs are selected from the input orbit_id for the orbit/time requested range in the following way:

- If the orbit_id mode is XO_ORBIT_INIT_USER_OSV_LIST_MODE, XO_ORBIT_INIT_ROF_MODE or XO_ORBIT_INIT_DORIS_MODE, all the OSVs in the input interval are fitted to the TLE by the Least Square method.
- In other cases (Predicted Orbit files), the input interval is populated with propagated OSV's. These OSV's are fitted to a single TLE by the Least Square method.

7.42.2 Calling sequence

For C programs, the call to **xo_osv_to_tle** is (input parameters are underlined, some may be input or output depending on the calling mode):

```
#include <explorer_orbit.h>
{
  xo_orbit_id    orbit_id = {NULL};
  xd_tle_rec     tle_rec;
  long               time_mode, time_ref, orbit0, orbit1;
      double          time0, time1;
  long           ierr[XO_NUM_ERR_OSV_TO_TLE], status;

  status = xo_osv_to_tle (&orbit_id,
                          &time_mode, &time_ref,
                          &time0, &time1,
                          &orbit0, &orbit1,
                           /* outputs */
                           &tle_rec,
                           ierr);
}
```

7.42.3 Input parameters

Table 116: Input parameters for xo_osv_to_tle

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	Structure that contains the orbit data	-	-

time_mode	long	-	time/orbit selection mode. For the XL_SEL_DEFAULT mode, the whole range of orbits stored in the orbit_id is selected	-	XO_SEL_TIME XO_SEL_ORBIT XO_SEL_DEFAUL T
time_ref	long	-	time reference (only used if time_mode is XO_SEL_TIME	-	Complete
time0	double	-	Start time	days	Start validity time for the orbit_id
time1	double	-	Output time	days	Stop validity time for the orbit_id
orbit0	long	-	Start orbit	-	First orbit stored in the orbit_id
orbit1	long	-	Stop orbit	-	Last orbit stored in the orbit_id

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

time_mode: See [LIB_SUM], section 6.2 (Time Initialization)

time_ref: See [LIB_SUM], section 6.2 (Time reference).

7.42.4 Output parameters

Table 117: Output parameters for xo_osv_to_tle

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_osv_to_tle	long	-	Main status flag	-	-1, 0, 1
tle_rec	xd_tle_rec	-	TLE record data	-	-
ierr	long	all	error array	-	-

7.42.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_osv_to_tle** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

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

Table 118: Error messages of xo_osv_to_tle function

Error type	Error message	Cause and impact	Error Code	Error No
ERR	Input orbit_id is initialised with an incorrect model	Computation not performed	XO_CFI_OSV_TO_TLE_WRONG_FILE_MODEL_ERR	0
ERR	The input time/orbit interval is not correct	Computation not performed	XO_CFI_OSV_TO_TLE_WRONG_INPUT_INTERVAL_ERR	1
ERR	Error in a time transformation	Computation not performed	XO_CFI_OSV_TO_TLE_TIME_TRANS_ERR	2
ERR	Incorrect input time mode	Computation not performed	XO_CFI_OSV_TO_TLE_WRONG_TIME_MODEL_ERR	3
ERR	Could not change from EF CS to TEME CS	Computation not performed	XO_CFI_OSV_TO_TLE_CHANGE_CS_ERR	4
ERR	Could not get keplerian elements for absolute orbit	Computation not performed	XO_CFI_OSV_TO_TLE_CART_TO_KEPLER_ERR	5
ERR	Error fitting OSVs to compute TLE	Computation not performed	XO_CFI_OSV_TO_TLE_FIT_ERR	6

7.43 xo_gen_osf_create

7.43.1 Overview

The **xo_gen_osf_create** CFI function creates a reference Orbit Scenario File (OSF) with one orbit change data structure using only user inputs in the calling interface. This data structure characterizes the reference orbit by means of the following parameters:

- Absolute orbit number
- Relative orbit number
- Cycle number
- Phase number
- Repeat cycle (days)
- Cycle length (orbits)
- Ascending crossing node longitude
- Mean local solar time of the ascending crossing node
- Mean local solar time drift (seconds per day)
- Time of the ascending crossing node (TAI, UTC and UT1)

In order to write files, xo_gen_osf_create function internally uses Data Handling functions. Please refer to [D_H_SUM], in particular sections 4.2 and 4.3, for further details.

Note: function xo_gen_osf_create is deprecated. It is recommended to use xo_gen_osf_create_2 instead.

7.43.2 Calling interface

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

```
#include <explorer_orbit.h>
{
  long sat_id;
  xl_model_id model_id = {NULL};
  xl_time_id time_id = {NULL};
  long abs_orbit_number, cycle_number, phase_number,
       repeat_cycle, cycle_length, drift_mode, version_number;
  double anx_long, inclination, mlst_drift, mlst, date;
  char output_dir[XD_MAX_STR], output_filename[XD_MAX_STR];
  char *file_class, *fh_system;
  long status, ierr[XO_ERR_VECTOR_MAX_LENGTH];
  status = xo_gen_osf_create (&sat_id, &model_id, &time_id,
                              &abs_orbit_number,
                              &cycle_number, &phase_number,
                              &repeat_cycle, &cycle_length,
                              &anx_long, &drift_mode,
```

```
                                    &inclination, &mlst_drift,
                                    &mlst, &date,
                                    output_dir, output_filename,
                                    file_class, &version_number,
                                    fh_system,
                                    ierr);

    /* Or, using the run_id */
    long run_id;

    status = xo_gen_osf_create_run (&run_id, &abs_orbit_number,
                                    &cycle_number, &phase_number,
                                    &repeat_cycle, &cycle_length,
                                    &anx_long, &drift_mode,
                                    &inclination, &mlst_drift,
                                    &mlst, &date,
                                    output_dir, output_filename,
                                    file_class, &version_number,
                                    fh_system,
                                    ierr);
}
```

7.43.3 Input parameters

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

Table 119: Input parameters of xo_gen_osf_create function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
model_id	xl_model_id	-	Model ID	-	Complete
time_id	xl_time_id*	-	Structure that contains the time correlations.	-	-
abs_orbit_number	long*	-	Orbit number in OSF first orbit change	-	>= 1
cycle_number	long*	-	Cycle number in OSF first orbit change	-	>= 1
phase_number	long*	-	Phase number in OSF first orbit change	-	>= 1
repeat_cycle	long*	-	Repeat cycle of the reference orbit	days	>= 1
cycle_length	long*	-	Cycle length of the reference orbit	orbits	>= 14

anx_long	double*	-	Reference orbit ascending node crossing longitude	deg	[-180, 180]
drift_mode	long*	-	Flag to select between drift in mean local solar time and inclination as input characterization of the reference orbit	-	[0,1]
inclination	double*	-	If *drift_mode* = *XO_NOSUNSYNC_INCLINATION* Inclination of the reference orbit	deg	[0,180]
mlst_drift	double*	-	If *drift_mode* = *XO_NOSUNSYNC_DRIFT* Drift in mean local solar time of the reference orbit: · MLST[N+1]=MLST[N]+MLST drift	seconds/day	TBD
mlst	double*	-	Mean local solar time at ascending node	decimal hours	[0,24)
date	double*	-	ANX date	decimal days	-
output_dir	char*	-	Directory where the resulting OSF is written (if empty (i.e. ““), the current directory is used)	-	-
output_filename	char*	-	Output OSF name if empty (i.e. ““), the software will generate the filename according to file name specification pre sented in [FORMATS]. In such case, the generated name is returned in this variable	-	-
file_class	char*	-	File class for output Orbit file	-	-
version_number	long*	-	Version number of output Orbit file	-	>= 1
fh_system	char*	-	System field of the output Orbit file fixed header	-	-

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

- Satellite ID: sat_id.
- Drift mode: mlst_drift.

This CFI can generate Orbit Scenario Files for both sun-synchronous orbits and quasi-sun-synchronous orbits.

Use `drift_mode=XO_NOSUNSYNC_DRIFT` and `mlst_drift = 0.0` for a sun-synchronous orbit.

Use any other combination for the general case of quasi-sun-synchronous orbit.

7.43.4 Output parameters

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

Table 120: Output parameters of xo_gen_osf_create function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
output_filename	char*	-	Name for output file. This is only an output parameter when it is empty (i.e. ""; see description of this parameter in Table 119)	-	-
ierr[XO_ERR_VECTOR_MAX_LENGTH]	long	all	Status vector	-	-

7.43.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_gen_osf_create** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_gen_osf_create** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 121: Error messages of xo_gen_osf_create function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Wrong input values	Wrong value of one or more of the following input parameters: abs_orbit_number, cycle_number, phase_number, repeat_cycle, cycle_length, mlst Computation not performed	XO_CFI_GEN_OSF_CREATE_INPUTS_ERR	0
ERR	Time ID is not initialized	Time correllations were not initialized. Computation not performed	XO_CFI_GEN_OSF_CREATE_TIME_INIT_ERR	1

ERR	Memory allocation error	Memory allocation error for the orbit change data structure Computation not performed	XO_CFI_GEN_OSF_CREATE_ALLOC_ERR	2
ERR	Wrong drift mode	Wrong drift mode flag value for charaterization of non-sun.synchronous orbits Computation not performed	XO_CFI_GEN_OSF_CREATE_DRIFT_MODE_ERR	3
ERR	Error calculating MLST drift	Error calculating MLST drift from inclination Computation not performed	XO_CFI_GEN_OSF_CREATE_DRIFT_CALC_ERR	4
ERR	Error calculating UTC of ANX	Error calculating the UTC time of the orbit ascending node Computation not performed	XO_CFI_GEN_OSF_CREATE_UTC_CALC_ERR	5
ERR	Error calculating TAI of ANX	Error calculating the TAI time of the orbit ascending node Computation not performed	XO_CFI_GEN_OSF_CREATE_TAI_CALC_ERR	6
ERR	Error calculating UT1 of ANX	Error calculating the UT1 time of the orbit ascending node Computation not performed	XO_CFI_GEN_OSF_CREATE_UT1_CALC_ERR	7
ERR	Error calculating the Fixed Header data	Error getting the data for the Fixed Header. Computation not performed	XO_CFI_GEN_OSF_CREATE_GET_FH_ERR	8
ERR	Error getting schema	Computation not performed	XO_CFI_GEN_OSF_CREATE_GET_SCHEMA_ERR	9
ERR	Error writing file to disk	Error writing the data structure to a file on disk Computation not performed	XO_CFI_GEN_OSF_CREATE_WRITE_ERR	10
WARN	Function xo_gen_osf_create is deprecated. Use xo_gen_osf_create_2 instead	Computation performed	XO_CFI_GEN_OSF_CREATE_DEPRECATED_WARN	11

7.43.6 Executable Program

The **gen_osf_create** executable program can be called from a Unix shell as:

```
gen_osf_create          -sat satellite_name
                        -orbit abs_orbit_number
                        -cyc cycle_number
                        -pha phase_number
                        -repcyc repeat_cycle(days)
                        -cyclen cycle_length(orbits)
                        -anx anx_long(deg)
                        { -mlstdr mlst_drift | -inc inclination }
                        -mlst mlst
                        -date anx_date
                        [-phinc]
                        [-dir dir_name] (current directory by default)
                        [-osf output_filename] (default: name generated automatically)
                        [-flcl file_class] (empty string by default)
                        [-vers version] (version = 1 by default)
                       [-eoffs ffs_version] (Earth Observation File Format Standard Version)
                        [-fhsys fh_system] (empty string by default)
                        [ -v ]
                        [ -xl_v ]
                        [ -xo_v ]
                        [ -help ]
                        [ -show]
                        [ -with_xslt ] (add xslt reference with default style sheet)
                         { (-tai TAI_time -gps GPS_time -utc UTC_time -ut1 UT1_time) |
                           (-tmod time_model -tfile time_file -trid time_reference
                            {(-tm0 time0 -tm1 time1) | (-orb0 orbit0 -orb1 orbit1) } ) }
```

Note that:

- Order of parameters does not matter.
- Bracketed parameters are not mandatory.
- Options between curly brackets and separated by a vertical bar are mutually exclusive.

- [**-phinc**] option for phase_increment. Default value for phase_increment is XO_NO_PHASE_INCREMENT. When the option is written, phase_increment is XO_PHASE_INCREMENT.
- [**-xl_v**] option for EO_LIB Verbose mode.
- [**-xo_v**] option for EO_ORBIT Verbose mode.
- [**-v**] option for Verbose mode for all libraries (default is Silent).
- [**-show**] displays the inputs of the function and the results.
- Possible values for *satellite_name*: ERS1, ERS2, ENVISAT, METOP1 , METOP2, METOP3, CRYOSAT, ADM, GOCE, SMOS, TERRASAR, EARTHCARE, SWARM_A, SWARM_B, SWARM_C, SENTINEL_1A, SENTINEL_1B, SENTINEL_2, SENTINEL_3, SENTINEL_1C, SENTINEL_2A, SENTINEL_2B, SENTINEL_2C, SENTINEL_3A, SENTINEL_3B, SENTINEL_3C, SEOSAT, JASON_CSA, JASON_CSB, METOP_SG_A1, METOP_SG_A2, METOP_SG_A3, METOP_SG_B1, METOP_SG_B2, METOP_SG_B3, SENTINEL_5P, BIOMASS, SENTINEL_5, SAOCOM_CS, FLEX, SENTINEL_6A, SENTINEL_6B, CIMR, ROSE-L, CHIME, CRISTAL, CO2M, LSTM, FORUM, TRUTHS, GENERIC, GENERIC_GEO, MTG.
- Possible values for *time_model*: USER, NONE , IERS_B_PREDICTED, IERS_B_RESTITUTED, FOS_PREDICTED, FOS_RESTITUTED, DORIS_PRELIMINARY, DORIS_PRECISE, DORIS_NAVIGATOR, OSF.
- Possible values for *ffs_version*: 0 (Default FFS), 1 (FFS version 1), 2 (FFS version 2), 3 (FFS version 3).
- Possible values for *time_reference*: UNDEF, TAI, UTC, UT1, GPS.
- The last three lines of parameters are used to initialize the time references. In order to do this, only one set of parameters should be introduced:
  - TAI, GPS, UTC and UT1 input times (as in xl_time_ref_init)
  - A file with time reference data, the time mode, the time reference name and a time range (as in xl_time_ref_init_file)

Example:

```
gen_osf_create -sat CRYOSAT -orbit 1 -cyc 1 -pha 1 -repcyc 2
               -cyclen 29 -inc 92 -mlst 21 -date 790 -anx 130
               -dir ./gen_osf -osf mpl_orb_sc_at_302
               -tai -1100.1 -utc -1100.099595
               -ut1 -1100.0995914352 -gps -1100.0997801
```

7.44 xo_gen_osf_create_2

7.44.1 Overview

The **xo_gen_osf_create_2** behaves the same way as **xo_gen_osf_create** with the difference that it supports Mean Local Solar Time non-linear parameters as input.

In order to write files, xo_gen_osf_create_2 function internally uses Data Handling functions. Please refer to [D_H_SUM], in particular sections 4.2 and 4.3, for further details.

7.44.2 Calling interface

The calling interface of the **xo_gen_osf_create_2** CFI function is the following (input parameters are underlined):

```
#include <explorer_orbit.h>
{
  long sat_id;
  xl_model_id model_id = {NULL};
  xl_time_id time_id = {NULL};
  xo_mission_info mission_info;
  double date;
  xo_ref_orbit_info ref_orbit_info;
  char output_dir[XD_MAX_STR], output_filename[XD_MAX_STR];
  char *file_class, *fh_system;
  long status, ierr[XO_ERR_VECTOR_MAX_LENGTH];
  status = xo_gen_osf_create_2(&sat_id, &model_id, &time_id,
                               &date,
                               &mission_info, &ref_orbit_info,
                               output_dir, output_filename,
                               file_class, &version_number,
                               fh_system,
                               ierr);

  /* Or, using the run_id */
  long run_id;

  status = xo_gen_osf_create_run_2 (&run_id, &date,
                               &mission_info, &ref_orbit_info,
                               output_dir, output_filename,
                               file_class, &version_number,
                               fh_system,
                               ierr);
}
```

7.44.3 Input parameters

The **xo_gen_osf_create2** CFI function has the following input parameters:

Table 122: Input parameters of xo_gen_osf_create_2 function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
model_id	xl_model_id	-	Model ID	-	Complete
time_id	xl_time_id*	-	Structure that contains the time correlations.	-	-
date	double*	-	ANX date	decimal days	-
mission_info	xo_mission_info*	-	Information about the orbit (equivalent to orbit information in xo_gen_osf_create, see table 119)	-	-
ref_orbit_info	xo_ref_orbit_info*	-	Struct with inputs for the function. The parameters are equivalent to the ones in xo_orbit_init_def (table 119) but also MLST non-linear terms can be introduced.	-	-
output_dir	char*	-	Directory where the resulting OSF is written (if empty (i.e. ““), the current directory is used)	-	-
output_filename	char*	-	Output OSF name If empty (i.e. ““), the software will generate the filename according to file name specification pre sented in [FORMATS]. In such case, the generated name is returned in this variable	-	-
file_class	char*	-	File class for output Orbit file	-	-
version_number	long*	-	Version number of output Orbit file	-	>= 1
fh_system	char*	-	System field of the output Orbit file fixed header	-	-

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

- Satellite ID: sat_id.
- Drift mode: mlst_drift.

This CFI can generate Orbit Scenario Files for both sun-synchronous orbits and quasi-sun-synchronous orbits.

Use `drift_mode=XO_NOSUNSYNC_DRIFT`, `mlst_drift = 0.0` and zero MLST non linear parameters for a sun-synchronous orbit.

Use any other combination for the general case of quasi-sun-synchronous orbit.

7.44.4 Output parameters

The output parameters of the **xo_gen_osf_create_2** CFI function are:

Table 123: Output parameters of xo_gen_osf_create_2 function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
output_filename	char*	-	Name for output file. This is only an output parameter when it is empty (i.e. “”; see description of this parameter in Table 122)	-	-
ierr[XO_ERR_VECTOR_MAX_LENGTH]	long	all	Status vector	-	-

7.44.5 Warnings and errors

Errors and warnings are the same as for function **xo_gen_osf_create** (see section 180).

7.45 xo_gen_osf_append_orbit_change

7.45.1 Overview

The **xo_gen_osf_append_orbit_change** CFI function appends an orbit change to an existing reference Orbit Scenario File (OSF). The user must provide in the calling interface the name of the existing OSF, the parameters describing the new orbit change and the output file name where the old OSF with the appended orbit change will be written. No output file is generated if the resulting orbit is discontinuous in terms of ascending node longitude, mean local solar time.

Note: function xo_gen_osf_append_orbit_change is deprecated. It is recommended to use xo_gen_osf_append_orbit_change_2 instead.

In order to read and write files, xo_gen_osf_append_orbit_change function internally uses Data Handling functions. Please refer to [D_H_SUM], in particular sections 4.2 and 4.3, for further details.

7.45.2 Calling interface

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

```
#include <explorer_orbit.h>
{
  long sat_id;
  xl_time_id time_id = {NULL};
  xl_model_id model_id = {NULL};
  long abs_orbit_number,        repeat_cycle, cycle_length,
       drift_mode, phase_increment, version_number;
  double anx_long, inclination, mlst_drift, mlst;
  char input_filename[XD_MAX_STR],
       output_dir[XD_MAX_STR], output_filename[XD_MAX_STR];
  char *file_class, *fh_system;
  long status, ierr[XO_ERR_VECTOR_MAX_LENGTH];
  status = xo_gen_osf_append_orbit_change (&sat_id, &model_id,
                                &time_id,
                                &input_filename,
                                    &abs_orbit_number,
                                &repeat_cycle, &cycle_length,
                                &anx_long, &drift_mode,
                                &inclination, &mlst_drift,
                                &mlst, &phase_increment,
                                output_dir, output_filename,
                                file_class, &version_number,
                                fh_system,
                                ierr);

  /* Or, using the run_id */
```

```
    long run_id;

    status = xo_gen_osf_append_orbit_change_run (&run_id,
                                    &input_filename,
                                        &abs_orbit_number,
                                    &repeat_cycle, &cycle_length,
                                    &anx_long, &drift_mode,
                                    &inclination, &mlst_drift,
                                    &mlst, &phase_increment,
                                    output_dir, output_filename,
                                    file_class, &version_number,
                                    fh_system,
                                     ierr);
}
```

7.45.3 Input parameters

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

Table 124: Input parameters of xo_gen_osf_append_orbit_change function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
model_id	long *	-	Model ID	-	-
time_id	xl_time_id*	-	Structure that contains the time correlations.	-	-
input_filename	char*	-	Input OSF to which the orbit change is appended		
abs_orbit_number	long*	-	Absolute orbit number of the new orbit change	-	> abs orbit number in input OSF last orbit change
repeat_cycle	long*	-	Repeat cycle of the new reference orbit	days	>= 1
cycle_length	long*	-	Cycle length of the new reference orbit	orbits	>= 14
anx_long	double*	-	Requested orbit ascending node crossing longitude	deg	[-180, 180]
drift_mode	long*	-	Flag to select between drift in mean local solar time and inclination as input characterization of the reference orbit	-	[0,1]

inclination	double*	-	If *drift_mode* = *XO_NOSUNSYNC_INCLINATION* Inclination of the reference orbit	deg	[0,180]
mlst_drift	double*	-	If *drift_mode* = *XO_NOSUNSYNC_DRIFT* Drift in mean local solar time of the reference orbit: · MLST[N+1]=MLST[N]+MLST drift	seconds/day	TBD
mlst	double*	-	Mean local solar time at ascending node	decimal hours	[0,24)

phase_increment	long*	-	If 1 then
			phase [N+1] = phase [N] + 1
			If 0 then
			phase [N+1] = phase [N]
				-	[0, 1]

output_dir	char*	-	Directory where the resulting OSF is written (if empty (i.e. ““), the current directory is used)	-	-
output_filename	char*	-	Output OSF name If empty (i.e. ““), the software will generate the filename according to file name specification pre sented in [FORMATS]. In such case, the generated name is returned in this variable	-	-
file_class	char*	-	File class for output Orbit file	-	-
version_number	long*	-	Version number of output Orbit file	-	>= 1
fh_system	char*	-	System field of the output Orbit file fixed header	-	-

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

- Satellite ID: sat_id. See [GEN_SUM].
- Drift mode: mlst_drift.
- Phase increment.

This CFI can append orbit changes for both sun-synchronous orbits and quasi-sun-synchronous orbits.

Use `drift_mode=XO_NOSUNSYNC_DRIFT` and `mlst_drift = 0.0` for a sun-synchronous orbit.

Use any other combination for the general case of quasi-sun-synchronous orbit.

7.45.4 Output parameters

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

Table 125: Output parameters of xo_gen_osf_append_orbit_change function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
output_filename	char*	-	Name for output file. This is only an output parameter when it is empty (i.e. ""; see description of this parameter in Table 124)	-	-
ierr[XO_ERR_VECTOR_MAX_LENGTH]	long	all	Status vector	-	-

7.45.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_gen_osf_append_orbit_change** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_gen_osf_append_orbit_change** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 126: Error messages of xo_gen_osf_append_orbit_change function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Wrong input values	Wrong value of one or more of the following input parameters: abs_orbit_number, repeat_cycle, cycle_length, mlst, phase_increment Computation not performed	XO_CFI_GEN_OSF_APPEND_INPUTS_ERR	0
ERR	Time ID is not initialized	Time correlations were not initialized. Computation not performed	XO_CFI_GEN_OSF_APPEND_TIME_INIT_ERR	1
ERR	Cannot read input OSF	Computation not performed	XO_CFI_GEN_OSF_APPEND_READ_IN_OSF_ERR	2

WARN	Very large ANX long jump (%lf deg)	Requested ANX long leads to an orbit discontinuity. Computation performed	XO_CFI_GEN_OSF_APPEND_ANX_LONG_WARN	3
WARN	Very large MLST jump (%lf hours)	Requested MLST leads to an orbit discontinuity. Computation performed	XO_CFI_GEN_OSF_APPEND_MLST_WARN	4
ERR	Wrong drift mode	Wrong drift mode flag value for charaterization of non-sun.synchronous orbits Computation not performed	XO_CFI_GEN_OSF_APPEND_DRIFT_MODE_ERR	5
ERR	Error calculating MLST drift	Error calculating MLST drift from inclination Computation not performed	XO_CFI_GEN_OSF_APPEND_DRIFT_CALC_ERR	6
ERR	Error calculating UTC of ANX	Error calculating the UTC time of the orbit ascending node Computation not performed	XO_CFI_GEN_OSF_APPEND_UTC_CALC_ERR	7
ERR	Error calculating TAI of ANX	Error calculating the TAI time of the orbit ascending node Computation not performed	XO_CFI_GEN_OSF_APPEND_TAI_CALC_ERR	8
ERR	Error calculating UT1 of ANX	Error calculating the UT1 time of the orbit ascending node Computation not performed	XO_CFI_GEN_OSF_APPEND_UT1_CALC_ERR	9
ERR	Memory allocation error	Computation not performed	XO_CFI_GEN_OSF_APPEND_ALLOC_ERR	10
ERR	Error calculating the Fixed Header data	Computation not performed	XO_CFI_GEN_OSF_APPEND_GET_FH_ERR	11
ERR	Error writing file to disk	Error writing the data structure to a file on disk Computation not performed	XO_CFI_GEN_OSF_APPEND_WRITE_ERR	12

7.45.6 Executable Program

The **gen_osf_append_orbit_change** executable program can be called from a Unix shell as:

```
gen_osf_append_orbit_change  -sat satellite_name
                             -inosf input_filename
                             -orbit abs_orbit_number
```

-repcyc repeat_cycle(days)
-cyclen cycle_length(orbits)
-anx anx_long(deg)
{ **-mlstdr** mlst_drift | **-inc** inclination }
-mlst mlst
[**-phinc**]
[**-dir** output_dir] (current directory by default)
[**-osf** output_filename] (default: name generated automatically)
[**-flcl** file_class] (empty string by default)
[**-vers** version] (version = 1 by default)
[**-eoffs** ffs_version] (Earth Observation File Format Standard Version)
[**-fhsys** fh_system] (empty string by default)
[**-v**]
[**-xl_v**]
[**-xo_v**]
[**-help**]
[**-show**]
[**-with_xslt]** (add xslt reference with default style sheet)
{ (**-tai** TAI_time **-gps** GPS_time **-utc** UTC_time **-ut1** UT1_time) |
(**-tmod** time_model **-tfile** time_file **-trid** time_reference
{(**-tm0** time0 **-tm1** time1) | (**-orb0** orbit0 **-orb1** orbit1) }) **}**

Note that:

- Order of parameters does not matter.
- Bracketed parameters are not mandatory.
- Options between curly brackets and separated by a vertical bar are mutually exclusive.
- [**-phinc**] option for phase_increment. Default value for phase_increment is xo_NO_PHASE_INCREMENT. When the option is written, phase_increment is xo_PHASE_INCREMENT.
- [**-xl_v**] option for EO_LIB Verbose mode.
- [**-xo_v**] option for EO_ORBIT Verbose mode.
- [**-v**] option for Verbose mode for all libraries (default is Silent).
- [**-show**] displays the inputs of the function and the results.
- Possible values for *satellite_name*: ERS1, ERS2, ENVISAT, METOP1 , METOP2, METOP3, CRYOSAT, ADM, GOCE, SMOS, TERRASAR, EARTHCARE, SWARM_A, SWARM_B,

SWARM_C, SENTINEL_1A, SENTINEL_1B, SENTINEL_2, SENTINEL_3, SENTINEL_1C, SENTINEL_2A, SENTINEL_2B, SENTINEL_2C, SENTINEL_3A, SENTINEL_3B, SENTINEL_3C, SEOSAT, JASON_CSA, JASON_CSB, METOP_SG_A1, METOP_SG_A2, METOP_SG_A3, METOP_SG_B1, METOP_SG_B2, METOP_SG_B3, SENTINEL_5P, BIOMASS, SENTINEL_5, SAOCOM_CS, FLEX, SENTINEL_6A, SENTINEL_6B, CIMR, ROSE-L, CHIME, CRISTAL, CO2M, LSTM, FORUM, TRUTHS, GENERIC, GENERIC_GEO, MTG.

- Possible values for *time_model*: USER, NONE , IERS_B_PREDICTED, IERS_B_RESTITUTED, FOS_PREDICTED, FOS_RESTITUTED, DORIS_PRELIMINARY, DORIS_PRECISE, DORIS_NAVIGATOR, OSF.
- Possible values for *ffs_version*: 0 (Default FFS), 1 (FFS version 1), 2 (FFS version 2), 3 (FFS version 3).
- Possible values for *time_reference*: UNDEF, TAI, UTC, UT1, GPS.
- The last three lines of parameters are used to initialize the time references. In order to do this, only one set of parameters should be introduced:

– TAI, GPS, UTC and UT1 input times (as in xl_time_ref_init)

– A file with time reference data, the time mode, the time reference name and a time range (as in xl_time_ref_init_file)

Example:

```
gen_osf_append_orbit_change -sat CRYOSAT
     -inosf CS_TEST_MPL_ORBREF_20020301T122001_99999999T999999_0001.EEF
     -orbit 30 -repcyc 366 -cyclen 5344 -anx 129.9986 -mlst 20.90083
     -inc 92 -dir ./gen_osf -osf mpl_orb_sc_at_303
     -tai -1100.1 -utc -1100.099595
     -ut1 -1100.0995914352 -gps -1100.0997801
```

7.46 xo_gen_osf_append_orbit_change_2

7.46.1 Overview

The **xo_gen_osf_append_orbit_change_2** CFI function appends an orbit change to an existing reference Orbit Scenario File (OSF) in the same way as **xo_gen_osf_append_orbit_change**, but allowing the introduction and management of Mean Local Solar Time non linear terms.

In order to read and write files, xo_gen_osf_append_orbit_change_2 function internally uses Data Handling functions. Please refer to [D_H_SUM], in particular sections 4.2 and 4.3, for further details.

7.46.2 Calling interface

The calling interface of the **xo_gen_osf_append_orbit_change_2** CFI function is the following (input parameters are underlined):

```
#include <explorer_orbit.h>
{
  long sat_id;
  xl_time_id time_id = {NULL};
  xl_model_id model_id = {NULL};
  long abs_orbit_number, phase_increment, version_number;
  char input_filename[XD_MAX_STR],
       output_dir[XD_MAX_STR], output_filename[XD_MAX_STR];
  char *file_class, *fh_system;
  long status, ierr[XO_ERR_VECTOR_MAX_LENGTH];
  status = xo_gen_osf_append_orbit_change_2 (&sat_id, &model_id,
                                  &time_id,
                                  &input_filename,
                                      &abs_orbit_number,
                                  &ref_orbit_info, &phase_increment,
                                  output_dir, output_filename,
                                  file_class, &version_number,
                                  fh_system,
                                  ierr);

  /* Or, using the run_id */
  long run_id;

  status = xo_gen_osf_append_orbit_change_run_2 (&run_id,
                                  &input_filename,
                                      &abs_orbit_number,
                                  &&ref_orbit_info, &phase_increment,
                                  output_dir, output_filename,
                                  file_class, &version_number,
                                  fh_system,
```

```
                                                    ierr);
    }
```

7.46.3 Input parameters

The **xo_gen_osf_append_orbit_change_2** CFI function has the following input parameters:

Table 127: Input parameters of xo_gen_osf_append_orbit_change_2 function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
model_id	long *	-	Model ID	-	-
time_id	xl_time_id*	-	Structure that contains the time correlations.	-	-
input_filename	char*	-	Input OSF to which the orbit change is appended		
abs_orbit_number	long*	-	Absolute orbit number of the new orbit change	-	> abs orbit number in input OSF last orbit change
ref_orbit_info	xo_ref_orbit_info*	-	Struct with inputs for the function. The parameters are equivalent to the ones in xo_orbit_init_def (see table 124) but also MLST non linear terms can be introduced.	-	-

phase_increment	long*	-	If 1 then
			phase [N+1] = phase [N] + 1
			If 0 then
			phase [N+1] = phase [N]
				-	[0, 1]

output_dir	char*	-	Directory where the resulting OSF is written (if empty (i.e. ““), the current directory is used)	-	-
output_filename	char*	-	Output OSF name if empty (i.e. ““), the software will generate the filename according to file name specification pre sented in [FORMATS]. In such case, the generated name is returned in this variable	-	-
file_class	char*	-	File class for output Orbit file	-	-
version_number	long*	-	Version number of output Orbit file	-	>= 1
fh_system	char*	-	System field of the output Orbit	-	-

			file fixed header		

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

- Satellite ID: sat_id. See [GEN_SUM].
- Drift mode: mlst_drift.
- Phase increment.

This CFI can append orbit changes for both sun-synchronous orbits and quasi-sun-synchronous orbits.

Use `drift_mode=XO_NOSUNSYNC_DRIFT, mlst_drift = 0.0` and zero MLST non linear terms for a sun-synchronous orbit.

Use any other combination for the general case of quasi-sun-synchronous orbit.

7.46.4 Output parameters

The output parameters of the **xo_gen_osf_append_orbit_change_2** CFI function are:

Table 128: Output parameters of xo_gen_osf_append_orbit_change_2 function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
output_filename	char*	-	Name for output file. This is only an output parameter when it is empty (i.e. “”; see description of this parameter in Table 127)	-	-
ierr[XO_ERR_VECTOR_MAX_LENGTH]	long	all	Status vector	-	-

7.46.5 Warnings and errors

Errors and warning are the same as in the function **xo_gen_osf_append_orbit_change**. See section 190 for details.

7.47 xo_gen_osf_change_repeat_cycle

7.47.1 Overview

Given a reference orbit from an existing OSF and a new target orbit (repeat cycle, cycle length, ascending node longitude and inclination or mean local solar time drift), the **xo_gen_osf_change_repeat_cycle** CFI function finds an optimum orbit change such that the target orbit can be reached from the found orbit change. This function will write a new OSF with the found orbit change appended to the content of the old OSF.

In order to read and write files, xo_gen_osf_change_repeat_cycle function internally uses Data Handling functions. Please refer to [D_H_SUM], in particular sections 4.2 and 4.3, for further details.

Note: function xo_gen_osf_change_repeat_cycle is deprecated. It is recommended to use xo_gen_osf_change_repeat_cycle_2 instead.

7.47.2 Calling interface

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

```
#include <explorer_orbit.h>
{
  long sat_id;
  xl_model_id model_id = {NULL};
  xl_time_id time_id = {NULL};
  long abs_orbit_number, search_direction, repeat_cycle,
       cycle_length, drift_mode, phase_increment, version_number;
  double anx_long, inclination, mlst_drift;
  char input_filename[XD_MAX_STR],
       output_dir[XD_MAX_STR], output_filename[XD_MAX_STR];
  char *file_class, *fh_system;
  long status, ierr[XO_ERR_VECTOR_MAX_LENGTH];

  status = xo_gen_osf_change_repeat_cycle (&sat_id, &model_id,
                                  &time_id, &input_filename,
                                 &abs_orbit_number,
                                  &search_direction,
                                  &repeat_cycle, &cycle_length,
                                  &anx_long, &drift_mode,
                                  &inclination, &mlst_drift,
                                  &phase_increment,
                                  output_dir, output_filename,
                                  file_class, &version_number,
                                  fh_system,
                                   ierr);
```

```
    /* Or, using the run_id */
    long run_id;

    status = xo_gen_osf_change_repeat_cycle_run (&run_id,
                                &input_filename, &abs_orbit_number,
                                &search_direction,
                                &repeat_cycle, &cycle_length,
                                &anx_long, &drift_mode,
                                &inclination, &mlst_drift,
                                &phase_increment,
                                output_dir, output_filename,
                                file_class, &version_number,
                                fh_system,
                                 ierr);
}
```

7.47.3 Input parameters

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

Table 129: Input parameters of xo_gen_osf_change_repeat_cycle function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
model_id	xl_model_id*	-	Model ID	-	-
time_id	xl_time_id*	-	Structure that contains the time correlations.	-	-
input_filename	char*	-	Input OSF to which the orbit change is appended		
abs_orbit_number	long*	-	Absolute orbit number from which the optimum transition search starts	-	> abs orbit number in input OSF last orbit change
search_direction	long*	-	Search for optimum transition after or before abs_orbit_number		{-1, 1}
repeat_cycle	long*	-	Repeat cycle of the new reference orbit	days	>= 1
cycle_length	long*	-	Cycle length of the new reference orbit	orbits	>= 14
anx_long	double*	-	Target orbit ascending node crossing longitude	deg	[-180, 180]

drift_mode	long*	-	Flag to select between drift in mean local solar time and inclination as input characterization of the reference orbit	-	[0,1]
inclination	double*	-	If *drift_mode* = *XO_NOSUNSYNC_INCLINATION* Inclination of the reference orbit	deg	[0,180]
mlst_drift	double*	-	If *drift_mode* = *XO_NOSUNSYNC_DRIFT* Drift in mean local solar time of the reference orbit: · MLST[N+1]=MLST[N]+MLST drift	seconds/day	TBD

phase_increment	long*	-	If 1 then
			phase [N+1] = phase [N] + 1
			If 0 then
			phase [N+1] = phase [N]
				-	[0, 1]

output_dir	char*	-	Directory where the resulting OSF is written (if NULL, the cur rent directory is used)	-	-
output_filename	char*	-	Output OSF name If empty (i.e. ““), the software will generate the filename according to file name specification presented in [FORMATS]. In such case, the generated name is returned in this variable	-	-
file_class	char*	-	File class for output Orbit file	-	-
version_number	long*	-	Version number of output Orbit file	-	>= 1
fh_system	char*	-	System field of the output Orbit file fixed header	-	-

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

- Satellite ID: sat_id.
- Search direction.
- Drift mode: mlst_drift.
- Phase increment.

This CFI can append orbit changes for both sun-synchronous orbits and quasi-sun-synchronous orbits.

Use `drift_mode=XO_NOSUNSYNC_DRIFT` and `mlst_drift = 0.0` for a sun-synchronous orbit.

Use any other combination for the general case of quasi-sun-synchronous orbit.

7.47.4 Output parameters

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

Table 130: Output parameters of xo_gen_osf_change_repeat_cycle function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
output_filename	char*	-	Name for output file. This is only an output parameter when it is empty (i.e. “”; see description of this parameter in Table 129)	-	-
ierr[XO_ERR_VECTOR_MAX_LENGTH]	long	all	Status vector	-	-

7.47.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_gen_osf_change_repeat_cycle** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_gen_osf_change_repeat_cycle** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 131: Error messages of xo_gen_osf_change_repeat_cycle function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Wrong input values	Wrong value of one or more of the following input parameters: abs_orbit_number, search_direction, repeat_cycle, cycle_length, phase_increment Computation not performed	XO_CFI_GEN_OSF_CHANGE_INPUTS_ERR	0
ERR	Time ID is not initialized	Computation not performed	XO_CFI_GEN_OSF_CHANGE_TIME_INIT_ERR	1
ERR	Cannot read input OSF	Computation not performed	XO_CFI_GEN_OSF_CHANGE_READ_IN_OSF_ERR	2

ERR	Wrong drift mode	Wrong drift mode flag value for charaterization of non-sun-synchronous orbits Computation not performed	XO_CFI_GEN_OSF_CHANGE_DRIFT_MODE_ERR	3
ERR	Error calculating MLST drift	Error calculating MLST drift from inclination Computation not performed	XO_CFI_GEN_OSF_CHANGE_DRIFT_CALC_ERR	4
ERR	No transition found	No optimum transition found keeping orbit continuity Computation not performed	XO_CFI_GEN_OSF_CHANGE_NO_TRANSITION_ERR	5
ERR	Error calculating UTC of ANX	Error calculating the UTC time of the orbit ascending node Computation not performed	XO_CFI_GEN_OSF_CHANGE_UTC_CALC_ERR	6
ERR	Error calculating TAI of ANX	Error calculating the TAI time of the orbit ascending node Computation not performed	XO_CFI_GEN_OSF_CHANGE_TAI_CALC_ERR	7
ERR	Error calculating UT1 of ANX	Error calculating the UT1 time of the orbit ascending node Computation not performed	XO_CFI_GEN_OSF_CHANGE_UT1_CALC_ERR	8
ERR	Memory allocation error	Computation not performed	XO_CFI_GEN_OSF_CHANGE_ALLOC_ERR	9
ERR	Error calculating the Fixed Header data	Computation not performed	XO_CFI_GEN_OSF_CHANGE_GET_FH_ERR	10
ERR	Error writing file to disk	Error writing the data structure to a file on disk Computation not performed	XO_CFI_GEN_OSF_CHANGE_WRITE_ERR	11
WARN	Function xo_gen_osf_change_repeat_cycle is deprecated. Use xo_gen_osf_change_repeat_cycle_2 instead	Computation performed	XO_GEN_OSF_CHANGE_DEPRECATED_WARN	12
ERR	Error in selecting schema	Computation not performed	XO_CFI_GEN_OSF_CHANGE_SELECT_SCHEMA_ERR	13

7.47.6 Executable Program

The **gen_osf_change_repeat_cycle** executable program can be called from a Unix shell as:

gen_osf_change_repeat_cycle **-sat** satellite_name

-inosf input_filename
-orbit abs_orbit_number
[**-back**]
-repcyc repeat_cycle(days)
-cyclen cycle_length(orbits)
-anx anx_long(deg)
{ **-mlstdr** mlst_drift | **-inc** inclination }
[**-phinc**]
[**-dir** output_dir] (current directory by default)
[**-osf** output_filename] (default: name generated automatically)
[**-flcl** file_class] (empty string by default)
[**-vers** version] (version = 1 by default)
[**-eoffs** ffs_version] (Earth Observation File Format Standard Version)
[**-fhsys** fh_system] (empty string by default)
[**-v**]
[**-xl_v**]
[**-xo_v**]
[**-help**]
[**-show**]
[**-with_xslt]** (add xslt reference with default style sheet)
{ (**-tai** TAI_time **-gps** GPS_time **-utc** UTC_time **-ut1** UT1_time) |
(**-tmod** time_model **-tfile** time_file **-trid** time_reference
{(**-tm0** time0 **-tm1** time1) | (**-orb0** orbit0 **-orb1** orbit1) }) **}**

Note that:

- Order of parameters does not matter.
- Bracketed parameters are not mandatory.
- Options between curly brackets and separated by a vertical bar are mutually exclusive.
- [**-back**] option for search_direction. Default value is XO_SEARCH_FORWARD. When the option is written, search_direction value is XO_SEARCH_BACKWARD.
- [**-phinc**] option for phase_increment. Default value is xo_NO_PHASE_INCREMENT. When the option is written, phase_increment value is xo_PHASE_INCREMENT.
- [**-xl_v**] option for EO_LIB Verbose mode.
- [**-xo_v**] option for EO_ORBIT Verbose mode.
- [**-v**] option for Verbose mode for all libraries (default is Silent).

- [**-show**] displays the inputs of the function and the results.
- Possible values for *satellite_name*: ERS1, ERS2, ENVISAT, METOP1 , METOP2, METOP3, CRYOSAT, ADM, GOCE, SMOS, TERRASAR, EARTHCARE, SWARM_A, SWARM_B, SWARM_C, SENTINEL_1A, SENTINEL_1B, SENTINEL_2, SENTINEL_3, SENTINEL_1C, SENTINEL_2A, SENTINEL_2B, SENTINEL_2C, SENTINEL_3A, SENTINEL_3B, SENTINEL_3C, SEOSAT, JASON_CSA, JASON_CSB, METOP_SG_A1, METOP_SG_A2, METOP_SG_A3, METOP_SG_B1, METOP_SG_B2, METOP_SG_B3, SENTINEL_5P, BIOMASS, SENTINEL_5, SAOCOM_CS, FLEX, SENTINEL_6A, SENTINEL_6B, CIMR, ROSE-L, CHIME, CRISTAL, CO2M, LSTM, FORUM, TRUTHS, GENERIC, GENERIC_GEO, MTG.
- Possible values for *time_model*: USER, NONE , IERS_B_PREDICTED, IERS_B_RESTITUTED, FOS_PREDICTED, FOS_RESTITUTED, DORIS_PRELIMINARY, DORIS_PRECISE, DORIS_NAVIGATOR, OSF.
- Possible values for *ffs_version*: 0 (Default FFS), 1 (FFS version 1), 2 (FFS version 2), 3 (FFS version 3).
- Possible values for *time_reference*: UNDEF, TAI, UTC, UT1, GPS.
- The last three lines of parameters are used to initialize the time references. In order to do this, only one set of parameters should be introduced:
  - TAI, GPS, UTC and UT1 input times (as in xl_time_ref_init)
  - A file with time reference data, the time mode, the time reference name and a time range (as in xl_time_ref_init_file)

Example:

```
gen_osf_change_repeat_cycle -sat CRYOSAT
     -inosf CS_TEST_MPL_ORBREF_20020301T122001_99999999T999999_0001.EEF
     -orbit 400 -repcyc 369 -cyclen 5344 -anx 286.524398  -inc 92
     -dir ./gen_osf -osf mpl_orb_sc_at_304
     -tai -1100.1 -utc -1100.099595
     -ut1 -1100.0995914352 -gps -1100.0997801
```

7.48 xo_gen_osf_change_repeat_cycle_2

7.48.1 Overview

Performs the same operations as **xo_gen_osf_change_repeat_cycle** but adding support for Mean Local Solar Time non-linear drift.

In order to read and write files, xo_gen_osf_change_repeat_cycle_2 function internally uses Data Handling functions. Please refer to [D_H_SUM], in particular sections 4.2 and 4.3, for further details.

7.48.2 Calling interface

The calling interface of the **xo_gen_osf_change_repeat_cycle_2** CFI function is the following (input parameters are underlined):

```
#include <explorer_orbit.h>
{
  long sat_id;
  xl_model_id model_id = {NULL};
  xl_time_id time_id = {NULL};
  long abs_orbit_number, search_direction, phase_increment,
version_number;
  xo_ref_orbit_info ref_orbit_info;
  char input_filename[XD_MAX_STR],
       output_dir[XD_MAX_STR], output_filename[XD_MAX_STR];
  char *file_class, *fh_system;
  long status, ierr[XO_ERR_VECTOR_MAX_LENGTH];

  status = xo_gen_osf_change_repeat_cycle_2 (&sat_id, &model_id,
                                &time_id, &input_filename,
                               &abs_orbit_number,
                               &search_direction,
                               &ref_orbit_info,
                               &phase_increment,
                               output_dir, output_filename,
                               file_class, &version_number,
                               fh_system,
                                ierr);

  /* Or, using the run_id */
  long run_id;

  status = xo_gen_osf_change_repeat_cycle_run_2 (&run_id,
                                &input_filename, &abs_orbit_number,
                                &search_direction,
                                &ref_orbit_info,
```

```
                                    &phase_increment,
                                    output_dir, output_filename,
                                    file_class, &version_number,
                                    fh_system,
                                     ierr);
    }
```

7.48.3 Input parameters

The **xo_gen_osf_change_repeat_cycle_2** CFI function has the following input parameters:

Table 132: Input parameters of xo_gen_osf_change_repeat_cycle_2 function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
model_id	xl_model_id*	-	Model ID	-	-
time_id	xl_time_id*	-	Structure that contains the time correlations.	-	-
input_filename	char*	-	Input OSF to which the orbit change is appended		
abs_orbit_number	long*	-	Absolute orbit number from which the optimum transition search starts	-	> abs orbit number in input OSF last orbit change
search_direction	long*	-	Search for optimum transition after or before abs_orbit_number		{-1, 1}
ref_orbit_info	xo_ref_orbit_info*	-	Struct with inputs for the function. The parameters are equivalent to the ones in xo_gen_osf_change_repeat_cycle (see table 129) but also MLST non-linear terms can be introduced.	-	-

phase_increment	long*	-	If 1 then
			phase [N+1] = phase [N] + 1
			If 0 then
			phase [N+1] = phase [N]
				-	[0, 1]

output_dir	char*	-	Directory where the resulting OSF is written (if NULL, the current directory is used)	-	-
output_filename	char*	-	Output OSF name If empty (i.e. “”), the software will	-	-

			generate the filename according to file name specification presented in [FORMATS]. In such case, the generated name is returned in this variable		
file_class	char*	-	File class for output Orbit file	-	-
version_number	long*	-	Version number of output Orbit file	-	>= 1
fh_system	char*	-	System field of the output Orbit file fixed header	-	-

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

- Satellite ID: sat_id.
- Search direction.
- Drift mode: mlst_drift.
- Phase increment.

This CFI can append orbit changes for both sun-synchronous orbits and quasi-sun-synchronous orbits.

Use `drift_mode=XO_NOSUNSYNC_DRIFT` and `mlst_drift = 0.0` for a sun-synchronous orbit.

Use any other combination for the general case of quasi-sun-synchronous orbit.

7.48.4 Output parameters

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

Table 133: Output parameters of xo_gen_osf_change_repeat_cycle_2 function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
output_filename	char*	-	Name for output file. This is only an output parameter when it is empty (i.e. “”; see description of this parameter in Table 132)	-	-
ierr[XO_ERR_VECTOR_MAX_LENGTH]	long	all	Status vector	-	-

7.48.5 Warnings and errors

The warnings and errors are the same as for function **xo_gen_osf_change_repeat_cycle** (see section 7.47.5).

7.49 xo_gen_osf_add_drift_cycle

7.49.1 Overview

Given a reference orbit from an existing OSF, a new requested orbit with a particular ascending node longitude and an orbit for the manoeuvre, the **xo_gen_osf_add_drift_cycle** CFI function fits a repeat cycle/ cycle length between the manoeuvre orbit (drift start) and the requested orbit (drift stop) such that the longitude of the ascending node at the drift stop orbit be the one requested.

The drift orbit is constrained by a maximum altitude difference with respect to the reference orbit.

Furthermore, if the reference orbit is sun-synchronous, the drift orbit shall also be sun-synchronous; but if the reference orbit is not sun-synchronous, the drift orbit shall keep the inclination constant.

This CFI appends two orbit changes to the existing OSF:

- The first one for the drift manoeuvre
- The second one for restoring the old reference orbit characteristics at the requested ascending node longitude

In order to read and write files, xo_gen_osf_add_drift_cycle function internally uses Data Handling functions. Please refer to [D_H_SUM], in particular sections 4.2 and 4.3, for further details.

7.49.2 Calling interface

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

```
#include <explorer_orbit.h>
{
  long sat_id;
  xl_model_id model_id = {NULL};
  xl_time_id time_id = {NULL};
  long drift_start_orbit, drift_stop_orbit,
       phase_inc_start, phase_inc_stop, version_number;
  double drift_stop_anx_long, max_altitude_change;
  char input_filename[XD_MAX_STR],
       output_dir[XD_MAX_STR], output_filename[XD_MAX_STR];
  char *file_class, *fh_system;
  long status, ierr[XO_ERR_VECTOR_MAX_LENGTH];

  status = xo_gen_osf_add_drift_cycle (&sat_id, &model_id,
                                       &time_id,
                                       &input_filename,
                                       &drift_start_orbit,
                                       &drift_stop_orbit,
                                       &drift_stop_anx_long,
                                       &max_altitude_change,
                                       &phase_inc_start, &phase_inc_stop,
```

```
                                    output_dir, output_filename,
                                    file_class, &version_number,
                                    fh_system,
                                     ierr);

    /* Or, using the run_id */
    long run_id;

    status = xo_gen_osf_add_drift_cycle_run (&run_id,
                                    &input_filename,
                                    &drift_start_orbit,
                                    &drift_stop_orbit,
                                    &drift_stop_anx_long,
                                    &max_altitude_change,
                                    &phase_inc_start, &phase_inc_stop,
                                    output_dir, output_filename,
                                    file_class, &version_number,
                                    fh_system,
                                     ierr);
}
```

7.49.3 Input parameters

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

Table 134: Input parameters of xo_gen_osf_add_drift_cycle function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
model_id	xl_model_id*	-	Model ID	-	-
time_id	xl_time_id*	-	Structure that contains the time correlations.	-	-
input_filename	char*	-	Input OSF to which the orbit changes are appended		
drift_start_orbit	long*	-	Absolute orbit number at the drift start	-	> abs orbit number in input OSF last orbit change
drift_stop_orbit	long*	-	Absolute orbit number at the drift stop	-	> drift_start_orbit
drift_stop_anx_long	double*	-	Drift stop orbit ascending node	deg	[-180, 180]

			crossing longitude		
max_altitude_change	double*	-	Maximum variation in altitude between the reference orbit and the drift orbit	m	

phase_inc_start	long*	-	Phase increment at drift start
			If 1 then
			phase [N+1] = phase [N] + 1
			If 0 then
			phase [N+1] = phase [N]
				-	[0, 1]

phase_inc_stop	long*	-	Phase increment at drift stop
			If 1 then
			phase [N+1] = phase [N] + 1
			If 0 then
			phase [N+1] = phase [N]
				-	[0, 1]

output_dir	char*	-	Directory where the resulting OSF is written (if empty (i.e. ““), the current directory is used)	-	-
output_filename	char*	-	Output OSF name If empty (i.e. ““), the software will generate the filename according to file name specification pre sented in [FORMATS]. In such case, the generated name is returned in this variable	-	-
file_class	char*	-	File class for output Orbit file	-	-
version_number	long*	-	Version number of output Orbit file	-	>= 1
fh_system	char*	-	System field of the output Orbit file fixed header	-	-

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

- Satellite ID: sat_id.
- Search direction.
- Drift mode: mlst_drift.
- Phase increment.

7.49.4 Output parameters

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

Table 135: Output parameters of xo_gen_osf_add_drift_cycle function

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

output_filename	char*	-	Name for output file. This is only an output parameter when it is empty (i.e. ""; see description of this parameter in Table 134)	-	-
ierr[XO_ERR_VECTOR_MAX_LENGTH]	long	all	Status vector	-	-

7.49.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_gen_osf_add_drift_cycle** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_gen_osf_add_drift_cycle** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 136: Error messages of xo_gen_osf_add_drift_cycle function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Wrong input values	Wrong value of one or more of the following input parameters: drift_start_orbit, drift_stop_orbit, phase_inc_start, phase_inc_stop, Computation not performed	XO_CFI_GEN_OSF_DRIFT_INPUTS_ERR	0
ERR	Time ID is not initialized	Computation not performed	XO_CFI_GEN_OSF_DRIFT_TIME_INIT_ERR	1
ERR	Cannot read input OSF	Computation not performed	XO_CFI_GEN_OSF_DRIFT_READ_IN_OSF_ERR	2
ERR	No drift orbit necessary	Computation not performed	XO_CFI_GEN_OSF_DRIFT_NO_ADD_ERR	3
ERR	Error calculating inclination	Error calculating inclination for a no sun-synchronous orbit in order to keep inclination constant during the drift phase Computation not performed	XO_CFI_GEN_OSF_DRIFT_INCL_CALC_ERR	4
ERR	No drift orbit found	No drift orbit has been found	XO_CFI_GEN_OSF_DRIFT	5

		that matches the drift start and stop ANX longitude Computation not performed	_NOT_FOUND_ERR	
ERR	Error calculating UTC of ANX	Error calculating the UTC time of the orbit ascending node Computation not performed	XO_CFI_GEN_OSF_DRIFT_UTC_CALC_ERR	6
ERR	Error calculating TAI of ANX	Error calculating the TAI time of the orbit ascending node Computation not performed	XO_CFI_GEN_OSF_DRIFT_TAI_CALC_ERR	7
ERR	Error calculating UT1 of ANX	Error calculating the UT1 time of the orbit ascending node Computation not performed	XO_CFI_GEN_OSF_DRIFT_UT1_CALC_ERR	8
ERR	Memory allocation error	Computation not performed	XO_GEN_OSF_DRIFT_ALLOC_ERR	9
ERR	Error calculating the Fixed Header data	Computation not performed	XO_GEN_OSF_DRIFT_GET_FH_ERR	10
ERR	Error writing file to disk	Error writing the data structure to a file on disk Computation not performed	XO_CFI_GEN_OSF_DRIFT_WRITE_ERR	11

7.49.6 Executable Program

The **gen_osf_add_drift_cycle** executable program can be called from a Unix shell as:

gen_osf_add_drift_cycle **-sat** satellite_name
-inosf input_filename
-drorb0 drift_start_orbit
-drorb1 drift_stop_orbit
-anx drift_stop_anx_long (deg)
-alt max_altitude_change (m)
[**-phinc0**]
[**-phinc1**]
[**-dir** output_dir] (current directory by default)
[**-osf** output_filename] (default: name generated automatically)
[**-flcl** file_class] (empty string by default)
[**-vers** version] (version = 1 by default)

[**-eoffs** ffs_version] (Earth Observation File Format Standard Version)
[**-fhsys** fh_system] (empty string by default)
[**-v**]
[**-xl_v**]
[**-xo_v**]
[**-help**]
[**-show**]
[**-with_xslt]** (add xslt reference with default style sheet)
{ (**-tai** TAI_time **-gps** GPS_time **-utc** UTC_time **-ut1** UT1_time) |
(**-tmod** time_model **-tfile** time_file **-trid** time_reference
{(**-tm0** time0 **-tm1** time1) | (**-orb0** orbit0 **-orb1** orbit1) })**}**

Note that:

- Order of parameters does not matter.
- Bracketed parameters are not mandatory.
- Options between curly brackets and separated by a vertical bar are mutually exclusive.
- [**-phinc0**] option for phase_inc_start. Default value is xo_NO_PHASE_INCREMENT. When the option is written, phase_inc_start value is xo_PHASE_INCREMENT.
- [**-phinc1**] option for phase_inc_stop. Default value is xo_NO_PHASE_INCREMENT. When the option is written, phase_inc_stop value is xo_PHASE_INCREMENT.
- [**-xl_v**] option for EO_LIB Verbose mode.
- [**-xo_v**] option for EO_ORBIT Verbose mode.
- [**-v**] option for Verbose mode for all libraries (default is Silent).
- [**-show**] displays the inputs of the function and the results.
- Possible values for *satellite_name*: ERS1, ERS2, ENVISAT, METOP1 , METOP2, METOP3, CRYOSAT, ADM, GOCE, SMOS, TERRASAR, EARTHCARE, SWARM_A, SWARM_B, SWARM_C, SENTINEL_1A, SENTINEL_1B, SENTINEL_2, SENTINEL_3, SENTINEL_1C, SENTINEL_2A, SENTINEL_2B, SENTINEL_2C, SENTINEL_3A, SENTINEL_3B, SENTINEL_3C, SEOSAT, JASON_CSA, JASON_CSB, METOP_SG_A1, METOP_SG_A2, METOP_SG_A3, METOP_SG_B1, METOP_SG_B2, METOP_SG_B3, SENTINEL_5P, BIOMASS, SENTINEL_5, SAOCOM_CS, FLEX, SENTINEL_6A, SENTINEL_6B, CIMR, ROSE-L, CHIME, CRISTAL, CO2M, LSTM, FORUM, TRUTHS, GENERIC, GENERIC_GEO, MTG.
- Possible values for *time_model*: USER, NONE , IERS_B_PREDICTED, IERS_B_RESTITUTED, FOS_PREDICTED, FOS_RESTITUTED, DORIS_PRELIMINARY, DORIS_PRECISE, DORIS_NAVIGATOR, OSF.
- Possible values for *ffs_version*: 0 (Default FFS), 1 (FFS version 1), 2 (FFS version 2), 3 (FFS version 3).
- Possible values for *time_reference*: UNDEF, TAI, UTC, UT1, GPS.

- The last three lines of parameters are used to initialize the time references. In order to do this, only one set of parameters should be introduced:

- TAI, GPS, UTC and UT1 input times (as in xl_time_ref_init)
- A file with time reference data, the time mode, the time reference name and a time range (as in xl_time_ref_init_file)

Example:

```
gen_osf_add_drift_cycle -sat CRYOSAT
-inosf CS_TEST_MPL_ORBREF_20020301T122001_99999999T999999_0001.EEF
-drorb0 30 -drorb1 2702 -anx 310 -alt 15000 -dir ./gen_osf
-osf mpl_orb_sc_at_305
-tai -1100.1 -utc -1100.099595 -ut1 -1100.0995914352 -gps -1100.0997801
```

7.50 xo_gen_rof

7.50.1 Overview

The **xo_gen_rof** CFI function creates a Restituted Orbit File (ROF) using as input one of the following reference file types:

- Orbit Scenario File
- FOS Predicted Orbit File
- DORIS Navigator File(s).
- FOS Restituted Orbit File
- DORIS Preliminary Orbit File
- DORIS Precise Orbit FileTime of the ascending crossing node (TAI, UTC and UT1)
- The accepted output file types are:
- FOS Restituted Orbit File
- DORIS Preliminary Orbit File
- DORIS Precise Orbit FileTime
- TLE File

The time interval between consecutive OSVs can be selected by the user by means of a parameter in the calling interface. A flag for precise location of OSVs at “integer intervals” (e.g. every exact minute) is also available. If the reference file and the Restituted Orbit File contain OSVs at the same time, these OSVs will be identical.

In order to read and write files, xo_gen_rof function internally uses Data Handling functions. Please refer to [D_H_SUM], in particular sections 4.2 and 4.3, for further details.

The value of the tag Time_Reference in variable header is set using the input parameter time_ref.Note: when using an OSF or Predicted Orbit file, the maximum time interval within the output Restituted orbit file is limited to 2 orbital periods before and after the middle point of the user requested time range.

7.50.2 Calling interface

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

```
#include <explorer_orbit.h>
{
  long sat_id;
  xl_model_id model_id = {NULL};
  xl_time_id time_id = {NULL};
  long   time_init, time_ref, start_orbit, stop_orbit,
             ref_filetype, rof_filetype, osv_precise, version_number;
  double start_time, stop_time, osv_interval;
  char   reference_file[XD_MAX_STR], output_dir[XD_MAX_STR],
             rof_filename[XD_MAX_STR], precise_conf_file[XD_MAX_STR];
```

```
    char  *file_class, *fh_system;
    long   status, ierr[XO_ERR_VECTOR_MAX_LENGTH];
    status = xo_gen_rof(&sat_id, &model_id, &time_id, &time_init,
                        &time_ref, &start_time,  &stop_time,
                        &start_orbit, &stop_orbit,
                        &osv_interval, &osv_precise,
                        &ref_filetype, reference_file,
                        precise_conf_file,
                        &rof_filetype, output_dir, rof_filename,
                        file_class, &version_number, fh_system,
                         /* output */
                         ierr);

    /* Or, using the run_id */
    long run_id;

    status = xo_gen_rof_run(&run_id, &time_init, &time_ref,
                            &start_time,  &stop_time,
                            &start_orbit, &stop_orbit,
                            &osv_interval, &osv_precise,
                            &ref_filetype, reference_file,
                            precise_conf_file,
                            &rof_filetype,
                        output_dir, rof_filename,
                            file_class, &version_number, fh_system,
                             /* output */
                             ierr);
}
```

7.50.3 Input parameters

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

Table 137: Input parameters of xo_gen_rof function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
model_id	xl_model_id*	-	Model ID	-	-
time_id	xl_time_id*	-	Structure that contains the time correlations. NOTE: If time_id is not initialized, then time correlations	-	-

			will be initialized internally using the input reference file.		
time_init	long*	-	Flag for selecting the time range of the initialisation.	-	Select either: · XO_SEL_ORBIT · XO_SEL_TIME
time_ref	long*	-	Time reference ID (see note in the ref_file type field)	-	Select either: · XO_TIME_TAI · XO_TIME_UTC · XO_TIME_UT1
start_time	double*	-	Processing time corresponding to the beginning of the required interval	Decimal days, MJD2000	[-18262.0,36524.0]
stop_time	double*	-	Processing time corresponding to the end of the required interval	Decimal days, MJD2000	[-18262.0,36524.0]
start_orbit	long*	-	Orbit number corresponding to the beginning of the required interval	orbits	>= 1
stop_orbit	long*	-	Orbit number corresponding to the end of the required interval	orbits	>= 1
osv_interval	double*	-	Interval between consecutive state vector. This parameter should be coherent with the osv_precise flag (see below). If osv_precise is set to: XO_OSV_PRECISE_MINUTE: osv will be forced to be a multiple of 60 seconds. XO_OSV_PRECISE_TEN_SECONDS: osv will be forced to be a multiple of 10 seconds.	secs	>=0
osv_precise	long*	-	Flag to indicate if state vectors should be placed at exact time locations	-	Complete
ref_filetype	long*	-	File type of the input reference file. (Note: When generating a ROF file from a DORIS NAVIGATOR file, the input times should be expressed in UTC)	-	Complete
reference_filename	char*	-	Reference File name	-	
precise_conf_file	char*	-	File with configuration parameters for precise propagator	-	If it is neither NULL nor "", precise propagation will be used

rof_filetype	long*	-	File type of the output reference file	-	XO_REF_FILETYP E_ROF XO_REF_FILETYP E_DORIS_PREM XO_REF_FILETYP E_DORIS_PREC
output_dir	char*	-	Directory where the resulting ROF is written (if NULL, the current directory is used)	-	-
rof_filename	char*	-	Output ROF name If empty (i.e. ““), the software will generate the filename according to file name specification pre sented in [FORMATS]. In such case, the generated name is returned in this variable	-	-
file_class	char*	-	File class for output Restituted file	-	-
version_number	long*	-	Version number of output Restituted file	-	>= 1
fh_system	char*	-	System field of the output Restituted file fixed header	-	-

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

- Satellite ID: sat_id.
- Time initialisation: time_init.
- Time reference: time_ref.
- OSV precise: osv_precise. See this SUM.
- File type: ref_filetype and rof_filetype. See this SUM.

7.50.4 Output parameters

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

Table 138: Output parameters of xo_gen_rof function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
rof_filename	char*	-	Name for the output file.	-	-

			This is only an output parameter when it is empty (i.e. ""; see description of this parameter in Table 137)		
ierr[XO_ERR_VECTOR_MAX_LENGTH]	long	all	Status vector	-	-

7.50.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_gen_rof** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_gen_rof** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 139: Error messages of xo_gen_rof function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Wrong satellite flag	Computation not performed	XO_CFI_GEN_ROF_WRONG_SAT_ID_ERR	0
ERR	Wrong input flag	Computation not performed	XO_CFI_GEN_ROF_WRONG_FLAG_ERR	1
ERR	Time ID is not initialized	Computation not performed	XO_CFI_GEN_ROF_TIME_I NIT_ERR	2
ERR	Could not initialise the time reference	Computation not performed	XO_CFI_GEN_ROF_TIME_I NITIALIZATION_ERR	3
ERR	Cannot initialise orbit ID	Computation not performed	XO_CFI_GEN_ROF_ORBIT_INIT_FILE_ERR	4
ERR	Cannot initialise the propagator	Computation not performed	XO_CFI_GEN_ROF_PROPAG_INIT_ERR	5
ERR	Could not perform a time <-> orbit transformation	Computation not performed	XO_CFI_GEN_ROF_TIME_ORBIT_ERR	6
ERR	Cannot initialise interpolation	Computation not performed	XO_CFI_GEN_ROF_INTERPOL_INIT_ERR	7
ERR	Cannot calculate state vector	Computation not performed	XO_CFI_GEN_ROF_CALCULATING_STATE_VECTOR_ERR	8
ERR	Cannot convert time to processing format	Computation not performed	XO_CFI_GEN_ROF_TIME_ERR	9

ERR	Cannot convert time from processing to external	Computation not performed	XO_CFI_GEN_ROF_TIME_TO_EXTERNAL_ERR	10
ERR	Cannot write ROF file to disk	Computation not performed	XO_CFI_GEN_ROF_WRITE _ERR	11
ERR	Error freeing memory	Computation not performed	XO_CFI_GEN_ROF_CLOSE_ERR	12
ERR	Memory allocation error	Computation not performed	XO_CFI_GEN_ROF_MEMORY_ERR	13
ERR	Error getting fixed header	Computation not performed	XO_CFI_GEN_ROF_GET_F H_ERR	14
ERR	OSV interval is not compatible with OSV Precise flag. The OSV Interval will be set to %f seconds.	Computation performed with a different value for the osv_interval	XO_CFI_GEN_ROF_WRONG_INTERVAL_WARN	15
ERR	OSV Interval is < 0	Computation not performed	XO_CFI_GEN_ROF_WRONG_OSV_INTERVAL_ERR	16
ERR	Error reading precise propagator configuration file	Computation not performed	XO_CFI_GEN_ROF_READ_PRECISE_FILE_ERR	17
ERR	Time reference ID is not allowed	Computation not performed	XO_CFI_GEN_ROF_TIME_ID_NOT_ALLOWED_ERR	18
ERR	With TLE Orbit File, the Time instance should be initialized	Computation not performed	XO_CFI_GEN_ROF_TIME_TLE_INITIALIZATION_ERR	19
ERR	Error in selecting schema	Computation not performed	XO_CFI_GEN_ROF_SELECT_SCHEMA_ERR	20

7.50.6 Executable Program

The **gen_rof** executable program can be called from a Unix shell as:

gen_rof **-sat** satellite_name
-tref time_ref
{ **-tstart** start_time **-tstop** stop_time (decimal days) |
-tastart start_time **-tastop** stop_time (CCSDSA format) |
-ostart start_orbit **-ostop** stop_orbit (orbits) }
-osvint osv_interval
[**-osvpre**]
-reftyp ref_file_type
-ref reference_file

-roftyp rof_file_type

[**-precfile** precise_conf_file] (empty string by default)

[**-dir** output_dir] (current directory by default)

[**-rof** output_filename] (default: name generated automatically)

[**-flcl** file_class] (empty string by default)

[**-vers** version] (version= 1 by default)

[**-eoffs** ffs_version] (Earth Observation File Format Standard Version)

[**-fhsys** fh_system] (empty string by default)

[**-v**]

[**-xl_v**]

[**-xo_v**]

[**-help**]

[**-show**]

[**-with_xslt]** (add xslt reference with default style sheet)

[(**-tai** TAI_time **-gps** GPS_time **-utc** UTC_time **-ut1** UT1_time) |

(**-tmod** time_model **-tfile** time_file **-trid** time_reference

{(**-tm0** time0 **-tm1** time1) | (**-orb0** orbit0 **-orb1** orbit1) }) **]**

Note that:

- Order of parameters does not matter.
- Bracketed parameters are not mandatory.
- Options between curly brackets and separated by a vertical bar are mutually exclusive.
- [**-osvpre**] option for osv_precise.Default value is xo_OSV_PRECISE_NO. When the option is written, ovs_precise value is xo_OSV_PRECISE_MINUTE.
- [**-xl_v**] option for EO_LIB Verbose mode.
- [**-xo_v**] option for EO_ORBIT Verbose mode.
- [**-v**] option for Verbose mode for all libraries (default is Silent).
- [**-show**] displays the inputs of the function and the results.
- Possible values for *satellite_name*: ERS1, ERS2, ENVISAT, METOP1 , METOP2, METOP3, CRYOSAT, ADM, GOCE, SMOS, TERRASAR, EARTHCARE, SWARM_A, SWARM_B, SWARM_C, SENTINEL_1A, SENTINEL_1B, SENTINEL_2, SENTINEL_3, SENTINEL_1C, SENTINEL_2A, SENTINEL_2B, SENTINEL_2C, SENTINEL_3A, SENTINEL_3B, SENTINEL_3C, SEOSAT, JASON_CSA, JASON_CSB, METOP_SG_A1, METOP_SG_A2, METOP_SG_A3, METOP_SG_B1, METOP_SG_B2, METOP_SG_B3, SENTINEL_5P, BIOMASS, SENTINEL_5, SAOCOM_CS, FLEX, SENTINEL_6A, SENTINEL_6B, CIMR, ROSE-L, CHIME, CRISTAL, CO2M, LSTM, FORUM, TRUTHS, GENERIC, GENERIC_GEO, MTG.

- Possible values for *time_model*: USER, NONE , IERS_B_PREDICTED, IERS_B_RESTITUTED, FOS_PREDICTED, FOS_RESTITUTED, DORIS_PRELIMINARY, DORIS_PRECISE, DORIS_NAVIGATOR, OSF.
- Possible values for *ref_file_type*: OSF, POF, DORISNAV, ROF, TLE, DORISPREM, DORISPREC.
- Possible values for *rof_file_type*: ROF, DORISPREM, DORISPREC.
- Possible values for *ffs_version*: 0 (Default FFS), 1 (FFS version 1), 2 (FFS version 2), 3 (FFS version 3).
- Possible values for *time_ref* and *time_reference*: UNDEF, TAI, UTC, UT1.
- The value of the tag Time_Reference in variable header is set using the input parameter *time_ref.*
- Time references need to be initialized only when using OSF as the type of the input reference file. The inputs needed for this issue are provided in the last three lines of parameters. Note that only one set of parameters should be introduced:

– TAI, GPS, UTC and UT1 input times (as in xl_time_ref_init)

– A file with time reference data, the time mode, the time reference name and a time range (as in xl_time_ref_init_file)

- Precise propagation is used if precfile is provided.

Example:

```
gen_rof   -sat CRYOSAT -tref TAI -ostart 1000 -ostop 1001
          -osvint 300  -reftyp OSF
          -ref
                 CS_TEST_MPL_ORBREF_20020301T122001_99999999T999999_0001.EEF
          -roftyp ROF -dir ./gen_rof/ -rof orb_res_file_at_306
          -tmod FOS_PREDICTED -tfile ./data/test.fpo -trid TAI
          -tm0 0 -tm1 10000
```

7.51 xo_gen_rof_prototype

7.51.1 Overview

The **xo_gen_rof_prototype** CFI function creates a Restituted Orbit File (ROF) using the following input parameters:

- Date (processing time) and orbit
- Longitude of the ascending node,
- Satellite Repeat Cycle and Cycle Length
- Mean local solar time at ascending node
- Drift of mean local solar time or the inclination

The time interval between consecutive OSVs can be selected by the user by means of a parameter in the calling interface.

A file with the configuration parameters for precise propagator can be introduced. In this case, the numeric propagator is used.

In order to write files, xo_gen_rof_prototype function internally uses Data Handling functions. Please refer to [D_H_SUM], in particular sections 4.2 and 4.3, for further details.

7.51.2 Calling interface

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

```
#include <explorer_orbit.h>
{
  long sat_id;
  xl_model_id model_id = {NULL};
  xl_time_id time_id = {NULL};
  long propag_model, time_ref, time_init_mode;
  long orbit0, drift_mode, irep, icyc, start_orbit, stop_orbit;
  double time0, start_time, stop_orbit, osv_interval;
  double ascmlst_drift, inclination, rlong, ascmlst;
  char   output_dir[XD_MAX_STR], rof_filename[XD_MAX_STR];
  char  *file_class, *fh_system;
  long status, ierr[XO_ERR_VECTOR_MAX_LENGTH], version_number;

  status = xo_gen_rof_prototype (&sat_id, &model_id, &time_id,
                                 &propag_model, &time_ref,
                                 &time0, &orbit0, &time_init_mode,
                                 &start_time, &start_orbit
                                 &stop_time, &stop_orbit,
                                 &drift_mode,
                                 &ascmlst_drift, &inclination,
```

```
                                        &irep, &icyc, &rlong, &ascmlst,
                                        &osv_interval
                                        output_dir,      rof_filename,
                                        file_class,
    &version_number,                                      fh_system,
                                    /* output */
                                   ierr);
      /* Or, using the run_id */
      long run_id;

      status = xo_gen_rof_prototype_run (&run_id,
                                        &propag_model, &time_ref,
                                        &time0, &orbit0, &time_init_mode,
                                        &start_time, &start_orbit
                                        &stop_time, &stop_orbit,
                                        &drift_mode,
                                        &ascmlst_drift, &inclination,
                                        &irep, &icyc, &rlong, &ascmlst,
                                        &osv_interval
                                        output_dir,      rof_filename,
                                        file_class,
    &version_number,                                      fh_system,
                                    /* output */
                                   ierr);
    }
```

7.51.3 Input parameters

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

Table 140: Input parameters of xo_gen_rof_prototype function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
model_id	xl_model_id*	-	Model ID	-	-
time_id	xl_time_id*	-	Structure that contains the time correlations.	-	-
propag_model	long*	-	Propagation model ID	-	Complete
time_ref	long*	-	Time reference ID	-	Complete
time0	double*	-	Reference time	Decimal days (Processing format)	[-18262.0,36524.0]

orbit0	long*	-	Absolute orbit number of the reference orbit	-	>= 0
time_init_mode	long*	-	Flag for selecting the time range of the initialisation.	-	Select either: · XO_SEL_ORBIT · XO_SEL_TIME
start_time	double*	-	Processing time corresponding to the beginning of the required interval	Decimal days, MJD2000	[-18262.0,36524.0]
start_orbit	long*	-	Orbit number corresponding to the beginning of the required interval	orbits	>= 1
stop_time	double*	-	Processing time corresponding to the end of the required interval	Decimal days, MJD2000	[-18262.0,36524.0]
stop_orbit	long*	-	Orbit number corresponding to the end of the required interval	orbits	>= 1
drift_mode	long*	-	Flag to select between drift in mean local solar time and incli nation as input characterization of the reference orbit	-	Complete
ascmlst_drift	double*	-	If *drift_mode* = *XO_NOSUNSYNC_MLST* Drift in mean local solar time of the reference orbit	seconds/day	TBD
inclination	double*	-	If *drift_mode* = *XO_NOSUNSYNC_INCLINATION* Inclination of the reference orbit	deg	[0,180]
irep	long *	-	Repeat cycle of the reference orbit The actual repeat cycle is calculated as per definition included in [MCD].	days	> 0
icyc	long *	-	Cycle length of the reference orbit	orbits	> 0
rlong	double*	-	Geocentric longitude of the [Earth fixed] ascending node (Earth fixed CS)	deg	[0,360)
ascmlst	double*	-	Mean local solar time at ascending node	hours	[0, 24)
osv_interval	double*	-	Interval between consecutive state vector	secs	>=0
output_dir	char*	-	Directory where the resulting ROF is written (if NULL, the current directory is used)	-	-

rof_filename	char*	-	Output ROF name If empty (i.e. “”), the software will generate the filename according to file name specification pre sented in [FORMATS]. In such case, the generated name is returned in this variable	-	-
file_class	char*	-	File class for output Restituted file	-	-
version_number	long*	-	Version number of output Restituted file	-	>= 1
fh_system	char*	-	System field of the output Restituted file fixed header	-	-

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

- Satellite ID: sat_id.
- Time initialisation: time_init.
- Time reference: time_ref.
- Drift Mode: drift_mode.

7.51.4 Output parameters

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

Table 141: Output parameters of xo_gen_rof_prototype function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
rof_filename	char*	-	Name for the output file. This is only an output parameter when it is empty (i.e. “”; see description of this parameter in Table 143)	-	-
ierr[XO_ERR_VECTOR_MAX_LENGTH]	long	all	Status vector	-	-

7.51.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_gen_rof_prototype** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_gen_rof_prototype** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 142: Error messages of xo_gen_rof_prototype function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Wrong satellite flag	Computation not performed	XO_CFI_GEN_ROF_PROTOTYPE_WRONG_SAT_ID_ERR	0
ERR	Time ID is not initialized	Computation not performed	XO_CFI_GEN_ROF_PROTOTYPE_TIME_ID_ERR	1
ERR	Wrong input flag	Computation not performed	XO_CFI_GEN_ROF_PROTOTYPE_WRONG_FLAG_ERR	2
ERR	Cannot initialise propagator	Computation not performed	XO_CFI_GEN_ROF_PROTOTYPE_PROPAG_INIT_DEF_ERR	3
ERR	Cannot calculate state vector	Computation not performed	XO_CFI_GEN_ROF_PROTOTYPE_CALCULATING_STATE_VECTOR_ER R	3
ERR	Cannot convert time in processing reference	Computation not performed	XO_CFI_GEN_ROF_PROTOTYPE_TIME_ERR	5
ERR	Cannot convert time from processing to external	Computation not performed	XO_CFI_GEN_ROF_PROTOTYPE_TIME_TO_EXTERNAL_ERR	6
ERR	Error freeing memory	Computation not performed	XO_CFI_GEN_ROF_PROTOTYPE_CLOSE_ERR	7
ERR	Error creating the fixed header	Computation not performed	XO_CFI_GEN_ROF_PROTOTYPE_GET_FH_ERR	8
ERR	Memory allocation error	Computation not performed	XO_CFI_GEN_ROF_PROTOTYPE_MEMORY_ERR	9
ERR	Cannot write ROF XML file	Computation not performed	XO_CFI_GEN_ROF_PROTOTYPE_WRITE_ERR	10
ERR	Error in selecting schema	Computation not performed	XO_CFI_GEN_ROF_PROTOTYPE_SELECT_SCHEMA_ERR	11

7.52 xo_gen_pof

7.52.1 Overview

The **xo_gen_pof** CFI function creates a Predicted Orbit File (POF) with one state vector per orbit using as input one of the following reference file types:

- Orbit Scenario File
- FOS Predicted Orbit File
- DORIS Navigator File(s).
- FOS Restituted Orbit File
- DORIS Preliminary Orbit File
- DORIS Precise Orbit FileTime of the ascending crossing node (TAI, UTC and UT1)
- TLE File

The location of the state vector within the orbit can be selected by the user by means of a parameter in the calling interface. If the reference file and the Predicted Orbit File contain OSVs at the same time, these OSVs will be identical.

A file with the configuration parameters for precise propagator can be introduced. In this case, the numeric propagator is used.

In order to read and write files, xo_gen_pof function internally uses Data Handling functions. Please refer to [D_H_SUM], in particular sections 4.2 and 4.3, for further details.

The value of the tag Time_Reference in variable header is set using the input parameter time_ref.

7.52.2 Calling interface

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

```
#include <explorer_orbit.h>
{
  long   sat_id;
  xl_model_id model_id = {NULL};
  xl_time_id time_id = {NULL};
  long   time_init, time_ref, start_orbit, stop_orbit,
             ref_filetype, pof_filetype, version_number;
  double start_time, stop_time, osv_location;
  char   reference_file[XD_MAX_STR], output_dir[XD_MAX_STR],
             pof_filename[XD_MAX_STR], precise_conf_file[XD_MAX_STR];
  char  *file_class, *fh_system;
  long   status, ierr[XO_ERR_VECTOR_MAX_LENGTH];
  status = xo_gen_pof(&sat_id, &model_id, &time_id,
                          &time_init, &time_ref,
                          &start_time, &stop_time,
                          &start_orbit, &stop_orbit, &osv_location,
```

```
                                &ref_filetype, reference_file,
                                precise_conf_file,
                                &pof_filetype, output_dir,
    pof_filename,
                                file_class, &version_number, fh_system,
                              /* output */
                                ierr);


    /* Or, using the run_id */
    long run_id;
    status = xo_gen_pof_run(&run_id,
                                &time_init, &time_ref,
                                &start_time,  &stop_time,
                                &start_orbit, &stop_orbit,
                                &osv_location,
                                &ref_filetype, reference_file,
                                precise_conf_file,
                             &pof_filetype, output_dir,
                                pof_filename,
                                file_class, &version_number, fh_system,
                              /* output */
                             ierr);
}
```

7.52.3 Input parameters

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

Table 143: Input parameters of xo_gen_pof function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
model_id	xl_model_id	-	Model ID	-	Complete
time_id	xl_time_id*	-	Structure that contains the time correlations. NOTE: If time_id is not initialized, then time correlations will be initialized internally using the input reference file	-	-
time_init	long*	-	Flag for selecting the time range of the initialisation.	-	Select either: · XO_SEL_ORBIT

					· XO_SEL_TIME
time_ref	long*	-	Time reference ID. (See note in the ref_filetype field)	-	Select either: · XO_TIME_TAI · XO_TIME_UTC · XO_TIME_UT1
start_time	double*	-	Processing time corresponding to the beginning of the required interval	Decimal days, MJD2000	[-18262.0,36524.0]
stop_time	double*	-	Processing time corresponding to the end of the required interval	Decimal days, MJD2000	[-18262.0,36524.0]
start_orbit	long*	-	Orbit number corresponding to the beginning of the required interval	orbits	>= 1
stop_orbit	long*	-	Orbit number corresponding to the end of the required interval	orbits	>= 1
osv_location	double*	-	Location of the state vector within the orbit	secs	>=0 < 1 nodal period
ref_filetype	long*	-	File type of the input reference file. (Note: When generating a POF file from a DORIS NAVIGATOR file, the input times should be expressed in UTC)	-	Complete
reference_filename	char*	-	Reference File name	-	
precise_conf_file	char*	-	File with precise propagator con figuration	-	If it is not neither NULL nor “”, precise propagation will be used
pof_filetype	long*	-	File type of the output reference file	-	XO_REF_FILETYPE_POF
output_dir	char*	-	Directory where the resulting POF is written (if NULL, the current directory is used)	-	-
pof_filename	char*	-	Output POF name if empty (i.e. “”), the software will generate the filename according to file name specification presented in [FORMATS]. In such case, the generated name is returned in this variable	-	-
file_class	char*	-	File class for output Predicted file	-	-
version_number	long*	-	Version number of output	-	>= 1

			Predicted file		
fh_system	char*	-	System field of the output Predicted file fixed header	-	-

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

- Satellite ID: sat_id.
- Time initialisation: time_init.
- Time reference: time_ref.
- File type: ref_filetype and pof_filetype. See section 6.2 in this SUM.

7.52.4 Output parameters

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

Table 144: Output parameters of xo_gen_pof function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
pof_filename	char*	-	Name for the output file. This is only an output parameter when it is empty (i.e. ""; see description of this parameter in Table 140)	-	-
ierr[XO_ERR_VECTOR_MAX_LENGTH]	long	all	Status vector	-	-

7.52.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_gen_pof** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_gen_pof** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 145: Error messages of xo_gen_pof function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Wrong satellite flag	Computation not performed	XO_CFI_GEN_POF_WRONG_SAT_ID_ERR	0
ERR	Wrong input flag	Computation not performed	XO_CFI_GEN_POF_WRONG_FLAG_ERR	1

ERR	Time ID is not initialized	Computation not performed	XO_CFI_GEN_POF_TIME_INIT_ERR	2
ERR	Could not initialise the time reference	Computation not performed	XO_CFI_GEN_POF_TIME_INITIALIZATION_ERR	3
ERR	Cannot initialise orbit	Computation not performed	XO_CFI_GEN_POF_ORBIT_INIT_FILE_ERR	4
ERR	Cannot initialise propagation	Computation not performed	XO_CFI_GEN_POF_PROPAG_INIT_ERR	5
ERR	Cannot initialise interpolation	Computation not performed	XO_CFI_GEN_POF_INTERPOL_INIT_ERR	6
ERR	Wrong interpol initialisation	Computation not performed	XO_CFI_GEN_POF_INTERNAL1_ERR	7
ERR	Cannot calculate state vector	Computation not performed	XO_CFI_GEN_POF_CALCULATING_STATE_VECTOR_ERR	8
ERR	Error freeing memory	Computation not performed	XO_CFI_GEN_POF_CLOSE_ERR	9
ERR	Time transformation error	Computation not performed	XO_CFI_GEN_POF_TIME_TRANS_ERR	10
ERR	Memory allocation error	Computation not performed	XO_CFI_GEN_POF_MEMORY_ERR	11
ERR	Error creating the fixed header	Computation not performed	XO_CFI_GEN_POF_GET_FH_ERR	12
ERR	Error writing POF file to disk	Computation not performed	XO_CFI_GEN_POF_WRITE_ERR	13
ERR	Error reading configuration file for precise propagation	Computation not performed	XO_CFI_GEN_POF_READ_ PRECISE_FILE_ERR	14
ERR	Error converting time to orbit or orbit to time	Computation not performed	XO_CFI_GEN_POF_ORBIT_TIME_ERR	15
ERR	Time reference ID is not allowed	Computation not performed	XO_CFI_GEN_POF_TIME_ID_NOT_ALLOWED_ERR	16

7.52.6 Executable Program

The **gen_pof** executable program can be called from a Unix shell as:

gen_pof **-sat** satellite_name
-tref time_ref
{ **-tstart** start_time **-tstop** stop_time (decimal days) |
-tastart start_time **-tastop** stop_time (CCSDSA format) |
-ostart start_orbit **-ostop** stop_orbit (orbits) }

-osvloc osv_location (secs)

-reftyp ref_file_type

-ref reference_file

-poftyp pof_file_type

[**-precfile** precise_conf_file] (empty string by default)

[**-dir** output_dir] (current directory by default)

[**-pof** output_filename] (default: name generated automatically)

[**-flcl** file_class] (empty string by default)

[**-vers** version] (version = 1 by default)

[**-eoffs** ffs_version] (Earth Observation File Format Standard Version)

[**-fhsys** fh_system] (empty string by default)

[**-v**]

[**-xl_v**]

[**-xo_v**]

[**-help**]

[**-show**]

[**-with_xslt]** (add xslt reference with default style sheet)

[(**-tai** TAI_time **-gps** GPS_time **-utc** UTC_time **-ut1** UT1_time) |

(**-tmod** time_model **-tfile** time_file **-trid** time_reference

{(**-tm0** time0 **-tm1** time1) | (**-orb0** orbit0 **-orb1** orbit1) })]

Note that:

- Order of parameters does not matter.
- Bracketed parameters are not mandatory.
- Options between curly brackets and separated by a vertical bar are mutually exclusive.
- [**-xl_v**] option for EO_LIB Verbose mode.
- [**-xo_v**] option for EO_ORBIT Verbose mode.
- [**-v**] option for Verbose mode for all libraries (default is Silent).
- [**-show**] displays the inputs of the function and the results.
- Possible values for *satellite_name*: ERS1, ERS2, ENVISAT, METOP1 , METOP2, METOP3, CRYOSAT, ADM, GOCE, SMOS, TERRASAR, EARTHCARE, SWARM_A, SWARM_B, SWARM_C, SENTINEL_1A, SENTINEL_1B, SENTINEL_2, SENTINEL_3, SENTINEL_1C, SENTINEL_2A, SENTINEL_2B, SENTINEL_2C, SENTINEL_3A, SENTINEL_3B, SENTINEL_3C, SEOSAT, JASON_CSA, JASON_CSB, METOP_SG_A1, METOP_SG_A2, METOP_SG_A3, METOP_SG_B1, METOP_SG_B2, METOP_SG_B3, SENTINEL_5P, BIOMASS, SENTINEL_5, SAOCOM_CS, FLEX, SENTINEL_6A, SENTINEL_6B, CIMR,

ROSE-L, CHIME, CRISTAL, CO2M, LSTM, FORUM, TRUTHS, GENERIC, GENERIC_GEO, MTG.

- Possible values for *time_model*: USER, NONE , IERS_B_PREDICTED, IERS_B_RESTITUTED, FOS_PREDICTED, FOS_RESTITUTED, DORIS_PRELIMINARY, DORIS_PRECISE, DORIS_NAVIGATOR, OSF.
- Possible values for *ref_file_type* and *pof_file_type*: OSF, POF, DORISNAV, ROF, TLE, DORISPREM, DORISPREC.
- Possible values for *ffs_version*: 0 (Default FFS), 1 (FFS version 1), 2 (FFS version 2), 3 (FFS version 3).
- Possible values for *time_ref* and *time_reference*: UNDEF, TAI, UTC, UT1.
- The value of the tag Time_Reference in variable header is set using the input parameter *time_ref*.
- Time references need to be initialized only when using OSF as the type of the input reference file. The inputs needed for this issue are provided in the last three lines of parameters. Note that only one set of parameters should be introduced:

– TAI, GPS, UTC and UT1 input times (as in xl_time_ref_init)

– A file with time reference data, the time mode, the time reference name and a time range (as in xl_time_ref_init_file)

- Precise propagation is used if precfile is provided.

Example:

```
gen_pof -sat CRYOSAT -tref TAI -ostart 13 -ostop 14 -osvloc 0 -reftyp OSF
     -ref CS_TEST_MPL_ORBREF_20020301T122001_99999999T999999_0001.EEF
-poftyp POF -dir ./gen_pof/ -pof orb_pre_file_at_307
         -tai -1100.1 -utc -1100.099595 -ut1 -1100.0995914352
         -gps -1100.0997801
```

7.53 xo_gen_oef

7.53.1 Overview

The **xo_gen_oef** CFI function creates an Orbit Event by merging an Orbit Scenario file (OSF) and a Predicted Orbit File.

In order to read and write files, xo_gen_oef function internally uses Data Handling functions. Please refer to [D_H_SUM], in particular sections 4.2 and 4.3, for further details.

Orbit Event File is deprecated and is only supported for Cryosat mission.

7.53.2 Calling interface

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

```
#include <explorer_orbit.h>
{
  char   oef[XD_MAX_STR], osf[XD_MAX_STR],
             pof[XD_MAX_STR];
  char  *file_class, *fh_system;
  long version_number;
  long   status, ierr[XO_NUM_ERR_GEN_OEF];
  status = xo_gen_oef(&oef, &osf, &pof,
                          file_class, &version_number, fh_system,
                          /* output */
                          ierr);
}
```

7.53.3 Input parameters

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

Table 146: Input parameters of xo_gen_oef function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
oef	char[]	-	Output OEF name. If empty (i.e. ""), the software will generate the filename according to file name specification presented in [FORMATS]. In such case, the generated name is returned in this variable	-	-
osf	char*	-	Orbit Scenario File name	-	-
pof	char*	-	Predicted Orbit File name	-	-
file_class	char*	-	File class for output file (dummy in the current version)	-	-
version_number	long*	-	Version number of output file (dummy	-	>= 1

			in the current version)		
fh_system	char*	-	System field of the output file fixed header (dummy in the current version)	-	-

7.53.4 Output parameters

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

Table 147: Output parameters of xo_gen_oef function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_gen_oef	long	-	Main status flag	-	-1, 0, +1
oef	char*	-	Name for the output file. This is only an output parameter when it is empty (i.e. ““; see description of this parameter in Table 146)	-	-
ierr[]	long	all	Status vector	-	-

7.53.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_gen_oef** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_gen_oef** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 148: Error messages of xo_gen_oef function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Could not open output file for writing	Computation not performed	XO_CFI_GEN_OEF_OPEN_FILE_ERR	0
ERR	Could not copy the Orbit Scenario file	Computation not performed	XO_CFI_GEN_OEF_COPY_FILE_ERR	1
ERR	Could not copy "List_of_OSVs" in the output file	Computation not performed	XO_CFI_GEN_OEF_COPY_NODE_ERR	2
ERR	Could not close output file	Computation not performed	XO_CFI_GEN_OEF_CLOSE_ERR	3

ERR	Error reading the fixed header from the Orbit Scenario file	Computation not performed	XO_CFI_GEN_OEF_READ_OSF_ERR	4
ERR	Error reading the fixed header from the Predicted Orbit file	Computation not performed	XO_CFI_GEN_OEF_READ_POF_ERR	5
ERR	Could not write the Orbit Event file	Computation not performed	XO_CFI_GEN_OEF_WRITE_ERR	6
ERR	Could not get the current time	Computation not performed	XO_CFI_GEN_OEF_CURRENT_TIME_ERR	7
WARN	Cannot write schema in the file	Computation performed. The output file does not contain the schema reference in the root tag	XO_CFI_GEN_OEF_SET_SCHEMA_WARN	8

7.53.6 Executable Program

The **gen_oef** executable program can be called from a Unix shell as:

gen_oef **-osf** name of the orbit scenario file
-pof
[**-oef**] (default: name generated automatically)
[**-flcl** file_class] (empty string by default)
[**-vers** version] (version = 1 by default)
[**-fhsys** fh_system] (empty string by default)
[**-v**]
[**-eoffs** ffs_version] (Earth Observation File Format Standard Version)
[**-xd_v**]
[**-xl_v**]
[**-xo_v**]
[**-help**]
[**-show**]

Note that:

- Order of parameters does not matter.
- Bracketed parameters are not mandatory.
- Options between curly brackets and separated by a vertical bar are mutually exclusive.
- [-xl_v] option for EO_LIB Verbose mode.

- [-xo_v] option for EO_ORBIT Verbose mode.
- [-v] option for verbose mode for all libraries (default is Silent).
- [-show] displays the inputs of the function and the results.
- Possible values for *ffs_version*: 0 (Default FFS), 1 (FFS version 1), 2 (FFS version 2), 3 (FFS version 3).

Example:

```
gen_oef -osf ./input_osf.xml -pof ./input_pof.xml
        -flcl OPER -vers 0 -show -v
```

7.54 xo_gen_dnf

7.54.1 Overview

The **xo_gen_dnf** CFI function creates a DORIS Navigator File using as input one of the following reference file types:

- Orbit Scenario File
- FOS Predicted Orbit File
- FOS Restituted Orbit File
- DORIS Navigator File(s).
- DORIS Preliminary Orbit File
- DORIS Precise Orbit FileTime of the ascending crossing node (TAI, UTC and UT1)
- The accepted output file types are:
- FOS Restituted Orbit File
- DORIS Preliminary Orbit File
- DORIS Precise Orbit FileTime

The time interval between consecutive OSVs can be selected by the user by means of a parameter in the calling interface. A flag for precise location of OSVs at “integer intervals” (e.g. every exact minute or every ten seconds) is also available. If the reference file and the DORIS Navigator File contain OSVs at the same time, these OSVs will be identical.

An optional control file can be introduced to correct the state vectors. This file contains the corrections for position and velocity in the along, across and radial directions. The format of this file is shown in [D_H_SUM].

A file with the configuration parameters for precise propagator can be introduced. In this case, the numeric propagator is used.

In order to read and write files, xo_gen_dnf function internally uses Data Handling functions. Please refer to [D_H_SUM], in particular sections 4.2 and 4.3, for further details.

Note: when using an OSF or Predicted Orbit file, the maximum time interval within the output Doris Navigator file is limited to 2 orbital periods before and after the middle point of the user requested time range.

7.54.2 Calling interface

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

```
#include <explorer_orbit.h>
{
  long   sat_id;
  xl_time_id time_id = {NULL};
  xl_model_id model_id = {NULL};
  long   time_init, time_ref, start_orbit, stop_orbit,
```

```
            ref_filetype, dnf_filetype, osv_precise, version_number;
    double start_time, stop_time, osv_interval;
    char   reference_file[XD_MAX_STR], output_dir[XD_MAX_STR],
            dnf_filename[XD_MAX_STR], ctrl_file[XD_MAX_STR],
            precise_conf_file[XD_MAX_STR];
    char  *file_class, *fh_system;
    long   status, ierr[XO_ERR_VECTOR_MAX_LENGTH];

    status = xo_gen_dnf(&sat_id, &model_id, &time_id,
                        &time_init, &time_ref,
                        &start_time, &stop_time,
                    &start_orbit, &stop_orbit,
                        &osv_interval, &osv_precise,
                        &ref_filetype, reference_file, ctrl_file,
                        precise_conf_file,
                        &dnf_filetype, output_dir, dnf_filename,
                        file_class, &version_number, fh_system,
                         /* output */
                         ierr);

    /* Or, using the run_id */
    long run_id;

    status = xo_gen_dnf_run(&run_id,
                            &time_init, &time_ref,
                            &start_time,  &stop_time,
                        &start_orbit, &stop_orbit,
                            &osv_interval, &osv_precise,
                            &ref_filetype, reference_file,ctrl_file,
                            precise_conf_file,
                            &dnf_filetype, output_dir, dnf_filename,
                            file_class, &version_number, fh_system,
                             /* output */
                             ierr);
}
```

7.54.3 Input parameters

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

Table 149: Input parameters of xo_gen_dnf function

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

sat_id	long *	-	Satellite ID	-	Complete
model_id	xl_model_id	-	Model ID	-	-
time_id	xl_time_id*	-	Structure that contains the time correlations. NOTE: If time_id is not initialized, then time correlations will be initialized internally using the input reference file	-	-
time_init	long*	-	Flag for selecting the time range of the initialisation.	-	Select either: · XO_SEL_ORBIT · XO_SEL_TIME
time_ref	long*	-	Time reference ID (see note in the ref_filetype field)	-	Complete
start_time	double*	-	Processing time corresponding to the beginning of the required interval	Decimal days, MJD2000	[-18262.0,36524.0]
stop_time	double*	-	Processing time corresponding to the end of the required interval	Decimal days, MJD2000	[-18262.0,36524.0]
start_orbit	long*	-	Orbit number corresponding to the beginning of the required interval	orbits	>= 1
stop_orbit	long*	-	Orbit number corresponding to the end of the required interval	orbits	>= 1
osv_interval	double*	-	Interval between consecutive state vector. This parameter should be coherent with the osv_precise flag (see below). If osv_precise is set to: xo_OSV_PRECISE_MINUTE: osv will be forced to be a multiple of 60 seconds. XO_OSV_PRECISE_TEN_SECONDS: osv will be forced to be a multiple of 10 seconds.	secs	>=0
osv_precise	long*	-	Flag to indicate if state vectors should be placed at exact time locations	-	Complete
ref_filetype	long*	-	File type of the input reference file. (Note: When generating a DNF file from another DORIS NAVIGA TOR file, the input times should be expressed in UTC)	-	Complete

reference_filename	char*	-	Reference File name	-	
ctrl_file	char*	-	Control File in xml format. This file contains the corrections for position and velocity in the along, across and radial directions together with the position accuracy (see [D_H_SUM].) If empty string (““), no corrections will be performed and the accuracy (quality index in the DNF)will be set to 1.	-	-
precise_conf_file	char*	-	File with precise propagator con figuration	-	If it is not neither NULL nor ““, precise propagation will be used
dnf_filetype	long*	-	File type of the output DORIS Navigator file	-	XO_REF_FILETY P E_DORIS_NAV
output_dir	char*	-	Directory where the resulting DNF is written (if NULL, the current directory is used)	-	-
dnf_filename	char*	-	Output DNF name if empty (i.e. ““), the software will generate the filename according to file name specification pre sented in [FORMATS]. In such case, the generated name is returned in this variable	-	-
file_class	char*	-	File class for output file (dummy in the current version)	-	-
version_number	long*	-	Version number of output file (dummy in the current version)	-	>= 1
fh_system	char*	-	System field of the output file fixed header (dummy in the current version)	-	-

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

- Satellite ID: sat_id.
- Time initialisation: time_init.
- Time reference: time_ref.
- OSV precise: osv_precise. See this SUM.
- File type: ref_filetype and rof_filetype. See this SUM.

7.54.4 Output parameters

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

Table 150: Output parameters of xo_gen_dnf function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
dnf_filename	char*	-	Name for the output file. This is only an output parameter when it is empty (i.e. ""; see description of this parameter in Table 149)	-	-
ierr[XO_ERR_VECTOR_MAX_LENGTH]	long	all	Status vector	-	-

7.54.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_gen_dnf** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_gen_dnf** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 151: Error messages of xo_gen_dnf function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Wrong satellite flag	Computation not performed	XO_CFI_GEN_DNF_WRONG_SAT_ID_ERR	0
ERR	Wrong input flag	Computation not performed	XO_CFI_GEN_DNF_WRONG_FLAG_ERR	1
ERR	Time ID is not initialized	Computation not performed	XO_CFI_GEN_DNF_TIME_INIT_ERR	2
ERR	Could not initialise the time reference	Computation not performed	XO_CFI_GEN_DNF_TIME_INITIALIZATION_ERR	3
ERR	Cannot initialise orbit ID	Computation not performed	XO_CFI_GEN_DNF_ORBIT_INIT_FILE_ERR	4
ERR	Cannot initialise the propagator	Computation not performed	XO_CFI_GEN_DNF_PROPAG_INIT_ERR	5

ERR	Cannot initialise interpolation	Computation not performed	XO_CFI_GEN_DNF_INTERPOL_INIT_ERR	6
ERR	Could not perform a time <-> orbit transformation	Computation not performed	XO_CFI_GEN_DNF_TIME_ORBIT_ERR	7
ERR	Error in a time transformation function	Computation not performed	XO_CFI_GEN_DNF_TIME_ERR	8
ERR	Memory allocation error	Computation not performed	XO_CFI_GEN_DNF_MEMORY_ERR	9
ERR	Cannot calculate state vector	Computation not performed	XO_CFI_GEN_DNF_CALCULATING_STATE_VECTOR_ERR	10
ERR	Error reading the Control File	Computation not performed	XO_CFI_GEN_DNF_READ_ CONTROL_FILE_ERR	11
ERR	Cannot correct state vector	Computation not performed	XO_CFI_GEN_DNF_CORRECT_OSV_ERR	12
ERR	Error changing state vector from EF to J2000	Computation not performed	XO_CFI_GEN_DNF_CHANGE_COORD_ERR	13
ERR	Error creating the DORIS header	Computation not performed	XO_CFI_GEN_DNF_COMPUTE_HEADER_ERR	14
ERR	Error freeing memory	Computation not performed	XO_CFI_GEN_DNF_CLOSE_ERR	15
ERR	Cannot write DORIS Data Block file	Computation not performed	XO_CFI_GEN_DNF_WRITE_FILE_ERR	16
WARN	OSV interval is not compatible with OSV Precise flag. The OSV Interval will be set to %f seconds.	Computation performed with a different value for the osv_interval	XO_CFI_GEN_DNF_WRONG_INTERVAL_WARN	17
ERR	Error reading precise propagator configuration file	Computation not performed	XO_CFI_GEN_DNF_READ_ PRECISE_FILE_ERR	18

7.54.6 Executable Program

The **gen_dnf** executable program can be called from a Unix shell as:

gen_dnf **-sat** satellite_name
-tref time_ref
{ -tstart start_time -tstop stop_time (decimal days) |
-tastart start_time -tastop stop_time (CCSDSA format) |
-ostart start_orbit -ostop stop_orbit (orbits) }
-osvint osv_interval
[**-osvpre**]

-reftyp ref_file_type

-ref reference_file

[**-ctrl** control_file]

[**-precfile** precise_conf_file] (empty string by default)

[**-dir** output_dir] (current directory by default)

[**-dnf** output_filename] (default: name generated automatically)

[**-flcl** file_class] (empty string by default)

[**-vers** version] (version = 1by default)

[**-eoffs** ffs_version] (Earth Observation File Format Standard Version)

[**-fhsys** fh_system] (empty string by default)

[**-v**]

[**-xl_v**]

[**-xo_v**]

[**-help**]

[**-show**]

[(**-tai** TAI_time **-gps** GPS_time **-utc** UTC_time **-ut1** UT1_time) |

(**-tmod** time_model **-tfile** time_file **-trid** time_reference

{(**-tm0** time0 **-tm1** time1) | (**-orb0** orbit0 **-orb1** orbit1) })]

Note that:

- Order of parameters does not matter.
- Bracketed parameters are not mandatory.
- Options between curly brackets and separated by a vertical bar are mutually exclusive.
- [**-osvpre**] option for osv_precise.Default value is xo_OSV_PRECISE_NO. When the option is written, ovs_precise value is xo_OSV_PRECISE_MINUTE.
- [**-xl_v**] option for EO_LIB Verbose mode.
- [**-xo_v**] option for EO_ORBIT Verbose mode.
- [**-v**] option for Verbose mode for all libraries (default is Silent).
- [**-show**] displays the inputs of the function and the results.
- Possible values for *satellite_name*: ERS1, ERS2, ENVISAT, METOP1 , METOP2, METOP3, CRYOSAT, ADM, GOCE, SMOS, TERRASAR, EARTHCARE, SWARM_A, SWARM_B, SWARM_C, SENTINEL_1A, SENTINEL_1B, SENTINEL_2, SENTINEL_3, SENTINEL_1C, SENTINEL_2A, SENTINEL_2B, SENTINEL_2C, SENTINEL_3A, SENTINEL_3B, SENTINEL_3C, SEOSAT, JASON_CSA, JASON_CSB, METOP_SG_A1, METOP_SG_A2, METOP_SG_A3, METOP_SG_B1, METOP_SG_B2, METOP_SG_B3, SENTINEL_5P, BIOMASS, SENTINEL_5, SAOCOM_CS, FLEX, SENTINEL_6A, SENTINEL_6B, CIMR, ROSE-L, CHIME, CRISTAL, CO2M, LSTM, FORUM, TRUTHS, GENERIC, GENERIC_GEO, MTG.

- Possible values for *time_model*: USER, NONE , IERS_B_PREDICTED, IERS_B_RESTITUTED, FOS_PREDICTED, FOS_RESTITUTED, DORIS_PRELIMINARY, DORIS_PRECISE, DORIS_NAVIGATOR, OSF.
- Possible values for *ref_file_type*: OSF, POF, DORISNAV, ROF, DORISPREM, DORISPREC.
- Possible values for *ffs_version*: 0 (Default FFS), 1 (FFS version 1), 2 (FFS version 2), 3 (FFS version 3).
- Possible values for *time_ref* and *time_reference*: UNDEF, TAI, UTC, UT1, GPS.
- Time references need to be initialized only when using OSF as the type of the input reference file. The inputs needed for this issue are provided in the last three lines of parameters. Note that only one set of parameters should be introduced:

– TAI, GPS, UTC and UT1 input times (as in xl_time_ref_init)

– A file with time reference data, the time mode, the time reference name and a time range (as in xl_time_ref_init_file)

- Precise propagation is used if precfile is provided.

Example:

```
gen_dnf -sat CRYOSAT -tref UTC -tstart 0.99650462962963
        -tstop 01386574074708 -osvint 20 -reftyp ROF
        -ref EARTH_EXPLORER_FRO_TO_DORIS_2000
        -ctrl CONTROL_FILE.xml -dir ./gen_dnf/ -dnf doris_nav_at_308
        -tai 0.000000 -utc -4.0509259e-4 -ut1 -4.1435185185e-4
             -gps 2.1991e-4 -show
```

7.55 xo_gen_tle

7.55.1 Overview

The **xo_gen_tle** CFI function creates TLE File using as input a Predicted or Restituted Orbit File. It is possible to select the way in which the TLE records are generated:

- Generate a TLE record per OSV in the orbit file (XO_ONE_TLE_PER_OSV).
- Find the best TLE record which fits to the OSVs in the orbit file (XO_FIT_TLE/XO_FIT_TLE_LIST).

These modes are explained in more detail in section 7.55.3.

In order to read and write files, xo_gen_tle function internally uses Data Handling functions. Please refer to [D_H_SUM], in particular sections 4.2 and 4.3, for further details.

7.55.2 Calling interface

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

```
#include <explorer_orbit.h>
{
  long   sat_id;
  xl_time_id time_id = {NULL};
  long   fit_mode, time_mode, time_ref, start_orbit, stop_orbit;
  double start_time, stop_time;
  char   reference_file[XD_MAX_STR], tle_filename[XD_MAX_STR];
  long   status, ierr[XO_ERR_VECTOR_MAX_LENGTH];

  status = xo_gen_tle (&sat_id, &fit_mode,
                       &time_mode, &time_ref,
                       &start_time, &stop_time,
                       &start_orbit, &stop_orbit,
                       reference_file, tle_filename,
                       ierr);
}
```

7.55.3 Input parameters

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

Table 152: Input parameters of xo_gen_tle function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete

fit_mode	long*	-	fitting mode	-	XO_FIT_TLE XO_FIT_TLE_LIST XO_ONE_TLE_PER_OSV
time_mode	long*	-	Flag for selecting the time range of the initialization.	-	Select either: XO_SEL_ORBIT XO_SEL_TIME XO_SEL_DEFAUL
time_ref	long*	-	Time reference for the input start_time and stop_time	-	Complete
start_time	double*	-	Processing time corresponding to the beginning of the required interval	Decimal days, MJD2000	[-18262.0,36524.0]
stop_time	double*	-	Processing time corresponding to the end of the required interval	Decimal days, MJD2000	[-18262.0,36524.0]
start_orbit	long*	-	Orbit number corresponding to the beginning of the required interval	orbits	>= 1
stop_orbit	long*	-	Orbit number corresponding to the end of the required interval	orbits	>= 1
reference_file	char*	-	Reference File name	-	-

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

- Fitting mode: fit_mode. See this SUM.
- Satellite ID: sat_id.
- Time initialisation: time_mode.
- Time reference: time_ref.

The behavior of the generator depends on the input arguments *fit_mode* and *input file type*:

- XO_ONE_TLE_PER_OSV: one TLE record will be computed for each Orbit State Vector in the input file and written into the output TLE file.
- XO_FIT_TLE: a single TLE record will be computed and written in the output TLE file by fitting a list of OSVs obtained as follows:
  - If the input is a POF, the list will be generated by propagating in the selected time interval,
  - otherwise (ROF), the list will be taken from the file in the selected time interval as it is.
- XO_FIT_TLE_LIST: a single TLE record will be computed and written in the output TLE file by fitting a list of OSVs taken from the file in the selected time interval as it is.

7.55.4 Output parameters

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

Table 153: Output parameters of xo_gen_tle function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
tle_filename	char*	-	Name for the output file.	-	-
ierr[XO_ERR_VECTOR_MAX_LENGTH]	long	all	Status vector	-	-

7.55.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_gen_tle** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_gen_tle** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 154: Error messages of xo_gen_tle function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Wrong satellite ID.	Computation not performed	XO_CFI_GEN_TLE_WRONG_SAT_ID_ERR	0
ERR	Wrong input time reference	Computation not performed	XO_CFI_GEN_TLE_WRONG_FLAG_ERR	1
ERR	Wrong input fitting mode	Computation not performed	XO_CFI_GEN_TLE_WRONG_FIT_MODE_ERR	2
ERR	Could not initialise the time correlations from input file: %s	Computation not performed	XO_CFI_GEN_TLE_TIME_INITIALIZATION_ERR	3
ERR	Could not initialise the orbit data from input file: %s	Computation not performed	XO_CFI_GEN_TLE_ORBIT_INIT_FILE_ERR	4
ERR	Memory allocation error	Computation not performed	XO_CFI_GEN_TLE_MEMORY_ERR	5
ERR	Could not generate the TLE for orbit %ld	Computation not performed	XO_CFI_GEN_TLE_OSV_TO_TLE_ERR	6
ERR	Could not close an ID	Computation not performed	XO_CFI_GEN_TLE_CLOSE_ERR	7
ERR	Could not write output file to disk	Computation not performed	XO_CFI_GEN_TLE_WRITE_FILE_ERR	8
ERR	Could not propagate orbit for	Computation not performed	XO_GEN_TLE_OSV_COM	9

	time		PUTE_ERR	
ERR	Error getting orbit for provided time	Computation not performed	XO_GEN_TLE_TIME_TO_ORBIT_ERR	10
ERR	Cannot generate TLE with input XD_USER_OSV_LIST_TYPE and Fit mode XO_ONE_TLE_PER_OSV	Computation not performed	XO_GEN_TLE_INVALID_INIT_MODE_ERR	11
WARN	Error generating the output filename. Creating file: %s	File generated with auxiliary name and not following the file format standard convention	XO_GEN_TLE_FILENAME_WARN	12

7.55.6 Executable Program

The **gen_tle** executable program can be called from a Unix shell as:

gen_tle **-sat** satellite_name
-tref time_ref
-**tastart** start_time -**tastop** stop_time (CCSDSA format) |
-ref reference_file
-fit_mode fit_mode
-tle tle_name
[-**tstep** num_days]
[-**eoffs** ffs_version] (Earth Observation File Format Standard Version)
[-**dir** output_dir] (current directory by default)
[-**v**]
[-**xl_v**]
[-**xo_v**]
[-**help**]
[-**show**]

Note that:

- Order of parameters does not matter.
- Bracketed parameters are not mandatory.
- Options between curly brackets and separated by a vertical bar are mutually exclusive.
- [-**xl_v**] option for EO_LIB Verbose mode.

- [**-xo_v**] option for EO_ORBIT Verbose mode.
- [**-v**] option for Verbose mode for all libraries (default is Silent).
- [**-show**] displays the inputs of the function and the results.
- Possible values for *satellite_name*: ERS1, ERS2, ENVISAT, METOP1 , METOP2, METOP3, CRYOSAT, ADM, GOCE, SMOS, TERRASAR, EARTHCARE, SWARM_A, SWARM_B, SWARM_C, SENTINEL_1A, SENTINEL_1B, SENTINEL_2, SENTINEL_3, SENTINEL_1C, SENTINEL_2A, SENTINEL_2B, SENTINEL_2C, SENTINEL_3A, SENTINEL_3B, SENTINEL_3C, SEOSAT, JASON_CSA, JASON_CSB, METOP_SG_A1, METOP_SG_A2, METOP_SG_A3, METOP_SG_B1, METOP_SG_B2, METOP_SG_B3, SENTINEL_5P, BIOMASS, SENTINEL_5, SAOCOM_CS, FLEX, SENTINEL_6A, SENTINEL_6B, CIMR, ROSE-L, CHIME, CRISTAL, CO2M, LSTM, FORUM, TRUTHS, GENERIC, GENERIC_GEO, MTG.
- Possible values for *ffs_version*: 0 (Default FFS), 1 (FFS version 1), 2 (FFS version 2), 3 (FFS version 3).
- Possible values for *time_ref* and *time_reference*: UNDEF, TAI, UTC, UT1.
- Possible values for *fit_mode*:
- ONE_TLE_PER_OSV: one TLE record will be computed for each Orbit State Vector in the input file and written into the output TLE file.
- FIT: a single TLE record will be computed and written in the output TLE file by fitting a list of OSVs obtained as follows:
- If the input is a POF, the list will be generated by propagating in the selected time interval,
- otherwise (ROF), the list will be taken from the file in the selected time interval as it is.
- FIT_LIST: a single TLE record will be computed and written in the output TLE file by fitting a list of OSVs taken from the file in the selected time interval as it is.
- If *tstep* is defined, one TLE record will be computed for each *tstep* time interval, from start_time until stop_time, and written in the output TLE file.

  Depending on the inputs the output will change:

  • ONE_TLE_PER_OSV: the generator will propagate and generate one OSV every *tstep* time, from start_time until stop_time, and then, for each OSV it will create one TLE record

  • FIT: One TLE to be generated every *tstep* time, from start_time until stop_time, by fitting a list of OSVs obtained as described above.

  • FIT_LIST: FIT_LIST: for each *tstep* interval, a single TLE record will be computed (and written in the output TLE file) by fitting a list of OSVs taken from the file within the selected *tstep* time interval.

Example:

```
gen_tle -sat SENTINEL_1A
        -tref UTC
        -tastart "2013-07-06T00:00:00.000000"
      -tastop "2013-07-06T23:59:00.000000"
        -tstep 0.5
        -ref OSV_LIST.EOF
```

```
-fit_mode FIT
-tle S1A_TLE_OUT.TLE
-show
```

7.56 xo_check_osf

7.56.1 Overview

The **xo_check_osf** CFI function checks the continuity of the orbital parameters at the transition from one orbital change and the next one in an Orbit Scenario file.

In order to read and write files, xo_check_osf function internally uses Data Handling functions. Please refer to [D_H_SUM], in particular sections 4.2 and 4.3, for further details.

7.56.2 Calling interface

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

```
#include <explorer_orbit.h>
{
  long   sat_id;
  xl_model_id model_id = {NULL};
  xl_time_id time_id = {NULL};
  char  *osf_file;
  long transition_number;
      double threshold[XO_NUM_CHECK_PARAMS],
             diffs[XO_NUM_CHECK_PARAMS];
  long status, ierr[XO_ERR_VECTOR_MAX_LENGTH];

  status = xo_check_osf(&sat_id, &model_id, &time_id,
                        osf_file, &transition_number,
                        threshold,
                          /* output */
                          diffs, ierr);

  /* Or, using the run_id */
  long run_id;

  status = xo_check_osf_run(&run_id,
                            osf_file, &transition_number,
                            threshold,
                              /* output */
                              diffs, ierr);
}
```

7.56.3 Input parameters

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

Table 155: Input parameters of xo_check_osf function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
model_id	xl_model_id*	-	Model ID	-	-
time_id	xl_time_id*	-	Structure that contains the time correlations.	-	-
osf_file	char*	-	Orbit Scenario file to be checked	-	-
transition_number	long*	-	Number of the transition to be checked. If 0, the last transition is checked	-	>1 <=Number of transitions
threshold	double [XO_NUM_CHECK_PARAMS]	0	Threshold for the time at ANX	s	>0
		1	Threshold for the ANX longitude	deg	
		2	Threshold for the MLST	s	
		3	Threshold for the osculating semi-axis major	m	
		4	Threshold for the osculating inclination	deg	
		5	Threshold for the nodal period	s	

7.56.4 Output parameters

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

Table 156: Output parameters of xo_check_osf function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_check_osf	long	-	Status	-	-1, 0, 1
diffs	double [XO_NUM_CHECK_PARAMS]	0	Difference for the time at ANX	s	>0
		1	Difference for the ANX longitude	deg	
		2	Difference for the MLST	s	
		3	Difference for the osculating semi-axis major	m	
		4	Difference for the osculating	deg	

			inclination		
		5	Difference for the nodal period	s	
ierr	long*	all	Status vector	-	

7.56.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_check_osf** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_check_osf** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 157: Error messages of xo_ckeck_osf function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Time correlations are not initialized	Computation not performed	XO_CFI_CHECK_OSF_TIME_INIT_ERR	0
ERR	Error reading the Orbit Scenario file	Computation not performed	XO_CFI_CHECK_OSF_OSF_READ_ERR	1
ERR	Wrong transition number	Computation not performed	XO_CFI_CHECK_OSF_WRONG_TRANSITION_ERR	2
ERR	Couldn't initialize the orbit	Computation not performed	XO_CFI_CHECK_OSF_ORBIT_INIT_ERR	3
ERR	Error in xo_orbit_info	Computation not performed	XO_CFI_CHECK_OSF_ORBIT_INFO_ERR	4
WARN	UTC at ANX exceeds the input threshold	Computation performed	XO_CFI_CHECK_OSF_UTC_WARN	5
WARN	ANX Longitude exceeds the input threshold	Computation performed	XO_CFI_CHECK_OSF_ANX_LONG_WARN	6
WARN	MLST exceeds the input threshold	Computation performed	XO_CFI_CHECK_OSF_MLST_WARN	7
WARN	Osculating semi-major axis exceeds the input threshold	Computation performed	XO_CFI_CHECK_OSF_OSC_A_WARN	8
WARN	Osculating inclination exceeds the input threshold	Computation performed	XO_CFI_CHECK_OSF_OSC_I_WARN	9
WARN	Nodal period exceeds the input threshold	Computation performed	XO_CFI_CHECK_OSF_TNOD_WARN	10

7.57 xo_check_oef

7.57.1 Overview

The **xo_check_oef** CFI function checks the consistency between the list of orbital changes and the list of state vectors in an Orbit Event file.

In order to read and write files, xo_check_oef function internally uses Data Handling functions. Please refer to [D_H_SUM], in particular sections 4.2 and 4.3, for further details.

Note: Orbit Event File is deprecated, only supported for CRYOSAT mission

7.57.2 Calling interface

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

```
#include <explorer_orbit.h>
{
  long   sat_id;
  xl_time_id time_id = {NULL};
  xl_model_id model_id = {NULL};
  char  *oef_file;
  long   time_mode, time_ref;
  double start_time, stop_time;
      long    start_orbit, stop_orbit;
  double threshold[XO_NUM_CHECK_PARAMS],
            max_diffs[XO_NUM_CHECK_PARAMS],
            rms[XO_NUM_CHECK_PARAMS];
  long status, ierr[XO_ERR_VECTOR_MAX_LENGTH];
  status = xo_check_oef(&sat_id, &model_id, &time_id,
                        &time_mode, &time_ref,
                        &start_time, &stop_time,
                        &start_orbit, &stop_orbit,
                        oef_file, threshold,
                          /* output */
                          max_diffs, rms, ierr);

  /* Or, using the run_id */
  long run_id;
  status = xo_check_oef_run(&run_id,
                            time_mode, &time_ref,
                            &start_time, &stop_time,
                            &start_orbit, &stop_orbit,
                            oef_file, threshold,
                             /* output */
                             max_diffs, rms, ierr);
}
```

7.57.3 Input parameters

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

Table 158: Input parameters of xo_check_oef function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
model_id	xl_model_id*	-	Model ID	-	-
time_id	xl_time_id*	-	Structure that contains the time correlations.	-	-
time_mode	long*	-	Flag for the input time range selection: whole file, time or orbits	-	XO_SEL_FILE, XO_SEL_TIME, XO_SEL_ORBIT
time_ref	long*	-	Time reference for the star_time and stop_time input parameters (only needed if time_mode is XO_SEL_TIME)	-	Complete
start_time	double*	-	Start time for the time range to be checked (only needed if time_mode is XO_SEL_TIME)	days	[-18262.0, 36524.0]
stop_time	double*	-	Stop time for the time range to be checked (only needed if time_mode is XO_SEL_TIME)	days	[-18262.0, 36524.0]
start_orbit	long*	-	Start orbit for the orbit range to be checked (only needed if time_mode is XO_SEL_ORBIT)	-	file range
stop_orbit	long*	-	Stop orbit for the orbit range to be checked (only needed if time_mode is XO_SEL_ORBIT)	-	file range
oef_file	char*	-	Orbit Event file to be checked	-	-
threshold	double [XO_NUM_CHECK_PARAMS]	0	Threshold for the time at ANX	s	>0
		1	Threshold for the ANX longitude	deg	
		2	Threshold for the MLST	s	
		3	Threshold for the osculating semi-axis major	m	
		4	Threshold for the osculating inclination	deg	
		5	Threshold for the nodal period	s	

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

- Satellite ID: sat_id.
- Time inputs selection: time_mode

- Time reference: time_ref.

7.57.4 Output parameters

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

Table 159: Output parameters of xo_check_oef function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_check_oef	long	-	Status	_	-1, 0, 1
max_diffs	double [XO_NUM_CHECK _PARAMS]	All	The following parameters are computed using the list of orbital changes and the list of state vectors. The maximum value of these differences for the requested interval are returned in this array.	_	>0
		0	Time at ANX	s	
		1	ANX longitude	deg	
		2	MLST	s	
		3	Osculating semi-axis major	m	
		4	Osculating inclination	deg	
		5	Nodal period	s	
rms	double [XO_NUM_CHECK _PARAMS]	All	The following parameters are computed using the list of orbital changes and the list of state vectors. The standard deviation of these differences for the requested interval are returned in this array.	_	>0
		1	ANX longitude	deg	
		2	MLST	s	
		3	Osculating semi-axis major	m	
		4	Osculating inclination	deg	
		5	Nodal period	s	
ierr	long*	all	Status vector	_	

7.57.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_check_oef** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_check_oef** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 160: Error messages of xo_ckeck_oef function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Error in xo_orbit_info	Computation not performed	XO_CFI_CHECK_OEF_ORBIT_INIT_ERR	0
ERR	Couldn't initialize the orbit	Computation not performed	XO_CFI_CHECK_OEF_ORBIT_INFO_ERR	1
ERR	Memory allocation error	Computation not performed	XO_CFI_CHECK_OEF_MEM_ERR	2
WARN	UTC at ANX exceeds the input threshold	Computation performed	XO_CFI_CHECK_OEF_UTC_WARN	3
WARN	ANX Longitude exceeds the input threshold	Computation performed	XO_CFI_CHECK_OEF_ANX_LONG_WARN	4
WARN	MLST exceeds the input threshold	Computation performed	XO_CHECK_OEF_MLST_W ARN	5
WARN	Osculating semi-major axis exceeds the input threshold	Computation performed	XO_CFI_CHECK_OEF_OSC_A_WARN	6
WARN	Osculating inclination exceeds the input threshold	Computation performed	XO_CFI_CHECK_OEF_OSC_I_WARN	7
WARN	Nodal period exceeds the input threshold	Computation performed	XO_CFI_CHECK_OEF_TNOD_WARN	8

7.58 xo_position_on_orbit_to_time

7.58.1 Overview

The **xo_position_on_orbit_to_time** calculates the time, position and velocity vectors in Earth-Fixed associated to a given position on orbit. This position on orbit is defined as the angle between the satellite position and the intersection of the orbital plane with a reference plane (the reference plane is the equator in GM2000, ToD or EF CS).
COMPATIBILITY NOTE: the output of this function is consistent with the calculation of orbit number and time from ANX within the EO CFI only when the input angle is compliant with [EO_OPS] and [MCD] i.e. either ToD or EF. Using other angle types (e.g. J2000) will result in an output time that cannot be used as input elsewhere in the EO CFI, notably in the xo_time_to_orbit function.

7.58.2 Calling Interface

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

```
#include <explorer_orbit.h>
{
  xo_orbit_id orbit_id = {NULL};
  long * abs_orbit_number,
         angle_type,
         deriv,
         time_ref;
  double * angle,
           angle_rate,
           angle_rate_rate,
           time;
  double pos[3],
         vel[3],
         acc[3];
  long status,
       ierr[XO_ERR_VECTOR_MAX_LENGTH];

  status = xo_position_on_orbit_to_time (&orbit_id,
                   &abs_orbit_number, &angle_type,
                   angle, &angle_rate,&angle_rate_rate,
                   &deriv, &time_ref,
                   /* Output */
                   &time, pos, vel, acc,
                   ierr);

}
```

7.58.3 Input Parameters

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

Table 161: Input parameters of xo_position_on_orbit_to_time function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id *	-	Orbit ID	-	Complete
abs_orbit_number	long *	-	Absolute orbit number	-	Complete
angle_type	long *	-	Type of angle	-	XL_ANGLE_TYPE_TRUE_LAT_TOD XL_ANGLE_TYPE_TRUE_LAT_GM2000 XL_ANGLE_TYPE_TRUE_LAT_EF
angle	double *	-	Angle describing the position in the orbit	-	-
angle_rate	double *	-	1st derivate from Angle	-	-
angle_rate_rate	double *	-	2nd derivate from Angle	-	-
deriv	long *	-	Derivative ID	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
time_ref	long *	-	Time Reference		Complete

7.58.4 Output Parameters

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

Table 162: Output parameters of xo_position_on_orbit_to_time function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
time	double *	-	Resulting time	Decimal days (processing format)	-
pos	double[3]	all	Satellite position vector (Earth Fixed CS)	m	-
vel	double[3]	all	Satellite velocity vector (Earth Fixed CS)	m/s	-
acc	double[3]	all	Satellite acceleration vector (Earth Fixed CS)	m/s2	-

ierr	long	.	Error vector	-	-

7.58.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xo_position_on_orbit_to_time** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_position_on_orbit_to_time** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 163: Error messages of xo_position_on_orbit_to_time function

Error type	Error message	Cause and impact	Error code	Error No
ERR	The Orbit Id was not initialized.	Computation not performed	XO_POSITION_ON_ORBIT_ORBIT_INIT_STATUS_ERR	0
ERR	Error calculating orbit info	Computation not performed	XO_POSITION_ON_ORBIT_ORBIT_INFO_ERR	1
ERR	Could not compute the ANX time for the requested orbit	Computation not performed	XO_POSITION_ON_ORBIT_ORBIT_TO_TIME_ERR	2
ERR	Could not compute orbit state vectors	Computation not performed	XO_POSITION_ON_ORBIT_OSV_COMPUTE_ERR	3
ERR	Error calculating satellite position	Computation not performed	XO_POSITION_ON_ORBIT_POSITION_ON_ORBIT_ERR	4
ERR	Error calculating time: maximum iterations reached	Computation not performed	XO_POSITION_ON_ORBIT_MAX_ITERATIONS_ERR	5

7.59 xo_orbit_data_filter

7.59.1 Overview

The function **xo_orbit_data_filter** filters data stored in the *xo_orbit_id_init_data* input structures and returns the result of such filtering in *xo_orbit_id_init_data* output structure and associated statistics in *xo_orbit_filter_report* report structure. The user can select the filter and set its configuration via the filter setting structure.

7.59.1.1 Outliers filter

The outliers filter works this way:

- For every input state vector time, a new state vector is computed using interpolation, taking as input for interpolation the 10 state vectors around that time.
- The interpolated state vector is compared with the one present in the file. If the difference between them (in position or velocity) is bigger than the input thresholds, the state vector is removed.

7.59.2 Calling Interface

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

```
#include <explorer_orbit.h>
{
  long ierr[XO_ERR_VECTOR_MAX_LENGTH];
  xo_orbit_id_init_data    orbit_data_in;
  xo_orbit_filter_settings filter_settings;
  xo_orbit_id_init_data    orbit_data_out;
  xo_orbit_filter_report    report;

  status = xo_orbit_data_filter(&orbit_data_in, &filter_settings,
                                &orbit_data_out, &report, ierr);

  status = xo_orbit_data_filter_close(&orbit_data_out, ierr);

}
```

Note that it is necessary to call xo_orbit_data_filter_close() to free the memory reserved inside the orbit_data_filter function for the output data.

7.59.3 Input Parameters

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

Table 164: Input parameters of xo_orbit_data_filter function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_data_in	xo_orbit_id_init_data		Input data structure	-	
filter_settings	xo_orbit_filter_settings		Input filter structure	-	

7.59.4 Output Parameters

The **xo_orbit_data_filter** CFI function has the following output parameters:

Table 165: Output parameters of xo_orbit_data_filter function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_data_out	xo_orbit_id_init_data		Output data structure with filtered samples	-	
report	xo_orbit_filter_report		Output Report	-	
ierr	long	.	Error vector	-	-

7.59.5 Warnings and errors

The next table lists the possible error messages that can be returned by the **xo_orbit_data_filter** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_orbit_data_filter** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 166: Error messages of xo_orbit_data_filter function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Unknown Filter provided	Input filter not recognized. Execution aborted	XO_ORBIT_DATA_FILTER_UNKNOWN_FILTER_ERR	0
ERR	The Input Orbit file does not contain any OSV	No OSV samples found in the input files. Execution aborted.	XO_ORBIT_DATA_FILTER_NO_OSV_ERR	1
ERR	Input file not supported: Orbit or Doris only	Input file is neither an Orbit nor a Doris file. Execution Aborted.	XO_ORBIT_DATA_FILTER_FILE_ERR	2
ERR	Error Interpolating OSV for orbit <orbit #>	Execution Aborted.	XO_ORBIT_DATA_FILTER_INTERPOL_ERR	3
ERR	Error analysing sample to validate OSV	Execution Aborted.	XO_ORBIT_DATA_FILTER_ANALYSE_OSV_SAMPLE_ERR	4
ERR	Could not find 10 valid OSV samples to interpolate point at <time>	Execution Aborted.	XO_ORBIT_DATA_FILTER_SAMPLES_ERR	5

7.60 xo_orbit_id_change

7.60.1 Overview

The function **xo_orbit_id_change** updates the orbit number in the state vectors of an orbit id that has been previously initialized with Predicted Orbit, Restituted Orbit or DORIS files. The correction to be applied to orbit numbers is calculated depending on the input provided in the *change_data* input parameter:

- With an OSF (*change_data.change_mode* = XO_ORBIT_ID_CHANGE_OSF). In this case, the time of the first state vector in the orbit id is used to compute the orbit number that the OSF file predicts for that time. All the state vector orbit numbers are corrected accordingly: if the orbit number and time of the first OSV in the orbit id are respectively N and T and the orbit number of the OSV computed at the same time T using the OSF is M, then the orbit numbers of all OSVs in the orbit id are increased of M-N. The field *change_data.eocfi_file* is of type xd_eocfi_file (only OSF file type is supported).
- With an input TIME+ORBIT (*change_data.change_mode* = XO_ORBIT_ID_CHANGE_TIME_ORBIT). In this case, if T and M are respectively time and orbit provided by the user, the function first computes the OSV at time T and the corresponding orbit number N, then the orbit numbers of all OSVs in the orbit id are increased of M-N. Time T shall be within the orbit id validity.

7.60.2 Calling Interface

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

```
#include <explorer_orbit.h>
{
  long ierr[XO_ERR_VECTOR_MAX_LENGTH];
  xo_orbit_id_change_data change_data;
  xo_orbit_id    orbit_id;

  status = xo_orbit_id_change(&orbit_id, &change_data, ierr);

}
```

7.60.3 Input Parameters

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

Table 167: Input parameters of xo_orbit_id_change function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*		Input orbit id	-	
change_data	xo_orbit_change_data*		Input orbit correction structure	-	

7.60.4 Output Parameters

The **xo_orbit_id_change** CFI function has the following output parameters:

Table 168: Output parameters of xo_orbit_id_change function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xo_orbit_id*	-	This parameter is also an output parameter, since the orbits of the state vectors stored internally are changed according to input parameters to the function		
ierr	long	.	Error vector	-	-

7.60.5 Warnings and errors

The next table lists the possible error messages that can be returned by the **xo_orbit_id_change** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_orbit_id_change** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 169: Error messages of xo_orbit_id_change function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Invalid Change mode provided.	No computation performed	XO_CFI_ORBIT_ID_CHANGE_MODE_ERR	0
ERR	Provided Orbit Id change data is not valid.	No computation performed	XO_CFI_ORBIT_ID_CHANGE_CHANGE_DATA_ERR	1
ERR	Provided Orbit Id is NULL.	No computation performed	XO_CFI_ORBIT_ID_CHANGE_ID_NULL_ERR	2
ERR	Could not initialize time id with the provided OSF.	No computation performed	XO_CFI_ORBIT_ID_CHANGE_TIME_ID_ERR	3
ERR	Could not initialize orbit id with the provided OSF.	No computation performed	XO_CFI_ORBIT_ID_CHANGE_ORBIT_ID_ERR	4
ERR	Error propagating time of first OSV of the Orbit file using the provided OSF.	No computation performed	XO_CFI_ORBIT_ID_CHANGE_PROPAG_ERR	5
ERR	Error propagating time %f using the provided OSF.	No computation performed	XO_CFI_ORBIT_ID_CHANGE_PROP_TIME_ERR	6

ERR	Error closing orbit_id initialized with the provided OSF for orbit change.	No computation performed	XO_CFI_ORBIT_ID_CHANGE_CLOSE_ORBIT_ID_ERR	7
ERR	Error closing time_id initialized with the provided Orbit file for orbit change.	No computation performed	XO_CFI_ORBIT_ID_CHANGE_CLOSE_TIME_ID_ERR	8

7.61 xo_osv_check

7.61.1 Overview

The function **xo_osv_check** provides a way to check if an orbit state vector (OSV) is compatible with the nominal orbit of a given satellite. The function checks that semi-major axis, inclination and eccentricity corresponding to the OSV are within certain tolerances. Two types of tolerances are defined (the values for each satellite can be found in section 5 of [MSC]):

- *Tight tolerances*: these are tolerances very close to the nominal orbit expected for the satellite. If the state vector values are inside the interval defined by these tolerances, the state vector is considered valid. If is outside the interval but inside the interval defined by loose tolerances (see next point), the state vector is considered correct but slightly away from nominal orbit. A warning is returned in this case (XO_CFI_OSV_CHECK_TIGHT_TOLERANCE_WARN).
- *Loose tolerances*: these are wider tolerances. If the state vector values are outside the interval defined by these tolerances, the state vector is considered wrong and an error is returned (XO_CFI_OSV_CHECK_LOOSE_TOLERANCE_ERR).

7.61.2 Calling Interface

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

```
#include <explorer_orbit.h>
{
  long ierr[XO_NUM_ERR_OSV_CHECK];
  xl_model_id model_id;
  xl_time_id time_id;
  long sat_id, time_ref;
  double time, pos[3], vel[3];

  status = xo_osv_check(&model_id, sat_id, &time_id, time_ref, time,
                        pos, vel, ierr);

}
```

7.61.3 Input Parameters

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

Table 170: Input parameters of xo_osv_check function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
model_id	xl_model_id*		Input model id	-	
sat_id	long		Input satellite id	-	See enum satellite in Table 2
time_id	xl_time_id*		Input time id	-	
time_ref	long		Time reference	-	See enum time reference in Table

					2
time	double		Time	days	
pos	double*		Position of the satellite in Earth Fixed reference frame	meters	
vel	double*		Velocity of the satellite in Earth Fixed reference frame	meters/second	

7.61.4 Output Parameters

The **xo_osv_check** CFI function has the following output parameters:

Table 171: Output parameters of xo_osv_check function

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

7.61.5 Warnings and errors

The next table lists the possible error messages that can be returned by the **xo_osv_check** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_osv_check** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 172: Error messages of xo_osv_check function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Error transforming to TOD frame	No computation performed	XO_CFI_OSV_CHECK_CHANGE_CART_CS_ERR	0
ERR	Orbit radius = 0	No computation performed	XO_CFI_OSV_CHECK_ORBIT_RADIUS_ZERO_ERR	1
ERR	Orbit velocity = 0	No computation performed	XO_CFI_OSV_CHECK_ORBIT_VEL_ZERO_ERR	2
ERR	Orbit semi-major axis undefined	No computation performed	XO_CFI_OSV_CHECK_SEMI_MAJOR_AXIS_ERR	3
WARN	Inclination = 0 or 180 deg	Computation performed	XO_CFI_OSV_CHECK_INCLINATION_WARN	4

ERR	Error getting satellite information	No computation performed	XO_CFI_OSV_CHECK_SAT_ARRAY_ERR	5
ERR	Loose tolerances are not met	No computation performed	XO_CFI_OSV_CHECK_LOOSE_TOLERANCE_ERR	6
WARN	Tight tolerances are not met	Computation performed	XO_CFI_OSV_CHECK_TIGHT_TOLERANCE_WARN	7

7.62 xo_orbit_id_check

7.62.1 Overview

The **xo_orbit_id_check** CFI function computes diagnostics data related to the OSVs contained in orbit id. 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 section7.62.3).

- 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).
- Number and indexes of OSVs whose orbit parameters are beyond loose tolerances for corresponding satellite.
- Number and indexes of OSVs whose orbit parameters are beyond tight tolerances for corresponding satellite.

For DORIS files only EF OSVS are checked, because they are the ones used by orbit initialization.

7.62.2 Calling Interface

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

```
#include <explorer_orbit.h>
{
  long ierr[XO_NUM_ERR_ORBIT_ID_CHECK];
  xo_orbit_id orbit_id;
  xd_orbit_file_diagnostics_settings diag_settings;
```

```
    xo_orbit_id_check_report report;

    status = xo_orbit_id_check(&orbit_id, &diag_settings, &report,
                                                        ierr);
}
```

7.62.3 Input Parameters

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

Table 173: Input parameters of xo_orbit_id_check function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	*xo_orbit_id**		Input orbit id	-	
diag_settings	xd_orbit_file_diagnostics_settings*		Input settings struct (see [D_H_SUM] for details)	-	

7.62.4 Output Parameters

The **xo_orbit_id_check** CFI function has the following output parameters:

Table 174: Output parameters of xo_orbit_id_check function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
report	xo_orbit_id_check_report*		Output data structure for diagnostics results		
ierr	long	.	Error vector	-	-

7.62.5 Warnings and errors

The next table lists the possible error messages that can be returned by the **xo_orbit_id_check** CFI function after translating the returned status vector into the equivalent list of error messages by calling the function of the EO_ORBIT software library **xo_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, mainly on the results vector.

The table is completed by the error code and value. These error codes can be obtained translating the status vector returned by the **xo_orbit_id_check** CFI function by calling the function of the EO_ORBIT software library **xo_get_code** (see [GEN_SUM]).

Table 175: Error messages of xo_orbit_id_check function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Orbit id is not initialised.	No computation performed	XO_CFI_ORBIT_ID_CHECK_ORBIT_ID_UNINITIALIZ	0

			ED_ERR	
ERR	Output report is null.	No computation performed	XO_ORBIT_ID_CHECK_NULL_OUTPUT_ERR	1
ERR	No OSVs found in given orbit id	No computation performed	XO_ORBIT_ID_CHECK_NO_OSVS_ERR	2
ERR	Couldn't retrieve time id from orbit id	No computation performed	XO_ORBIT_ID_CHECK_NO_TIME_ID_ERR	3
ERR	Error in xd_orbit_file_diagnostics()	No computation performed	XO_ORBIT_ID_CHECK_FILE_DIAGNOSTICS_ERR	4
ERR	Error in xo_osv_check()	No computation performed	XO_ORBIT_ID_CHECK_OSV_CHECK_ERR	5
ERR	Memory allocation error	No computation performed	XO_ORBIT_ID_CHECK_MEM_ALLOC_ERR	6

7.63 xo_orbit_get_mode

7.63.1 Overview

The **xo_orbit_get_mode** CFI function returns the orbit model with which **xo_orbit_id** has been initialized.

7.63.2 Calling interface

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

```
#include <explorer_orbit.h>
{
        xo_orbit_id orbit_id;
        long orbit_mode;
        orbit_mode = xo_orbit_get_mode(&orbit_id);
}
```

7.63.3 Input parameters

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

Table 37: Input parameters of xo_orbit_get_mode function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_id	xo_orbit_id*	-	Structure for orbit initiaization	-	-

7.63.4 Output parameters

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

Table 38: Output parameters of xo_orbit_get_mode function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xo_orbit_get_mode	long	-	Orbit initialization mode. The value corresponds to the ones in enumeration XO_Orbit_init_mode	-	-

7.63.5 Warnings and errors

This function does not return any error/warning.

8 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 10.04)	16 GB
MACIN64	Intel Core i7 4 cores @2,6 GHz	MACOSX 10.12	16 GB
WINDOWS64	Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz 3.19 GHz	Microsoft Windows 10 (or superior)	32 GB
WINDOWS7_64	Intel(R) Xeon(R)CPU ES-2630 @ 2.40GHz 2.40GHz	Microsoft Windows 7	16 GB

The table below shows the time (in milliseconds - ms) each function takes to be run under each platform:

Function ID	WINDOWS7_64	WINDOWS64	LINUX64	LINUX64_LEGACY	MACIN64
xo_orbit_init_def	0.271500	0.255000	0.197000	0.138000	0.090000
xo_osv_compute_extra	0.080900	0.066500	0.050000	0.056000	0.031000
xo_osv_compute * (PROPAGATION)	0.038800	0.047800	0.015000	0.015000	0.010000
xo_orbit_cart_init	0.091300	0.097300	0.039000	0.035000	0.025000
xo_orbit_init_file * 3 OSVs read.	0.699800	0.838300	0.377000	0.287000	0.292000
xo_osv_compute * (INTERPOLATION)	0.017600	0.020700	0.007000	0.007000	0.005000
xo_orbit_to_time	0.052177	0.032023	0.071690	0.038540	0.022140
xo_time_to_orbit	0.154500	0.095000	0.214000	0.115000	0.065000
xo_orbit_rel_from_abs	0.000043	0.000025	0.000040	0.000020	0.000020
xo_orbit_info	0.055200	0.035300	0.075000	0.039000	0.023000
xo_orbit_abs_from_rel	0.000047	0.000026	0.000040	0.000030	0.000020
xo_orbit_abs_from_phase	0.000032	0.000018	0.000020	0.000020	0.000010
xo_orbit_init_status	0.000005	0.000003	0.000000	0.000000	0.000010
xo_orbit_get_sat_id	0.000004	0.000003	0.000004	0.000003	0.000003

xo_orbit_get_mode	0.000004	0.000003	0.000000	0.000010	0.000010
xo_orbit_get_osf_rec	0.000693	0.000435	0.000280	0.000220	0.000200
xo_orbit_set_osf_rec	0.000665	0.000463	0.000530	0.000340	0.000400
xo_orbit_get_val_time	0.000009	0.000004	0.000010	0.000010	0.000010
xo_orbit_set_val_time	0.000010	0.000006	0.000010	0.000000	0.000000
xo_orbit_get_propag_config	0.000071	0.000042	0.000030	0.000020	0.000020
XO_Interpol_get_id_data	0.000043	0.000026	0.000020	0.000020	0.000020
xo_gen_osf_create_2	1.350000	2.280000	0.900000	0.500000	0.500000
xo_gen_osf_append_orbit_change	1.860000	2.810000	1.300000	0.800000	0.900000
xo_gen_osf_change_repeat_cycle	2.460000	3.160000	1.600000	1.800000	1.200000
xo_gen_osf_add_drift_cycle	2.800000	3.970000	2.400000	3.600000	1.400000
xo_gen_rof	32.290001	42.889999	32.299999	25.700001	30.000000
xo_gen_pof * 16 OSVs generated	3.730000	5.080000	3.700000	2.300000	2.500000
xo_gen_dnf	147.789993	126.230003	234.399994	182.899994	214.199997
xo_gen_oef * 16 OSVs generated	3.920000	6.500000	3.400000	4.100000	2.800000
xo_check_osf	1.046000	1.055600	0.789000	0.553000	0.487000
xo_check_oef	4.515500	4.400200	3.594000	2.511000	2.225000
xo_gen_tle	3.370000	4.020000	1.700000	1.500000	1.200000
xo_orbit_init_def_2	0.270000	0.250000	0.200000	0.000000	0.100000
xo_orbit_init_file_precise * Includes xo_orbit_close	258.549988	173.050003	165.500000	107.500000	100.000000
xo_osv_compute * (xo_orbit_init_file_precise)	8.750000	7.350000	5.500000	5.000000	3.500000
xo_orbit_cart_init_precise * Includes xo_orbit_close	256.619995	170.910004	164.199997	107.500000	98.900002
xo_osv_compute * (xo_orbit_cart_init_precise)	8.700000	7.250000	5.500000	5.500000	3.000000
xo_orbit_set_osv	0.117043	0.124699	0.049153	0.041973	0.030760
xo_orbit_set_precise_propag_config	0.000535	0.000197	0.000840	0.000470	0.000630
xo_osv_to_tle	0.227000	0.226300	0.119000	0.106000	0.089000
xo_orbit_init_geo	0.008600	0.010700	0.003000	0.002000	0.003000
xo_position_on_orbit_to_time	0.206800	0.219900	0.082000	0.080000	0.057000
xo_orbit_data_filter * Processing 1600 OSV	0.000400	0.000800	0.001000	0.000000	0.000000
xo_orbit_data_filter	0.100000	0.100000	0.000000	0.000000	0.100000

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”.

9 LIBRARY PRECAUTIONS

The following precautions shall be taken into account when using EO_ORBIT software library:

- When a message like

<LIBRARY NAME> >>> ERROR in *xo_function*: Internal computation error # *n*

or

<LIBRARY NAME> >>> WARNING in *xo_function*: Internal computation warning # *n*

appears, run the program in *verbose* mode for a complete description of warnings and errors, and call for maintenance if necessary.