Earth Observation Mission CFI Software

EO_POINTING SOFTWARE USER MANUAL

Code:	EO-MA-DMS-GS-0005
Issue:	4.21
Date:	23/06/2021

	Name	Function	Signature
Prepared by:	EOCFI Team	Project Engineers	
Checked by:	Inês Estrela	Project Manager	
Approved by:	Antonio Gutierrez	Division Head	

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/01/026-024
Electronic file name:	eo-ma-dms-gs-005-21

DOCUMENT STATUS LOG

Issue	Change Description	Date	Approval
0.1	First draft version	23/05/02	
1.0	First release	19/07/02	
2.0	Second release	29/11/02	
2.1	Maintenance release	13/05/03	
2.2	Added the following functions: • xp_target_extra_aux • xp_target_extra_target_to_sun • xp_target_extra_ef_to_sat • xp_converterAdded the following functions: • xp_target_extra_aux • xp_target_extra_target_to_sun • xp_target_extra_ef_to_sat • xp_converterAdded the following functions: • xp_target_extra_aux • xp_target_extra_target_to_sun • xp_target_extra_ef_to_sat • xp_converter	30/09/03	
3.0	Completely new initialization strategy and attitude functions.	21/07/04	
3.1	New attitude models implemented. New multitarget functions.New attitude models implemented. New multitarget functions.New attitude models implemented. New multitarget functions.	13/10/04	
3.2	DEM Model implementation.DEM Model implementation.DEM Model implementation.	15/11/04	
3.3	New features: • xp_target_travel_time • New atttitude models (Cryosat YSM, ADM model) • New attitude files for initialization • Idenfier accessorsNew features:	11/07/05	

	• xp_target_travel_time • New atttitude models (Cryosat YSM, ADM model) • New attitude files for initialization • Idenfier accessorsNew features: • xp_target_travel_time • New atttitude models (Cryosat YSM, ADM model) • New attitude files for initialization • Identifier accessors		
3.4	New features: • New attitude model for ADM • xp_dem_compute • xp_target_reflected and xp_target_extra_specular_reflection (only interface definition) • xp_target_extra_target_to_moon • New axis for the generic attitude models: – XP_SC_EF_VEL_VEC – XP_ORBIT_POLENew features: • New attitude model for ADM • xp_dem_compute • xp_target_reflected and xp_target_extra_specular_reflection (only interface definition) • xp_target_extra_target_to_moon • New axis for the generic attitude models: – XP_SC_EF_VEL_VEC – XP_ORBIT_POLENew features: • New attitude model for ADM • xp_dem_compute • xp_target_reflected and xp_target_extra_specular_reflection (only interface definition) • xp_target_extra_target_to_moon • New axis for the generic attitude models: – XP_SC_EF_VEL_VEC – XP_ORBIT_POLE	18/11/05	
3.5	Maintenance release	26/05/06	

	New features: • Aberration correction for Cryosat attitude based on star-trackersMaintenance release New features: • Aberration correction for Cryosat attitude based on star-trackersMaintenance release New features: • Aberration correction for Cryosat attitude based on star-trackers		
3.6	Maintenance release New features: • New attitude models for SENTINEL 1A and 1B • New Axis defined: XP_INERTIAL_POS_VEC_CORRECTED and XP_INERTIAL_VEL_VEC_ROTATEDMaintenance release New features: • New attitude models for SENTINEL 1A and 1B • New Axis defined: XP_INERTIAL_POS_VEC_CORRECTED and XP_INERTIAL_VEL_VEC_ROTATEDMaintenance release New features: • New attitude models for SENTINEL 1A and 1B • New Axis defined: XP_INERTIAL_POS_VEC_CORRECTED and XP_INERTIAL_VEL_VEC_ROTATED	24/11/06	
3.7	Maintenance release New features: • Function expcfi_check_libs • Library version for MAC OS X on Intel (32 and 64-bits)Maintenance release New features: • Function expcfi_check_libs • Library version for MAC OS X on Intel (32 and 64-bits)Maintenance release New features:	13/07/07	

	• Function expcfi_check_libs • Library version for MAC OS X on Intel (32 and 64-bits)		
3.7.2	Maintenance release New features: • Support for missalignment for attitude frame • Azimuth and Elevation definition for attitude frames • Improvement in the quaternion interpolationMaintenance release New features: • Support for missalignment for attitude frame • Azimuth and Elevation definition for attitude frames • Improvement in the quaternion interpolationMaintenance release New features: • Support for missalignment for attitude frame • Azimuth and Elevation definition for attitude frames • Improvement in the quaternion interpolation	13/07/07	
4.0	Maintenance releaseMaintenance releaseMaintenance release	19/01/09	
4.1	Maintenance release. New features: • Pointing functions support DEM GETASSEv2 • Sentinel-1 attitude model (roll steering) • Instrument offsets for attitude computations	07/05/10	
4.2	Maintenance release. New features: • Support to DEM ACE2 9SEC		
4.3	Maintenance release. New features: • Raytracing model in target fucntions determined by input atmos_id • New attitude model for SENTINEL2 (XP_MODEL_SENTINEL2)		

4.4	Maintenance release. New features: • Support for GEO satellites: - New Yaw flip attitude - New function xp_target_sc • Option to use a memory cache for DEM computations. New function to configure cache • New function xp_target_list_inter		
4.5	Maintenance release New features: • New functions: - xp_target_list_extra_vector - xp_target_list_extra_main - xp_target_list_extra_aux - xp_target_list_extra_ef_target - xp_target_list_extra_specular_reflection - xp_target_list_extra_target_to_moon - xp_target_list_extra_target_to_sun • New DEM algorithm of maximum heights		
4.6	Maintenance release New features: • New function xp_attitude_define. • Internal improvements for runtime performance in DEM computations.		
4.7	Maintenance release	03/28/14	
4.8	Maintenance release New features: • Added support for Earth Fixed input in initialization of satellite nominal attitude with harmonics.	29/10/2014	

4.9	Maintenance release	23/04/2015	
4.10	Maintenance release New features: • Support for DEM ACE2 30 secs • Target functions: possibility of considering light travel time in target computation • Run-time improvements in target functions	29/10/2015	
4.11	Maintenance release New features: • Support for DEM ACE2 3 secs • New functions xp_gen_attiude_data and xp_gen_attitude_file	15/04/2016	
4.12	Maintenance release	03/11/2016	
4.13	Maintenance release	05/04/2017	
4.14	Maintenance release New features: • Support for MetOp-SG attitude law • New function xp_free_target_id_data	16/11/2017	
4.15	Maintenance release	20/04/2018	

4.16	Maintenance release New features: • Support for DEM ACE2 5 minutes	09/11/2018	
4.17	Maintenance release New features: • xp_dem_get_cell_value • xp_dem_get_cell_geod	10/05/2019	
4.18	Maintenance release New features: • Support for TanDEM-X 90 m DEM	08/11/2019	
4.19	Maintenance release New features: • Support Reference_Frame tag in Attitude Angles files • Support for GDEM V3	29/05/2019	
4.20	Maintenance release	30/11/2020	
4.21	Maintenance release	23/06/2021	

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 SCOPE

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

2 ACRONYMS, NOMENCLATURE AND TERMINOLOGY

2.1 Acronyms

ANX	Ascending Node Crossing
AOCS	Attitude and Orbit Control Subsystem
ASCII	American Standard Code for Information Interchange
CFI	Customer Furnished Item
CS	Coordinate System
DRS	Data Relay Satellite
ESA	European Space Agency
ESTEC	European Space Technology and Research Centre
GPL	GNU Public Library
GPS	Global Positioning System
GS	Ground Station
H/W	Hardware
IERS	International Earth Rotation Service
I/F	Interface
LOS	Line Of Sight
LUT	Look-Up Table
OBT	On-board Binary Time
OSF	Orbit Scenario File
RAM	Random Access Memory
SBT	Satellite Binary Time
SRAR	Satellite Relative Actual Reference
SSP	Sub Satellite Point
SUM	Software User Manual
S/W	Software
TAI	International Atomic Time
UTC	Coordinated Universal Time
UT1	Universal Time UT1
WGS[84]	World Geodetic System 1984

2.2 Nomenclature

CFI A group of CFI functions, and related software and documentation that will be distributed by ESA to the users as an independent unit

CFI function A single function within a CFI that can be called by the user

Library A software library containing all the CFI functions included within a CFI plus the supporting functions used by those CFI functions (transparently to the user)

2.3 Note on Terminology

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

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

3 APPLICABLE AND REFERENCE DOCUMENTS

3.1 Applicable Documents

No applicable documents.

3.2 Reference Documents

[MCD]	Earth Observation Mission CFI Software. Conventions Document. EO-MA-DMS-GS-0001.
[MSC]	Earth Observation Mission CFI Software. Mission Specific Customizations. 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.
[LOS_ALG]	LOS Intersection. PE-TN-ESA-SY-0043

The latest applicable version of [MCD], [GEN_SUM], [F_H_SUM], [D_H_SUM], [LIB_SUM] is v4.20 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 the CFI functions required to perform accurate computation of pointing parameters from and to a satellite for various types of targets.

It includes a set of functions to initialize the attitude of the platform and the instruments. The values provided by these functions are later used by all the other functions of the library.

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, coordinate systems, 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 usage and the error handling functions.

4.1.1 Attitude Data Flow

The following figure shows the typical data flow for the attitude functions. First, the different transformations between the various reference frames are initialised. Then, given the spacecraft position, the attitude is calculated:

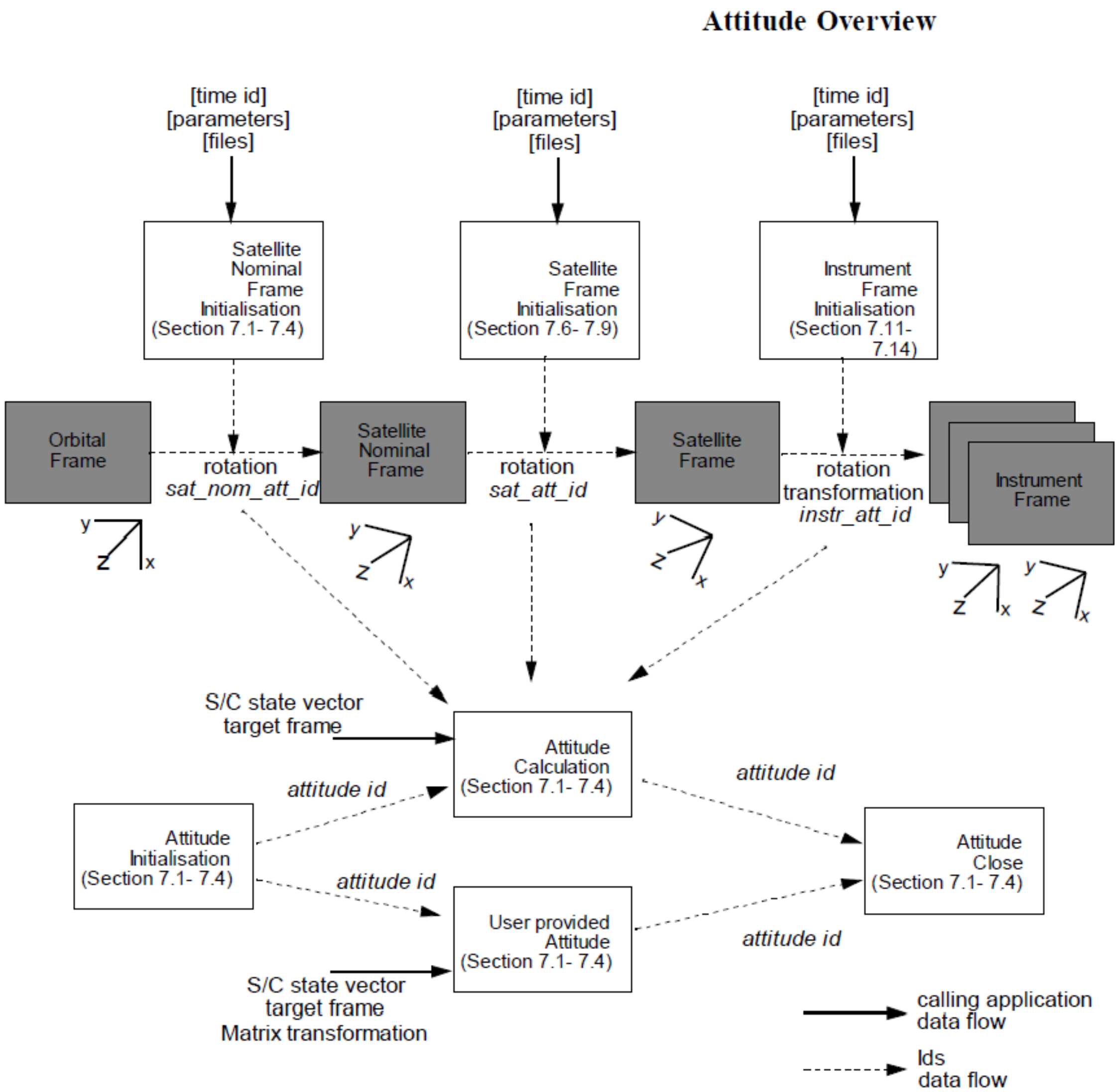


Figure 1: Attitude Initialization Overview

Each different transformation can be initialised with different models (note that all the attitudes can be initialized at the same time using the function xp_atttitude_define (see section 7.49) and an Attitude definition file):

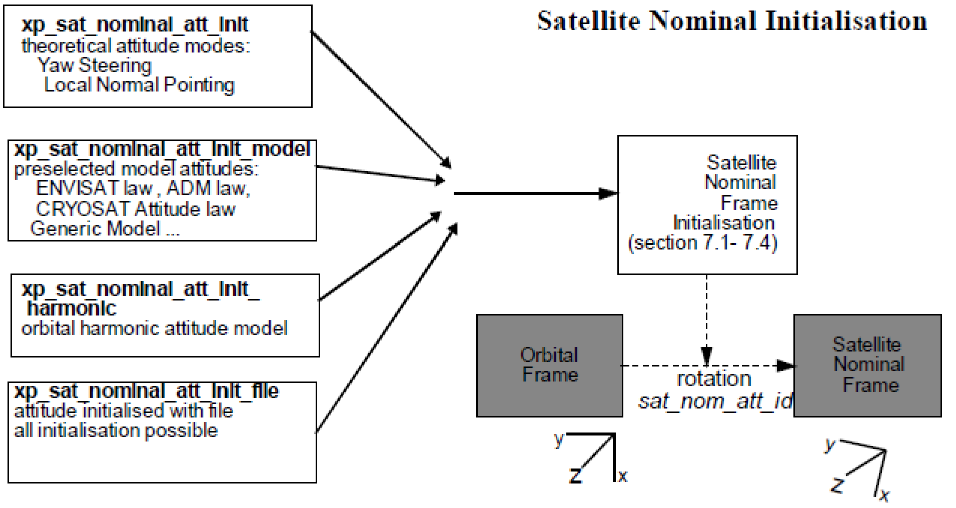


Figure 2: Satellite Nominal Initialization

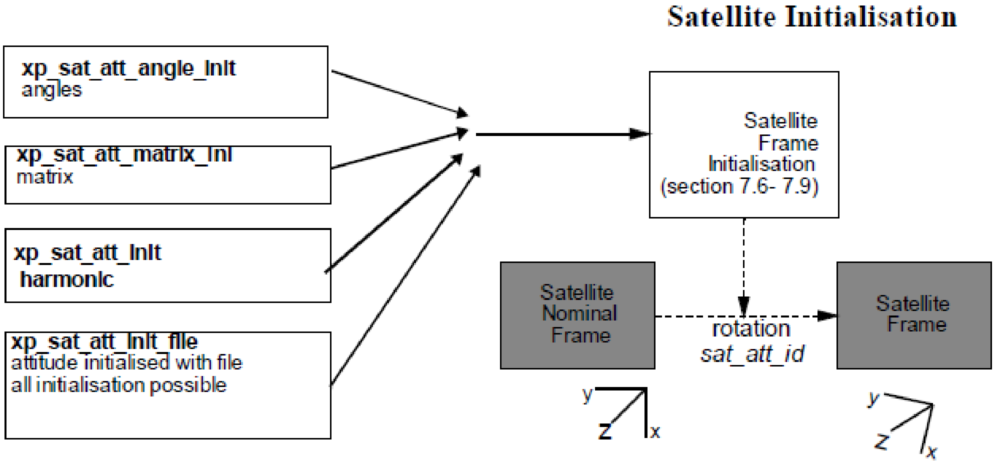


Figure 3: Satellite Initialization

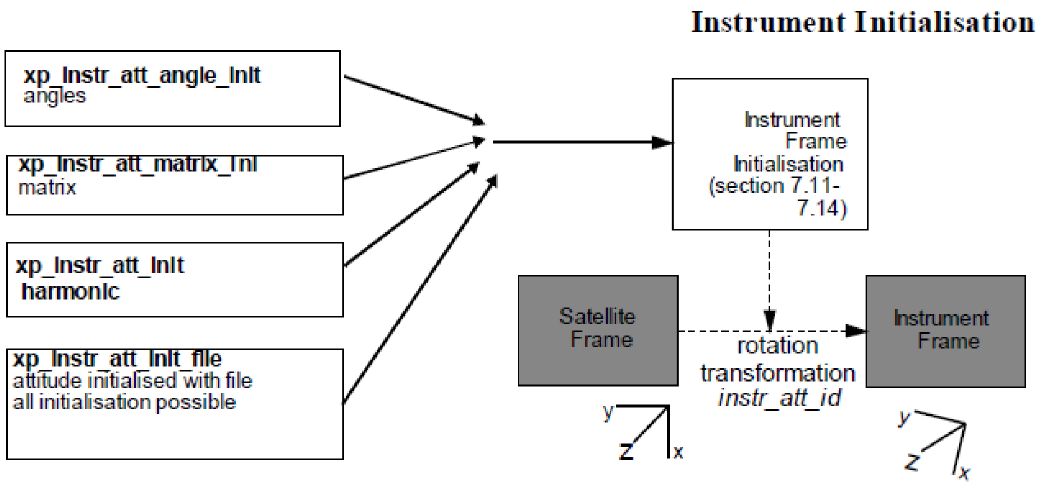


Figure 4: Instrument Initialization

4.1.2 Geolocation Routines Data Flow

The following figure shows the tipical data flow for the geolocation routines functions. First, the attitude should be calculated, and, if needed, the refraction and Digital Elevation Models initialised.

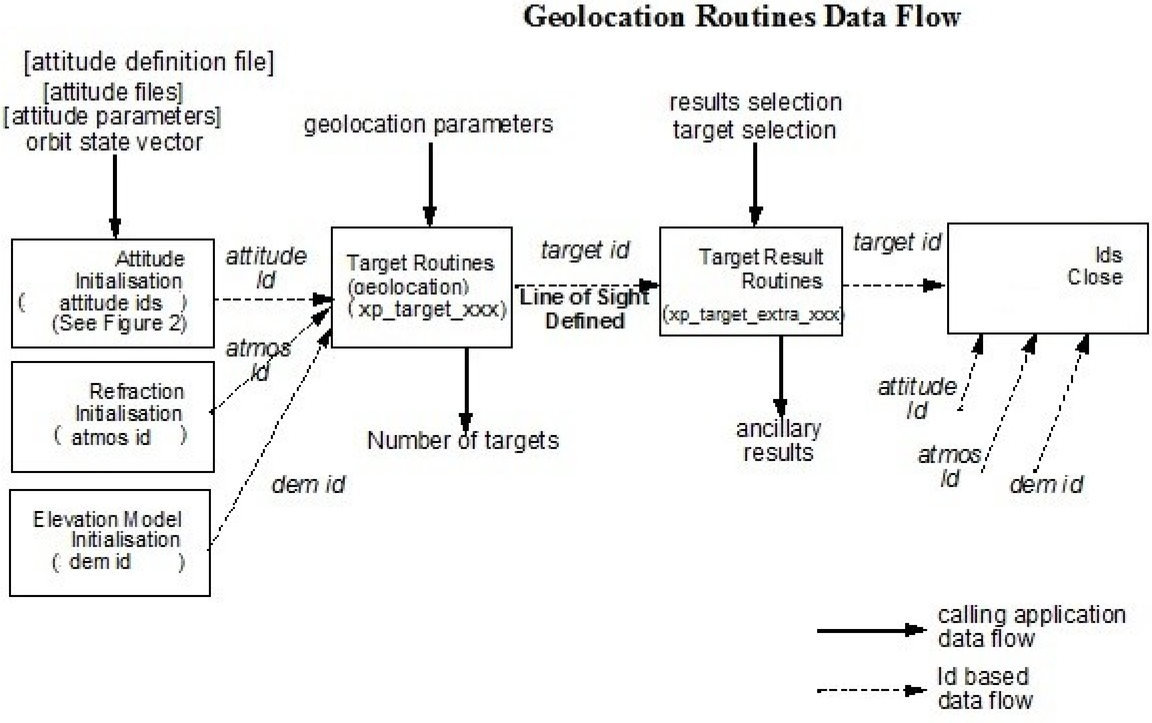
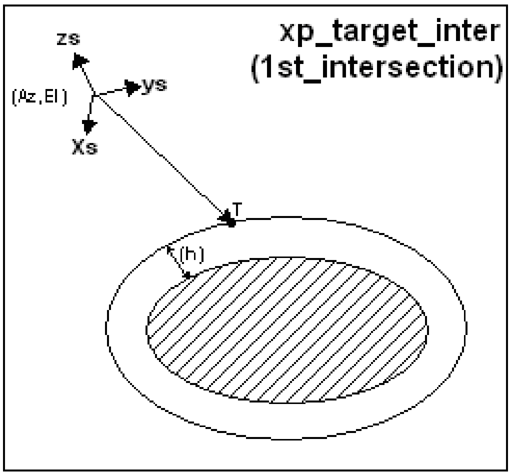


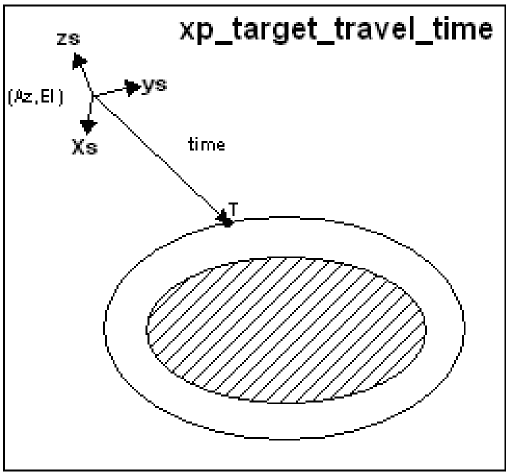
Figure 5: Geolocation Routines Calling Sequence

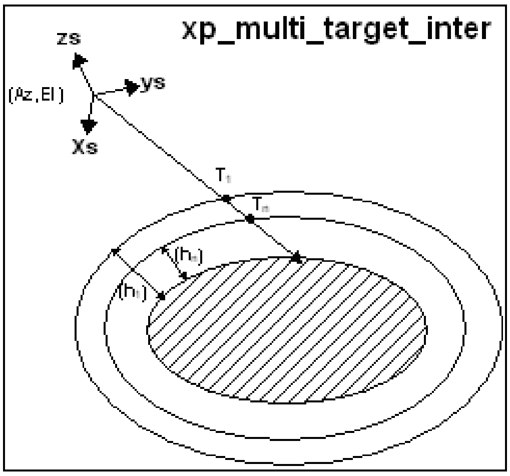
The table below and the diagrams on the next pages describe the various **xp_target_<function>.**

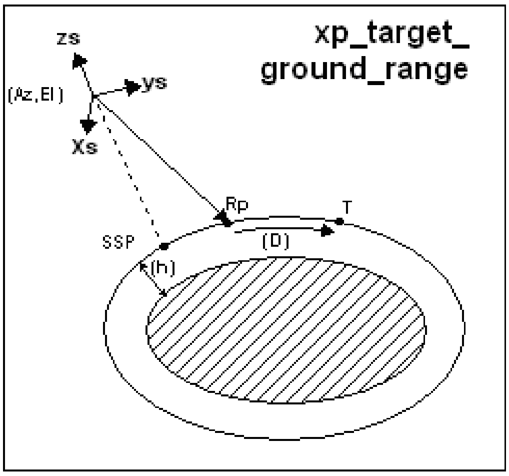
Table 1: xp_target functions

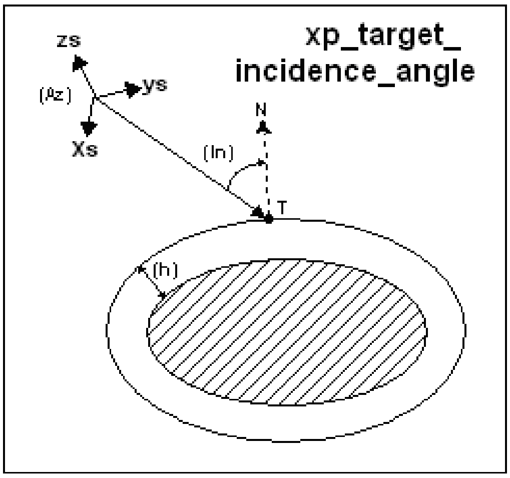
xp_target_ <function>	Description
xp_(multi)_target_ inter xp_target_list_inter	It calculates the intersection point(s) of the line of sight defined by an elevation and an azimuth angle (or a set of them) expressed in the input Attitude frame, with a surface(s) located at a certain geodetic altitude(s) over the Earth.
xp_(multi)_target_ travel_time	It calculates the point of the line or sight from the satellite (defined by an elevation and an azimuth angle expressed in the selected Attitude Frame) at a given travel time(s) along the (curved) line of sight.
xp_target_ground _range	It calculates the location of a point that is placed on a surface at a certain geodetic altitude over the Earth, that lays on the plane defined by theS/C position, the nadir and a reference point, and that is at a certain distance or ground range measured along that surface from that reference point. This reference point is calculated being the intersection of the previous surface with the line of sight defined by an elevation and azimuth angle in the input Attitude coordinate system.
xp_target_inciden ce_angle	It calculates the location of a point that is placed on a surface at a certain geodetic altitude over the Earth and that is seen from the S/C on a line of sight that forms a certain azimuth angle in the input Attitude frame and that intersects that surface with a certain incidence angle.
xp_target_range	It calculates the location of a point that is placed on a surface at a certain geodetic altitude over the Earth, that is seen from the S/C on a line of sight that forms a certain azimuth angle in the input Attitude frame, and that is at a certain range or slant-range from the S/C.
xp_target_range_r ate	It calculates the location of a point that is placed on a surface at a certain geodetic altitude over the Earth, that is at a certain range from S/C, and whose associated Earth-fixed target has a certain range-rate value.
xp_target_tangent	It calculates the location of the tangent point over the Earth that is located on the line of sight defined by an elevation and azimuth angles expressed in the input Attitude frame.
xp_target_altitude	It calculates the location of the tangent point over the Earth that is located on a surface at a certain geodetic altitude over the Earth and that is on a line of sight that forms a certain azimuth angle in the input Attitude frame.
xp_target_star	It calculates the location of the tangent point over the Earth that is located on the line of sight that points to a star defined by its right ascension and declination coordinates.
xp_target_generic	The cartesian state vector of the target is taken as an input.
xp_target_tangent _sun	It calculates the location of the tangent point over the Earth that is located on the line of sight that points to the Sun
xp_target_tangent _moon	It calculates the location of the tangent point over the Earth that is located on the line of sight that points to the Moon
xp_target_station	It calculates the most relevant observation parameters of the link between the satellite and a ground station
xp_target_sc	It calculates the most relevant observation parameters of the link between one satellite and another Satellite.

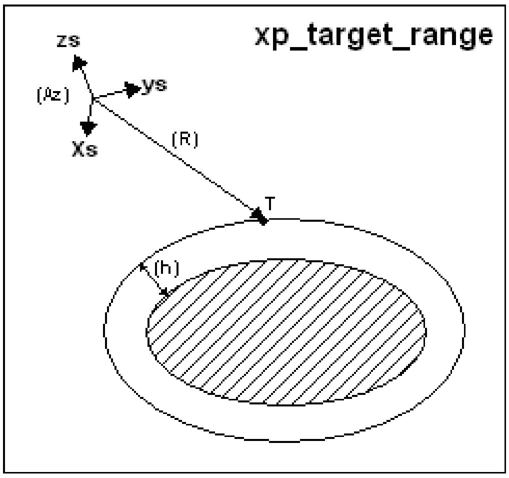

xp_target_inter
(1st_intersection)
zs
ys
Xs
(Az,El)
T
(h)

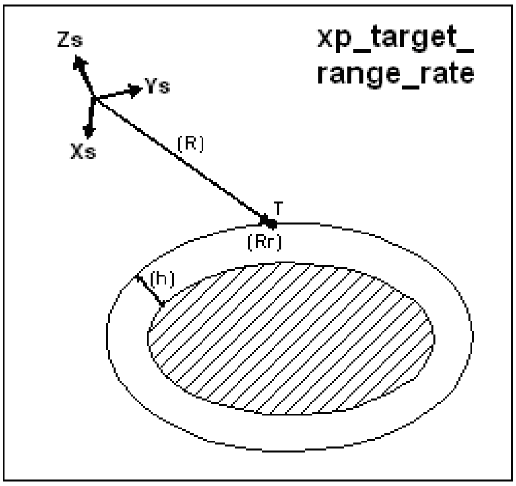

xp_target_travel_time
zs
ys
Xs
(Az,El)
time
T

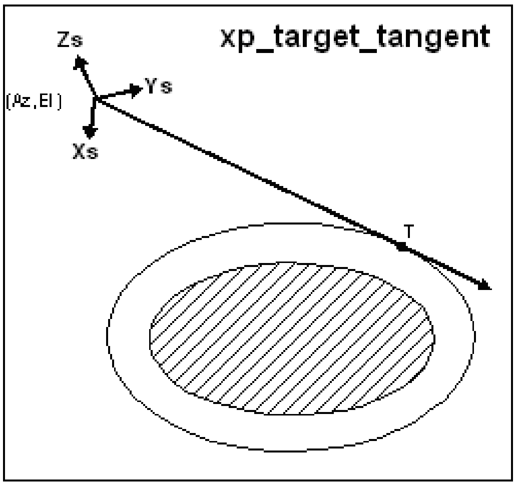

xp_multi_target_inter
zs
ys
Xs
(Az,El)

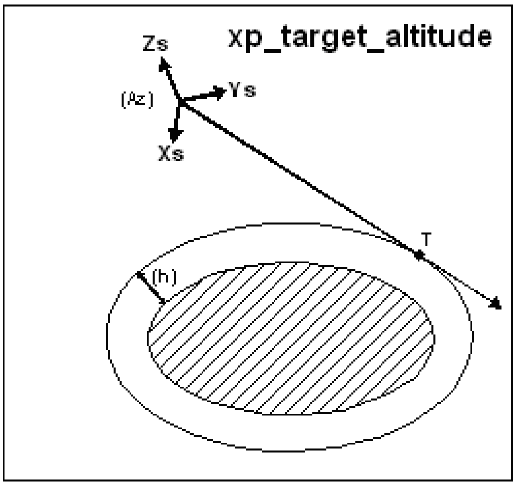

xp_target_
ground_range
zs
ys
Xs
(Az,El)
Rp
T
SSP
(D)
(h)

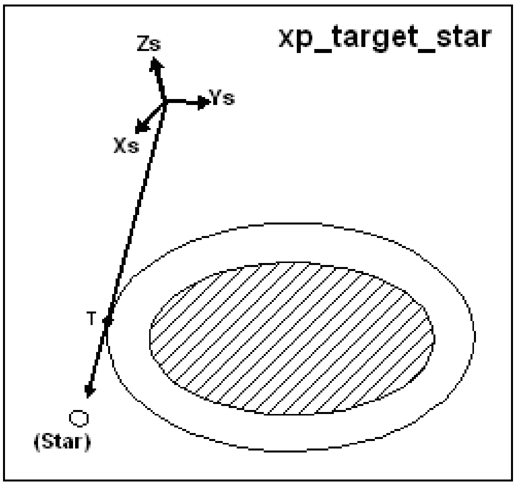

xp_target_
incidence_angle
zs
ys
Xs
(Az)
N
(In)
T
(h)

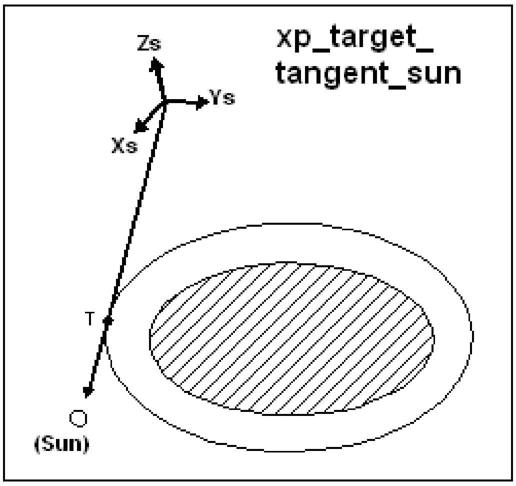

xp_target_range
zs
ys
Xs
(Az)
(R)
T
(h)

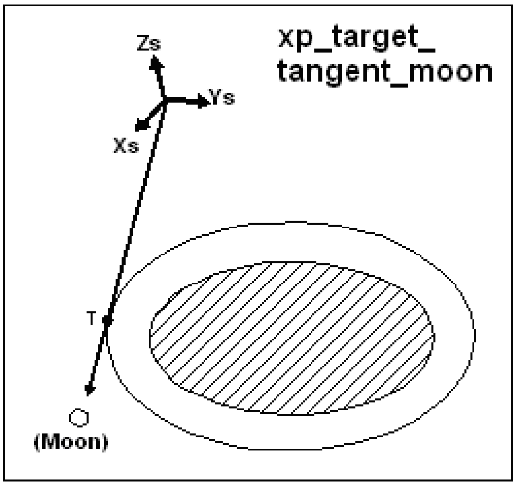

xp_target_
range_rate
Zs
Ys
Xs
(R)
T
(Rr)
(h)

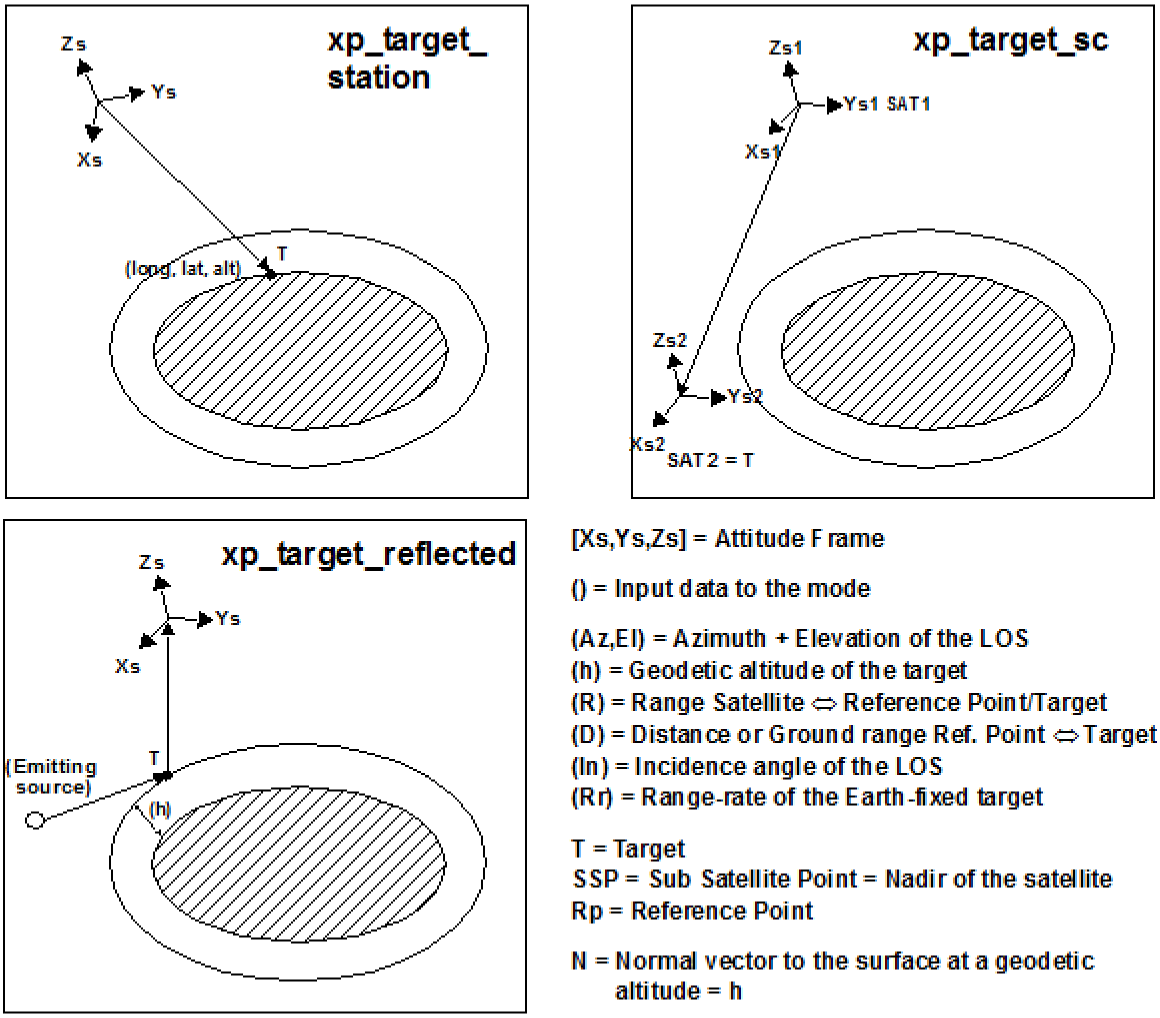

xp_target_tangent
Zs
Ys
(Az,El)
Xs
T


xp_target_altitude
Zs
(Az)
Ys
Xs
T
(h)


xp_target_star
Zs
Ys
Xs
T
(Star)


xp_target_
tangent_sun
Zs
Ys
Xs
T
(Sun)


xp_target_
tangent_moon
Zs
Ys
Xs
T
(Moon)



As it can be seen from the list of functions, there are some functions that calculate several targets (xp_multi_target_xxxx, xp_target_list_inter). The number of targets found by the functions is returned through the interface.

In addition to these “user” targets, two other categories of targets can be defined, “LOS” targets and “DEM” targets.

4.1.2.1 LOS targets

The idea is to get information about all the ray path points computed by a specific target routine along the Line of Sight (LOS) trajectory.

For every target routine, the output parameter num_los_target will return the number of points in the path.

It applies when the variable "target_type" is equal to XP_LOS_TARGET_TYPE.

1. Start point of LOS

The spacecraft position (Instrument CS) shall be considered as the start point for the LOS path.

2. Stop point of LOS

The stop point for the LOS path will be different depending on the selected target function; nominally it will be the resulting target point.

- xp_target_inter, xp_target_list_inter and xp_multi_target_inter: 1st or 2nd intersection point (Point corresponding to the last altitude for the multi_target routine)
- xp_target_ground_range: Target point
- xp_target_incidence_angle: Target point
- xp_target_range: Target point
- xp_target_range_rate: Target point
- xp_target_tangent: Two different cases to consider depending on whether refraction is selected or not:
  - No refraction mode: Tangent point
  - Refraction mode:
    - The 2nd intersection point with a surface located at Refraction Model Maximum Height (geodetic altitude) over the Earth if tangent height <= Refraction Model Maximum Height
    - The tangent point if tangent height > Refraction Model Maximum Height
- xp_target_altitude: Point at selected altitude
- xp_target_star: Two different cases to consider depending on whether refraction is selected or not:
  - No refraction mode: Tangent point
  - Refraction mode:
    - The 2nd intersection point with a surface located at Refraction Model Maximum Height (geodetic altitude) over the Earth if tangent height <= Refraction Model Maximum Height
    - The tangent point if tangent height > Refraction Model Maximum Height
- xp_target_station: Ground Station position
- xp_target_generic: Target position
- xp_target_reflected: Reflection point
- xp_target_travel_time and xp_multi_target_travel_time: Point at selected travel time (Point corresponding to the last travel time for the multi_target routine)
- xp_target_tangent_sun: Tangent point
- xp_target_tangent_moon: Tangent point
- xp_target_sc: Target position.

4.1.2.2 DEM targets

A DEM Target is defined as the intersection of a line of sight with the Earth Surface defined using a digital elevation model (DEM).

A DEM Target is calculated using as line of sight the LOS targets that has been computed previously with a target routine (Note that such LOS consist in a polygonal line, no necessarily a straight line). Consequently, to get a DEM target it is necessary to follow these steps:

- Initialize the DEM model using the xp_dem_init routine and a configuration file (Section 7.60).
- One call to the target routine for getting the LOS targets.

- One call to the target extra routine requesting the DEM target.

The digital elevation model of the Earth consists in a set of points defining a grid for which a measure of the altitude over the Earth reference ellipsoid is given. The altitude of the points within each cell of the grid is computed by the CFI using a bilinear interpolation with the points of the corner of the cell. Details about the bilinear algorithm used to compute the intersection can be seen in [LOS_ALG].

4.1.2.3 Light propagation model

When the light propagation model is enabled, the target functions keep into account the time spent by a generic signal traveling at the speed of light to:

- in the TRANSMITTER mode: go from the satellite to the target;
- in the RECEIVER mode: go from the target to the satellite.

Two distinct times are considered:

1) The satellite time (T) is the time provided as input to the target function. It is:

- in the TRANSMITTER mode: the time at which the satellite (instrument) emits the signal towards the target;
- in the RECEIVER mode: the time at which the satellite (instrument) receives the signal emitted by the target.

2) The target time is the satellite time T plus or minus the light travel time between satellite and target (dT). It is:

- in the TRANSMITTER mode: T+dT, i.e. the target receives the signal sent by the satellite with a delay dT;
- in the RECEIVER mode: T-dT, i.e. the satellite receives the signal emitted by the target with a delay dT.

dT is calculated as the light travel time from the satellite to target calculated with dT=0. When the light propagation model is not activated, it is assumed dT=0, therefore target and satellite are considered at the same time T.

According to the definitions above, the Line of Sight (LOS) can be defined as the segment joining satellite and target at their correspondent times.

For the following functions the calculation method is slightly different:

- xp_target_range: the input range is used to calculate the light travel time;
- xp_target_travel_time: the input travel time is used as dT;
- xp_target_generic, xp_target_sc: in this case the input is the target at time T+/-dT. The function estimates the target at T to compute the line of sight parameters also considering (if provided as inputs) velocity and acceleration of the target.

Target geometric properties (returned by the extra functions) are evaluated considering the two distinct times, for example:

- The target position (i.e. position in EF co-ordinates and geodetic co-ordinates) is evaluated at time T-dT;

- Direction from satellite to target and viceversa are evaluated considering the satellite position at time T and target position at time T+/-dT;
- Direction from target to e.g. sun/moon take into account the target position at T+/-dT and other celestial bodies at the same time.

Following figure 6. shows an example using the xp_target_inter function.

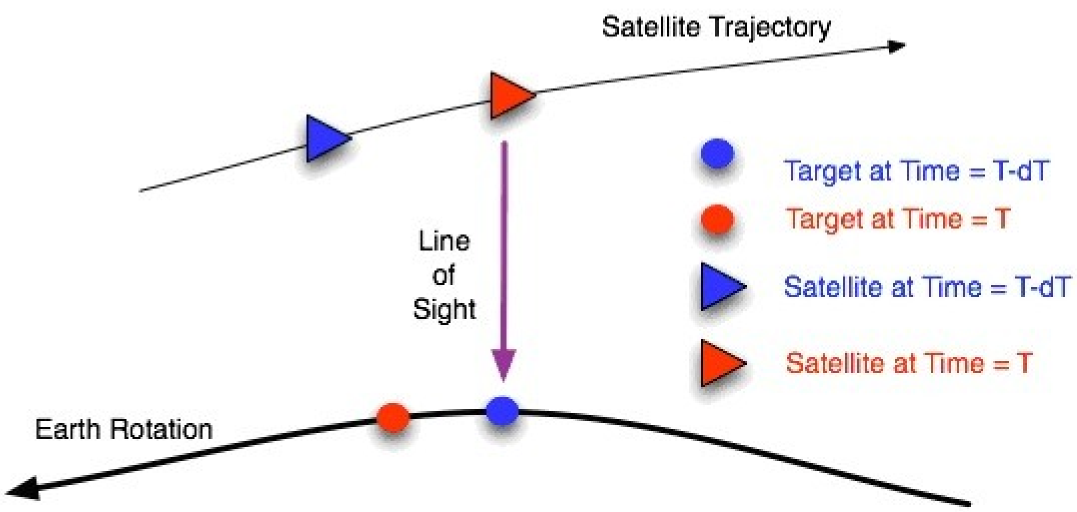


Figure 6: Example of light travel time with xp_target_inter

The light propagation mode is set to RECEIVER and input azimuth,elevation are **0, 90 deg** (assuming a local normal pointing). The signal is emitted by the target at time T-dT (blue point, let's assume at geodetic co-ordinates **(lon,lat,h)** and EF co-ordinates **(X,Y,Z)**) and is received by the satellite at time T (red triangle). Due to Earth rotation, at time T the observed target has moved to the red point.

Here are some examples of results from xp_target_extra… functions:

- xp_target_extra_vector:

  - Target position: the vector **(X,Y,Z)**, i.e. the target point considered at T-dT;

  - Direction LOS: the vector corresponding to the purple line in Fig. X (it is the line joining satellite position at time T and target position at time T-dT);
- xp_target_extra_main:

  - Target geodetic co-ordinates: **(lon,lat,h)**

  - Satellite to target azimuth, elevation: **0,90**, i.e. the same azimuth and elevation used as input for xp_target_inter.

  - Target to satellite azimuth, elevation: **0,90**, this is the view direction from the target at time T-dT to the satellite at time T.

The same results are given by the xp_target_extra… functions if the xp_target_generic is called with input target at EF co-ordinates **(X,Y,Z)** and velocity set to zero.

To activate the light propagation mode the *model_id* structure must be initialized using the function xl_model_init as follow:

1) for TRANSMITTER mode:

```
#include <explorer_lib.h>
{
  long mode, models[XL_NUM_MODEL_TYPES_ENUM];
  xl_model_id model_id = {NULL};
  long ierr[XL_NUM_ERR_MODEL_INIT], status;

  mode = XL_MODEL_CONFIG;
  ...
  models[XL_MODEL_TYPE_LIGHT_PROPAGATION] = XL_MODEL_LIGHT_PROPAGATION_TRANSMITTER;

  status =  xl_model_init (&mode, models,
                           &model_id,
                           ierr)
}
```

2) for RECEIVER mode:

```
#include <explorer_lib.h>
{
  long mode, models[XL_NUM_MODEL_TYPES_ENUM];
  xl_model_id model_id = {NULL};
  long ierr[XL_NUM_ERR_MODEL_INIT], status;

  mode = XL_MODEL_CONFIG;
  ...
  models[XL_MODEL_TYPE_LIGHT_PROPAGATION] = XL_MODEL_LIGHT_PROPAGATION_RECEIVER;

  status =  xl_model_init (&mode, models,
                           &model_id,
```

```
                               ierr)
}
```

5 LIBRARY INSTALLATION

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

6 LIBRARY USAGE

The EO_POINTING 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).

In order to improve run-time performance, some functions (e.g. xp_target_list_extra_vector, xp_target_list_extra_main, xp_target_list_extra_aux, xp_target_list_extra_ef_target, xp_target_list_extra_target_to_sun, xp_target_list_extra_target_to_moon, xp_target_list_extra_specular_reflection) perform their computations in multi-threading mode.

The multi-threading code of the Pointing functions uses the OpenMP API (see http://en.wikipedia.org/wiki/OpenMP).

OpenMP is not supported in the clang compiler, therefore such functions work in single-thread mode in MacOS.

The following is required to compile and link a Software application that uses the EO_POINTING 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_pointing.h (for a C application)

2) use the following compile and link options:

   Linux platforms:

   -I*cfi_include_dir* -L*cfi_lib_dir* -lexplorer_pointing

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

   MacOS platforms (openmp is not supported):

   -I*cfi_include_dir* -L*cfi_lib_dir* -lexplorer_pointing

   -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_pointing.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 /openmp

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

To avoid problems in linking a user application with the `EO_POINTING` 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 `XP_` or `xp_`.

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

Table 2: CFI functions included within EO_POINTING library

Function Name	Enumeration value	Long
Main CFI Functions		
xp_sat_nominal_att_init	XP_SAT_NOMINAL_ATT_INIT_ID	0
xp_sat_nominal_att_init_model	XP_SAT_NOMINAL_ATT_INIT_MODEL_ID	1
xp_sat_nominal_att_init_harmonic	XP_SAT_NOMINAL_ATT_INIT_HARMONIC_ID	2
xp_sat_nominal_att_init_file	XP_SAT_NOMINAL_ATT_INIT_FILE_ID	3
xp_sat_nominal_att_close	XP_SAT_NOMINAL_ATT_CLOSE_ID	4
xp_sat_att_angle_init	XP_SAT_ATT_ANGLE_INIT_ID	5
xp_sat_att_matrix_init	XP_SAT_ATT_MATRIX_INIT_ID	6
xp_sat_att_init_harmonic	XP_SAT_ATT_INIT_HARMONIC_ID	7
xp_sat_att_init_file	XP_SAT_ATT_INIT_FILE_ID	8
xp_sat_att_quat_plus_matrix_init	XP_SAT_ATT_QUAT_PLUS_MATRIX_INIT_ID	9
xp_sat_att_quat_plus_angle_init	XP_SAT_ATT_QUAT_PLUS_ANGLE_INIT_ID	10
xp_sat_att_close	XP_SAT_ATT_CLOSE_ID	11
xp_instr_att_angle_init	XP_INSTR_ATT_ANGLE_INIT_ID	12
xp_instr_att_matrix_init	XP_INSTR_ATT_MATRIX_INIT_ID	13
xp_instr_att_init_harmonic	XP_INSTR_ATT_INIT_HARMONIC_ID	14
xp_instr_att_init_file	XP_INSTR_ATT_INIT_FILE_ID	15
xp_instr_att_close	XP_INSTR_ATT_CLOSE_ID	16
xp_change_frame	XP_CHANGE_FRAME_ID	17
xp_attitude_init	XP_ATTITUDE_INIT_ID	18
xp_attitude_compute	XP_ATTITUDE_COMPUTE_ID	19
xp_attitude_user_set	XP_ATTITUDE_USER_SET_ID	20
xp_attitude_close	XP_ATTITUDE_CLOSE_ID	21

xp_set_az_el_definition	XP_SET_AZ_EL_DEFINITION_ID	22
xp_atmos_init	XP_ATMOS_INIT_ID	23
xp_atmos_close	XP_ATMOS_CLOSE_ID	24
xp_dem_init	XP_DEM_INIT_ID	25
xp_dem_compute	XP_DEM_COMPUTE_ID	26
xp_dem_close	XP_DEM_CLOSE_ID	27
xp_dem_get_info	XP_DEM_GET_INFO_ID	28
xp_dem_id_configure	XP_DEM_ID_CONFIGURE_ID	29
xp_dem_get_cell_value	XP_DEM_GET_CELL_VALUE_ID	30
xp_dem_get_cell_geod	XP_DEM_GET_CELL_GEOD_ID	31
xp_target_inter	XP_TARGET_INTER_ID	32
xp_target_travel_time	XP_TARGET_TRAVEL_TIME_ID	33
xp_target_ground_range	XP_TARGET_GROUND_RANGE_ID	34
xp_target_incidence_angle	XP_TARGET_INCIDENCE_ANGLE_ID	35
xp_target_range	XP_TARGET_RANGE_ID	36
xp_target_range_rate	XP_TARGET_RANGE_RATE_ID	37
xp_target_tangent	XP_TARGET_TANGENT_ID	38
xp_target_altitude	XP_TARGET_ALTITUDE_ID	39
xp_target_star	XP_TARGET_STAR_ID	40
xp_target_station	XP_TARGET_STATION_ID	41
xp_target_drs	XP_TARGET_DRS_ID	42
xp_target_generic	XP_TARGET_GENERIC_ID	43
xp_target_reflected	XP_TARGET_REFLECTED_ID	44
xp_target_sc	XP_TARGET_SC_ID	45
xp_multi_target_inter	XP_MULTI_TARGET_INTER_ID	46
xp_multi_target_travel_time	XP_MULTI_TARGET_TRAVEL_TIME_ID	47
xp_target_list_inter	XP_TARGET_LIST_INTER_ID	48
xp_target_extra_vector	XP_TARGET_EXTRA_VECTOR_ID	49
xp_target_extra_main	XP_TARGET_EXTRA_MAIN_ID	50
xp_target_extra_aux	XP_TARGET_EXTRA_AUX_ID	51
xp_target_extra_ef_target	XP_TARGET_EXTRA_EF_TARGET_ID	52
xp_target_extra_target_to_sun	XP_TARGET_EXTRA_TARGET_TO_SUN_ID	53
xp_target_extra_target_to_moon	XP_TARGET_EXTRA_TARGET_TO_MOON_ID	54
xp_target_extra_specular_reflectio n	XP_TARGET_EXTRA_SPECULAR_REFLECTIO N_ID	55
xp_target_list_extra_vector	XP_TARGET_LIST_EXTRA_VECTOR_ID	56
xp_target_list_extra_main	XP_TARGET_LIST_EXTRA_MAIN_ID	57

xp_target_list_extra_aux	XP_TARGET_LIST_EXTRA_AUX_ID	58
xp_target_list_extra_ef_target	XP_TARGET_LIST_EXTRA_EF_TARGET_ID	59
xp_target_list_extra_specular_reflection	XP_TARGET_LIST_EXTRA_SPECULAR_REFLECT_ID	60
xp_target_list_extra_target_to_moon	XP_TARGET_LIST_EXTRA_TARGET_TO_SUN_ID	61
xp_target_list_extra_target_to_sun	XP_TARGET_LIST_EXTRA_TARGET_TO_MOON_ID	62
xp_target_tangent_sun	XP_TARGET_TANGENT_SUN_ID	63
xp_target_tangent_moon	XP_TARGET_TANGENT_MOON_ID	64
xp_target_close	XP_TARGET_CLOSE_ID	65
xp_gen_dem_max_atitude	XP_GEN_DEM_MAX_ALTITUDE_ID	66
xp_gen_dem_altitude_from_ellipsoid	XP_GEN_DEM_ALTITUDE_FROM_ELLIPSOID_ID	67
xp_attitude_define	XP_ATTITUDE_DEFINE_ID	68
xp_attitude_transform	XP_ATTITUDE_TRANSFORM_ID	69
xp_run_init	XP_RUN_INIT_ID	70
Error Handling Functions		
xp_verbose	not applicable	
xp_silent		
xp_get_code		
xp_get_msg		
xp_print_msg		

Notes about the table:

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

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

6.1 Usage hints

The runtime performances of some of the CFI functions are improved to a large extent if they are called two consecutive times keeping constant some of their inputs.

Nevertheless, although the user may not need to call the CFI functions two consecutive times with the same inputs, there are internal functions that are actually called in those conditions, and thus improving the runtime performances of the former.

Thus, the runtime improvement is achieved with any sequence of calls to those CFI functions, not only with a sequence of calls to the same function.

In fact, the time, position, velocity, acceleration vectors, AOCS and mispointing angles do not need to keep exactly constant as long as the difference between two consecutive calls lays within the following thresholds:

- Time: 0.0864 microsec
- Position vector: 0.6e-3 m
- Velocity vector: 0.6e-6 m/s
- Acceleration vector: 0.6e-9 m/s^2
- AOCS: 5e-9 deg
- Mispointing angles: 5e-9 deg
- Mispointing angles-rate: 5e-12 deg
- Mispointing angles-rate-rate: 5e-15 deg

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 (XP_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_POINTING routines, as shown in the table below. The enumerations presented in [GEN_SUM], [F_H_SUM] and [LIB_SUM] are also applicable.

Table 3: Enumerations within EO_POINTING library

Input	Description	Enumeration value	Long
Time Initialization Mode	Initialization from file (data-driven)	XP_SEL_FILE	0
	Initialization within a time range	XP_SEL_TIME	1
	Initialization within a range of orbits	XP_SEL_ORBIT	2
	(not used in POINTING)	XP_SEL_DEFAULT	3
Earth Intersection Mode	No intersection with Earth geoid	XP_NO_INTER	0
	First intersection with Earth geoid	XP_INTER_1ST	1
	Second intersection with Earth geoid	XP_INTER_2ND	2
AOCS mode	Geocentric pointing	XP_AOCS_GPM	0
	Local normal pointing	XP_AOCS_LNP	1
	Yaw steering + local normal pointing	XP_AOCS_YSM	2
	Zero-Doppler YSM	XP_AOCS_ZDOPPLER	3
Satellite Nominal Atti tude Model	Generic model	XP_MODEL_GENERIC	0
	Envisat model	XP_MODEL_ENVISAT	1
	Cryosat model	XP_MODEL_CRYOSAT	2

	ADM model	XP_MODEL_ADM	3
	SENTINEL 1 model	XP_MODEL_SENTINEL1	4
	SENTINEL 2 model	XP_MODEL_SENTINEL2	5
	Geostationary satellite model	XP_MODEL_GEO	6
	MetOp-SG	XP_MODEL_METOPSG	7
Axis enumeration	X axis	XP_X_AXIS	0
	-X axis	XP_NEG_X_AXIS	1
	Y axis	XP_Y_AXIS	2
	-Y axis	XP_NEG_Y_AXIS	3
	Z axis	XP_Z_AXIS	4
	-Z axis	XP_NEG_Z_AXIS	5
Axis target	Sun pointing	XP_SUN_VEC	0
	Moon pointing	XP_MOON_VEC	1
	Earth pointing	XP_EARTH_VEC	2
	Nadir pointing	XP_NADIR_VEC	3
	Inertial velocity pointing	XP_INERTIAL_VEL_VEC	4
	Earth Fixed velocity pointing	XP_EF_VEL_VEC	5
	Inertial target pointing	XP_INERTIAL_TARGET_VEC	6
	Spacecraft Earth Fixed velocity	XP_EF_TARGET_VEC	7
	Earth Fixed target pointing	XP_SC_EF_VEL_VEC	8
	Orbit Pole	XP_ORBIT_POLE	9
	Corrected Satellite Position (ToD)	XP_INERTIAL_POS_VEC_CORRECTED	10
	Rotated Inertial velocity vector (ToD)	XP_INERTIAL_VEL_VEC_ROTATED	11
	North (EF)	XP_EF_NORTH	12
	South (EF)	XP_EF_SOUTH	13
Mode Flag	Flag for location calculus	XP_MODE_FLAG_LOCATION	0
	Flag for direction calculus	XP_MODE_FLAG_DIRECTION	1
Frame Flag	Selection of coordinate frame	XP_FRAME_FLAG_EXT	0
	Selection of attitude frame	XP_FRAME_FLAG_SAT	1
Angle Type	True Latitude (TOD)	XP_ANGLE_TYPE_TRUE_LAT_TOD	0
	True latitude (EF)	XP_ANGLE_TYPE_TRUE_LAT_EF	1
Attittude Frame ID	No attitude frame defined	XP_NONE_ATTITUDE	-1
	Satellite Orbital Reference Frame	XP_SAT_ORBITAL_REF	0
	Satellite Nominal Attitude Frame	XP_SAT_NOMINAL_ATT	1
	Satellite Attitude Frame	XP_SAT_ATT	2
	Instrument(s) Attitude Frame(s)	XP_INSTR_ATT	3

Target Type	User Target	XP_USER_TARGET_TYPE	0
	Line of Sight Target	XP_LOS_TARGET_TYPE	1
	DEM Target	XP_DEM_TARGET_TYPE	2
Source Type	Star	XP_SOURCE_STAR	0
	Sun	XP_SOURCE_SUN	1
	Moon	XP_SOURCE_MOON	2
	Generic source	XP_SOURCE_GENERIC	3
Atmosphere Initialization Mode	No refraction mode	XP_NO_REF_INIT	0
		XP_STD_INIT	1
	User defined mode (n-z table, see section 10)	XP_USER_INIT	2
	Predefined LUT mode	XP_PRED_INIT	3
	Standard refraction mode (US76)	XP_STD_INIT_N	10
	User defined mode (n-z table, see section 10))	XP_USER_INIT_N	20
	Predefined LUT mode	XP_PRED_INIT_N	30
	User's predefined refraction LUTs	XP_US76_INIT	300
		XP_TROPIC_INIT	301
		XP_MID_SUM_INIT	302
		XP_MID_WIN_INIT	303
		XP_SUBAR_SUM_INIT	304
		XP_SUBAR_WIN_INIT	305
		XP_LUT_INIT	400
		XP_US76_INIT_N	3000
		XP_TROPIC_INIT_N	3001
		XP_MID_SUM_INIT_N	3002
		XP_MID_WIN_INIT_N	3003
		XP_SUBAR_SUM_INIT_N	3004
		XP_SUBAR_WIN_INIT_N	3005
		XP_LUT_INIT_N	4000
Attitude file type	Attitude generic file containing a list for angles or quaternions	XP_ATTITUDE_GENERIC_FILE_MODEL	0
	CryoSat Star Tracker File	XP_ATTITUDE_STAR_TRACKER_FILE_MODEL	1
	Frame based on satellite initialized with quaternions and a rotation to the satellite frame (rotation matrix or angles), not from a file.	XP_ATTITUDE_QUATERNION_NO_FILE_MODEL	2
Target extra main	Geocentric longitude and latitude.	XP_TARG_EXTRA_MAIN_GEO	1

results choice	Geodetic altitude and latitude.		
	Geocentric longitude and latitude rates. Geodetic altitude and latitude rates.	XP_TARG_EXTRA_MAIN_GEO_D	2
	Geocentric longitude and latitude rate rates. Geodetic altitude and latitude rate rates.	XP_TARG_EXTRA_MAIN_GEO_2D	4
	Target to satellite azimuth and eleva tion (Topocentric CS)	XP_TARG_EXTRA_MAIN_TARG2SA T_TOP	8
	Target to satellite azimuth and eleva tion rates (Topocentric CS)	XP_TARG_EXTRA_MAIN_TARG2SA T_TOP_D	16
	Target to satellite azimuth and eleva tion rate rates (Topocentric CS)	XP_TARG_EXTRA_MAIN_TARG2SA T_TOP_2D	32
	Satellite to target azimuth and eleva tion (Topocentric CS)	XP_TARG_EXTRA_MAIN_SAT2TAR G_TOP	64
	Satellite to target azimuth and eleva tion rates (Topocentric CS)	XP_TARG_EXTRA_MAIN_SAT2TAR G_TOP_D	128
	Satellite to target azimuth and eleva tion rate rates (Topocentric CS)	XP_TARG_EXTRA_MAIN_SAT2TAR G_TOP_2D	256
	Satellite to target azimuth and eleva tion (Attitude Frame)	XP_TARG_EXTRA_MAIN_SAT2TAR G_ATTITUDE	512
	Satellite to target azimuth and eleva tion rates (Attitude Frame)	XP_TARG_EXTRA_MAIN_SAT2TAR G_ATTITUDE_D	1024
	Satellite to target azimuth and eleva tion rate rates (Attitude Frame)	XP_TARG_EXTRA_MAIN_SAT2TAR G_ATTITUDE_2D	2048
	Target to satellite azimuth and elevation (Attitude Frame). Only meaningful for xp_target_sc	XP_TARG_EXTRA_MAIN_TARG2SA T_ATTITUDE	4096
	Target to satellite azimuth and elevation rates (Attitude Frame). Only meaningful for xp_target_sc	XP_TARG_EXTRA_MAIN_TARG2SA T_ATTITUDE_D	8192
	Target to satellite azimuth and elevation rate rates (Attitude Frame). Only meaningful for xp_target_sc	XP_TARG_EXTRA_MAIN_TARG2SA T_ATTITUDE_2D	16384
	All parameters	XP_TARG_EXTRA_MAIN_ALL	32767
Target extra aux results choice	Minimum distance from the nadir of the target to the ground track.	XP_TARG_EXTRA_AUX_DIST_NAD _TARG_GT	1
	Radius of curvature in the look direc tion at the nadir of the target.	XP_TARG_EXTRA_AUX_RAD_CUR	2
	Minimum distance rate from the nadir of the target to the ground track.	XP_TARG_EXTRA_AUX_DIST_NAD _TARG_GT_D	4
	Minimum distance rate rate from the nadir of the target to the ground track.	XP_TARG_EXTRA_AUX_DIST_NAD _TARG_GT_2D	8

	Radius of curvature rate in the look direction at the nadir of the target.	XP_TARG_EXTRA_AUX_RAD_CUR_D	16
	Radius of curvature rate rate in the look direction at the nadir of the target.	XP_TARG_EXTRA_AUX_RAD_CUR_2D	32
	Target Nadir Velocity relative to the Earth. (Topocentric CS)	XP_TARG_EXTRA_AUX_TARGET_NADIR_VEL	64
	Mean Local Solar Time at target.	XP_TARG_EXTRA_AUX_MLST	128
	True Local Solar Time at target.	XP_TARG_EXTRA_AUX_TLST	256
	Distance from the nadir of the target to the satellite nadir (Earth fixed CS)	XP_TARG_EXTRA_AUX_DIST_NAD_TARG_SAT_NAD	512
	Distance rate from the nadir of the tar get to the satellite nadir (Earth fixed CS)	XP_TARG_EXTRA_AUX_DIST_NAD_TARG_SAT_NAD_D	1024
	Distance rate rate from the nadir of the target to the satellite nadir (Earth fixed CS)	XP_TARG_EXTRA_AUX_DIST_NAD_TARG_SAT_NAD_2D	2048
	R.A. and declination at which the look direction from the satellite to the target point after crossing the atmosphere.	XP_TARG_EXTRA_AUX_LOOK_DIR	4096
	Distance from the SSP to the point on the ground track nearest to the nadir of the target. (Earth fixed CS)	XP_TARG_EXTRA_AUX_DIST_SSP_MIN_DIST_GT	8192
	Distance rate from the SSP to the point on the ground track nearest to the nadir of the target. (Earth fixed CS)	XP_TARG_EXTRA_AUX_DIST_SSP_MIN_DIST_GT_D	16384
	Distance rate rate from the SSP to the point on the ground track nearest to the nadir of the target. (Earth fixed CS)	XP_TARG_EXTRA_AUX_DIST_SSP_MIN_DIST_GT_2D	32768
	All parameters	XP_TARG_EXTRA_AUX_ALL	65535
Satellite Nominal Atti tude Mode	Satellite Nominal Attitude initialised with AOCS mode	XP_SAT_NOMINAL_ATT_INIT_MODE	0
	Satellite Nominal Attitude initialised with Model	XP_SAT_NOMINAL_ATT_INIT_MODEL_MODE	1
	Satellite Nominal Attitude initialised with Harmonics	XP_SAT_NOMINAL_ATT_INIT_HARMONIC_MODE	2
	Satellite Nominal Attitude initialised with a File	XP_SAT_NOMINAL_ATT_INIT_FILE_MODE	3
Satellite Attitude Mode	Satellite Attitude initialised with angles	XP_SAT_ATT_ANGLE_INIT_MODE	0
	Satellite Attitude initialised with matri ces	XP_SAT_ATT_MATRIX_INIT_MODE	1
	Satellite Attitude initialised with Har monics	XP_SAT_ATT_INIT_HARMONIC_MODE	2
	Satellite Attitude initialised with a File	XP_SAT_ATT_INIT_FILE_MODE	3

Instrument Attitude Mode	Instrument Attitude initialised with angles	XP_INSTR_ATT_ANGLE_INIT_MODE	0
	Instrument Attitude initialised with matrices	XP_INSTR_ATT_MATRIX_INIT_MODE	1
	Instrument Attitude initialised with Harmonics	XP_INSTR_ATT_INIT_HARMONIC_MODE	2
	Instrument Attitude initialised with a File	XP_INSTR_ATT_INIT_FILE_MODE	3
Attitude Mode	Attitude not calculated	XP_ATTITUDE_INIT_NO_DATA_MODE	0
	Attitude calculated	XP_ATTITUDE_COMPUTE_MODE	1
	Attitude defined by the user	XP_ATTITUDE_USER_SET_MODE	2
Target Mode	Target calculated with Inter (1st) function	XP_TARGET_INTER_1ST_MODE	0
	Target calculated with Inter (2nd) function	XP_TARGET_INTER_2ND_MODE	1
	Target calculated with Travel Time (1st) function	XP_TARGET_TRAVEL_TIME_1ST_MODE	2
	Target calculated with Travel Time (2nd) function	XP_TARGET_TRAVEL_TIME_2ND_MODE	3
	Target calculated with Ground Range function	XP_TARGET_GROUND_RANGE_MODE	4
	Target calculated with Incidence Angle function	XP_TARGET_INCIDENCE_ANGLE_MODE	5
	Target calculated with Range function	XP_TARGET_RANGE_MODE	6
	Target calculated with Range Rate function	XP_TARGET_RANGE_RATE_MODE	7
	Target calculated with Tangent function	XP_TARGET_TANGENT_MODE	8
	Target calculated with Altitude function	XP_TARGET_ALTITUDE_MODE	9
	Target calculated with Star function	XP_TARGET_STAR_MODE	10
	Target calculated with Tangent to Sun function	XP_TARGET_TANGENT_SUN_MODE	11
	Target calculated with Tangent to Moon function	XP_TARGET_TANGENT_MOON_MODE	12
	Target calculated with Station function	XP_TARGET_STATION_MODE	13
	Target calculated with DRS function	XP_TARGET_DRS_MODE	14
	Target calculated with Generic function	XP_TARGET_GENERIC_MODE	15
	Target calculated with S/C function	XP_TARGET_SC_MODE	16
	Target calculated with Multi Inter (1st) function	XP_MULTI_TARGET_INTER_1ST_MODE	17

	Target calculated with Multi Inter (2nd) function	XP_MULTI_TARGET_INTER_2ND_MODE	18
	Target calculated with Multi Travel Time (1st) function	XP_MULTI_TARGET_TRAVEL_TIME_1ST_MODE	19
	Target calculated with Multi Travel Time (2nd) function	XP_MULTI_TARGET_TRAVEL_TIME_2ND_MODE	20
	Target calculated with Target reflected function	XP_TARGET_REFLECTED_MODE	21
	Target calculated with Target List Inter (1st) function	XP_TARGET_LIST_INTER_1ST_MODE	22
	Target calculated with Target List Inter (2nd) function	XP_TARGET_LIST_INTER_2ND_MODE	23
DEM configuration operations	XP_LOAD_TILE_SET	Load a set of tiles identified by a rectangular region	0
	XP_CLEAR_CACHE	Unload all the tiles in cache but do not free memory	1
	XP_FREE_CACHE	Unload all the tiles in cache but and free memory	2
	XP_SET_MAX_SIZE	Set new maximum size for cache	3
Azimuth elevation type	XP_AZ_EL_LIST	List of azimuth/elevation points	0
	XP_AZ_EL_STRIP	Strip of points with fixed azimuth	1
	XP_AZ_EL_GRID	Grid of azimuth/elevation points	2
Yaw flip attitude configuration	XP_AUTOMATIC_FLIP_MODE	The mode will be updated automatically to Winter or Summer mode	0
	XP_WINTER_MODE	Yaw flip Winter mode	1
	XP_SUMMER_MODE	Yaw flip Summer mode	2
Set of DEM selection	XP_ALL_DEM	All DEM tiles will be computed	0
	XP_DEM_SET	Only tiles in input range will be computed	1

6.3 Data Structures

The aim of the current section is to present the data structures that are used in the EO_POINTING 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 4: EO_POINTING structures

Structure name	Data		
	Variable Name	C type	Description
xp_param_model_str	model_param	double [XP_NUM_MODEL_PARAM]	Model Parameters
	model_enum	long	Model type
xp_harmonic_data	num_terms	long [3]	Number of harmonics coefficient(pitch, roll and yaw)
	harmonic_type_pitch	long [XP_MAX_NUM_HARMONIC]	Harmonic type
	harmonic_type_roll	long [XP_MAX_NUM_HARMONIC]	Harmonic type
	harmonic_type_yaw	long [XP_MAX_NUM_HARMONIC]	Harmonic type
	harmonic_coef_pitch	double [XP_MAX_NUM_HARMONIC]	Harmonic coefficient
	harmonic_coef_roll	double [XP_MAX_NUM_HARMONIC]	Harmonic coefficient
	harmonic_coef_yaw	double [XP_MAX_NUM_HARMONIC]	Harmonic coefficient
xp_harmonic_model_str	angle_type	long	Angle type
	harmonic	xp_harmonic_data	Harmonic data
xp_att_data_rec	time_ref	long	Time reference
	time	double	Time for the quaternions/angles
	quaternion	double [4]	Quaternions
	angles	double [3]	Angles
	file_model	long	File model
xp_sat_nom_att_file_model_str	val_time0	double	Validity start time
	val_time1	double	Validity stop time
	data_type	long	0 = quaternions 1 = angles
	inertial_frame	long	initial reference frame: Inertial reference frame
	lines	long	number of records in the attitude lists

	max_gap	double	Maximum gap between consequtive data
	att_data	xp_att_data_rec*	array with the angle/quaternion records
xp_angle_model_str	pitch	double	Pitch
	roll	double	Roll
	yaw	double	Yaw
xp_matrix_model_str	att_matrix	double [3][3]	Attitude matrix
xp_star_tracker	quaternion[4]	float	Quaternions
	time	double	Quaternion time in TAI
	status	unsigned char	Quaternions status
xp_star_tracker_aux	star_tr_id	long	Star tracker Id (1,2 or 3)
	aberr_correction	long	Aberration correction flag: -1 = Aberration correction with trans posed matrix 0 = No aberration 1 = Aberration correction
	str_att_rot	double [3][3]	Satellite attitude frame to star tracker rotation matrix
xp_sat_att_file_model_str	file_model	long	file model
	val_time0	double	Validity start time
	val_time1	double	Validity stop time
	data_type	long	0 = quaternions 1 = angles
	inertial_frame	long	initial reference frame
	lines	long	number of records in the attitude lists
	max_gap	double	Maximum time gap between angles or quaternions
	att_data	xp_att_data_rec*	Array with the angle/quaternion records
	aux_data	xp_star_tracker_aux	Data from the auxiliary file
	tm_data	xp_star_tracker*	Cryosat Star Tracker attitude data
	rot_to_sat	double**	Matrix with the rotation from the frame based on the satellite to the satellite frame.
xp_instr_att_file_model_str	file_model	long	File model
	val_time0	double	Validity start time
	val_time1	double	Validity stop time

	data_type	long	0 = quaternions 1 = angles
	inertial_frame	long	initial reference frame
	lines	long	number of records in the attitude lists
	max_gap	double	Maximum gap between quaternions/ angles
	att_data	xp_att_data_rec*	array with the angle/quaternion records
xp_quat_plus_angle_model_str	inertial_frame	long	Inertial reference frame
	num_quat	long	Number of quaternions
	quat	xd_att_rec*	List of quaternions
	angles	double[3]	Rotation angles
xp_quat_plus_matrix_model_str	inertial_frame	long	Inertial reference frame
	num_quat	long	Number of quaternions
	quat	xd_att_rec*	List of quaternions
	rot_matrix	double[3][3]	Rotation matrix
xp_attitude_id_data	model	long	Attitude model
	time_ref	long	Time reference
	time	double	Time
	sat_vector	xl_cord	Satellite vector (EF)
	source_frame	long	Source reference frame (according to the extended reference frames enumeration in [LIB_SUM])
	target_frame	long	Target reference frame according to the Attitude Frame ID enumeration, defined in the current document (see Table 3)
	sat_mat	xl_cs_tra	Attitude matrix. Provides transformation from source to target frame
	offset	double [3]	Instrument offset from initial reference frame
	attitude_EF	xl_cs_tra	Attitude matrix in EF: position and orientation of the instrument/satellite frame.
xp_atmos_id_data	atm_max_alt_std	double	Standard atmosphere geometric altitude
	atm_max_alt_user	double	User atmosphere geometric altitude
xp_dem_ace	dir	char [100]	Directory for the the DEM files
(deprecated)	res_X	double	Interval between points along X-axis
	res_Y	double	Interval between points along Y-axis

	res_unit	double	Conversion factor from x,y units to the res_X, res_Y units. for example if res_X is given in seconds and X in degrees => res_unit=3600
	X_num_points	long	Number of points along X-axis (columns)
	Y_num_points	long	Number of points along Y-axis (lines)
	x_range	double	Longitude of the x-axis for one file (grid)
	y_range	double	longitude of the y-axis for one file (grid)
	data_size	long	Size in bytes of the data stored in the files
	data_type	long	data type (int, long, float, double)
	north_alt	double[4]	Altitude at the North pole cell
	south_alt	double[4]	Altitude at the South pole cell
xp_dem_id_data	model	long	DEM model
	dem_data	xp_dem_ace *	DEM configuration data (deprecated)
	dem_user_params	xd_dem_user_params	DEM user parameters (see [D_H_SUM])
	dem_metadata	xd_dem_metadata	DEM metadata (see [D_H_SUM])
xp_dem_info	dem_model	long	DEM model (according to XD_Dem_model_enum in [D_H_SUM])
	data_source	long	Source flag. According to the dem _model, the value is one of the following enumerations (see [D_H_SUM]): • DEM Data Source Types for GETASSE30 v1, v2 and v3 • DEM Data Source Types for ACE2 • DEM Data Source Types for GDEM v2 • For GDEM v3 the range for the data source is a wide range as follows: *Value* *Description* 0-50 GDEM V3 (0 to 50+ scenes) 60-110 GDEM V2 (0 to 50+ scenes) 131-184 PRISM (1 to 54 scenes)

			201-223 SRTM (1 to 23 swaths) 231 SRTM V3 from initial GDEM V3 232 SRTM V2 from initial GDEM V3 233 SRTM V2 from GDEM V2 234 SRTM with NGA fill from GDEM V2 241 NED from GDEM V2 (USA) 242 NED from initial GDEM V3 (USA) 243 CDED from GDEM V2 (Canada) 244 CDED from initial GDEM V3 (Canada) 245 Alaska DEM from GDEM V2 246 Alaska DEM from initial GDEM V3 250 Interpolated (described in “Reference Data for Number of Scenes Layer” in https://lpdaac.usgs.gov/products/astgtmv003/)
xp_generic_data	time_ref	long	Time reference
	time	double	Time
	sat_vector	xl_cord	Satellite state vector (see [LIB_SUM])
	iray	long	Refraction model
	freq	double	Frequency
	deriv	long	Derivative flag according to deriva tives enumeration in [LIB_SUM]
xp_target_data	tar_vector	xl_cord	target vector
	z_tan	xl_par_der	Tangent altitude
	range	xl_par_der	target range
	time	xl_par_der	time
	tar_sat_vector	xl_cord	target to satellite vector
	sat_tar_vector	xl_cord	satellite to target vector
xp_target_str	num_target	long	Number of targets
	target	xp_target_data *	target data
xp_target_id_data	generic_data	xp_generic_data	Target generic data
	earth_crossed	long	Flag to indicate if the Earth is crossed
	atm_crossed	long	Flag to indicate if the atmosphere is crossed
	user	xp_target_str	User target
	los	xp_target_str	LOS target

	earth	xp_target_str	Earth target
	exit_atm_vector	xl_cord	Pointing vector at exit from atmos phere
xp_latlon_area	lon_min	double	Minimum longitude of rectangular area
	lon_max	double	Maximum longitude of rectangular area
	lat_min	double	Minimum latitude of rectangular area
	lat_max	double	Maximum latitude of rectangular area
xp_dem_id_config	command	long	Operation to be executed (DEM configuration enum)
	max_cache_size	long	Maximum cache size (MegaBytes)
	area	xp_latlon_area	Rectangular area
xp_azimuth_elevation	azimuth	double	Azimuth [deg]
	elevation	double	Elevation [deg]
	azimuth_rate	double	Azimuth rate [deg/s]
	elevation_rate	double	Elevation rate [deg/s]
xp_azimuth_elevation_list	num_rec	long	Number of azimuth/elevation points
	az_el_list	xp_azimuth_elevation*	Array of azimuth/elevation points
xp_azimuth_elevation_strip	azimuth	double	Fixed azimuth of strip [deg]
	min_elevation	double	Lower elevation of strip points [deg]
	max_elevation	double	Maximum elevation of strip points [deg]
	step_elevation	double	Step between elevation points [deg]
xp_azimuth_elevation_grid	min_azimuth	double	Lower azimuth of grid points [deg]
	max_azimuth	double	Maximum azimuth of grid points [deg]
	step_azimuth	double	Step between azimuth points [deg]
	min_elevation	double	Lower elevation of grid points [deg]
	max_elevation	double	Maximum elevation of grid points [deg]
	step_elevation	double	Step between elevation points [deg]
xp_azimuth_elevation_input_union	azimuth_elevation_list	xp_azimuth_elevation_list	List of points
	azimuth_elevation_strip	xp_azimuth_elevation_strip	Strip of points
	azimuth_elevation_grid	xp_azimuth_elevation_grid	Grid of points
xp_instrument_data	type	long	Type of instrument data (see Azimuth elevation type enumeration)
	azimuth_elevation_input_union	xp_azimuth_elevation_input_union	Azimuth/elevation points

	signal_frequency	double	Signal frequency
xp_target_output	num_user_target	long	Number of user targets
	num_los_target	long*	Array of length num_user_targets with the number of LOS targets corresponding to every user target
xp_target_list_input_info	input_type	long	List/Grid/Strip (see XP_Az_el_type_enum)
	azimuth	double	Azimuth used as input to compute the target
	elevation	double	Elevation used as input to compute the target
	azimuth_idx	long	Azimuth index in azimuth/elevation grid
	elevation_idx	long	Elevation index in azimuth/elevation grid
xp_target_input_info_union	target_list_input_info	xp_target_list_input_info	Target input information union
xp_target_input_info	target_function	long	It defines the function used to compute the target
	target_input_info_union	xp_target_input_info_union	Target input information
xp_target_extra_vector_results	status	long	Target computation status
	target_input_info	xp_target_input_info	Input information used to compute the target
	vector_results	double[]	Results
	vector_results_rate	double[]	Results rate
	vector_results_rate_rate	double[]	Results rate rate
xp_target_extra_vector_results_list	num_rec	long	Number of targets computed
	extra_vector_results	xp_target_extra_vector_results*	Array with results for every target
xp_target_extra_main_results	status	long	Target computation status
	target_input_info	xp_target_input_info	Input information used to compute the target
	main_results	double[]	Results
	main_results_rate	double[]	Results rate
	main_results_rate_rate	double[]	Results rate rate
xp_target_extra_main_results_list	num_rec	long	Number of targets computed
	extra_main_results	xp_target_extra_main_results*	Array with results for every target

xp_target_extra_aux_results	status	long	Target computation status
	target_input_info	xp_target_input_info	Input information used to compute the target
	aux_results	double[]	Results
	aux_results_rate	double[]	Results rate
	aux_results_rate_rate	double[]	Results rate rate
xp_target_extra_aux_results_list	num_rec	long	Number of targets computed
	extra_aux_results	xp_target_extra_aux_results*	Array with results for every target
xp_target_extra_ef_target_results	status	long	Target computation status
	target_input_info	xp_target_input_info	Input information used to compute the target
	ef_target_results	double[]	Results
	ef_target_results_rate	double[]	Results rate
	ef_target_results_rate_rate	double[]	Results rate rate
xp_target_extra_ef_target_results_list	num_rec	long	Number of targets computed
	extra_ef_target_results	xp_target_extra_ef_target_results*	Array with results for every target
xp_target_extra_spec_reflec_target_results	status	long	Target computation status
	target_input_info	xp_target_input_info	Input information used to compute the target
	spec_reflec_results	double[]	Results
	spec_reflec_results_rate	double[]	Results rate
	spec_reflec_results_rate_rate	double[]	Results rate rate
xp_target_extra_spec_reflec_target_results_list	num_rec	long	Number of targets computed
	extra_spec_reflec_target_results	xp_target_extra_spec_reflec_target_results*	Array with results for every target
xp_target_extra_moon_target_results	status	long	Target computation status
	target_input_info	xp_target_input_info	Input information used to compute the target
	moon_results	double[]	Results
	moon_rate	double[]	Results rate
	moon_rate_rate	double[]	Results rate rate
xp_target_extra_moon_target_results_list	num_rec	long	Number of targets computed
	extra_moon_target_results	xp_target_extra_moon_target_results*	Array with results for every target
xp_target_extra_su	status	long	Target computation status

n_target_results	target_input_info	xp_target_input_info	Input information used to compute the target
	sun_results	double[]	Results
	sun_rate	double[]	Results rate
	sun_rate_rate	double[]	Results rate rate
xp_target_extra_sun_target_results_list	num_rec	long	Number of targets computed
	extra_sun_target_results	xp_target_extra_sun_target_results*	Array with results for every target
xp_gen_dem_alt_from_ellipsoid_inputs	set_type	long	DEM set selected for computation (see DEM set enum)
	lon_min	double	Minimum longitude of DEM set
	lon_max	double	Maximum longitude of DEM set
	lat_min	double	Minimum latitude of DEM set
	lat_max	double	Maximum latitude of DEM set
	verbose	long	If == 0, no log message will be printed. If > 0, a log message will be printed every “verbose” points processed in DEM.
xp_attitude_def	type	long	Attitude type. Possible values: XP_NONE_ATTITUDE: No attitude defined XP_SAT_NOMINAL_ATT: Satellite Nominal attitude target. XP_SAT_ATT: Satellite attitude target. XP_INSTR_ATT: Instrument attitude target.
	sat_nom_trans_id	xp_sat_nom_trans_id	Satellite Nominal attitude
	sat_trans_id	xp_sat_trans_id	Satellite attitude target
	instr_trans_id	xp_instr_trans_id	Instrument attitude target
xp_transform_cfg	ref_frame	long	New reference frame. See enumeration “Reference frame” in [D_H_SUM]
	time_id	xl_time_id	Time Id.
	model_id	xl_model_id	Model Id.

7 CFI FUNCTIONS DESCRIPTION

The following sections describe each CFI function.

The calling interfaces are described for C users.

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 xp_sat_nominal_att_init

7.1.1 Overview

The **xp_sat_nominal_att_init** CFI function initialises the AOCS mode for a given satellite. The initialised mode will be stored in the *sat_nom_trans_id* output structure.

7.1.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     long aocs_mode;
     xp_sat_nom_trans_id sat_nom_trans_id = {NULL};
     long ierr[XP_NUM_ERR_NOM_ATT_INIT_DEF], status;

     status = xp_sat_nominal_att_init(&aocs_mode,
                                      &sat_nom_trans_id, ierr);
}
```

The XP_NUM_ERR_SAT_NOM_ATT_INIT constant is defined in the file *explorer_pointing.h*.

7.1.3 Input Parameters

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

Table 5: Input parameters of xp_sat_nominal_att_init function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
aocs_mode	long *	-	AOCS Mode ID	-	Complete

It is possible to use enumeration values rather than integer values for some of the input arguments:
AOCS Mode ID: aocs_mode. See current document, Table 3.

7.1.4 Output Parameters

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

Table 6: Output parameters of xp_sat_nominal_att_init

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_nom_trans_id	xp_sat_nom_trans_id*	-	Structure that contains the Satellite nominal Transformation	-	-
ierr	long	-	Error vector	-	-

7.1.5 Warnings and Errors

Next table lists the possible error messages that can be returned by the **xp_sat_nominal_att_init** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_POINTING software library **xp_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 by translating the error vector returned by the **xp_sat_nominal_att_init** function by calling the function of the EO_POINTING software library **xp_get_code** (see [GEN_SUM])

Table 7: Error messages of xp_sat_nominal_att_init function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Memory allocation error	No calculation performed	XP_CFI_SAT_NOMINAL_ATT_INIT_MEMORY_ERR	0

7.2 xp_sat_nominal_att_init_model

7.2.1 Overview

The **xp_sat_nominal_att_init_model** CFI function initialises the satellite nominal attitude model for a given satellite. The initialised model will be stored in the *sat_nom_trans_id* output structure.

7.2.2 Calling Interface

The calling interface of the **xp_sat_nominal_att_init_model** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
     long model_enum;
     double model_param[XP_NUM_MODEL_PARAM];
     xp_sat_nom_trans_id sat_nom_trans_id = {NULL};
     long ierr[XP_NUM_ERR_SAT_NOM_ATT_INIT_MODEL], status;

     status = xp_sat_nominal_att_init_model(&model_enum,
                                             model_param,
                                             &sat_nom_trans_id, ierr);
}
```

The `XP_NUM_ERR_SAT_NOM_ATT_INIT_MODEL` and `XP_NUM_MODEL_PARAM` constants are defined in the file *explorer_pointing.h.*

7.2.3 Input Parameters

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

Table 8: Input parameters of xp_sat_nominal_att_init_model function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
model_enum	long *	-	Sat Nom Attitude Model ID	-	Complete
model_param	double	-	Model dependant parameters	-	Complete

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

- Satellite Nominal Attitude Model ID: model_enum. See current document, Table 3.
- Model dependant parameters: model_param. See current document, Table 9.

Table 9: Model parameters depending on the attitude model

Attitude Model	Array Element	Description (Reference)	Unit (Format)
XP_MODEL_GENERIC	[0]	First Axis enumeration value	-
	[1]	First Target enumeration value	-
	[2]	First Vector[0]	- or deg
	[3]	First Vector[1]	- or deg
	[4]	First Vector[2]	- or deg
	[5]	Second Axis enumeration value	
	[6]	Second Target enumeration value	
	[7]	Second Vector[0]	- or deg
	[8]	Second Vector[1]	- or deg
	[9]	Second Vector[2]	- or deg
XP_MODEL_ENVISAT	[0]	AOCS Cx parameter [pitch]	deg
	[1]	AOCS Cy parameter [roll]	deg
	[2]	AOCS Cz parameter [yaw]	deg
XP_MODEL_CRYOSAT	[0]	Local Normal Z Coefficient	-
XP_MODEL_ADM	[0]	Scan Angle	deg
	[1]	Scan Limit	deg
	[2]	Velocity Offset	m/s
XP_MODEL_SENTINEL1	[0]	Local Normal Coefficient	-
	[1]	Earth's angular velocity vector	rad/s
	[2]	Antenna bore sight off nadir angle at reference altitude (Θ_{ref})	deg

	[3]	Reference altitude (H_{ref})	km
	[4]	Roll steering sensitivity versus altitude (α_{roll})	deg/km
	[5]	h_0	m
	[6]	h_1	m
	[7]	h_2	m
	[8]	h_3	m
	[9]	h_4	m
	[10]	φ_1	rad
	[11]	φ_2	rad
	[12]	φ_3	rad
	[13]	φ_4	rad
XP_MODEL_SENTINEL2	[0]	Earth's angular velocity vector	rad/s
XP_MODEL_GEO	[0]	Flag (enumeration values can be used): XP_AUTOMATIC_FLIP_MODE = enable the automatic Yaw Flip XP_WINTER_MODE = Winter XP_SUMMER_MODE = Summer	-
XP_MODEL_METOPSG	[0]	Local Normal Coefficient	-
	[1]	Earth's angular velocity vector	rad/s

7.2.3.1 Generic Model description

The generic model builds the reference frames from the specified direction vectors.

The model parameters are:

- first_axis: It can be any of {XP_X_AXIS, XP_NEG_X_AXIS, XP_Y_AXIS, XP_NEG_Y_AXIS, XP_Z_AXIS, XP_NEG_Z_AXIS}
- first_target: It can be any of {XP_SUN_VEC, XP_MOON_VEC, XP_EARTH_VEC, XP_NADIR_VEC, XP_INERTIAL_VEL_VEC, XP_EF_VEL_VEC, XP_INERTIAL_TARGET_VEC, XP_EF_TARGET_VEC, XP_SC_EF_VEL_VEC, XP_ORBIT_POLE, XP_INERTIAL_POS_VEC_CORRECTED, XP_INERTIAL_VEL_VEC_ROTATED, XP_EF_NORTH, XP_EF_SOUTH}
- first_vector[3]: contains either:
  - dummies
  - [long, lat, alt] if first target = XP_EF_TARGET_VEC
  - [ra, decl, parallax] if first target = XP_INERTIAL_TARGET_VEC
  - correction coefficients if first target = XP_INERTIAL_POS_VEC_CORRECTED
  - rotation vector if first target = XP_INERTIAL_VEL_VEC_ROTATED

- second_axis: It can be any of {XP_X_AXIS, XP_NEG_X_AXIS, XP_Y_AXIS, XP_NEG_Y_AXIS,XP_Z_AXIS, XP_NEG_Z_AXIS}
- second_target: : It can be any of {XP_SUN_VEC, XP_MOON_VEC, XP_EARTH_VEC, XP_NADIR_VEC, XP_INERTIAL_VEL_VEC, XP_EF_VEL_VEC, XP_INERTIAL_TARGET_VEC, XP_EF_TARGET_VEC, XP_SC_EF_VEL_VEC, XP_ORBIT_POLE, XP_INERTIAL_POS_VEC_CORRECTED, XP_INERTIAL_VEL_VEC_ROTATED, XP_EF_NORTH, XP_EF_SOUTH}
- second_vector[3]: contains either:
  - dummies
  - [long, lat, alt] if second target= XP_EF_TARGET_VEC
  - [ra, decl, parallax] if fsecond target=XP_INERTIAL_TARGET_VEC
  - correction coefficients if second target=XP_INERTIAL_POS_VEC_CORRECTED
  - rotation vector if second target = XP_INERTIAL_VEL_VEC_ROTATED

It is necessary to define a convention for each target type (e.g, always from Satellite to XXX):
- XP_SUN_VEC: Unit direction vector from Satellite to Sun
- XP_MOON_VEC: Unit direction vector from Satellite to Moon
- XP_EARTH_VEC: Unit direction vector from Satellite to Earth centre (opposite to Satellite Position Vector)
- XP_NADIR_VEC: Unit direction vector from Satellite to Nadir point
- XP_INERTIAL_VEL_VEC: Inertial Velocity vector (in TOD)
- XP_EF_VEL_VEC: Earth Fixed Velocity vector
- XP_INERTIAL_TARGET_VEC: Unit direction vector from Satellite to a target defined by a given [ra, decl, parallax]. The annual parallax is used in case we are pointing to a close object (for instance, the Moon), in order to get the distance. For stars, parallax=0 shall be used, meaning infinite distance. Units: degrees
- XP_EF_TARGET_VEC: Unit direction vector from Satellite to a target defined by a given [long, lat, alt]
- XP_SC_EF_VEL_VEC: Satellite Earth Fixed Velocity vector
- XP_ORBIT_POLE: Unit direction vector normal to the orbital plane (computed as the cross product of the Satellite Position vector and its Velocity vector)
- XP_INERTIAL_POS_VEC_CORRECTED: Unit Satellite position vector in ToD corrected by coefficients (e.g to approximate the local normal direction)
- XP_INERTIAL_VEL_VEC_ROTATED: Inertial Velocity vector in ToD rotated (e.g correcting for the Earth rotation)
- XP_EF_NORTH: Unit direction vector pointing North (in Earth Fixed)
- XP_EF_SOUTH: Unit direction vector pointing South (in Earth Fixed)

With these parameters, the calculation is done as follows:
- Compute the unit direction vector specified by first_target
  - Assign the calculated first target vector to the first axis vector
- Compute the unit direction vector specified by second_target
  - Cross-product of the first axis vector and the second target vector
  - Assign the resulting vector to the second axis vector
  - Complete the right-handed frame

The following are some examples:

Sun-Fixed Reference Frame

- model_param = {XP_X_AXIS, XP_SUN_VEC, 0.0, 0.0, 0.0, XP_Z_AXIS, XP_EARTH_VEC, 0.0, 0.0, 0.0}
  Then:
- X-axis = Unit vector from Satellite to Sun (Sun Vector)
- Z-axis = Unit cross product: X-axis x (Unit vector from Satellite to Earth (Earth Vector))
- Y-axis = Z-axis x X-axis (completing the right-handed frame)

Yaw Steering Mode

- model_param={XP_NEG_Z_AXIS, XP_NADIR_VEC, 0.0, 0.0, 0.0, XP_X_AXIS, XP_SC_EF_VEL_VEC, 0.0, 0.0, 0.0}
  Then:
- Z-axis = -(Unit vector from Satellite to Nadir (Nadir Vector))
- X-axis = Unit cross product: Z-axis x (Satellite Earth-Fixed Velocity Vector)
- Y-axis = Z-axis x X-axis (completing the right-handed frame)

7.2.3.2 Sentinel-1 Model parameters description

The parameters for the Sentinel-1 attitude model corresponds to the roll steering law:

$$\theta_{offNadir} = \theta_{ref} - \alpha_{roll}(H - H_{ref})$$

where the actual altitude of the satellite is approximated by the harmonic function:

$$H(t) = h_0 + \sum_{n=1}^{N} h_n \cdot \sin(n \cdot \omega_{orb} \cdot (t - t_{ANX}) + \phi_n)$$

The first fourth terms of the series are considered.

Consult [MSC] for more information.

7.2.3.3 Sentinel-2 Model description

Sentinel 2 model is implemented as generic model with the following definitions:

- First axis: XP_NEG_Z_AXIS; first target = XP_EARTH_VEC.

- Second axis: XP_X_AXIS; second target = XP_INERTIAL_VEL_VEC_ROTATED

7.2.3.4 Yaw flip attitude Model description

Yaw Flip model is implemented as generic model with the following definitions:

1. For WINTER mode:

   - First axis: XP_NEG_Z_AXIS; first target = XP_NADIR_VEC.

   - Second axis: XP_X_AXIS; second target = XP_EF_SOUTH

2. For SUMMER mode:

   - First axis: XP_NEG_Z_AXIS; first target = XP_NADIR_VEC.

   - Second axis: XP_X_AXIS; second target = XP_EF_NORTH

3. For AUTOMATIC Yaw Flip, the attitude is set to WINTER or SUMMER mode depending on the Sun position: if the Sun position is above the equatorial plane, SUMMER mode is selected; if the Sun position is below the equatorial plane, WINTER mode is selected.

7.2.3.5 MetOp-SG Model description

MetOp-SG model is identical to the ideal YSM law with the following definitions:

- The Z axis that is computed with an approximation for the local normal vector using an altitude dependent correction factor.

- The input parameters are the local normal coefficient and the Earth's rotation speed.

- First axis: XP_Z_AXIS; first target = XP_INERTIAL_POS_VEC_CORRECTED

- Second axis: XP_X_AXIS; second target = XP_INERTIAL_VEL_VEC_ROTATED

7.2.4 Output Parameters

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

Table 10: Output parameters of xp_sat_nominal_att_init_model

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_nom_trans_id	xp_sat_nom_trans_id*	-	Structure that contains the Satellite nominal Transformation	-	-
ierr	long	-	Error vector	-	-

7.2.5 Warnings and Errors

Next table lists the possible error messages that can be returned by the **xp_sat_nominal_att_init_model** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_POINTING software library **xp_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 by translating the error vector returned by the **xp_sat_nominal_att_init_model** function by calling the function of the EO_POINTING software library **xp_get_code** (see [GEN_SUM])

Table 11: Error messages of xp_sat_nominal_att_init_model function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Memory allocation error	No calculation performed	XP_CFI_SAT_NOMINAL_ATT_INIT_MODEL_MEMORY_ERR	0

7.3xp_sat_nominal_att_init_harmonic

7.3.1 Overview

The **xp_sat_nominal_init_harmonic** CFI function initialises the satellite orbital to satellite nominal attitude mispointing angles (i.e. roll, pitch, yaw) for a given satellite with a user-provided set of values. The initialised values will be stored in the *sat_nom_trans_id* output structure.

The mispointing angle (attitude angle in the formula) will be calculated by functions using such sat_nominal_trans_id (i.e. xp_attitude_compute or xp_change_frame) according to the following formula (the "angle" variable will be calculated as in xl_position_on_orbit (see [LIB_SUM]), using as inputs:

- the input state vector in EF passed to such functions;
- the angle_type passed as input to xp_sat_nominal_att_init_harmonic.

7.3.2 Calling Interface

The calling interface of the **xp_sat_nominal_att_init_harmonic** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
    long angle_type, num_terms[3];
    long harmonic_type_pitch[XP_MAX_NUM_HARMONIC],
         harmonic_type_roll[XP_MAX_NUM_HARMONIC],
         harmonic_type_yaw[XP_MAX_NUM_HARMONIC];
    double harmonic_coef_pitch[XP_MAX_NUM_HARMONIC],
        harmonic_coef_roll[XP_MAX_NUM_HARMONIC],
        harmonic_coef_yaw[XP_MAX_NUM_HARMONIC];
    xp_sat_nom_trans_id sat_nom_trans_id = {NULL};
    long ierr[XP_NUM_ERR_SAT_NOM_ATT_INIT_HARMONIC], status;

    status =     xp_sat_nominal_att_init_harmonic(&angle_type,
                                  num_terms,
                                  harmonic_type_pitch,
                                  harmonic_type_roll,
                                  harmonic_type_yaw,
                                  harmonic_coef_pitch,
                                  harmonic_coef_roll,
                                  harmonic_coef_yaw,
                                  &sat_nom_trans_id,
```

```
                                              ierr);
   }
```

The XP_NUM_ERR_SAT_NOM_ATT_INIT_HARMONIC and XP_MAX_NUM_HARMONIC constants are defined in the file *explorer_pointing.h.*

7.3.3 Input Parameters

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

Table 12: Input parameters of xp_sat_nominal_att_init_harmonic function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
angle_type	long *	-	Type of angle	-	XP_ANGLE_TYPE_TRUE_LAT_TOD XP_ANGLE_TYPE_TRUE_LAT_EF
num_terms[3]	long	[0]	Number of elements used in vectors harmonic_type_pitch and harmonic_coef_pitch	-	>=0
		[1]	Number of elements used in vectors harmonic_type_roll and harmonic_coef_roll	-	>=0
		[2]	Number of elements used in vectors harmonic_type_yaw and harmonic_coef_yaw	-	>=0
harmonic_type_pitch	long[XP_MAX_NUM_HARMONIC]	all	Type of coefficients: =0 for the bias parameter <0 for the sinus coefficients (-n means that correponds to the sinus coefficient of order n) >0 for the cosinus coefficients (+n means that correponds to the cosinus coefficient of order n)	-	
harmonic_type_roll	long[XP_MAX_NUM_HARMONIC]	all	Type of coefficients: =0 for the bias parameter <0 for the sinus coefficients (-n means that correponds to the sinus coefficient of order n) >0 for the cosinus coefficients (+n means that correponds to the cosinus coefficient of order n)	-	-
harmonic_type_yaw	long[XP_MAX_NUM_HARMONIC]	all	Type of coefficients: =0 for the bias parameter <0 for the sinus coefficients (-n means that correponds to the sinus coefficient of order n)	-	-

			>0 for the cosinus coefficients (+n means that correponds to the cosinus coefficient of order n)		
harmonic_coef_pitch	double[XP_MAX_NUM_HARMONIC]	all	Bias, sinus and cosinus coeficients for the pitch angle	deg	-
harmonic_coef_roll	double[XP_MAX_NUM_HARMONIC]	all	Bias, sinus and cosinus coeficients for the roll angle	deg	-
harmonic_coef_yaw	double[XP_MAX_NUM_HARMONIC]	all	Bias, sinus and cosinus coeficients for the yaw angle	deg	-

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

- Angle Type: See current document, Table 3.

7.3.4 Output Parameters

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

Table 13: Output parameters of xp_sat_nominal_att_init_harmonic

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_nom_trans_id	xp_sat_nom_trans_id*	-	Structure that contains the Satellite nominal Transformation	-	-
ierr	long	-	Error vector	-	-

7.3.5 Example

For the satellite ERS:

pitch = -0.16725*cos(true_lat)*sin(true_lat)*2=-0.16725*sin(2*true_lat)

num_terms[0]=1

harmonic_type_pitch={-2} harmonic_coef_pitch={-0.16725}

roll = 0.05012*sin(true_lat)

num_terms[1]=1

harmonic_type_roll={-1} harmonic_coef_roll={0.05012} yaw= 3.9163*cos(true_lat)

num_terms[2]=1

harmonic_type_yaw={+1} harmonic_coef_yaw={3.9163}

7.3.6 Warnings and Errors

Next table lists the possible error messages that can be returned by the **xp_sat_nominal_att_init_harmonic** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_POINTING software library **xp_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 by translating the error vector returned by the **xp_sat_nominal_att_init_harmonic** function by calling the function of the EO_POINTING software library **xp_get_code** (see [GEN_SUM])

Table 14: Error messages of xp_sat_nominal_att_init_harmonic function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Memory allocation error	No calculation performed	XP_CFI_SAT_NOMINAL_ATT_INIT_HARMONIC_MEMORY_ERR	0

7.4 xp_sat_nominal_att_init_file

7.4.1 Overview

The **xp_sat_nominal_att_init_file** CFI function initialises the satellite nominal attitude rotation for a given satellite reading values from the attitude file(s). The validity time or orbital range for the attitude angles can be specified by the user. The initialised values will be stored in the *sat_nom_trans_id* output structure.

The computed rotation will be given between the inertial reference frame specified in the file (tag <Reference_Frame>) and the satellite nominal attitude frame. In case that the files contain rotation angles (pitch, roll, yaw) and if the reference frame is based on the satellite (“SATELLITE_ACTUAL”, “SATELLITE” or “SATELLITE_RELATIVE”), then the rotation will be defined between the satellite orbital frame and the satellite nominal attitude frame.

In order to read files, xp_sat_nominal_att_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.

7.4.1.1 Initialisation with several files.

When the attitude frame is initialized with more than one file (see parameter *attitude_file* in next section), the user has to provide the files sorted from lower to higher precedence. This way the possible overlap between files is solved as follows:

attitude_file[0] is the file with lower precedence

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

the attitude data (angles or quaternions) from files with lower precedence that are after the data of a file with higher precedence are skipped.

The maximum gap allowed for the whole set of attitude data, is taken from the first file (*attidude_file[0]*).

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

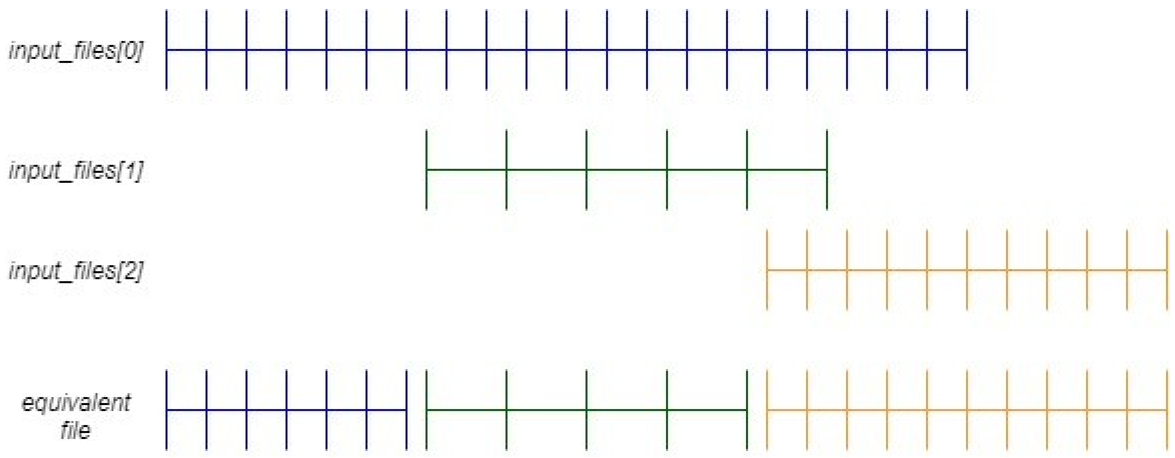


Figure 7: Overlapping attitude files

7.4.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     xl_time_id time_id = {NULL};
     long n_files, time_init_mode, time_ref;
     char **attitude_file;
     double time0, time1;
     double val_time0, val_time1;
     xp_sat_nom_trans_id sat_nom_trans_id = {NULL};
     long ierr[XP_NUM_ERR_SAT_NOM_ATT_INIT_FILE], status;

     status = xp_sat_nominal_att_init_file(&time_id, &n_files,
          attitude_file, &time_init_mode, &time_ref, &time0, &time1,
          &val_time0, &val_time1, &sat_nom_trans_id, ierr);
}
```

The `XP_NUM_ERR_SAT_NOM_ATT_INIT_FILE` constant is defined in the file *explorer_pointing.h.*

7.4.3 Input Parameters

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

Table 15: Input parameters of xp_sat_nominal_att_init_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
time_id	xl_time_id*	-	Structure that contains the time correlations.	-	-
n_files	long *	-	Number of reference data files	-	> 0
attitude_file	char **	-	Filenames of the reference data files. In case multiple files are used, the files should be time ordered and all files should be of the same type (see section 7.4.1.1). The supported Attitude File format are: - Generic Attitude File. - Attitude Ephemeris Message Files. In this case only one file is allowed and only the first segment of the file is taken for the initalisation. Those files are described in [D_H_SUM].	-	-
time_init_mode	long *	-	Flag for selecting the time range of the initialization.	-	Select either: · XP_SEL_TIME · XP_SEL_FILE
time_ref	long *	-	Time reference ID	-	Complete
time0	double*	-	If:*time_init_mode=XP_SEL_TIME* S Start of the time range defined by[time0,time1]	Decimal days (Processing format)	[-18262.0,36524.0]
time1	double*	-	If: *time_init_mode=XP_SEL_TIME* End of the time range defined by [time0,time1]	Decimal days (Process ing format)	[-18262.0,36524.0] > time0

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].
- Time Init Mode ID: time_init_mode. See current document, Table 3.

7.4.4 Output Parameters

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

Table 16: Output parameters of xp_sat_nominal_att_init_file

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
val_time0	double*	-	Validity start time of the initialization	Decimal days (Processing format)	[-18262.0,36524.0]
val_time1	double*	-	Validity end time of the initialization	Decimal days (Processing format)	[-18262.0,36524.0]
sat_nom_trans_id	xp_sat_nom _trans_id*	-	Structure that contains the Satellite nominal Transformation	-	-
ierr	long	-	Error vector	-	-

7.4.5 Warnings and Errors

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

Table 17: Error messages of xp_sat_nominal_att_init_file function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Memory allocation error	No calculation performed	XP_CFI_SAT_NOMINAL_AT T_INIT_FILE_MEMORY_ER R	0
ERR	Wrong input time reference	No calculation performed	XP_CFI_SAT_NOMINAL_AT T_INIT_FILE_WRONG_TIME _REF_ERR	1
ERR	Error opening attitude file: %s	No calculation performed	XP_CFI_SAT_NOMINAL_AT T_INIT_FILE_OPEN_FILES_ ERR	2
ERR	Error reading generic attitude files	No calculation performed	XP_CFI_SAT_NOMINAL_AT T_INIT_FILE_READ_ATT_FI	3

			LES_ERR	
ERR	Could not perform a time transformation	No calculation performed	XP_CFI_SAT_NOMINAL_ATT_INIT_FILE_TIME_CONV_ ERR	4
ERR	Unsupported attitude file format detected	No calculation performed	XP_CFI_SAT_NOMINAL_ATT_INIT_FILE_WRONG_FILE_FORMAT_ERR	5
WARN	AEM with more than 1 segment used for initialization. Only 1st segment will be used	Attitude is initialised using the 1st segment in the AEM file.	XP_CFI_SAT_NOMINAL_ATT_INIT_FILE_ONLY_SEGMENT0_WARN	6

7.5 xp_sat_nominal_att_close

7.5.1 Overview

The **xp_sat_nominal_att_close** CFI function cleans up any memory allocation performed by the satellite nominal attitude initialization functions.

7.5.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
    xp_sat_nom_trans_id sat_nom_trans_id = {NULL};
    long ierr[XP_NUM_ERR_SAT_NOM_ATT_CLOSE], status;

    status =     xp_sat_nominal_att_close(&sat_nom_trans_id, ierr);
}
```

The `XP_NUM_ERR_SAT_NOM_ATT_CLOSE` constant is defined in the file *explorer_pointing.h*.

7.5.3 Input Parameters

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

Table 18: Input parameters of xp_sat_nominal_att_close function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_nom_trans_id	xp_sat_nom _trans_id*	-	Structure that contains the Satellite Nom. Trans.	-	-

7.5.4 Output Parameters

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

Table 19: Output parameters of xp_sat_nominal_att_close

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

7.5.5 Warnings and Errors

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

Table 20: Error messages of xp_sat_nominal_att_close function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Could not close the Id. as it is not initialized or it is being used	No calculation performed	XP_CFI_SAT_NOMINAL_A TT_CLOSE_WRONG_ID_E RR	0

7.6 xp_sat_nominal_att_get_aocs

7.6.1 Overview

The **xp_sat_nominal_att_get_aocs** CFI function returns AOCS mode used for the satellite nominal attitude initialization.

7.6.2 Calling interface

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

```
#include <explorer_lib.h>
{
       xp_sat_nom_trans_id sat_nom_trans_id;
       long status, aocs_model;
       status = xp_sat_nominal_att_get_aocs (&sat_nom_trans_id,
                                              &aocs_model);
}
```

7.6.3 Input parameters

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

Table 21: Input parameters of xp_sat_nominal_att_get_aocs function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_nom_trans_id	xp_sat_nom_trans_id *	-	Satellite nominal transformation ID.	-	-

7.6.4 Output parameters

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

Table 22: Output parameters of xp_sat_nominal_att_get_aocs function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_sat_nominal_att_get_aocs	long	-	Status flag	-	-
aocs_model	long	-	AOCS model	-	-

7.6.5 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 sat_nom_trans_id was not initialised.
- The sat_nom_trans_id initialization does not allow the use of this function.

7.7 xp_sat_nominal_att_set_aocs

7.7.1 Overview

The **xp_sat_nominal_att_set_aocs** CFI function changes the AOCS mode used for the satellite nominal attitude initialization.

7.7.2 Calling interface

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

```
#include <explorer_lib.h>
{
       xp_sat_nom_trans_id sat_nom_trans_id;
       long status, aocs_model;
       status = xp_sat_nominal_att_set_aocs (&sat_nom_trans_id,
                                             &aocs_model);
}
```

7.7.3 Input parameters

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

Table 23: Input parameters of xp_sat_nominal_att_set_aocs function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_nom_trans_id	xp_sat_nom_trans_id *	-	Satellite nominal transformation ID (input / output parameter)	-	-
aocs_model	long	-	AOCS model	-	-

7.7.4 Output parameters

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

Table 24: Output parameters of xp_sat_nominal_att_set_aocs function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_sat_nominal_att	long	-	Status flag	-	-

_set_aocs					
sat_nom_trans_id	xp_sat_nom_trans_id *	-	Satellite nominal transformation ID (input / output parameter)	-	-

7.7.5 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 sat_nom_trans_id was not initialised.
- The sat_nom_trans_id initialization does not allow the use of this function.

7.8 xp_sat_nominal_att_get_param

7.8.1 Overview

The **xp_sat_nominal_att_get_param** CFI function returns parameters used for the satellite nominal attitude initialization.

7.8.2 Calling interface

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

```
#include <explorer_lib.h>
{
      xp_sat_nom_trans_id sat_nom_trans_id;
      long status;
      xp_param_model_str data;
      status = xp_sat_nominal_att_get_param (&sat_nom_trans_id,
                                             &data);
}
```

7.8.3 Input parameters

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

Table 25: Input parameters of xp_sat_nominal_att_get_param function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_nom_trans_id	xp_sat_nom_trans_id *	-	Satellite nominal transformation ID.	-	-

7.8.4 Output parameters

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

Table 26: Output parameters of xp_sat_nominal_att_get_param function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_sat_nominal_att_get_param	long	-	Status flag	-	-

data	xp_param_model_str	-	Attitude initialization data	-	-

7.8.5 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 sat_nom_trans_id was not initialised.
- The sat_nom_trans_id initialization does not allow the use of this function.

7.9 xp_sat_nominal_att_set_param

7.9.1 Overview

The **xp_sat_nominal_att_set_param** CFI function changes the parameters used for the satellite nominal attitude initialization.

7.9.2 Calling interface

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

```
#include <explorer_lib.h>
{
        xp_sat_nom_trans_id sat_nom_trans_id;
        long status;
        xp_param_model_str data;
        status = xp_sat_nominal_att_set_param (&sat_nom_trans_id,
                                              &data);
}
```

7.9.3 Input parameters

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

Table 27: Input parameters of xp_sat_nominal_att_set_param function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_nom_trans_id	xp_sat_nom_trans_id *	-	Satellite nominal transformation ID (input / output parameter)	-	-
data	xp_param_model_str	-	Attitude initialization data	-	-

7.9.4 Output parameters

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

Table 28: Output parameters of xp_sat_nominal_att_set_param function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_sat_nominal_att_set_param	long	-	Status flag	-	-
sat_nom_trans_id	xp_sat_nom_trans_id *	-	Satellite nominal transformation ID (input / output parameter)	-	-

7.9.5 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 sat_nom_trans_id was not initialised.
- The sat_nom_trans_id initialization does not allow the use of this function.

7.10 xp_sat_nominal_att_get_harmonic

7.10.1 Overview

The **xp_sat_nominal_att_get_harmonic** CFI function returns harmonic data used for the satellite nominal attitude initialization.

7.10.2 Calling interface

The calling interface of the **xp_sat_nominal_att_get_harmonic** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_lib.h>
{
       xp_sat_nom_trans_id sat_nom_trans_id;
       long status;
       xp_harmonic_model_str data;
       status = xp_sat_nominal_att_get_harmonic (&sat_nom_trans_id,
                                                  &data);
}
```

7.10.3 Input parameters

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

Table 29: Input parameters of xp_sat_nominal_att_get_harmonic function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_nom_trans_id	xp_sat_nom_trans_id *	-	Satellite nominal transformation ID.	-	-

7.10.4 Output parameters

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

Table 30: Output parameters of xp_sat_nominal_att_get_harmonic function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_sat_nominal_att_get_harmonic	long	-	Status flag	-	-

data	xp_harmonic_model_str	-	Attitude initialization 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 sat_nom_trans_id was not initialised.
- The sat_nom_trans_id initialization does not allow the use of this function.

7.11 xp_sat_nominal_att_set_harmonic

7.11.1 Overview

The **xp_sat_nominal_att_set_harmonic** CFI function changes the harmonic data used for the satellite nominal attitude initialization.

7.11.2 Calling interface

The calling interface of the **xp_sat_nominal_att_set_harmonic** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_lib.h>
{
        xp_sat_nom_trans_id sat_nom_trans_id;
        long status;
        xp_harmonic_model_str data;
        status = xp_sat_nominal_att_set_harmonic (&sat_nom_trans_id,
                                                  &data);
}
```

7.11.3 Input parameters

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

Table 31: Input parameters of xp_sat_nominal_att_set_harmonic function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_nom_trans_id	xp_sat_nom_trans_id *	-	Satellite nominal transformation ID (input / output parameter)	-	-
data	xp_harmonic_model_str	-	Attitude initialization data	-	-

7.11.4 Output parameters

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

Table 32: Output parameters of xp_sat_nominal_att_set_harmonic function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_sat_nominal_att_set_harmonic	long	-	Status flag	-	-
sat_nom_trans_id	xp_sat_nom_trans_id *	-	Satellite nominal transformation ID (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 sat_nom_trans_id was not initialised.
- The sat_nom_trans_id initialization does not allow the use of this function.

7.12 xp_sat_nominal_att_get_file

7.12.1 Overview

The **xp_sat_nominal_att_get_file** CFI function returns initialization data from the satellite nominal attitude Id. when it was initialised with a file.

7.12.2 Calling interface

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

```
#include <explorer_lib.h>
{
        xp_sat_nom_trans_id sat_nom_trans_id;
        long status;
        xp_file_model_str data;
        status = xp_sat_nominal_att_get_file (&sat_nom_trans_id,
                                              &data);
}
```

7.12.3 Input parameters

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

Table 33: Input parameters of xp_sat_nominal_att_get_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_nom_trans_id	xp_sat_nom_trans_id *	-	Satellite nominal transformation ID.	-	-

7.12.4 Output parameters

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

Table 34: Output parameters of xp_sat_nominal_att_get_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_sat_nominal_att_get_file	long	-	Status flag	-	-

data	xp_file_model_str	-	Attitude initialization data	-	-

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 sat_nom_trans_id was not initialised.
- The sat_nom_trans_id initialization does not allow the use of this function.

7.13 xp_sat_nominal_att_set_file

7.13.1 Overview

The **xp_sat_nominal_att_set_file** CFI function changes the initialization data for the satellite nominal attitude Id, when it was initialised with a file.

If quaternions are introduced, it is checked that they are normalized.

7.13.2 Calling interface

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

```
#include <explorer_lib.h>
{
       xp_sat_nom_trans_id sat_nom_trans_id;
       long status;
       xp_file_model_str data;
       status = xp_sat_nominal_att_set_file (&sat_nom_trans_id,
                                             &data);
}
```

7.13.3 Input parameters

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

Table 35: Input parameters of xp_sat_nominal_att_set_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_nom_trans_id	xp_sat_nom_trans_id *	-	Satellite nominal transformation ID (input / output parameter)	-	-
data	xp_file_model_str	-	Attitude initialization data	-	-

7.13.4 Output parameters

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

Table 36: Output parameters of xp_sat_nominal_att_set_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_sat_nominal_att_set_file	long	-	Status flag	-	-
sat_nom_trans_id	xp_sat_nom_trans_id *	-	Satellite nominal transformation ID (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 sat_nom_trans_id was not initialised.
- The sat_nom_trans_id initialization does not allow the use of this function.

7.14 xp_sat_att_angle_init

7.14.1 Overview

The **xp_sat_att_angle_init** CFI function initialises the satellite nominal attitude to satellite attitude mispointing angles for a given satellite with a user-provided set of values. The initialised values will be stored in the *sat_trans_id* output structure.

7.14.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
    double ang[3];
    xp_sat_trans_id sat_trans_id = {NULL};
    long ierr[XP_NUM_ERR_MISP_ANGLE_INIT_DEF], status;

    status = xp_sat_att_angle_init(ang, &sat_trans_id, ierr);
}
```

The `XP_NUM_ERR_SAT_ATT_ANGLE_INIT` constant is defined in the file *explorer_pointing.h.*

7.14.3 Input Parameters

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

Table 37: Input parameters of xp_sat_att_angle_init function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
ang	double[3]	[0]	Pitch mispointing angle (Satellite Nominal Attitude Frame)	deg	If no better value, assume 0.0
		[1]	Roll mispointing angle (Satellite Nominal Attitude Frame)	deg	If no better value, assume 0.0
		[2]	Yaw mispointing angle (Satellite Nominal Attitude Frame)	deg	If no better value, assume 0.0

7.14.4 Output Parameters

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

Table 38: Output parameters of xp_sat_att_angle_init

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_trans_id	xp_sat_trans_id*	-	Structure that contains the Satellite Transformation	-	-
ierr	long	-	Error vector	-	-

7.14.5 Warnings and Errors

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

Table 39: Error messages of xp_sat_att_angle_init function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Memory allocation error	No calculation performed	XP_CFI_SAT_ATT_ANGLE_INIT_MEMORY_ERR	0

7.15 xp_sat_att_matrix_init

7.15.1 Overview

The **xp_sat_att_matrix_init** CFI function initializes misalignment matrix between the satellite nominal attitude frame and satellite attitude frame with a user-provided matrix. The initialized values will be stored in the *sat_trans_id* output structure. It is checked that the input matrix is orthonormal.

Note on matrix notation:

If XYZ are the axes of the original reference frame, and X'Y'Z' are the axes of the rotated frame, the rows of the rotation matrix are respectively X, Y and Z axes expressed in X'Y'Z' system.

In the C representation, M[0][], M[1][], M[2][] are respectively 1st, and and 3rd row of a rotation matrix M.

The rotation matrix M satisfies the following equivalence:

$$\mathbf{V} = M*\mathbf{V'}$$

where **V'** is a vector expressed in the X'Y'Z' reference system and **V** is expressed in the XYZ reference system.

7.15.2 Calling Interface

The calling interface of the **xp_sat_att_matrix_init** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
    double att_matrix[3][3];
    xp_sat_trans_id sat_trans_id = {NULL};
    long ierr[XP_NUM_ERR_SAT_ATT_MATRIX_INIT], status;

    status = xp_sat_att_matrix_init(att_matrix,
                                    &sat_trans_id, ierr);
}
```

The `XP_NUM_ERR_SAT_ATT_MATRIX_INIT` constant is defined in the file *explorer_pointing.h.*

7.15.3 Input Parameters

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

Table 40: Input parameters of xp_sat_att_matrix_init function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
att_matrix	double[3][3]	all	Mispointing Matrix	-	-

7.15.4 Output Parameters

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

Table 41: Output parameters of xp_sat_att_matrix_init

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_trans_id	xp_sat_trans_id*	-	Structure that contains the Satellite Transformation	-	-
ierr	long	-	Error vector	-	-

7.15.5 Example

TBD

7.15.6 Warnings and Errors

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

Table 42: Error messages of xp_sat_att_matrix_init function

Error	Error message	Cause and impact	Error code	Error

type				No
ERR	Memory allocation error	No calculation performed	XP_CFI_SAT_ATT_MATRIX_INIT_MEMORY_ERR	0
ERR	Matrix not orthonormal	No calculation performed The CFI performs a check, with a tolerance of 10^{-6}, that the product of the input matrix and its transposed is the unitary matrix	XP_CFI_SAT_ATT_MATRIX_INIT_ORTHONORMAL_ERR	1

7.16 xp_sat_att_init_harmonic

7.16.1 Overview

The **xp_sat_att_init_harmonic** CFI function initialises the satellite nominal orbital to satellite attitude mispointing angles (i.e. roll, pitch, yaw) for a given satellite with a user-provided set of values. The initialised values will be stored in the *sat_trans_id* output structure.

The mispointing angle (attitude angle in the formula) will be calculated by functions using such sat_trans_id (i.e. xp_attitude_compute or xp_change_frame) according to the following formula (the "angle" variable will be calculated as in xl_position_on_orbit (see [LIB_SUM]), using as inputs:

- the input state vector in EF passed to such functions;
- the angle_type passed as input to xp_sat_att_init_harmonic.

7.16.2 Calling Interface

The calling interface of the **xp_sat_att_init_harmonic** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
     long angle_type, num_terms[3];
     long harmonic_type_pitch[XP_MAX_NUM_HARMONIC],
           harmonic_type_roll[XP_MAX_NUM_HARMONIC],
           harmonic_type_yaw[XP_MAX_NUM_HARMONIC];
     double harmonic_coef_pitch[XP_MAX_NUM_HARMONIC],
          harmonic_coef_roll[XP_MAX_NUM_HARMONIC],
          harmonic_coef_yaw[XP_MAX_NUM_HARMONIC];
     xp_sat_trans_id sat_trans_id = {NULL};
     long ierr[XP_NUM_ERR_SAT_ATT_INIT_HARMONIC], status;

     status = xp_sat_att_init_harmonic(&angle_type, num_terms,
                                   harmonic_type_pitch,
                                   harmonic_type_roll,
                                   harmonic_type_yaw,
                                   harmonic_coef_pitch,
                                   harmonic_coef_roll,
                                   harmonic_coef_yaw,
                                   &sat_trans_id, ierr);
```

```
}
```

The `XP_NUM_ERR_SAT_ATT_INIT_HARMONIC` and `XP_MAX_NUM_HARMONIC` constants are defined in the file *explorer_pointing.h.*

7.16.3 Input Parameters

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

Table 43: Input parameters of xp_sat_att_init_harmonic function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
angle_type	long *	-	Type of angle	-	XP_ANGLE_TYPE_TRUE_LAT_TOD XP_ANGLE_TYPE_TRUE_LAT_EF
num_terms[3]	long	[0]	Number of elements used in vectors harmonic_type_pitch and harmonic_coef_pitch	-	>=0
		[1]	Number of elements used in vectors harmonic_type_roll and harmonic_coef_roll	-	>=0
		[2]	Number of elements used in vectors harmonic_type_yaw and harmonic_coef_yaw	-	>=0
harmonic_type_pitch	long[XP_MAX_NUM_HARMONIC]	all	Type of coefficients: =0 for the bias parameter <0 for the sinus coefficients (-n means that correponds to the sinus coefficient of order n) >0 for the cosinus coefficients (+n means that correponds to the cosinus coefficient of order n)	-	
harmonic_type_roll	long[XP_MAX_NUM_HARMONIC]	all	Type of coefficients: =0 for the bias parameter <0 for the sinus coefficients (-n means that correponds to the sinus coefficient of order n) >0 for the cosinus coefficients (+n means that correponds to the cosinus coefficient of order n)	-	-
harmonic_type_yaw	long[XP_MAX_NUM_HARMONIC]	all	Type of coefficients: =0 for the bias parameter <0 for the sinus coefficients (-n means that correponds to the sinus coefficient of order n)	-	-

			>0 for the cosinus coefficients (+n means that correponds to the cosinus coefficient of order n)		
harmonic_coef_pitch	double[XP_MAX_NUM_HARMONIC]	all	Bias, sinus and cosinus coeficients for the pitch angle	deg	
harmonic_coef_roll	double[XP_MAX_NUM_HARMONIC]	all	Bias, sinus and cosinus coeficients for the roll angle	deg	
harmonic_coef_yaw	double[XP_MAX_NUM_HARMONIC]	all	Bias, sinus and cosinus coeficients for the yaw angle	deg	

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

- Angle Type: See current document, Table 3.

7.16.4 Output Parameters

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

Table 44: Output parameters of xp_sat_att_init_harmonic

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_trans_id	xp_sat_trans_id*	-	Structure that contains the Satellite Transformation	-	-
ierr	long	-	Error vector	-	-

7.16.5 Warnings and Errors

Next table lists the possible error messages that can be returned by the **xp_sat_att_init_harmonic** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_POINTING software library **xp_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 by translating the error vector returned by the **xp_sat_att_init_harmonic** function by calling the function of the EO_POINTING software library **xp_get_code** (see [GEN_SUM])

Table 45: Error messages of xp_sat_att_init_harmonic function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Memory allocation error	No calculation performed	XP_CFI_SAT_ATT_INIT_HARMONIC_MEMORY_ERR	0

7.17 xp_sat_att_init_file

7.17.1 Overview

The **xp_sat_att_init_file** CFI function initialises the satellite attitude rotation for a given satellite reading values from the attitude file(s). The validity time or orbital range for the attitude angles can be specified by the user. The initialised values will be stored in the *sat_trans_id* output structure. The quaternions that could be read from the file are checked to be normalized.

The computed rotation will be given between the inertial reference frame specified in the file (tag <Reference_Frame>) and the satellite attitude frame. In case that the files contain rotation angles (pitch, roll, yaw) and if the reference frame is based on the satellite (“SATELLITE_ACTUAL”, “SATELLITE” or “SATELLITE_RELATIVE”), then the rotation will be defined between the satellite nominal attitude frame and the satellite attitude frame.

In order to read files, xp_sat_att_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.

7.17.1.1 Initialisation with several files.

When the attitude frame is initialized with more than one file (see parameter *attitude_file* in next section), the user has to provide the files sorted from lower to higher precedence. This way the possible overlap between files is solved as follows:

attitude_file[0] is the file with lower precedence

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

the attitude data (angles or quaternions) from files with lower precedence that are after the data of a file with higher precedence are skipped.

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

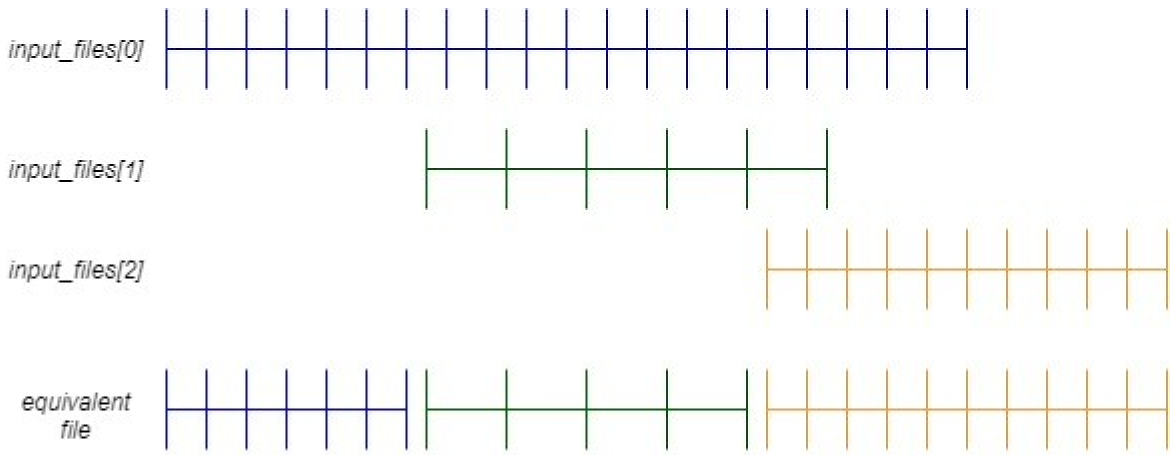


Figure 8: Overlapping attitude files

7.17.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     xl_time_id time_id = {NULL};
     long n_files, time_init_mode, time_ref;
     char **attitude_file *auxiliary_file;
     double time0, time1;
     double val_time0, val_time1;
     xp_sat_trans_id sat_trans_id = {NULL};
     long ierr[XP_NUM_ERR_SAT_ATT_INIT_FILE], status;

     status = xp_sat_att_init_file(&time_id, &n_files,
                  attitude_file, auxiliary_file,
                  &time_init_mode, &time_ref, &time0, &time1,
                  &val_time0, &val_time1, &sat_trans_id, ierr);
}
```

The `XP_NUM_ERR_SAT_ATT_INIT_FILE` constant is defined in the file *explorer_pointing.h.*

7.17.3 Input Parameters

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

Table 46: Input parameters of xp_sat_att_init_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
time_id	xl_time_id*	-	Structure that contains the time correlations.	-	-
n_files	long *	-	Number of reference data files	-	> 0
attitude_file	char **	-	Filenames of the reference data files. In case multiple files are used, the files should be time ordered and all files should be of the same type (see section 7.17.1.1). The supported Attitude Files are: - Generic Attitude Files - Star Tracker files., The function assumes that all the input files belong to the same Star- Tracker. As a consequence of this assumption only the Star-Tracker identifier of the first file provided in the list is read. Note that the Star-Tracker identification number should be either 1, 2 or 3 (no inter nal check is performed) - Attitude Ephemeris Message Files. In this case only one file is allowed and only the first segment of the file is taken for the initalisation. Those files are described in [D_H_SUM] .	-	-
auxiliary_file	char **	-	Filename of an auxiliary file con taining the Star-Tracker misaligne ment matrices	-	-
time_init_mode	long *	-	Flag for selecting the time range of the initialization.	-	Select either: · XP_SEL_TIME · XP_SEL_FILE
time_ref	long *	-	Time reference ID	-	Complete
time0	double*	-	If: *time_init_mode=XP_SEL_TIME* Start of the time range defined by [time0,time1]	Decimal days (Process	[-18262.0,36524.0]

				ing format)	
time1	double*	-	If: *time_init_mode=XP_SEL_TIME* End of the time range defined by [time0,time1]	Decimal days (Process ing format)	[-18262.0,36524.0] > time0

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].
- Time Init Mode ID: time_init_mode. See current document, Table 3.

7.17.4 Output Parameters

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

Table 47: Output parameters of xp_sat_att_init_file

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
val_time0	double*	-	Validity start time of the initialization	Decimal days (Processing format)	[-18262.0,36524.0]
val_time1	double*	-	Validity end time of the initialization	Decimal days (Processing format)	[-18262.0,36524.0]
sat_trans_id	xp_sat_trans_id*	-	Structure that contains the Satellite Transformation	-	-
ierr	long	-	Error vector	-	-

7.17.5 Warnings and Errors

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

Table 48: Error messages of xp_sat_att_init_file function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Memory allocation error	No calculation performed	XP_CFI_SAT_ATT_INIT_FILE_MEMORY_ERR	0

ERR	Error opening attitude file: %s	No calculation performed	XP_CFI_SAT_ATT_INIT_FILE_OPEN_FILES_ERR	1
ERR	Error reading input star tracker files	No calculation performed	XP_CFI_SAT_ATT_INIT_FILE_READ_FILES_ERR	2
ERR	Error reading generic attitude files	No calculation performed	XP_CFI_SAT_ATT_INIT_FILE_READ_ATT_FILES_ERR	3
ERR	No data has been read from the files	No calculation performed	XP_CFI_SAT_ATT_INIT_FILE_NO_READ_DATA_ERR	4
ERR	Error reading auxiliary file	No calculation performed	XP_CFI_SAT_ATT_INIT_FILE_READ_AUX_FILE_ERR	5
ERR	Wrong input time reference	No calculation performed	XP_CFI_SAT_ATT_INIT_FILE_WRONG_TIME_REF_ERR	6
ERR	Could not perform a time transformation	No calculation performed	XP_CFI_SAT_ATT_INIT_FILE_TIME_REF_ERR	7
ERR	Could not find word "SPH_DESCRIPTOR" in attitude file	No calculation performed	XP_CFI_SAT_ATT_INIT_FILE_READ_STR_ID_ERR	8
ERR	Quaternion is not normalized	No calculation performed	XP_CFI_SAT_ATT_INIT_FILE_QUAT_UNITARY_ERR	9
WARN	AEM with more than 1 segment used for initialization. Only 1st segment will be used	Attitude is initialised using the 1st segment in the AEM file.	XP_CFI_SAT_ATT_INIT_FILE_ONLY_SEGMENT0_WARN	10

7.18 xp_sat_att_quat_plus_matrix_init

7.18.1 Overview

The **xp_sat_att_quat_plus_matrix_init** CFI function initialises the satellite attitude angles using the input quaternions, and stores the rotation matrix from the satellite-based reference frame defined by the quaternions to the satellite frame, that must be provided by the user. The initialised values will be stored in the *sat_trans_id* output structure. The input quaternions are checked to be normalized, and the input matrix is checked to be orthonormal.

Note on matrix notation:

If XYZ are the axes of the original reference frame, and X'Y'Z' are the axes of the rotated frame, the rows of the rotation matrix are respectively X, Y and Z axes expressed in X'Y'Z' system.

In the C representation, M[0][], M[1][], M[2][] are respectively 1st, and and 3rd row of a rotation matrix M.

The rotation matrix M satisfies the following equivalence:

$$\mathbf{V} = M*\mathbf{V'}$$

where **V’** is a vector expressed in the X’Y’Z’ reference system and **V** is expressed in the XYZ reference system.

7.18.2 Calling Interface

The calling interface of the **xp_sat_att_quat_plus_matrix_init** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
      long inertial_frame;
      long num_rec;
      xd_att_rec *quaternions;
      double **matrix;
      xp_sat_trans_id sat_trans_id = {NULL};
      long ierr[XP_NUM_ERR_SAT_ATT_QUAT_PLUS_MATRIX_INIT], status;

      status = xp_sat_att_quat_plus_matrix_init( &inertial_frame,
                  &num_rec, quaternions, matrix, sat_trans_id, ierr);
}
```

The `XP_NUM_ERR_SAT_ATT_QUAT_PLUS_MATRIX_INIT` constant is defined in the file *explorer_pointing.h.*

7.18.3 Input Parameters

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

Table 49: Input parameters of xp_sat_att_quat_plus_matrix_init function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
inertial_frame	long*	-	Inertial reference frame.	-	Defined in XD_Cs_enum
num_rec	long *	-	Number of quaternions	-	> 0
quaternions	xd_att_rec *	-	Quaternions that give the rotation from the inertial reference frame to the frame based on the satellite.	-	-
matrix	double**	-	Rotation matrix from the frame based on the satellite to the satel lite frame.	-	-

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

- Inertial frame. See [D_H_SUM].

7.18.4 Output Parameters

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

Table 50: Output parameters of xp_sat_att_quat_plus_matrix_init

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_trans_id	xp_sat_trans _id*	-	Structure that contains the Satellite Transformation	-	-
ierr	long	-	Error vector	-	-

7.18.5 Warnings and Errors

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

Table 51: Error messages of xp_sat_att_quat_plus_matrix_init function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Memory allocation error	No calculation performed	XP_CFI_SAT_ATT_QUAT_PLUS_MATRIX_INIT_MEMORY_ERR	0
ERR	Quaternion is not normalized	No calculation performed	XP_CFI_SAT_ATT_QUAT_PLUS_MATRIX_INIT_QUAT_UNITARY_ERR	1
ERR	Matrix is not orthonormal	No calculation performed. The CFI performs a check, with a tolerance of 10^{-6}, that the product of the input matrix and its transposed is the unitary matrix.	XP_CFI_SAT_ATT_QUAT_PLUS_MATRIX_INIT_MATRIX_ORTHONORMAL_ERR	2

7.19 xp_sat_att_quat_plus_angle_init

7.19.1 Overview

The **xp_sat_att_quat_plus_angle_init** CFI function initialises the satellite attitude angles using the input quaternions, and stores the rotation matrix from the satellite-based reference frame defined by the quaternions to the satellite frame, calculated with the input angles. The initialised values will be stored in the *sat_trans_id* output structure. The input quaternions are checked to be normalized.

7.19.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     long inertial_frame;
     long num_rec;
     xd_att_rec *quaternions;
     double angles[3];
     xp_sat_trans_id sat_trans_id = {NULL};
     long ierr[XP_NUM_ERR_SAT_ATT_QUAT_PLUS_ANGLE_INIT], status;

     status = xp_sat_att_quat_plus_angle_init( &inertial_frame,
              &num_rec, quaternions, angles, sat_trans_id, ierr);
}
```

The XP_NUM_ERR_SAT_ATT_QUAT_PLUS_ANGLE_INIT constant is defined in the file *explorer_pointing.h.*

7.19.3 Input Parameters

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

Table 52: Input parameters of xp_sat_att_quat_plus_matrix_init function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
inertial_frame	long*	-	Inertial reference frame.	-	Defined in XD_Cs_enum
num_rec	long *	-	Number of quaternions	-	> 0
quaternions	xd_att_rec *	-	Quaternions that give the rotation from the inertial reference frame to the frame based on the satellite.	-	-
angles	double[3]	-	Angles that define the rotation from the frame based on the satellite to the satellite frame.	-	-

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

- Inertial frame. See [D_H_SUM].

7.19.4 Output Parameters

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

Table 53: Output parameters of xp_sat_att_quat_plus_angle_init

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_trans_id	xp_sat_trans_id*	-	Structure that contains the Satellite Transformation	-	-
ierr	long	-	Error vector	-	-

7.19.5 Warnings and Errors

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

Table 54: Error messages of xp_sat_att_quat_plus_angle_init function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Memory allocation error	No calculation performed	XP_CFI_SAT_ATT_QUAT_PLUS_ANGLE_INIT_MEMORY_ERR	0
ERR	Error calculating rotation matrix for eurler angles	No calculation performed	XP_CFI_SAT_ATT_QUAT_PLUS_ANGLE_EULER_TO_MATRIX_ERR	1
ERR	Quaternion is not normalized	No calculation performed The CFI performs a check, with a tolerance of 10^{-6}, that the product of the input matrix and its transposed is the unitary matrix.	XP_SAT_ATT_QUAT_PLUS_ANGLE_INIT_QUAT_UNITARY_ERR	2

7.20 xp_sat_att_close

7.20.1 Overview

The **xp_sat_att_close** CFI function cleans up any memory allocation performed by the satellite attitude initialization functions.

7.20.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     xp_sat_trans_id sat_trans_id = {NULL};
     long ierr[XP_NUM_ERR_SAT_ATT_CLOSE], status;

     status = xp_sat_att_close(&sat_trans_id, ierr);
}
```

The `XP_NUM_ERR_SAT_ATT_CLOSE` constant is defined in the file *explorer_pointing.h.*

7.20.3 Input Parameters

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

Table 55: Input parameters of xp_sat_att_close function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_trans_id	xp_sat_trans _id*	-	Structure that contains the Sat. Trans.	-	-

7.20.4 Output Parameters

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

Table 56: Output parameters of xp_sat_att_close

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

7.20.5 Warnings and Errors

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

Table 57: Error messages of xp_sat_att_close function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Could not close the Id. as it is not initialized or it is being used	No calculation performed	XP_CFI_SAT_ATT_CLOSE_W RONG_ID_ERR	0

7.21 xp_sat_att_get_angles

7.21.1 Overview

The **xp_sat_att_get_angles** CFI function returns angle data used for the satellite attitude initialization.

7.21.2 Calling interface

The calling interface of the **xp_sat_att_get_angles** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_lib.h>
{
        xp_sat_trans_id sat_trans_id;
        long status;
        xp_angle_model_str data;
        status = xp_sat_att_get_angles (&sat_trans_id,
                                        &data);
}
```

7.21.3 Input parameters

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

Table 58: Input parameters of xp_sat_att_get_angles function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_trans_id	xp_sat_trans_id *	-	Satellite transformation ID.	-	-

7.21.4 Output parameters

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

Table 59: Output parameters of xp_sat_att_get_angles function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_sat_att_get_angles	long	-	Status flag	-	-
data	xp_angle_model_str	-	Attitude initialization data	-	-

7.21.5 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 sat_trans_id was not initialised.
- The sat_trans_id initialization does not allow the use of this function.

7.22 xp_sat_att_set_angles

7.22.1 Overview

The **xp_sat_att_set_angles** CFI function changes the harmonic data used for the satellite attitude initialization.

7.22.2 Calling interface

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

```
#include <explorer_lib.h>
{
        xp_sat_trans_id sat_trans_id;
        long status;
        xp_angle_model_str data;
        status = xp_sat_att_set_angles (&sat_trans_id,
                                        &data);
}
```

7.22.3 Input parameters

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

Table 60: Input parameters of xp_sat_att_set_angles function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_trans_id	xp_sat_trans_id *	-	Satellite transformation ID (input / output parameter)	-	-
data	xp_angle_model_str	-	Attitude initialization data	-	-

7.22.4 Output parameters

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

Table 61: Output parameters of xp_sat_att_set_angles function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_sat_att_set_angles	long	-	Status flag	-	-
sat_trans_id	xp_sat_trans_id *	-	Satellite transformation ID (input / output parameter)	-	-

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 sat_trans_id was not initialised.
- The sat_trans_id initialization does not allow the use of this function.

7.23 xp_sat_att_get_matrix

7.23.1 Overview

The **xp_sat_att_get_matrix** CFI function returns the matrix data used for the satellite attitude initialization.

Note on matrix notation:

If XYZ are the axes of the original reference frame, and X'Y'Z' are the axes of the rotated frame, the rows of the rotation matrix are respectively X, Y and Z axes expressed in X'Y'Z' system.

In the C representation, M[0][], M[1][], M[2][] are respectively 1st, and and 3rd row of a rotation matrix M.

The rotation matrix M satisfies the following equivalence:

$$\mathbf{V} = M*\mathbf{V'}$$

where **V'** is a vector expressed in the X'Y'Z' reference system and **V** is expressed in the XYZ reference system.

7.23.2 Calling interface

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

```
#include <explorer_lib.h>
{
       xp_sat_trans_id sat_trans_id;
       long status;
       xp_matrix_model_str data;
       status = xp_sat_att_get_matrix (&sat_trans_id,
                                       &data);
}
```

7.23.3 Input parameters

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

Table 62: Input parameters of xp_sat_att_get_matrix function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_trans_id	xp_sat_trans_id *	-	Satellite transformation ID.	-	-

7.23.4 Output parameters

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

Table 63: Output parameters of xp_sat_att_get_matrix function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_sat_att_get_matrix	long	-	Status flag	-	-
data	xp_matrix_model_str	-	Attitude initialization data	-	-

7.23.5 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 sat_trans_id was not initialised.
- The sat_trans_id initialization does not allow the use of this function.

7.24 xp_sat_att_set_matrix

7.24.1 Overview

The **xp_sat_att_set_matrix** CFI function changes matrix data used for the satellite attitude initialization. It is checked that the input matrix is orthonormal.

Note on matrix notation:

If XYZ are the axes of the original reference frame, and X'Y'Z' are the axes of the rotated frame, the rows of the rotation matrix are respectively X, Y and Z axes expressed in X'Y'Z' system.

In the C representation, M[0][], M[1][], M[2][] are respectively 1st, and and 3rd row of a rotation matrix M.

The rotation matrix M satisfies the following equivalence:

$$\mathbf{V} = M*\mathbf{V'}$$

where **V’** is a vector expressed in the X’Y’Z’ reference system and **V** is expressed in the XYZ reference system.

7.24.2 Calling interface

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

```
#include <explorer_lib.h>
{
        xp_sat_trans_id sat_trans_id;
        long status;
        xp_matrix_model_str data;
        status = xp_sat_att_set_matrix (&sat_trans_id,
                                        &data);
}
```

7.24.3 Input parameters

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

Table 64: Input parameters of xp_sat_att_set_matrix function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_trans_id	xp_sat_trans_id *	-	Satellite transformation ID (input / output parameter)	-	-
data	xp_angle_mod	-	Attitude initialization	-	-

	el_str		data		

7.24.4 Output parameters

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

Table 65: Output parameters of xp_sat_att_set_matrix function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_sat_att_set_matrix	long	-	Status flag	-	-
sat_trans_id	xp_sat_trans_id *	-	Satellite transformation ID (input / output parameter)	-	-

7.24.5 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 sat_trans_id was not initialised.
- The sat_trans_id initialization does not allow the use of this function.

7.25 xp_sat_att_get_harmonic

7.25.1 Overview

The **xp_sat_att_get_harmonic** CFI function returns harmonic data used for the satellite attitude initialization.

7.25.2 Calling interface

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

```
#include <explorer_lib.h>
{
        xp_sat_trans_id sat_trans_id;
        long status;
        xp_harmonic_model_str data;
        status = xp_sat_att_get_harmonic (&sat_trans_id,
                                          &data);
}
```

7.25.3 Input parameters

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

Table 66: Input parameters of xp_sat_att_get_harmonic function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_trans_id	xp_sat_trans_id *	-	Satellite transformation ID.	-	-

7.25.4 Output parameters

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

Table 67: Output parameters of xp_sat_att_get_harmonic function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_sat_att_get_harmonic	long	-	Status flag	-	-

data	xp_harmonic_model_str	-	Attitude initialization data	-	-

7.25.5 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 sat_trans_id was not initialised.
- The sat_trans_id initialization does not allow the use of this function.

7.26 xp_sat_att_set_harmonic

7.26.1 Overview

The **xp_sat_att_set_harmonic** CFI function changes the harmonic data used for the satellite attitude initialization.

7.26.2 Calling interface

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

```
#include <explorer_lib.h>
{
        xp_sat_trans_id sat_trans_id;
        long status;
        xp_harmonic_model_str data;
        status = xp_sat_att_set_harmonic (&sat_trans_id,
                                          &data);
}
```

7.26.3 Input parameters

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

Table 68: Input parameters of xp_sat_att_set_harmonic function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_trans_id	xp_sat_trans_id *	-	Satellite transformation ID (input / output parameter)	-	-
data	xp_harmonic_model_str	-	Attitude initialization data	-	-

7.26.4 Output parameters

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

Table 69: Output parameters of xp_sat_att_set_harmonic function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_sat_att_set_harmonic	long	-	Status flag	-	-
sat_trans_id	xp_sat_trans_id *	-	Satellite transformation ID (input / output parameter)	-	-

7.26.5 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 sat_trans_id was not initialised.
- The sat_trans_id initialization does not allow the use of this function.

7.27 xp_sat_att_get_file

7.27.1 Overview

The **xp_sat_att_get_file** CFI function returns satellite attitude data from the satellite attitude Id. that was initialized with a file.

7.27.2 Calling interface

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

```
#include <explorer_lib.h>
{
       xp_sat_trans_id sat_trans_id;
       long status;
       xp_sat_att_file_model_str data;
       status = xp_sat_att_get_file (&sat_trans_id,
                                     &data);
}
```

7.27.3 Input parameters

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

Table 70: Input parameters of xp_sat_att_get_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_trans_id	xp_sat_trans_id *	-	Satellite transformation ID.	-	-

7.27.4 Output parameters

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

Table 71: Output parameters of xp_sat_att_get_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_sat_att_get_file	long	-	Status flag	-	-
data	xp_sat_att_file_m	-	Attitude initialization	-	-

	odel_str		data		

7.27.5 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 sat_trans_id was not initialised.
- The sat_trans_id initialization does not allow the use of this function.

7.28 xp_sat_att_set_file

7.28.1 Overview

The **xp_sat_att_set_file** CFI function changes the initalization data in the satellite attidude Id. when it was initialised with a file. Quaternions are checked to be normalized.

7.28.2 Calling interface

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

```
#include <explorer_lib.h>
{
       xp_sat_trans_id sat_trans_id;
       long status;
       xp_sat_att_file_model_str data;
       status = xp_sat_att_set_file (&sat_trans_id,
                                     &data);
}
```

7.28.3 Input parameters

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

Table 72: Input parameters of xp_sat_att_set_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_trans_id	xp_sat_trans_id *	-	Satellite transformation ID (input / output parameter)	-	-
data	xp_sat_att_file_model_str	-	Attitude initialization data	-	-

7.28.4 Output parameters

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

Table 73: Output parameters of xp_sat_att_set_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_sat_att_set_file	long	-	Status flag	-	-
sat_trans_id	xp_sat_trans_id *	-	Satellite transformation ID (input / output parameter)	-	-

7.28.5 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 sat_trans_id was not initialised.
- The sat_trans_id initialization does not allow the use of this function.

7.29 xp_sat_att_get_quat_plus_angle

7.29.1 Overview

The **xp_sat_att_get_quat_plus_angle** CFI function returns satellite attitude data from the satellite attitude Id. that was initialized with quaternions and angles.

7.29.2 Calling interface

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

```
#include <explorer_lib.h>
{
        xp_sat_trans_id sat_trans_id;
        long status;
        xp_quat_plus_angle_model_str data;
        status = xp_sat_att_get_quat_plus_angle (&sat_trans_id,
                                               &data);
}
```

7.29.3 Input parameters

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

Table 74: Input parameters of xp_sat_att_get_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_trans_id	xp_sat_trans_id *	-	Satellite transformation ID.	-	-

7.29.4 Output parameters

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

Table 75: Output parameters of xp_sat_att_get_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_sat_att_get_quat_plus_angle	long	-	Status flag	-	-

data	xp_quat_plus_angle_model_str	-	Attitude initialization data	-	-

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 sat_trans_id was not initialised.
- The sat_trans_id initialization does not allow the use of this function.
- There was an error in the calculation of the angles from the rotation matrix.

7.30 xp_sat_att_set_quat_plus_angle

7.30.1 Overview

The **xp_sat_att_set_quat_plus_angle** CFI function changes the initalization data in the satellie attidude Id. when it was initialised with quaternions and angles. The input quaternions are checked to be normalized.

7.30.2 Calling interface

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

```
#include <explorer_lib.h>
{
        xp_sat_trans_id sat_trans_id;
        long status;
        xp_quat_plus_angle_model_str data;
        status = xp_sat_att_set_quat_plus_angle (&sat_trans_id,
                                               &data);
}
```

7.30.3 Input parameters

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

Table 76: Input parameters of xp_sat_att_set_quat_plus_angle function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_trans_id	xp_sat_trans_id *	-	Satellite transformation ID (input / output parameter)	-	-
data	xp_quat_plus_angle_model_str	-	Attitude initialization data	-	-

7.30.4 Output parameters

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

Table 77: Output parameters of xp_sat_att_set_quat_plus_angle function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_sat_att_set_quat_plus_angle	long	-	Status flag	-	-
sat_trans_id	xp_sat_trans_id *	-	Satellite transformation ID (input / output parameter)	-	-

7.30.5 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 sat_trans_id was not initialised.
- The sat_trans_id initialization does not allow the use of this function.
- There was an error in the calculation of the rotation matrix from angles.

7.31 xp_sat_att_get_quat_plus_matrix

7.31.1 Overview

The **xp_sat_att_get_quat_plus_matrix** CFI function returns satellite attitude data from the satellite attitude Id. that was initialized with quaternions and a rotation matrix.

Note on matrix notation:

If XYZ are the axes of the original reference frame, and X'Y'Z' are the axes of the rotated frame, the rows of the rotation matrix are respectively X, Y and Z axes expressed in X'Y'Z' system.

In the C representation, M[0][], M[1][], M[2][] are respectively 1st, and and 3rd row of a rotation matrix M.

The rotation matrix M satisfies the following equivalence:

$$\mathbf{V} = M*\mathbf{V'}$$

where **V'** is a vector expressed in the X'Y'Z' reference system and **V** is expressed in the XYZ reference system.

7.31.2 Calling interface

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

```
#include <explorer_lib.h>
{
       xp_sat_trans_id sat_trans_id;
       long status;
       xp_quat_plus_matrix_model_str data;
       status = xp_sat_att_get_quat_plus_matrix (&sat_trans_id,
                                                   &data);
}
```

7.31.3 Input parameters

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

Table 78: Input parameters of xp_sat_att_get_quat_plus_matrix function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_trans_id	xp_sat_trans_id *	-	Satellite transformation ID.	-	-

7.31.4 Output parameters

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

Table 79: Output parameters of xp_sat_att_get_quat_plus_matrix function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_sat_att_get_quat_plus_matrix	long	-	Status flag	-	-
data	xp_quat_plus_matrix_model_str	-	Attitude initialization data	-	-

7.31.5 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 sat_trans_id was not initialised.
- The sat_trans_id initialization does not allow the use of this function.

7.32 xp_sat_att_set_quat_plus_matrix

7.32.1 Overview

The **xp_sat_att_set_quat_plus_matrix** CFI function changes the initalization data in the satellie attitude Id. when it was initialised with quaternions and a rotation matrix. The input quaternions are checked to be normalized, and the input matrix is checked to be orthonormal.

Note on matrix notation:

If XYZ are the axes of the original reference frame, and X'Y'Z' are the axes of the rotated frame, the rows of the rotation matrix are respectively X, Y and Z axes expressed in X'Y'Z' system.

In the C representation, M[0][], M[1][], M[2][] are respectively 1st, and and 3rd row of a rotation matrix M.

The rotation matrix M satisfies the following equivalence:

$$\mathbf{V} = M*\mathbf{V'}$$

where **V’** is a vector expressed in the X’Y’Z’ reference system and **V** is expressed in the XYZ reference system.

7.32.2 Calling interface

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

```
#include <explorer_lib.h>
{
       xp_sat_trans_id sat_trans_id;
       long status;
       xp_quat_plus_matrix_model_str data;
       status = xp_sat_att_set_quat_plus_matrix (&sat_trans_id,
                                                 &data);
}
```

7.32.3 Input parameters

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

Table 80: Input parameters of xp_sat_att_set_quat_plus_matrix function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_trans_id	xp_sat_trans_id *	-	Satellite transformation ID (input / output parameter)	-	-

data	xp_quat_plus_matrix_model_str	-	Attitude initialization data	-	-

7.32.4 Output parameters

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

Table 81: Output parameters of xp_sat_att_get_quat_plus_matrix function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_sat_att_set_quat_plus_matrix	long	-	Status flag	-	-
sat_trans_id	xp_sat_trans_id *	-	Satellite transformation ID (input / output parameter)	-	-

7.32.5 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 sat_trans_id was not initialised.
- The sat_trans_id initialization does not allow the use of this function.

7.33 xp_instr_att_angle_init

7.33.1 Overview

The **xp_instr_att_angle_init** CFI function initialises the instrument attitude mispointing angles for a given satellite and instrument with a user-provided set of values. The initialised values will be stored in the *instr_trans_id* output structure.

7.33.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
    double ang[3], offset[3];
    xp_instr_trans_id instr_trans_id = {NULL};
    long ierr[XP_NUM_ERR_INSTR_ATT_ANGLE_INIT], status;

    status = xp_instr_att_angle_init(ang, offset,
                                     &instr_trans_id, ierr);
}
```

The `XP_NUM_ERR_INSTR_ATT_ANGLE_INIT` constant is defined in the file *explorer_pointing.h.*

7.33.3 Input Parameters

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

Table 82: Input parameters of xp_instr_att_angle_init function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
ang	double[3]	[0]	Pitch mispointing angle (Satellite Attitude Frame)	deg	If no better value, assume 0.0
		[1]	Roll mispointing angle (Satellite Attitude Frame)	deg	If no better value, assume 0.0
		[2]	Yaw mispointing angle (Satellite Attitude Frame)	deg	If no better value, assume 0.0
offset	double[3]	all	Instrument Frame Origin position vector (Satellite Attitude Frame)	m	-

7.33.4 Output Parameters

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

Table 83: Output parameters of xp_instr_att_angle_init

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
instr_trans_id	xp_instr_trans_id*	-	Structure that contains the Instrument Transformation	-	-
ierr	long	-	Error vector	-	-

7.33.5 Warnings and Errors

Next table lists the possible error messages that can be returned by the **xp_instr_att_angle_init** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_POINTING software library **xp_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 by translating the error vector returned by the **xp_instr_att_angle_init** function by calling the function of the EO_POINTING software library **xp_get_code** (see [GEN_SUM])

Table 84: Error messages of xp_instr_att_angle_init function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Memory allocation error	No calculation performed	XP_CFI_INSTR_ATT_ANGLE_INIT_MEMORY_ERR	0

7.34 xp_instr_att_matrix_init

7.34.1 Overview

The **xp_instr_att_matrix_init** CFI function initialises the instrument attitude mispointing angles for a given satellite and instrument with a user-provided matrix. The initialised values will be stored in the *instr_trans_id* output structure. Input matrix is checked to be orthonormal.

Note on matrix notation:

If XYZ are the axes of the original reference frame, and X'Y'Z' are the axes of the rotated frame, the rows of the rotation matrix are respectively X, Y and Z axes expressed in X'Y'Z' system.

In the C representation, M[0][], M[1][], M[2][] are respectively 1st, and and 3rd row of a rotation matrix M.

The rotation matrix M satisfies the following equivalence:

$$\mathbf{V} = M*\mathbf{V'}$$

where **V’** is a vector expressed in the X’Y’Z’ reference system and **V** is expressed in the XYZ reference system.

7.34.2 Calling Interface

The calling interface of the **xp_instr_att_matrix_init** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
     double att_matrix[3][3], offset[3];
     xp_instr_trans_id instr_trans_id = {NULL};
     long ierr[XP_NUM_ERR_INSTR_ATT_MATRIX_INIT], status;

     status = xp_instr_att_matrix_init(att_matrix, offset,
                                       &instr_trans_id, ierr);
}
```

The `XP_NUM_ERR_INSTR_ATT_MATRIX_INIT` constant is defined in the file *explorer_pointing.h.*

7.34.3 Input Parameters

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

Table 85: Input parameters of xp_instr_att_matrix_init function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
att_matrix	double[3][3]	all	Mispointing Matrix	-	-
offset	double[3]	all	Instrument Frame Origin posi tion vector (Satellite Attitude Frame)	m	-

7.34.4 Output Parameters

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

Table 86: Output parameters of xp_instr_att_matrix_init

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
instr_trans_id	xp_instr_trans _id*	-	Structure that contains the Instrument Transformation	-	-
ierr	long	-	Error vector	-	-

7.34.5 Warnings and Errors

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

Table 87: Error messages of xp_instr_att_matrix_init function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Memory allocation error	No calculation performed	XP_CFI_INSTR_ATT_MATRIX _INIT_MEMORY_ERR	0

ERR	Matrix is not orthonormal	No calculation performed. The CFI performs a check, with a tolerance of 10^{-6}, that the product of the input matrix and its transposed is the unitary matrix.	XP_CFI_INSTR_ATT_MATRIX_INIT_MATRIX_ORTHONORMAL_ERR	1

7.35 xp_instr_att_init_harmonic

7.35.1 Overview

The **xp_instr_att_init_harmonic** CFI function initialises the instrument attitude mispointing angles (i.e. roll, pitch, yaw) for a given satellite and instrument with a user-provided set of values. The initialised values will be stored in the *instr_trans_id* output structure.

The mispointing angle (attitude angle in the formula) will be calculated by functions using such instr_trans_id (i.e. xp_attitude_compute or xp_change_frame) according to the following formula (the "angle" variable will be calculated as in xl_position_on_orbit (see [LIB_SUM]), using as inputs:

- the input state vector in EF passed to such functions;

- the angle_type passed as input to xp_instr_att_init_harmonic.

7.35.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     long angle_type, num_terms[3];
     long harmonic_type_pitch[XP_MAX_NUM_HARMONIC],
          harmonic_type_roll[XP_MAX_NUM_HARMONIC],
          harmonic_type_yaw[XP_MAX_NUM_HARMONIC];
     double harmonic_coef_pitch[XP_MAX_NUM_HARMONIC],
         harmonic_coef_roll[XP_MAX_NUM_HARMONIC],
         harmonic_coef_yaw[XP_MAX_NUM_HARMONIC];
     double offset[3];
     xp_instr_trans_id instr_trans_id = {NULL};
     long ierr[XP_NUM_ERR_INSTR_ATT_INIT_HARMONIC], status;

     status = xp_instr_att_init_harmonic(&angle_type, num_terms,
                                         harmonic_type_pitch,
                                         harmonic_type_roll,
                                         harmonic_type_yaw,
                                         harmonic_coef_pitch,
                                         harmonic_coef_roll,
                                         harmonic_coef_yaw,
```

```
                                 offset,
                                 &instr_trans_id, ierr);
  }
```

The XP_NUM_ERR_INSTR_ATT_INIT_HARMONIC and XP_MAX_NUM_HARMONIC constants are defined in the file *explorer_pointing.h.*

7.35.3 Input Parameters

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

Table 88: Input parameters of xp_instr_att_init_harmonic function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
angle_type	long *	-	Type of angle	-	XP_ANGLE_TYPE_TRUE_LAT_TOD XP_ANGLE_TYPE_TRUE_LAT_EF
num_terms[3]	long	[0]	Number of elements used in vectors harmonic_type_pitch and harmonic_coef_pitch	-	>=0
		[1]	Number of elements used in vectors harmonic_type_roll and harmonic_coef_roll	-	>=0
		[2]	Number of elements used in vectors harmonic_type_yaw and harmonic_coef_yaw	-	>=0
harmonic_type_pitch	long[XP_MAX_NUM_HARMONIC]	all	Type of coefficients: =0 for the bias parameter <0 for the sinus coefficients (-n means that correponds to the sinus coefficient of order n) >0 for the cosinus coefficients (+n means that correponds to the cosinus coefficient of order n)	-	
harmonic_type_roll	long[XP_MAX_NUM_HARMONIC]	all	Type of coefficients: =0 for the bias parameter <0 for the sinus coefficients (-n means that correponds to the sinus coefficient of order n) >0 for the cosinus coefficients (+n means that correponds to the cosinus coefficient of order n)	-	-
harmonic_type_yaw	long[XP_MAX_NUM_HARMONIC]	all	Type of coefficients: =0 for the bias parameter <0 for the sinus coefficients (-n means that correponds to the sinus coefficient of order n)	-	-

			>0 for the cosinus coefficients (+n means that correponds to the cosinus coefficient of order n)		
harmonic_coef_pitch	double[XP_MAX_NUM_HARMONIC]	all	Bias, sinus and cosinus coeficients for the pitch angle	deg	
harmonic_coef_roll	double[XP_MAX_NUM_HARMONIC]	all	Bias, sinus and cosinus coeficients for the roll angle	deg	
harmonic_coef_yaw	double[XP_MAX_NUM_HARMONIC]	all	Bias, sinus and cosinus coeficients for the yaw angle	deg	
offset	double[3]	all	Instrument Frame Origin position vector (Satellite Attitude Frame)	m	-

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

- Angle Type: See current document, Table 3.

7.35.4O utput Parameters

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

Table 89: Output parameters of xp_instr_att_init_harmonic

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
instr_trans_id	xp_instr_trans_id*	-	Structure that contains the Instrument Transformation	-	-
ierr	long	-	Error vector	-	-

7.35.5 Warnings and Errors

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

Table 90: Error messages of xp_instr_att_init_harmonic function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Memory allocation error	No calculation performed	XP_CFI_INSTR_ATT_INIT_HARMONIC_MEMORY_ERR	0

7.36 xp_instr_att_init_file

7.36.1 Overview

The **xp_instr_att_init_file** CFI function initialises the instrument attitude rotation for a given satellite reading values from the attitude file(s). The validity time or orbital range for the attitude angles can be specified by the user. The initialised values will be kept in memory and used by other CFI functions.

The computed rotation will be given between the inertial reference frame specified in the file (tag <Reference_Frame>) and the instrument frame. In case that the files contain rotation angles (pitch, roll, yaw) and if the reference frame is based on the satellite (“SATELLITE_ACTUAL”, “SATELLITE” or “SATELLITE_RELATIVE”), then the rotation will be defined between the satellite attitude frame and the instrument frame.

In order to read files, xp_instr_att_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.

7.36.1.1 Initialisation with several files.

When the attitude frame is initialized with more than one file (see parameter *instrument_file* in next section), the user has to provide the files sorted from lower to higher precedence. This way the possible overlap between files is solved as follows:

instrument_file[0] is the file with lower precedence

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

the attitude data (angles or quaternions) from files with lower precedence that are after the data of a file with higher precedence are skipped.

The maximum gap allowed for the whole set of attitude data, is taken from the first file (*instrument_file*[0]).

Example: The following figure represents three instrument files. Every vertical line represents the positon in time of the attidude data within the file. The initilazation with these files is equivalent to initialize with a single “equivalent file” in the following way :

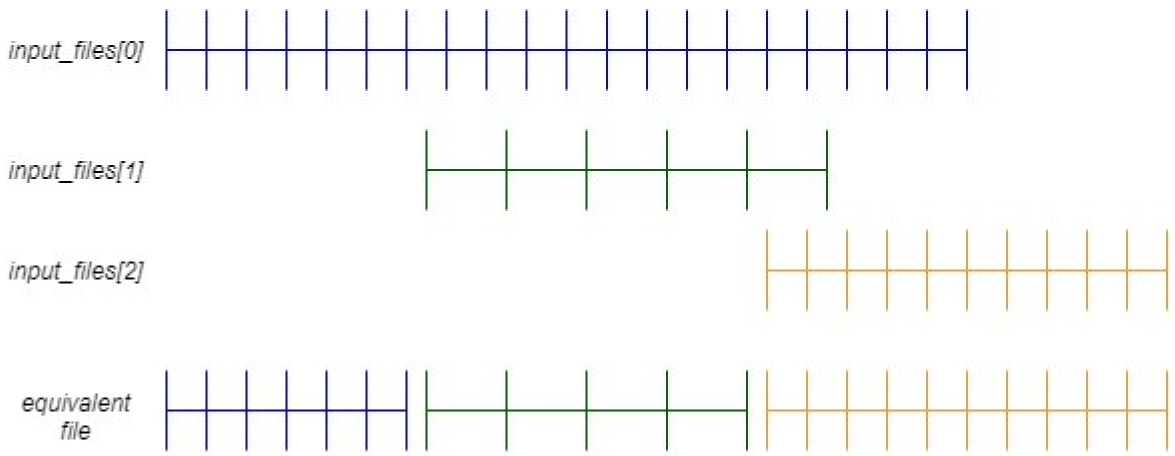


Figure 9: Overlapping instrument files

7.36.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
    xl_time_id time_id = {NULL};
    long n_files, time_init_mode, time_ref;
    char **instrument_file;
    double time0, time1;
    double val_time0, val_time1;
    xp_instr_trans_id instr_trans_id = {NULL};
    long ierr[XP_NUM_ERR_INSTR_ATT_INIT_FILE], status;

    status = xp_instr_att_init_file(&time_id,
                 &n_files, instrument_file,
                 &time_init_mode, &time_ref, &time0, &time1,
                 &val_time0, &val_time1, &instr_trans_id, ierr);
}
```

The `XP_NUM_ERR_INSTR_ATT_INIT_FILE` constant is defined in the file *explorer_pointing.h.*

7.36.3 Input Parameters

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

Table 91: Input parameters of xp_instr_att_init_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
time_id	xl_time_id*	-	Structure that contains the time correlations.	-	-
n_files	long *	-	Number of reference data files	-	> 0
instrument_file	char **	-	Filenames of the reference data files. In case multiple files are used, the files should be time ordered and all files should be of the same type (see section 7.36.1.1). The supported Attitude File format are: - Generic Attitude files. - Attitude Ephemeris Message Files. In this case only one file is allowed and only the first segment of the file is taken for the initalisation. Those files are described in [D_H_SUM] .	-	-
time_init_mode	long *	-	Flag for selecting the time range of the initialization.	-	Select either: · XP_SEL_TIME · XP_SEL_FILE
time_ref	long *	-	Time reference ID	-	Complete
time0	double*	-	If:*time_init_mode=XP_SEL_TIME* Start of the time range defined by [time0,time1]	Decimal days (Processing format)	[-18262.0,36524.0]
time1	double*	-	If:*time_init_mode=XP_SEL_TIME* End of the time range defined by [time0,time1]	Decimal days (Processing format)	[-18262.0,36524.0] > time0

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].
- Time Init Mode ID: time_init_mode. See current document, Table 3.

7.36.4 Output Parameters

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

Table 92: Output parameters of xp_instr_att_init_file

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
val_time0	double*	-	Validity start time of the initialization	Decimal days (Processing format)	[-18262.0,36524.0]
val_time1	double*	-	Validity end time of the initialization	Decimal days (Processing format)	[-18262.0,36524.0]
instr_trans_id	xp_instr_trans_id*	-	Structure that contains the Instrument Transformation	-	-
ierr	long	-	Error vector	-	-

7.36.5 Warnings and Errors

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

Table 93: Error messages of xp_instr_att_init_file function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Memory allocation error	No calculation performed	XP_CFI_INSTR_ATT_INIT_FILE_MEMORY_ERR	0
ERR	Wrong input time reference	No calculation performed	XP_CFI_INSTR_ATT_INIT_FILE_WRONG_TIME_REF_ERR	1
ERR	Error opening attitude file: %s	No calculation performed	XP_CFI_INSTR_ATT_INIT_FILE_OPEN_FILES_ERR	2

ERR	Error reading generic attitude files	No calculation performed	XP_CFI_INSTR_ATT_INIT_ FILE_READ_ATT_FILES_E RR	3
ERR	Could not perform a time transformation	No calculation performed	XP_CFI_INSTR_ATT_INIT_ FILE_TIME_CONV_ERR	4
ERR	Unsupported attitude file format detected	No calculation performed	XP_CFI_INSTR_ATT_INIT_FIL E_WRONG_FILE_FORMAT_E RR	5
WARN	AEM with more than 1 segment used for initialization. Only 1st segment will be used	Attitude is initialised using the 1st segment in the AEM file.	XP_CFI_INSTR_ATT_INIT_FIL E_ONLY_SEGMENT0_WARN,	6

7.37 xp_instr_att_close

7.37.1 Overview

The **xp_instr_att_close** CFI function cleans up any memory allocation performed by the instrument attitude initialization functions.

7.37.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     xp_instr_trans_id instr_trans_id = {NULL};
     long ierr[XP_NUM_ERR_INSTR_ATT_CLOSE], status;

     status = xp_instr_att_close(&instr_trans_id, ierr);
}
```

The `XP_NUM_ERR_INSTR_ATT_CLOSE` constant is defined in the file *explorer_pointing.h.*

7.37.3 Input Parameters

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

Table 94: Input parameters of xp_instr_att_close function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
instr_trans_id	xp_instr_trans_id*	-	Structure that contains the Instr. Trans.	-	-

7.37.4 Output Parameters

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

Table 95: Output parameters of xp_instr_att_close

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

7.37.5 Warnings and Errors

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

Table 96: Error messages of xp_instr_att_close function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Could not close the Id. as it is not initialized or it is being used	No calculation performed	XP_CFI_INSTR_ATT_CLOSE_WRONG_ID_ERR	0

7.38 xp_instr_att_get_angles

7.38.1 Overview

The **xp_instr_att_get_angles** CFI function returns the angle data used for the instrument attitude initialization.

7.38.2 Calling interface

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

```
#include <explorer_lib.h>
{
       xp_instr_trans_id instr_trans_id;
       long status;
       xp_angle_model_str data;
       status = xp_instr_att_get_angles (&instr_trans_id,
                                         &data);
}
```

7.38.3 Input parameters

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

Table 97: Input parameters of xp_instr_att_get_angles function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
instr_trans_id	xp_instr_trans_id *	-	Instrument transformation ID.	-	-

7.38.4 Output parameters

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

Table 98: Output parameters of xp_instr_att_get_angles function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_instr_att_get_angles	long	-	Status flag	-	-

data	xp_angle_model _str	-	Attitude initialization data	-	-

7.38.5 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 instr_trans_id was not initialised.
- The instr_trans_id initialization does not allow the use of this function.

7.39 xp_instr_att_set_angles

7.39.1 Overview

The **xp_instr_att_set_angles** CFI function changes the harmonic data used for the satellite attitude initialization.

7.39.2 Calling interface

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

```
#include <explorer_lib.h>
{
       xp_instr_trans_id instr_trans_id;
       long status;
       xp_angle_model_str data;
       status = xp_instr_att_set_angles (&instr_trans_id,
                                           &data);
}
```

7.39.3 Input parameters

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

Table 99: Input parameters of xp_instr_att_set_angles function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
instr_trans_id	xp_instr_trans_id *	-	Instrument transformation ID (input / output parameter)	-	-
data	xp_angle_model_str	-	Attitude initialization data	-	-

7.39.4 Output parameters

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

Table 100: Output parameters of xp_instr_att_set_angles function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_instr_att_set_angles	long	-	Status flag	-	-
instr_trans_id	xp_instr_trans_id *	-	Instrument transformation ID (input / output parameter)	-	-

7.39.5 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 instr_trans_id was not initialised.
- The instr_trans_id initialization does not allow the use of this function.

7.40 xp_instr_att_get_matrix

7.40.1 Overview

The **xp_instr_att_get_matrix** CFI function returns the matrix data used for the satellite attitude initialization.

Note on matrix notation:

If XYZ are the axes of the original reference frame, and X'Y'Z' are the axes of the rotated frame, the rows of the rotation matrix are respectively X, Y and Z axes expressed in X'Y'Z' system.

In the C representation, M[0][], M[1][], M[2][] are respectively 1st, and and 3rd row of a rotation matrix M.

The rotation matrix M satisfies the following equivalence:

V = M***V'**

where **V'** is a vector expressed in the X'Y'Z' reference system and **V** is expressed in the XYZ reference system.

7.40.2 Calling interface

The calling interface of the **xp_instr_att_get_matrix** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_lib.h>
{
       xp_instr_trans_id instr_trans_id;
       long status;
       xp_matrix_model_str data;
       status = xp_instr_att_get_matrix (&instr_trans_id,
                                         &data);
}
```

7.40.3 Input parameters

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

Table 101: Input parameters of xp_instr_att_get_matrix function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
instr_trans_id	xp_instr_trans_id *	-	Instrument transformation ID.	-	-

7.40.4 Output parameters

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

Table 102: Output parameters of xp_instr_att_get_matrix function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_instr_att_get_matrix	long	-	Status flag	-	-
data	xp_matrix_model_str	-	Attitude initialization data	-	-

7.40.5 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 instr_trans_id was not initialised.
- The instr_trans_id initialization does not allow the use of this function.

7.41 xp_instr_att_set_matrix

7.41.1 Overview

The **xp_instr_att_set_matrix** CFI function changes matrix data used for the satellite attitude initialization. The matrix is checked to be orthonormal.

Note on matrix notation:

If XYZ are the axes of the original reference frame, and X'Y'Z' are the axes of the rotated frame, the rows of the rotation matrix are respectively X, Y and Z axes expressed in X'Y'Z' system.

In the C representation, M[0][], M[1][], M[2][] are respectively 1st, and and 3rd row of a rotation matrix M.

The rotation matrix M satisfies the following equivalence:

V = M***V'**

where **V'** is a vector expressed in the X'Y'Z' reference system and **V** is expressed in the XYZ reference system.

7.41.2 Calling interface

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

```
#include <explorer_lib.h>
{
       xp_instr_trans_id instr_trans_id;
       long status;
       xp_matrix_model_str data;
       status = xp_instr_att_set_matrix (&instr_trans_id,
                                         &data);
}
```

7.41.3 Input parameters

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

Table 103: Input parameters of xp_instr_att_set_matrix function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
instr_trans_id	xp_instr_trans_id *	-	Instrument transformation ID (input / output parameter)	-	-
data	xp_angle_mod	-	Attitude initialization	-	-

	el_str		data		

7.41.4 Output parameters

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

Table 104: Output parameters of xp_instr_att_set_matrix function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_instr_att_set_matrix	long	-	Status flag	-	-
instr_trans_id	xp_instr_trans_id *	-	Instrument transformation ID (input / output parameter)	-	-

7.41.5 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 instr_trans_id was not initialised.
- The instr_trans_id initialization does not allow the use of this function.

7.42 xp_instr_att_get_harmonic

7.42.1 Overview

The **xp_instr_att_get_harmonic** CFI function returns harmonic data used for the satellite attitude initialization.

7.42.2 Calling interface

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

```
#include <explorer_lib.h>
{
        xp_instr_trans_id instr_trans_id;
        long status;
        xp_harmonic_model_str data;
        status = xp_instr_att_get_harmonic (&instr_trans_id,
                                            &data);
}
```

7.42.3 Input parameters

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

Table 105: Input parameters of xp_instr_att_get_harmonic function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
instr_trans_id	xp_instr_trans_id *	-	Instrument transformation ID.	-	-

7.42.4 Output parameters

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

Table 106: Output parameters of xp_instr_att_get_harmonic function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_instr_att_get_harmonic	long	-	Status flag	-	-

data	xp_harmonic_model_str	-	Attitude initialization data	-	-

7.42.5 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 instr_trans_id was not initialised.
- The instr_trans_id initialization does not allow the use of this function.

7.43 xp_instr_att_set_harmonic

7.43.1 Overview

The **xp_instr_att_set_harmonic** CFI function changes the harmonic data used for the satellite attitude initialization.

7.43.2 Calling interface

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

```
#include <explorer_lib.h>
{
        xp_instr_trans_id instr_trans_id;
        long status;
        xp_harmonic_model_str data;
        status = xp_instr_att_set_harmonic (&instr_trans_id,
                                            &data);
}
```

7.43.3 Input parameters

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

Table 107: Input parameters of xp_instr_att_set_harmonic function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
instr_trans_id	xp_instr_trans_id *	-	Instrument transformation ID (input / output parameter)	-	-
data	xp_harmonic_model_str	-	Attitude initialization data	-	-

7.43.4 Output parameters

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

Table 108: Output parameters of xp_instr_att_set_harmonic function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_instr_att_set_harmonic	long	-	Status flag	-	-
instr_trans_id	xp_instr_trans_id *	-	Instrument transformation ID (input / output parameter)	-	-

7.43.5 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 instr_trans_id was not initialised.
- The instr_trans_id initialization does not allow the use of this function.

7.44 xp_instr_att_get_file

7.44.1 Overview

The **xp_instr_att_get_file** CFI function returns satellite attitude data from the satellite attitude Id. that was initialized with a file.

7.44.2 Calling interface

The calling interface of the **xp_instr_att_get_file** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_lib.h>
{
        xp_instr_trans_id instr_trans_id;
        long status;
        xp_instr_att_file_model_str data;
        status = xp_instr_att_get_file (&instr_trans_id,
                                        &data);
}
```

7.44.3 Input parameters

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

Table 109: Input parameters of xp_instr_att_get_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
instr_trans_id	xp_instr_trans_id *	-	Instrument transformation ID.	-	-

7.44.4 Output parameters

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

Table 110: Output parameters of xp_instr_att_get_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_instr_att_get_file	long	-	Status flag	-	-
data	xp_instr_att_file_	-	Attitude initialization	-	-

	model_str		data		

7.44.5 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 instr_trans_id was not initialised.
- The instr_trans_id initialization does not allow the use of this function.

7.45 xp_instr_att_set_file

7.45.1 Overview

The **xp_instr_att_set_file** CFI function changes the initialization data in the satellite attitude Id. when it was initialised with a file. Quaternions are checked to be normalized.

7.45.2 Calling interface

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

```
#include <explorer_lib.h>
{
        xp_instr_trans_id instr_trans_id;
        long status;
        xp_instr_att_file_model_str data;
        status = xp_instr_att_set_file (&instr_trans_id,
                                        &data);
}
```

7.45.3 Input parameters

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

Table 111: Input parameters of xp_instr_att_set_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
instr_trans_id	xp_instr_trans_id *	-	Instrument transformation ID (input / output parameter)	-	-
data	xp_instr_att_file_model_str	-	Attitude initialization data	-	-

7.45.4 Output parameters

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

Table 112: Output parameters of xp_instr_att_set_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_instr_att_set_file	long	-	Status flag	-	-
instr_trans_id	xp_instr_trans_id *	-	Instrument transformation ID (input / output parameter)	-	-

7.45.5 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 instr_trans_id was not initialised.
- The instr_trans_id initialization does not allow the use of this function.

7.46 xp_instr_att_get_offset

7.46.1 Overview

The **xp_instr_att_get_offset** CFI function allows the user to retrieve the offsets associated with an xp_instr_trans_id.

7.46.2 Calling interfaces

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

```
#include <explorer_pointing.h>
{
     xp_instr_trans_id *instr_trans_id;
     double offset[3];
     long status;

     status = xp_instr_att_get_offset(instr_trans_id, offset);
}
```

7.46.3 Input Parameters

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

Table 113: Input parameters of xp_instr_att_get_offset function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
instr_trans_id	xp_instr_trans_id *	-	Instrument trans id	-	-

7.46.4 Output Parameters

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

Table 114: Output parameters of xp_instr_att_get_offset

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
offset	double[3]	-	Offsets associated with instr_trans_id	-	-

7.46.5 Warnings and Errors

The **xp_instr_att_get_offset** CFI function has no warnings and errors defined.

7.47 xp_instr_att_set_offset

7.47.1 Overview

The **xp_instr_att_set_offset** CFI function allows the user to set the offsets associated with an xp_instr_trans_id.

7.47.2 Calling interfaces

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

```
#include <explorer_pointing.h>
{
    xp_instr_trans_id *instr_trans_id;
    double offset[3];
    long status;

    status = xp_instr_att_set_offset(instr_trans_id, offset);
}
```

7.47.3 Input Parameters

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

Table 115: Input parameters of xp_instr_att_set_offset function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
instr_trans_id	xp_instr_trans_id *	-	Instrument trans id	-	-
offset	double[3]	-	New offsets for instr_trans_id	-	-

7.47.4 Output Parameters

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

Table 116: Output parameters of xp_instr_att_set_offset

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
instr_trans_id	xp_instr_trans_id *	-	Instrument trans id	-	-

7.47.5 Warnings and Errors

The **xp_instr_att_set_offset** CFI function has no warnings and errors defined.

7.48 xp_set_az_el_definition

7.48.1 Overview

The **xp_set_az_el_definition** function sets an user-defined azimuth/elevation in a satellite nominal attitude id, satellite attitude id or instrument attitude id.

7.48.2 Calling interface

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

```
#include <explorer_lib.h>
{
       void *att_frame_id;
       xl_az_el_definition azel_def;
       long ierr[XP_NUM_ERR_SET_AZ_EL_DEFINITION];
       status = xp_set_az_el_definition (att_frame_id,
                                            &azel_def,
                                        ierr);
}
```

7.48.3 Input parameters

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

Table 117: Input parameters of xp_instr_att_set_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
att_frame_id	void*	-	Attitude where the definition will be inserted.	-	It must be a Satellite Nominal id (xp_sat_nom_trans_id*), satellite attitude id (xp_sat_trans_id*) or instrument attitude id (xp_instr_trans_id*).

azel_def	xl_az_el_definition	-	Azimuth/elevation definition	-	-

7.48.4 Output parameters

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

Table 118: Output parameters of xp_instr_att_set_file function

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

7.48.5 Warnings and errors

This function returns error if the input id is not initialized, it is not of the correct type, or there is a problem with the azimuth/elevation definition introduced by the user. In Table 119 are summarized the possible errors.

Table 119: Error messages of xp_set_az_el_definition function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Argument ID is not initialized.	No calculation performed	XP_CFI_SET_AZ_EL_DEF_ID_NOT_INITIALIZED_ERR	0
ERR	Argument ID is not a satellite nominal, satellite or instrument attitude ID.	No calculation performed	XP_CFI_SET_AZ_EL_DEF_NOT_ATTITUDE_ID_ERR	1
ERR	Azimuth axis are nor perpendicular.	No calculation performed	XP_CFI_SET_AZ_EL_DEF_NOT_PERPENDICULAR_AZIMUTH_AXIS_E RR	2
ERR	Elevation axis not perpendicular to azimuth plane.	No calculation performed	XP_CFI_SET_AZ_EL_DEF_NOT_PERPENDICULAR_ELEVATION_AXIS_ERR	3

7.49 xp_attitude_define

7.49.1 Overview

The **xp_attitude_define** CFI function initializes the satellite nominal attitude, satellite attitude and instrument attitude according to the input data.

The input data is stored in a structure of type xd_attitude_definition_data (see section 6.3 of [D_H_SUM]). The user can fill this structure within his application program or by reading an atttitude definition file using function xd_read_att_def (see [D_H_SUM]).

7.49.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
    xd_attitude_definition_data data;
    xp_sat_nom_trans_id sat_nom_trans_id;
    xp_sat_trans_id sat_trans_id;
    xp_instr_trans_id instr_trans_id;
    long ierr[XP_NUM_ERR_ATTITUDE_DEFINE], status;

    status = xp_attitude_define(&data,
                                &sat_nom_trans_id,
                                &sat_trans_id,
                                &instr_trans_id, ierr);
}
```

The `XP_NUM_ERR_ATTITUDE_DEFINE` constant is defined in the file *explorer_pointing.h.*

7.49.3 Input Parameters

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

Table 120: Input parameters of xp_attitude_define function

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

7.49.4 Output Parameters

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

Table 121: Output parameters of xp_attitude_define

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_nom_trans_id	xp_sat_nom_trans _id*	-	Structure that contains the Satellite nominal Transformation	-	-
sat_trans_id	xp_sat_trans _id*	-	Structure that contains the Satellite Transformation	-	-
instr_trans_id	xp_instr_trans _id*	-	Structure that contains the Instrument Transformation	-	-
ierr	long	-	Error vector	-	-

7.49.5 Warnings and Errors

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

Table 122: Error messages of xp_attitude_define function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Error initializing satellite nominal attitude	No computation performed	XP_ATTITUDE_DEFINE_SAT_NOM_TRANS_INIT	0
ERR	Error initializing satellite attitude	No computation performed	XP_ATTITUDE_DEFINE_SAT_TRANS_INIT	1
ERR	Error initializing instrument attitude	No computation performed	XP_ATTITUDE_DEFINE_INSTR_TRANS_INIT	2

7.50 xp_run_init

7.50.1 Overview

The **xp_run_init** CFI function adds to the *run id* the s*at_nom_trans_id*, *sat_trans_id*, *instr_trans_id*, *atmos Id* and *dem Id.*

7.50.2 Calling interface

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

```
#include <explorer_pointing.h>
{
     long run_id;
     xp_sat_nom_trans_id sat_nom_trans_id = {NULL};
     xp_sat_trans_id     sat_trans_id = {NULL};
     xp_instr_trans_id   instr_trans_id = {NULL};
     xp_atmos_id         atmos_id = {NULL};
     xp_dem_id           dem_id = {NULL};
     long ierr[XP_NUM_ERR_RUN_INIT], status;
     status = xp_run_init (&run_id, &sat_nom_trans_id,
                             &sat_trans_id, &instr_trans_id,
                             &atmos_id, &dem_id,
                             ierr);
}
```

7.50.3 Input parameters

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

Table 123: Input parameters of xp_run_init function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
run_id	long *	-	Run ID	-	>=0
sat_nom_trans_id	xp_sat_nom_trans_id*	-	Structure that contains the Sat. Nom. Trans.	-	-
sat_trans_id	xp_sat_trans_id*	-	Structure that contains the Sat. Trans.	-	-
instr_trans_id	xp_instr_trans_id*	-	Structure that contains the Instr. Trans.	-	-
atmos_id	xp_atmos_id*	-	Structure that contains the atmosphere initialization.	-	-
dem_id	xp_dem_id*	-	Structure that contains the DEM initialization.	-	-

7.50.4 Output parameters

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

Table 124: Output parameters of xp_run_init function

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

7.50.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xp_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_POINTING software library **xp_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 **xp_run_init** function by calling the function of the EO_POINTING software library **xp_get_code** (see [GEN_SUM]).

Table 125: Error messages of xl_run_init function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Wrong input run_id. It is not correctly initialized	No calculation performed	XP_CFI_RUN_INIT_STATUS_ERR	0
ERR	Memory allocation error	No calculation performed	XP_CFI_RUN_INIT_MEMORY_ERR	1
ERR	Incompatible input Ids	No calculation performed	XP_CFI_RUN_INIT_INCONSISTENCY_ERR	2

7.51 xp_run_get_ids

7.51.1 Overview

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

7.51.2 Calling interface

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

```
#include <explorer_pointing.h>
{
     long run_id;
     xp_sat_nom_trans_id sat_nom_trans_id = {NULL};
     xp_sat_trans_id     sat_trans_id = {NULL};
     xp_instr_trans_id   instr_trans_id = {NULL};
     xp_atmos_id         atmos_id = {NULL};
     xp_dem_id           dem_id = {NULL};
     xp_run_get_ids  (&run_id,
                      &sat_nom_trans_id,
                      &sat_trans_id,
                      &instr_trans_id,
                      &atmos_id,
                      &dem_id);
}
```

7.51.3 Input parameters

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

Table 126: Input parameters of xp_run_get_ids function

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

7.51.4 Output parameters

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

Table 127: Output parameters of xp_run_get_ids function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xl_run_get_ids	void	-	-	-	-
sat_nom_trans_id	xp_sat_nom_trans_id*	-	Structure that contains the Sat. Nom. Trans.	-	-
sat_trans_id	xp_sat_trans_id*	-	Structure that contains the Sat. Trans.	-	-
instr_trans_id	xp_instr_trans_id*	-	Structure that contains the Instr. Trans.	-	-
atmos_id	xp_atmos_id*	-	Structure that contains the atmosphere initialization.	-	-
dem_id	xp_dem_id*	-	Structure that contains the DEM initialization.	-	-

7.51.5 Warnings and errors

TBW

7.52 xp_run_close

7.52.1 Overview

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

7.52.2 Calling interface

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

```
#include <explorer_pointing.h>
{
        long run_id;
      xp_run_close (&run_id);
}
```

7.52.3 Input parameters

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

Table 128: Input parameters of xp_run_close function

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

7.52.4 Output parameters

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

Table 129: Output parameters of xp_run_close function

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

7.52.5 Warnings and errors

This function does not return errors nor warnings.

7.53 xp_attitude_init

7.53.1 Overview

The **xp_attitude_init** CFI function creates an empty *attitude Id*.

7.53.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     xp_attitude_id attitude_id = {NULL};
     long ierr[XP_NUM_ERR_ATTITUDE_INIT], status;

     status = xp_attitude_init(&attitude_id, ierr);
}
```

The `XP_NUM_ERR_ATTITUDE_INIT` constant is defined in the file *explorer_pointing.h.*

7.53.3 Input Parameters

The **xp_attitude_init** CFI function has no input parameters.

7.53.4 Output Parameters

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

Table 130: Output parameters of xp_attitude_init

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude.	-	-
ierr	long	-	Error vector	-	-

7.53.5 Warnings and Errors

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

Table 131: Error messages of xp_attitude_init function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Memory allocation error	No calculation performed	XP_CFI_ATTITUDE_INIT_MEMORY_ERR	0

7.54 xp_attitude_compute

7.54.1 Overview

The **xp_attitude_compute** CFI function calculates the Attitude Frame for a given S/C state vector.

Note: a correction can be applied in order to compensate the travel time of Sun light travel time. This correction is not applied with default model. To activate this correction, the Sun model in xl_model_id must be initialized with the enum XL_MODEL_SUN_TRAVEL_TIME using the function xl_model_init (see [LIB_SUM]).

7.54.2 Calling interface

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

```
#include <explorer_pointing.h>
{
    xl_model_id          model_id = {NULL};
    xl_time_id           time_id  = {NULL};
    xp_sat_nom_trans_id sat_nom_trans_id = {NULL};
    xp_sat_trans_id      sat_trans_id = {NULL};
    xp_instr_trans_id    instr_trans_id = {NULL};
    xp_attitude_id       attitude_id = {NULL};
    long time_ref, target_frame;
    double time, pos[3], vel[3], acc[3];
    long ierr[XP_NUM_ERR_ATTITUDE_COMPUTE];

    status =xp_attitude_compute(&model_id, &time_id,
                                &sat_nom_trans_id,
                                &sat_trans_id,
                                &instr_trans_id,
                                &attitude_id,
                                  /* input/output */
                                &time_ref, &time, pos, vel, acc,
                                &target_frame,
                                ierr);

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

```
    status = xp_attitude_compute_run(&run_id,
                                     &attitude_id,
                                       /* input/output */
                                     &time_ref, &time, pos, vel, acc,
                                     &target_frame,
                                     ierr);
}
```

The XP_NUM_ERR_ATTITUDE_COMPUTE constant is defined in the file *explorer_pointing.h*.

7.54.3 Input parameters

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

Table 132: Input parameters of xp_attitude_compute function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
model_id	xl_model_id*	-	Model ID.	-	-
time_id	xl_time_id*	-	Structure that contains the time correlations.	-	-
sat_nom_trans_id	xp_sat_nom_trans_id*	-	Structure that contains the Sat. Nom. Trans.	-	-
sat_trans_id	xp_sat_trans_id*	-	Structure that contains the Sat. Trans.	-	-
instr_trans_id	xp_instr_trans_id*	-	Structure that contains the Instr. Trans.	-	-
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude (input/output)	-	-
time_ref	long *	-	Time reference ID	-	Complete
time	double	-	Time in Processing Format	Decimal days, MJD2000	[-18262.0,36524.0]
pos[3]	double	all	Satellite position vector (Earth Fixed CS)	m	-
vel[3]	double	all	Satellite velocity vector (Earth Fixed CS)	m/s	-
acc[3]	double	all	Satellite acceleration vector (Earth Fixed CS)	m/s^2	-
target_frame	long *	-	Attitude FrameID	-	Complete

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].
- Attitude Frame ID: attitude_frame_id. See current document, Table 3.

7.54.4 Output parameters

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

Table 133: Output parameters of xp_attitude_compute function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude. (input/output)	-	-
ierr	long	-	Error vector	-	-

7.54.5 Warnings and errors

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

Table 134: Error messages of xp_attitude_compute function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Time Id. not initialized	No calculation performed	XP_CFI_ATTITUDE_COMPUTE_TIME_STATUS_ERR	0
ERR	Instrument Trans. Id. not initialized	No calculation performed	XP_CFI_ATTITUDE_COMPUTE_INSTR_TRANS_STATUS_ERR	1
ERR	Satellite Att. Trans. not initialized	No calculation performed	XP_CFI_ATTITUDE_COMPUTE_SAT_TRANS_STATUS_ERR	2
ERR	Satellite Nom. Trans not initialized	No calculation performed	XP_CFI_ATTITUDE_COMPUTE_SAT_NOM_TRANS_STATUS_ERR	3
ERR	Attitude Id. not initialized	No calculation performed	XP_CFI_ATTITUDE_COMPUTE_ATTITUDE_STATUS_ERR	4
ERR	Wrong input time reference	No calculation performed	XP_CFI_ATTITUDE_COMPUTE_WRONG_TIME_REF_ERR	5

ERR	Attitude Id is being used by another Id.	No calculation performed	XP_CFI_ATTITUDE_COMPUTE_BEING_USED_ERR	6
ERR	Could not compute orbit reference frame	No calculation performed	XP_CFI_ATTITUDE_COMPUTE_ORB_REF_ERR	7
ERR	Could not calculate AOCS parametes	No calculation performed	XP_CFI_ATTITUDE_COMPUTE_AOCS_CALC_ERR	8
ERR	Could not compute Sat. Nom. Trans frame	No calculation performed	XP_CFI_ATTITUDE_COMPUTE_SAT_NOM_TRANS_ERR	9
ERR	"Could not calculate the true latitude"	No calculation performed	XP_CFI_ATTITUDE_COMPUTE_TRUE_LAT_ER R	10
ERR	Could not calculate harmonic angles	No calculation performed	XP_CFI_ATTITUDE_COMPUTE_HARMONIC_CALC_ERR	11
ERR	Could not compute Sat.Trans. frame	No calculation performed	XP_CFI_ATTITUDE_COMPUTE_SAT_TRANS_E RR	12
ERR	Error computing direction cosine matrix from Sat. Att. to BJ2000	No calculation performed	XP_CFI_ATTITUDE_COMPUTE_ATT_TO_J2000_ERR	13
ERR	Could not compute Instrument Trans. frame	No calculation performed	XP_CFI_ATTITUDE_COMPUTE_INSTR_TRANS_ERR	14
ERR	Memory allocation error	No calculation performed	XP_CFI_ATTITUDE_COMPUTE_MEMORY_ERR	15
ERR	Both input targets are the same	No calculation performed	XP_CFI_ATTITUDE_COMPUTE_SAME_TARGETS_ERR	16
ERR	Error occured during call to XP_Vec_Find	No calculation performed	XP_CFI_ATTITUDE_COMPUTE_VEC_FIND_ERR	17
ERR	Error occured during call to XP_Create_Base	No calculation performed	XP_CFI_ATTITUDE_COMPUTE_CREATE_BASE_ERR	18
ERR	Error occured trying to get the interpolated value for the quaternions/angles	No calculation performed	XP_CFI_ATTITUDE_COMPUTE_ATT_FILE_INTER_ERR	19
ERR	Error in a change of coordinate frame	No calculation performed	XP_CFI_ATTITUDE_COMPUTE_CHANGE_CS_ERR	20
WARN	Warning raised by XP_Vec_Find	Calculation performed	XP_CFI_ATTITUDE_COMPUTE_VEC_FIND_WARN	21
ERR	Error computing Sun position	No calculation performed	XP_ATTITUDE_COMPUT	22

			E_SUN_ERR	
ERR	Error in a change of time system	No calculation performed	XP_ATTITUDE_COMPUTE_CHANGE_TIME_ERR	23

7.55 xp_attitude_user_set

7.55.1 Overview

The **xp_attitude_user_set** CFI function assigns a user defined Attitude Frame to the *attitude Id*. Input matrix is checked to be orthonormal.

Note on matrix notation:

If XYZ are the axes of the original reference frame, and X'Y'Z' are the axes of the rotated frame, the rows of the rotation matrix are respectively X, Y and Z axes expressed in X'Y'Z' system.

In the C representation, M[0][], M[1][], M[2][] are respectively 1st, and and 3rd row of a rotation matrix M.

The rotation matrix M satisfies the following equivalence:

$$\mathbf{V} = M*\mathbf{V'}$$

where **V’** is a vector expressed in the X’Y’Z’ reference system and **V** is expressed in the XYZ reference system.

7.55.2 Calling interface

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

```
#include <explorer_pointing.h>
{
    xl_model_id         model_id = {NULL};
    xl_time_id          time_id  = {NULL};
    xp_attitude_id      attitude_id = {NULL};
    long time_ref, target_frame;
    double time, pos[3], vel[3], acc[3];
    double matrix[3][3];
    double matrix_rate[3][3];
    double matrix_rate_rate[3][3];
    double offset[3],;
    long ierr[XP_NUM_ERR_ATTITUDE_USER_SET];

    long xp_attitude_user_set(&model_id, &time_id,
                              &attitude_id,
                                /* input / output */
                              &time_ref, &time, pos, vel, acc,
                              &target_frame,
                              matrix, matrix_rate, matrix_rate_rate,
                              offset,
```

```
                                    ierr);

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

      long xp_attitude_user_set_run(&run_id,
                                    &attitude_id,
                                      /* input / output */
                                    &time_ref, &time, pos, vel, acc,
                                    &target_frame,
                                    matrix, matrix_rate, matrix_rate_rate,
                                    offset, ierr);
  }
```

The `XP_NUM_ERR_ATTITUDE_USER_SET` constant is defined in the file *explorer_pointing.h.*

7.55.3 Input parameters

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

Table 135: Input parameters of xp_attitude_user_set function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
model_id	xl_model_id*	-	Model ID	-	-
time_id	xl_time_id*	-	Structure that contains the time correlations.	-	-
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude (input/output)	-	-
time_ref	long *	-	Time reference ID	-	Complete
time	double	-	Time in Processing Format	Decimal days, MJD2000	[-18262.0,36524.0]
pos[3]	double	all	Satellite position vector (Earth Fixed CS)	m	-
vel[3]	double	all	Satellite velocity vector (Earth Fixed CS)	m/s	-
acc[3]	double	all	Satellite acceleration vector (Earth Fixed CS)	m/s^2	-
target_frame	long *	-	Attitude FrameID	-	Complete
matrix[3][3]	double	all	Matrix representing the	-	-

			transformation from ToD to target_frame		
matrix_rate [3][3]	double	all	Matrix representing the transformation rate from ToD to target_frame	-	-
matrix_rate_rate [3][3]	double	all	Matrix representing the transformation rate rate from ToD to target_frame	-	-
offset[3]	double	all	Offset in the instrument frame origin	m	-

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].
- Attitude Frame ID: attitude_frame_id. See current document, Table 3.

7.55.4 Output parameters

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

Table 136: Output parameters of xp_attitude_user_set function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude. (input/output)	-	-
ierr	long	-	Error vector	-	-

7.55.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xp_attitude_user_set** CFI function after translating the returned extended status flag into the equivalent list of error messages by calling the function of the EO_POINTING software library **xl_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 **xp_attitude_user_set** function by calling the function of the EO_POINTING software library **xl_get_code** (see [GEN_SUM]).

Table 137: Error messages of xp_attitude_user_set function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Time Id. not initialized	No calculation performed	XP_CFI_ATTITUDE_USER_SET_TIME_STATUS	0

			_ERR	
ERR	Wrong input target frame	No calculation performed	XP_CFI_ATTITUDE_USER_SET_WRONG_TARGET_FRAME_ERR	1
ERR	Attitude Id. not initialized	No calculation performed	XP_CFI_ATTITUDE_USER_SET_ATTITUDE_STATUS_ERR	2
ERR	Attitude Id is being used by another Id	No calculation performed	XP_CFI_ATTITUDE_USER_SET_BEING_USED_ERR	3
ERR	Could not compute orbit reference frame	No calculation performed	XP_CFI_ATTITUDE_USER_SET_ORB_REF_ERR	4
ERR	Memory allocation error	No calculation performed	XP_CFI_ATTITUDE_USER_SET_MEMORY_ERR	5
ERR	Matrix is not orthonormal	No calculation performed. The CFI performs a check, with a tolerance of 10^{-6}, that the product of the input matrix and its transposed is the unitary matrix.	XP_CFI_ATTITUDE_USER_SET_MATRIX_ORTHONORMAL_ERR,	6

7.56 xp_get_attitude_data

7.56.1 Overview

The **xp_get_attitude_data** CFI function computes the quaternions or attitude angles (roll, pitch, yaw) that define the rotation between two reference frames:

- A source reference frame (given as input).
- The attitude reference frame given by the input attitude_id. Note that the attitude_id has to be previously computed using the functions xp_attitude_compute or xp_attitude_user_set.

7.56.2 Calling interface

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

```
#include <explorer_pointing.h>
{
    xp_attitude_id      attitude_id = {NULL};
    long data_type;
    long source_ref_type;
    long source_ref;
    xd_att_rec att_rec;
    long ierr[XP_NUM_ERR_GET_ATTITUDE_DATA];
    long xp_get_attitude_data(&attitude_id,
                              &data_type,
                              &source_ref_type,
                              &source_ref,
                              /* output */
                              &att_rec,
                              ierr);
```

The XP_NUM_ERR_GET_ATTITUDE_DATA constant is defined in the file *explorer_pointing.h.*

7.56.3 Input parameters

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

Table 138: Input parameters of xp_get_attitude_data function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
attitude_id	xp_attitude_i d*	-	Structure that contains	-	-

			the Attitude		
data_type	long	-	Requested data type: angles or quaternions.	-	XD_ATT_QUATERNIONS XD_ATT_ANGLES
source_ref_type	long	-	Source reference type: External or Satellite.	-	XP_FRAME_FLAG_EXT XP_FRAME_FLAG_SAT
source_ref	long	-	Source reference CS	-	XL_BM2000, XL_HM2000, XL_GM2000, XL_MOD, XL_TOD, XL_PEF, XL_EF XL_LIF, XL_GALACTIC, XL_SAT_ORBITAL_REF, XL_SAT_NOMINAL_ATT, XL_SAT_ATT, XL_INSTR_ATT

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

- data_type: See XD_Attitude_data_type_enum [D_H_SUM].
- source_ref_type: See enumeration XP_Frame_flag_enum in current document (Table 3).
- source_ref: See enumeration XL_CS_rl_enum and XL_Attitude_fr_enum in [LIB_SUM].

7.56.4 Output parameters

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

Table 139: Output parameters of xp_get_attitude_data function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
att_rec	xd_att_rec	-	Structure containing the attitude angles/quaternions	-	-
ierr	long	-	Error vector	-	-

7.56.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xp_get_attitude_data** CFI function after translating the returned extended status flag into the equivalent list of error messages by calling the function of the EO_POINTING software library **xp_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 **xp_get_attitude_data** function by calling the function of the EO_POINTING software library **xp_get_code** (see [GEN_SUM]).

Table 140: Error messages of xp_get_attitude_data function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Wrong input satellite source frame	No calculation performed	XP_CFI_GET_ATT_DATA_WRONG_SAT_CS_ERR	0
ERR	Wrong input exteranl source frame	No calculation performed	XP_CFI_GET_ATT_DATA_WRONG_EXT_CS_ERR	1
ERR	Wrong inpute source frame type. Should be external or satellite	No calculation performed	XP_CFI_GET_ATT_DATA_WRONG_FRAME_ERR	2
ERR	Error getting atttide data from input attitude Id.	No calculation performed	XP_CFI_GET_ATT_DATA_GET_ATT_DATA_ERR	3
ERR	Error computing rotation matrix	No calculation performed	XP_CFI_GET_ATT_DATA_GET_ROTATION_MATRIX_ERR, ERR	4
ERR	Error in attitude initialization	No calculation performed	XP_CFI_GET_ATT_DATA_GET_ATT_INIT_ERR	5
ERR	Error in attitude compute	No calculation performed	XP_CFI_GET_ATT_DATA_GET_ATT_COMPUTE_ERR	6
ERR	Error in matrix inversion	No calculation performed	XP_CFI_GET_ATT_DATA_MATRIX_INV_ERR	7
ERR	Could not close the attitude id.	No calculation performed	XP_CFI_GET_ATT_DATA_CLOSE_ATT_ERR	8
ERR	Could not compute the Euler angles	No calculation performed	XP_CFI_GET_ATT_DATA_ANGLE_COMP_ERR	9
ERR	Could not compute the quaternions	No calculation performed	(XP_CFI_GET_ATT_DATA_QUAT_COMP_ERR	10

7.57 xp_gen_attitude_data

7.57.1 Overview

The **xp_gen_attitude_data** function computes a list of quaternions or attitude angles (roll, pitch, yaw) at an interval given by the user (with a regular time separation) that define the rotation between two reference frames:

- A source reference frame (given as input).
- The attitude frame given by the input attitude definition file.

7.57.2 Calling interface

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

```
#include <explorer_pointing.h>
{
    xo_orbit_id      orbit_id = {NULL};
    xp_attitude_def  att_def;
    xo_time_interval time_interval;
    double           time_step;
    long             data_type;
    long             source_ref_type;
    long             source_ref;
    xd_att_file      *att_file;
    long ierr[XP_NUM_ERR_GEN_ATTITUDE_DATA];

    long xp_gen_attitude_data(&orbit_id,
                              &att_def,
                              &time_interval,
                              &time_step,
                              &data_type,
                              &source_ref_type,
                              &source_ref,
                              /* output */
                              &att_file,
                              ierr);

    /* Or, using the run_id */
    long run_id;
    long xp_gen_attitude_data_run(&run_id,
```

```
                                      &time_interval,
                                      &time_step,
                                      &data_type,
                                      &source_ref_type,
                                      &source_ref,
                                      /* output */
                                      &att_file,
                                      ierr);
}
```

The XP_NUM_ERR_GEN_ATTITUDE_DATA constant is defined in the file *explorer_pointing.h.*

7.57.3 Input parameters

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

Table 141: Input parameters of xp_gen_attitude_data function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xp_orbit_id*	-	Structure that contains the satellite orbit data	-	-
att_def	xp_attitude_def*	-	Structure defining the attitude frames. It also defines the destination frame.	-	
time_interval	xo_time_interval*	-	start-stop time interval for the data generation	-	-
time_step	double*	-	Time step between records in the output file data.	seconds	-
data_type	long	-	Requested data type: angles or quaternions.	-	XD_ATT_QUATERNIONS XD_ATT_ANGLES
source_ref_type	long	-	Source reference type: External or Satellite.	-	XP_FRAME_FLAG_EXT XP_FRAME_FLAG_SAT
source_ref	long	-	Source reference CS	-	XL_BM2000, XL_HM2000, XL_GM2000, XL_MOD, XL_TOD, XL_PEF,

					XL_EF XL_LIF, XL_GALACTIC, XL_SAT_ORBITAL_REF, XL_SAT_NOMINAL_ATT, XL_SAT_ATT, XL_INSTR_ATT

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

- data_type: See XD_Attitude_data_type_enum [D_H_SUM].
- source_ref_type: See enumeration XP_Frame_flag_enum in current document (Table 143).
- source_ref: See enumeration XL_CS_rl_enum and XL_Attitude_fr_enum in [LIB_SUM].

7.57.4 Output parameters

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

Table 142: Output parameters of xp_gen_attitude_data function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
att_file	xd_att_file	-	Structure containing the list of attitude angles/quaternions	-	-
ierr	long	-	Error vector	-	-

7.57.5 Warnings and errors

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

This table also indicates the type of message return ed, 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 **xp_gen_attitude_data** function by calling the function of the EO_POINTING software library **xp_gen_code** (see [GEN_SUM]).

Table 143: Error messages of xp_gen_attitude_data function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Wrong input orbit Id. It is not initialized	No calculation performed	XP_CFI_GEN_ATT_DATA_ORBIT_INIT_ERR	0

ERR	Error in attitude initialization	No calculation performed	XP_CFI_GEN_ATT_DATA_ATT_INIT_ERR	1
ERR	Wrong input attitude definition structure: Attitude type has a wrong value	No calculation performed	XP_CFI_GEN_ATT_DATA_ATT_TARGET_TYPE_ERR	2
ERR	Wrong input attitude definition structure: Instrument attitude is not initialized	No calculation performed	XP_CFI_GEN_ATT_DATA_INSTR_ATT_ERR	3
ERR	Wrong input attitude definition structure: Satellite attitude is not initialized	No calculation performed	XP_CFI_GEN_ATT_DATA_SAT_ATT_ERR	4
ERR	Wrong input attitude definition structure: Satellite nominal attitude is not initialized	No calculation performed	XP_CFI_GEN_ATT_DATA_SAT_NOM_ATT_ERR	5
ERR	Could not compute the start interval UTC time	No calculation performed	XP_CFI_GEN_ATT_DATA_START_INTERVAL_ERR	6
ERR	Could not compute the stop interval UTC time	No calculation performed	XP_CFI_GEN_ATT_DATA_STOP_INTERVAL_ERR	7
ERR	Wrong input time step. It cannot be negative	No calculation performed	XP_CFI_GEN_ATT_DATA_TIME_STEP_ERR	8
ERR	Wrong input data type. It should be quaternions or angles	No calculation performed	XP_CFI_GEN_ATT_DATA_DATA_TYPE_ERR	9
ERR	Wrong input satellite source frame	No calculation performed	XP_CFI_GEN_ATT_DATA_SAT_CS_ERR	10
ERR	Wrong input exteranl source frame	No calculation performed	XP_CFI_GEN_ATT_DATA_EXTERNAL_CS_ERR	11
ERR	Wrong input source frame type. Should be external or satellite	No calculation performed	XP_CFI_GEN_ATT_DATA_CS_TYPE_ERR	12
ERR	Memory allocation error	No calculation performed	XP_CFI_GEN_ATT_DATA_MEM_ALLOC_ERR	13
ERR	Error computing the satellite state vector	No calculation performed	XP_CFI_GEN_ATT_DATA_OSV_COMP_ERR	14
ERR	Error in attitude compute	No calculation performed	XP_CFI_GEN_ATT_DATA_ATT_COMP_ERR	15
ERR	Could not compute the attitude data	No calculation performed	XP_CFI_GEN_ATT_DATA_GET_ATT_DATA_ERR	16

7.58 xp_gen_attitude_file

7.58.1 Overview

The **xp_gen_attitude_file** function creates an attitude file with the list of quaternions or attitude angles (roll, pitch, yaw) at an interval given by the user (with a regular time separation) that define the rotation between two reference frames:

- A source reference frame (given as input).
- The attitude frame given by the input attitude definition file.

7.58.2 Calling interface

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

```
#include <explorer_pointing.h>
{
    xo_orbit_id       orbit_id = {NULL};
    xp_attitude_def   att_def;
    xo_time_interval  time_interval;
    double            time_step;
    long              data_type;
    long              source_ref_type;
    long              source_ref;
    char              *output_dir,
    char              *file_class,
    long              *version_number,
    char              *fh_system,
    char              filename[XD_MAX_STR],
    long ierr[XP_NUM_ERR_GEN_ATTITUDE_FILE];

    long xp_gen_attitude_file(&orbit_id,
                              &att_def,
                              &time_interval,
                              &time_step,
                              &data_type,
                              &source_ref_type,
                              &source_ref,
                              output_dir,
                              file_class,
                              &version_number,
```

```
                                fh_system,
                                /* input/output */
                                filename,
                                ierr);

    /* Or, using the run_id */
    long run_id;
    long xp_gen_attitude_data_run(&run_id,
                                   &time_interval,
                                   &time_step,
                                   &data_type,
                                   &source_ref_type,
                                   &source_ref,
                                   output_dir,
                                   file_class,
                                   &version_number,
                                   fh_system,
                                   /* input/output */
                                   filename,
                                   ierr);
}
```

The XP_NUM_ERR_GEN_ATTITUDE_FILE constant is defined in the file *explorer_pointing.h.*

7.58.3 Input parameters

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

Table 144: Input parameters of xp_gen_attitude_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
orbit_id	xp_orbit_id*	-	Structure that contains the satellite orbit data	-	-
att_def	xp_attitude_def*	-	Structure defining the attitude frames. It also defines the destination frame.	-	
time_interval	xo_time_interval*	-	start-stop time interval for the data generation	-	-
time_step	double*	-	Time step between records in the output file	seconds	-

			data.		
data_type	long	-	Requested data type: angles or quaternions.	-	XD_ATT_QUATERNIONS XD_ATT_ANGLES
source_ref_type	long	-	Source reference type: External or Satellite.	-	XP_FRAME_FLAG_EXT XP_FRAME_FLAG_SAT
source_ref	long	-	Source reference CS	-	XL_BM2000, XL_HM2000, XL_GM2000, XL_MOD, XL_TOD, XL_PEF, XL_EF XL_LIF, XL_GALACTIC, XL_SAT_ORBITAL_REF, XL_SAT_NOMINAL_ATT, XL_SAT_ATT, XL_INSTR_ATT
output_dir	char	-	Directory for the output file. If empty, the current directory is chosen.	-	-
file_class	char	-	File class	-	-
version_number	long	-	File version	-	-
fh_system	char	-	system	-	-
filename	char	-	Output file name. If emtpy, the file is generated autmatically and returned to this variable.	-	-

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

- data_type: See XD_Attitude_data_type_enum [D_H_SUM].
- source_ref_type: See enumeration XP_Frame_flag_enum in current document (Table 146).
- source_ref: See enumeration XL_CS_rl_enum and XL_Attitude_fr_enum in [LIB_SUM].

7.58.4 Output parameters

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

Table 145: Output parameters of xp_gen_attitude_file function

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

7.58.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xp_gen_attitude_file** CFI function after translating the returned extended status flag into the equivalent list of error messages by calling the function of the EO_POINTING software library **xp_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 **xp_gen_attitude_file** function by calling the function of the EO_POINTING software library **xp_gen_code** (see [GEN_SUM]).

Table 146: Error messages of xp_gen_attitude_file function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Error generating attitude data	No calculation performed	XP_CFI_GEN_ATT_FILE_GENDATA_ERR	0
ERR	Could not get the Fixed header data	No calculation performed	XP_CFI_GEN_ATT_FILE_GENFHR_ERR	1
ERR	No data generated for the requested interval	No calculation performed	XP_CFI_GEN_ATT_FILE_NO_DATA_ERR	2
ERR	Memory allocation error	No calculation performed	XP_CFI_GEN_ATT_FILE_MEM_ERR	3
ERR	Error writing attitude file to disk	No calculation performed	XP_CFI_GEN_ATT_FILE_WRITE_ERR	4

7.58.6 Executable Program

The gen_attitude executable program can be called form a shell as:

gen_attitude

-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)
        }
        -orbtyp "orbit file type"
        -orbf  "orbit file name"
        -atdef "attitude definition file name"
       -attyp "attitude data type, angles or quaternions"
       -tstep "Time step between generated attitude records (seconds)”
       -cs "source reference frame"
       [-dir output_dir] (default: current directory)
       [-atf 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 ]
       [ -xd_v ]
       [ -xl_v ]
       [ -xo_v ]
       [ -xp_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.
- [-xd_v] option for EXPLORER_DATA_HANDLING Verbose mode.
- [-xl_v] option for EXPLORER_LIB Verbose mode.
- [-xo_v] option for EXPLORER_ORBIT Verbose mode.
- [-xp_v] option for EXPLORER_POINTING 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_1C,
  SENTINEL_2A, SENTINEL_2B, SENTINEL_2C,
  SENTINEL_3A, SENTINEL_3B, SENTINEL_3C,
  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,
  SEOSAT, GENERIC.
- Possible values for time_model: USER, NONE, IERS_B_PREDICTED,
  IERS_B_RESTITUTED,FOS_PREDICTED,
  FOS_RESTITUTED, DORIS_PRELIMNARY, DORIS_PRECISE,
  DORIS_NAVIGATOR, OSF.
- Possible values for time_ref and time_reference: UNDEF, TAI, UTC, UT1.
- Possible values for "orbit 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 "Attitude data type": ANGLES, QUATERNIONS.
- Possible values for "source reference frame": GALACTIC (= Galactic CS)
  BM2000 (= Barycentric Mean of 2000.0 CS)
  HM2000 (= Heliocentric Mean of 2000.0 CS)
  GM2000 (= Geocentric Mean of 2000.0 CS)
  MOD (= Mean of Date CS)
  TOD (= True of Date CS)
  PEF (= Pseudo Earth Fixed CS)
  EF (= Earth Fixed CS)
  LIF (= Launch Inertial CS)
  ORBITAL (= Satellite orbital frame CS)
  NOM_ATT (= Satellite nominal attitude CS)
  ATT (= Satellite attitude CS)
  INSTR (= Satellite instrument CS)

- Time references need to be initialized.

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:

1. TAI, GPS, UTC and UT1 input times (as in xl_time_ref_init) \n");

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

7.59 xp_attitude_close

7.59.1 Overview

The **xp_attitude_close** CFI function cleans up any memory allocation performed by the Attitude functions.

7.59.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     xp_attitude_id attitude_id = {NULL};
     long ierr[XP_NUM_ERR_ATTITUDE_CLOSE], status;

     status = xp_attitude_close(&attitude_id, ierr);
}
```

The XP_NUM_ERR_ATTITUDE_CLOSE constant is defined in the file *explorer_pointing.h.*

7.59.3 Input Parameters

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

Table 147: Input parameters of xp_attitude_close function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
attitude_id	xp_attitude_id *	-	Structure that contains the Attitude.	-	-

7.59.4 Output Parameters

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

Table 148: Output parameters of xp_attitude_close

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

7.59.5 Warnings and Errors

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

Table 149: Error messages of xp_attitude_close function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Could not close Attitude Id. The Attitude Id. is not initialized or it is being used	No calculation performed	XP_CFI_ATTITUDE_CLOSE_WRONG_ID_ERR	0

7.60 xp_attitude_get_id_data

7.60.1 Overview

The **xp_attitude_get_id_data** CFI function returns attitude initialization data.

7.60.2 Calling interface

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

```
#include <explorer_lib.h>
{
        xp_attitude_id attitude_id;
        long status;
        xp_attitude_id_data data;
        status = xp_attitude_get_id_data (&attitude_id,
                                          &data);
}
```

7.60.3 Input parameters

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

Table 150: Input parameters of xp_attitude_get_id_data function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
attitude_id	xp_attitude_id *	-	Structure that contains the Attitude.	-	-

7.60.4 Output parameters

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

Table 151: Output parameters of xp_attitude_get_id_data function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_attitude_get_id_data	long	-	Status flag	-	-
data	xp_attitude_id_data	-	Attitude initialization data	-	-

7.60.5 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 attitude_id was not initialised.
- The attitude_id initialization does not allow the use of this function.

7.61 xp_attitude_set_id_data

7.61.1 Overview

The **xp_attitude_set_id_data** CFI function changes the harmonic data used for the satellite attitude initialization. Input matrix is checked to be orthonormal.

7.61.2 Calling interface

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

```
#include <explorer_lib.h>
{
       xp_attitude_id attitude_id;
       long status;
       xp_attitude_id_data data;
       status = xp_attitude_set_id_data (&attitude_id,
                                          &data);
}
```

7.61.3 Input parameters

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

Table 152: Input parameters of xp_attitude_set_id_data function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
attitude_id	xp_attitude_id *	-	Structure that contains the Attitude (input / output parameter)	-	-
data	xp_attitude_id_data	-	Attitude initialization data	-	-

7.61.4 Output parameters

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

Table 153: Output parameters of xp_attitude_set_id_data function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_attitude_set_id_data	long	-	Status flag	-	-
attitude_id	xp_attitude_id *	-	Structure that contains the Attitude. (input / output parameter)	-	-

7.61.5 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 attitude_id was not initialised.
- The attitude_id initialization does not allow the use of this function.

7.62 xp_attitude_get_model_id

7.62.1 Overview

The **xp_attitude_get_model_id** CFI function retrieves the model ID from the input attitude ID.

7.62.2 Calling interface

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

```
#include <explorer_lib.h>
{
        xp_attitude_id attitude_id = {NULL};
        xl_model_id model_id;
        model_id =     xp_attitude_get_model_id (&attitude_id);
}
```

7.62.3 Input parameters

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

Table 154: Input parameters of xp_attitude_get_model_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
attitude_id	xp_attitude_id *	-	Structure that contains the attitude	-	-

7.62.4 Output parameters

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

Table 155: Output parameters of xp_attitude_get_model_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_attitude_get_model_id	long	-	Status flag	-	-

7.62.5 Warnings and errors

This function does not return any error/warning code. If there is an error, then the returned model ID will be set to NULL (no initialised)

The possible causes of error are:

- The attitude_id was not initialised.

7.63 xp_attitude_get_time_id

7.63.1 Overview

The **xp_attitude_get_time_id** CFI function retrieves the time_id from the input attitude_id.

7.63.2 Calling interface

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

```
#include <explorer_lib.h>
{
        xp_attitude_id attitude_id = {NULL};
        xl_time_id time_id;
        time_id = xp_attitude_get_time_id (&attitude_id);
}
```

7.63.3 Input parameters

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

Table 156: Input parameters of xp_attitude_get_time_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
attitude_id	xp_attitude_id *	-	Structure that contains the attitude	-	-

7.63.4 Output parameters

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

Table 157: Output parameters of xp_attitude_get_time_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
time_id	xl_time_id	-	time_id used for the attitude_id initialization	-	-

7.63.5 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).

7.64 xp_attitude_get_sat_nom_trans_id

7.64.1 Overview

The **xp_attitude_get_sat_nom_trans_id** CFI function retrieves the snmt_id from the input attitude_id.

7.64.2 Calling interface

The calling interface of the **xp_attitude_get_sat_nom_trans_id** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_lib.h>
{
        xp_attitude_id attitude_id = {NULL};
        xp_sat_nom_trans_id snmt_id;
        snmt_id = xp_attitude_get_sat_nom_trans_id (&attitude_id);
}
```

7.64.3 Input parameters

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

Table 158: Input parameters of xp_attitude_get_sat_nom_trans_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
attitude_id	xp_attitude_id *	-	Structure that contains the attitude	-	-

7.64.4 Output parameters

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

Table 159: Output parameters of xp_attitude_get_sat_nom_trans_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
snmt_id	xp_sat_nom_trans_id	-	snmt_id used for the attitude_id initialization	-	-

7.64.5 Warnings and errors

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

7.65 xp_attitude_get_sat_trans_id

7.65.1 Overview

The **xp_attitude_get_sat_trans_id** CFI function retrieves the st_id from the input attitude_id.

7.65.2 Calling interface

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

```
#include <explorer_lib.h>
{
       xp_attitude_id attitude_id = {NULL};
       xp_sat_trans_id st_id;
       st_id = xp_attitude_get_sat_trans_id (&attitude_id);
}
```

7.65.3 Input parameters

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

Table 160: Input parameters of xp_attitude_get_sat_trans_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
attitude_id	xp_attitude_id *	-	Structure that contains the attitude	-	-

7.65.4 Output parameters

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

Table 161: Output parameters of xp_attitude_get_sat_trans_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
st_id	xp_sat_trans_id	-	st_id used for the attitude_id	-	-

			initialization		

7.65.5 Warnings and errors

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

7.66 xp_attitude_get_instr_id

7.66.1 Overview

The **xp_attitude_get_instr_id** CFI function retrieves the inst_id from the input attitude_id.

7.66.2 Calling interface

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

```
#include <explorer_lib.h>
{
        xp_attitude_id attitude_id = {NULL};
        xp_instr_trans_id inst_id;
        inst_id = xp_attitude_get_instr_id (&attitude_id);
}
```

7.66.3 Input parameters

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

Table 162: Input parameters of xp_attitude_get_instr_id function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
attitude_id	xp_attitude_id *	-	Structure that contains the attitude	-	-

7.66.4 Output parameters

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

Table 163: Output parameters of xp_attitude_get_instr_id function

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

inst_id	xp_instr_trans_id	-	inst_id used for the attitude_id initialization	-	-

7.66.5 Warnings and errors

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

7.67 xp_change_frame

7.67.1 Overview

The **xp_change_frame** CFI function changes the coordinate or attitude frame of a location or direction by keeping the location or direction in inertial space identical. Both all coordinate frames and all attitude frames are supported.

When changing the frame for a location (mode_flag = XP_MODE_FLAG_LOCATION), the difference between the frame origins is taken into account.

When changing the frame for a direction (mode_flag = XP_MODE_FLAG_DIRECTION), the output of the function is a direction, that does not depend on the origin of reference frame of the input vector. Therefore, in this specific case, the instrument offsets are not taken into account.

7.67.2 Calling interface

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

```
#include <explorer_pointing.h>
{
    xl_model_id model_id = {NULL};
    long sat_id, mode_flag, frame_flag_in, frame_id_in,
         frame_flag_out, frame_id_out, time_ref;
    xl_time_id          time_id = {NULL};
    xp_sat_nom_trans_id sat_nom_trans_id = {NULL};
    xp_sat_trans_id     sat_trans_id = {NULL};
    xp_instr_trans_id   instr_trans_id = {NULL};
    double time;
    double pos[3], vel[3], acc[3];
    long deriv;
    double vec_in[3], vec_rate_in[3], vec_rate_rate_in[3];
```

```
    double vec_out[3], vec_rate_out[3], vec_rate_rate_out[3];
    long ierr[XP_NUM_ERR_CHANGE_FRAME], status;
    status = xp_change_frame (&sat_id, &model_id,
                              &time_id,
                              &sat_nom_trans_id,
                              &sat_trans_id,
                              &instr_trans_id,
                              &mode_flag,
                              &frame_flag_in, &frame_id_in,
                              &frame_flag_out, &frame_id_out,
                              &time_ref, &time,
                              pos, vel, acc, &deriv,
                              vec_in, vec_rate_in, vec_rate_rate_in,
                              vec_out, vec_rate_out, vec_rate_rate_out,
                              ierr);

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

    status =    xp_change_frame_run (&run_id,
                              &mode_flag,
                              &frame_flag_in, &frame_id_in,
                              &frame_flag_out, &frame_id_out,
                              &time_ref, &time,
                              pos, vel, acc, &deriv,
                              vec_in, vec_rate_in, vec_rate_rate_in,
                              vec_out, vec_rate_out, vec_rate_rate_out,
                              ierr);

}
```

The `XP_NUM_ERR_CHANGE_FRAME` constant is defined in the file *explorer_pointing.h.*

7.67.3 Input parameters

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

Table 164: Input parameters of xp_change_frame 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.	-	-
sat_nom_trans_id	xp_sat_nom_trans_id*	-	Structure that contains the Sat. Nom. Trans.	-	-
sat_trans_id	xp_sat_trans_id*	-	Structure that contains the Sat. Trans.	-	-
instr_trans_id	xp_instr_trans_id*	-	Structure that contains the Instr. Trans.	-	-
mode_flag	long *	-	Selection of location or direction calculus		Complete
frame_flag_in	long *	-	Selection of Coordinate or Attitude Frame on input		Complete
frame_id_in	long *		Coordinate Frame id or Attitude Frame id on input		Complete
frame_flag_out	long *	-	Selection of Coordinate or Attitude Frame on output		Complete
frame_id_out	long *		Coordinate Frame id or Attitude Frame id on output		Complete
time_ref	long *	-	Time reference ID	-	Complete
time	double	-	Time in Processing Format	Decimal days, MJD2000	[-18262.0,36524.0]
pos[3]	double	all	Satellite position vector (Earth Fixed CS)	m	-
vel[3]	double	all	Satellite velocity vector (Earth Fixed CS)	m/s	-
acc[3]	double	all	Satellite acceleration vector (Earth Fixed CS)	m/s^2	-
deriv	long *	-	Derivative ID	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND

vec_in[3]	double	all	Position (direction) vector (Frame in)	m or -	-
vec_rate_in[3]	double	all	Velocity (direction) vector (Frame in)	m/s or 1/s	-
vec_rate_rate_in[3]	double	all	Acceleration (direction) vector (Frame in)	m/s2 or 1/s2	-

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].
- Selection of location or direction calculus: mode_flag. See current document, Table 3.
- Selection of Coordinate or Attitude Frame: frame_flag. See current document, Table 3.

7.67.4 Output parameters

The output parameters of the **xp_change_frame** CFI function are

Table 165: Output parameters of xp_change_frame function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
vec_out[3]	double	all	Position (direction) vector (Frame out)	m or -	-
vec_rate_out[3]	double	all	Velocity (direction) vector (Frame out)	m/s or 1/s	-
vec_rate_rate_out[3]	double	all	Acceleration (direction) vector (Frame out)	m/s^2 or $1/s^2$	-
ierr	long	-	Error vector	-	-

7.67.5 Warnings and errors

Next table lists the possible error messages that can be returned by the **xp_change_frame** CFI function after translating the returned extended status flag into the equivalent list of error messages by calling the function of the EO_POINTING software library **xp_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 **xp_change_frame** function by calling the function of the EO_POINTING software library **xp_get_code** (see [GEN_SUM]).

Table 166: Error messages of xp_change_frame function

Error	Error message	Cause and impact	Error code	Error

type				No
ERR	Could not initialize the attitude	No calculation performed	XP_CHANGE_FRAME_ ATTITUDE_INIT_ERR	0
ERR	Frame input flag is not correct	No calculation performed	XP_CHANGE_FRAME_I NPUT_FRAME_ERR	1
ERR	Frame output flag is not correct	No calculation performed	XP_CHANGE_FRAME_ OUTPUT_FRAME_ERR	2
ERR	Error calling xl_change_cart_cs	No calculation performed	XP_CHANGE_FRAME_ CHANGE_CART_CS_ER R	3
ERR	Could not compute the attitude	No calculation performed	XP_CHANGE_FRAME_ ATTITUDE_COMP_ERR	4
ERR	The Attitude Id could not be closed	No calculation performed	XP_CHANGE_FRAME_ ATTITUDE_CLOSE_ER R	5

7.68 xp_atmos_init

7.68.1 Overview

The **xp_atmos_init** CFI function initialises the atmospheric model for a given satellite. The initialised values will be stored in the *atmos_id* output structure.

7.68.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     long atmos_mode, atmos_model;
     char atmos_file[XL_MAX_STR];
     xp_atmos_id atmos_id = {NULL};
     long ierr[XP_NUM_ERR_ATMOS_INIT], status;

     status = xp_atmos_init(&atmos_mode, &atmos_model,   atmos_file,
                            &atmos_id, ierr);
}
```

The `XP_NUM_ERR_ATMOS_INIT` constant is defined in the file *explorer_pointing.h.*

7.68.3 Input Parameters

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

Table 167: Input parameters of xp_atmos_init function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
atmos_mode	long *	-	Atmosphere initialization mode	-	Complete
atmos_model	long *	-	Not Used in the current implementation.	-	-
atmos_file	char[]	-	File used for atmosphere initialization. It is required when the input atmos_mode is: - User initalization mode (n-z table, see section 10) - User LUT mode	-	Complete

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

- Atmosphere Initialization Mode: atmos_mode. See current document, Table 3 .

7.68.4 Output Parameters

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

Table 168: Output parameters of xp_atmos_init

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
atmos_id	xp_atmos_id*	-	Structure that contains the atmosphere initialization.	-	-
ierr	long	-	Error vector	-	-

7.68.5 Warnings and Errors

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

Table 169: Error messages of xp_atmos_init function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Atmosphere Mode ID is not correct	No calculation performed	XP_CFI_ATMOS_INIT_MODE_ID_ERR	0
ERR	Atmosphere Model ID is not correct	No calculation performed	XP_CFI_ATMOS_INIT_MODEL_ID_ERR	1
ERR	Atmosphere initialization file could not be opened	No calculation performed	XP_CFI_ATMOS_INIT_FILE_NOT_OPEN_ERR	2
ERR	Unable to store atmosphere initialization file (not enough memory	No calculation performed	XP_CFI_ATMOS_INIT_MEMORY_ERR	3
ERR	Error while reading atmosphere initialization file	No calculation performed	XP_CFI_ATMOS_INIT_FILE_READING_ERR	4

7.69 xp_atmos_close

7.69.1 Overview

The **xp_atmos_close** CFI function cleans up any memory allocation performed by the **xp_atmos_init** functions.

7.69.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     xp_atmos_id atmos_id = {NULL};
     long ierr[XP_NUM_ERR_ATMOS_CLOSE], status;

     status = xp_atmos_close(&atmos_id, ierr);
}
```

The `XP_NUM_ERR_ATMOS_CLOSE` constant is defined in the file *explorer_pointing.h.*

7.69.3 Input Parameters

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

Table 170: Input parameters of xp_atmos_close function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
atmos_id	xp_atmos_id*	-	Structure that contains the atmosphere initialization.	-	-

7.69.4 Output Parameters

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

Table 171: Output parameters of xp_atmos_close

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

7.69.5 Warnings and Errors

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

Table 172: Error messages of xp_atmos_close function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Could not close the Atmos. Id. as it is not initialized or it is being used	No calculation performed	XP_CFI_ATMOS_CLOSE_ WRONG_ID_ERR	0

7.70 xp_atmos_get_id_data

7.70.1 Overview

The **xp_atmos_get_id_data** CFI function returns atmospheric initialization data.

7.70.2 Calling interface

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

```
#include <explorer_lib.h>
{
        xp_atmos_id atmos_id;
        long status;
        xp_atmos_id_data data;
        status = xp_atmos_get_id_data (&atmos_id, &data);
}
```

7.70.3 Input parameters

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

Table 173: Input parameters of xp_atmos_get_id_data function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
atmos_id	xp_atmos_id *	-	Atmospheric Id.	-	-

7.70.4 Output parameters

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

Table 174: Output parameters of xp_atmos_get_id_data function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_atmos_get_id_data	long	-	Status flag	-	-
data	xp_atmos_id_data	-	Atmospheric initialization data	-	-

7.70.5 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 atmos_id was not initialised.

7.71 xp_dem_init

7.71.1 Overview

The **xp_dem_init** CFI function initializes the digital elevation model (DEM). The DEM is initialized using the DEM configuration file (see [D_H_SUM]) which contains some characteristics that can be configured (see [MCD] for further details about the DEM models).

Finally the initalisation values will be stored in the *dem_id* output structure.

7.71.2 Location of DEM dataset

The DEM files are looked for in the directory specified in the field Directory in the DEM configuration file (see [D_H_SUM]). If this field is empty, the DEM files are looked for in the directory where the DEM configuration file is placed.

7.71.3Access to DEM dataset

Depending on the Cache_Type field in the DEM configuration file (see [D_H_SUM]), one of the following methods is used to access the DEM dataset:

- FIFO_CACHE (default): memory is reserved for holding DEM data. As soon as an altitude value is requested and is not yet available in memory, the corresponding data file is loaded in memory. When the maximum size of reserved memory (configurable with the field Cache_Max_Size) is exceeded, memory is made available with a First In – First Out policy, that is memory correspondent to the file loaded earliest is made available for the file to be loaded.
- PRELOAD_CACHE: memory is reserved for holding DEM data. The user shall load in memory the needed files in advance via the function xp_dem_id_configure (see section 7.76). Request of a value not available in memory would result in an error.
- NO_CACHE: no memory is reserved for holding DEM data. The dataset is accessed via a single direct I/O access to the file storing the requested value.

In the case of GDEM, due to the special structure of the tiles, loading them to memory can take much time, so it is not recommended the use of cache methods.

Choice of the method that best fits user’s needs depends on many aspects including HW/SW setup and the type of user application:

1) The FIFO_CACHE is recommended for user applications able to request to the operating system a large amount of physical memory and that require making a large numbers of DEM computations per DEM area i.e. when several DEM computations are done reading the same file or small set of files covering the same region.
2) The PRELOAD_CACHE is recommended for multithreading applications. Note that memory holding DEM data can be shared amongst several threads. In the FIFO_CACHE, as memory content can change at runtime, mutual exclusion mechanisms are implemented in order to avoid threads to access inconsistent data. Such mechanisms are not needed in the PRELOAD_CACHE methods and therefore multithreading applications may run more efficiently. However the user is requested to estimate the area (in terms of the longitude/latitude boundaries) that will be requested during computations.

3) The above methods using memory to hold DEM data do not improve performance (or make even performance worse) of applications running with low amount of physical memory available or when DEM request is sporadic per DEM area. In all these cases, the user is recommended to set Cache_Type to NO_CACHE. For example, applications making sporadic accesses and in different DEM areas will not benefit of the caching methods, as the advantage of having a fast access to data is lost by the disadvantage of continuously load new files correspondent to different areas.

The default configuration is (i.e. when fields are not provided in the file):

- Cache_Type = FIFO_CACHE
- Cache_Max_Size = 2 GigaBytes

Values of Cache_Type and Cache_Max_Size can be changed at runtime under certain conditions using the function xp_dem_id_configure.

Memory is allocated using the malloc() C library function. Therefore performance of DEM access using caching strictly depends on the implementation of such library and on memory management from the Operating System. Performance of access to memory depends on many factors that can be tuned by the user. For the sake of example, if, as it normally happens in Linux systems, the memory request is larger than a given threshold size, the memory will be allocated in the virtual memory space and this may result in several page faults at runtime, leading to inefficiencies in the execution. The user can improve this by tuning the threshold size (i.e. using the mallopt() C library, if available). In order to get the best advantages from the caching methods, the user is therefore recommended to evaluate and tackle platform specific issues to memory allocation and management.

7.71.4 DEM maximum altitude algorithm

If the mini tile configuration is provided (MiniTile_Configuration tag) in the DEM configuration file (see [D_H_SUM]), the maximum altitude algoritm is used to compute DEM intersection. This algorithm consists in the following:

- The tiles are divided in sub-sections (mini-tiles), according to the configuration provided in DEM configuration file: Lon_Size and Lat_Size tags. For example, if the tile is 15 degrees long in longitude and 15 degrees long in latitude, and Lon_Size and Lat_Size are 5 degrees and 5 degrees respectively, the tile would have 9 mini-tiles equal in size between them.
- In the DEM configuration of mini-tiles, also the tag Filename is provided, which corresponds to the name (or path) of the binary file which contains the maximum altitude corresponding to each mini-tile. This file can be generated with function xp_gen_dem_max_altitude_file. If no path is provided, the binary file in looked for in current directory and the DEM directory.
- Internally, the algorithm checks if the altitude of the rays when crossing above each mini-tile is higher or lower than maximum altitude contained in the mini-tile. If the altitude is lower, the mini-tile is computed to look for an intersection; if not, the mini-tile is skipped and the following mini-tile is checked.

Note: the algorithm is not executed if the difference in latitude/longitude of start and end points in ray search is less than mini-tile size.

7.71.5 DEM Geoid computation

The DEM ACE2 and GDEM V2 files provide the altitude with respect to its reference geoid. In the internal DEM computations, the altitude is transformed to altitude over the reference ellipsoid. To perform this operation, a number of harmonics must be used, which can be configured with the following DEM configuration user tags (see [D_H_SUM]):

- Geoid_Computation tag: this field can take the values:
  - "Enabled": geoid computation is performed.
  - "Disabled": geoid computation is not performed.
- Geoid_Nof_Harmonics tag: the number of harmonics to be used in geoid computation.

If this fields are not provided, the default values are:

- Geoid_Computation: Enabled
- Geoid_Nof_Harmonics: 30

The computation precision increases with the number of harmonics (maximum is 360 harmonics) but the runtime performance gets worse. The computation of the geoid at runtime can be avoided by generating offline a DEM dataset storing altitudes w.r.t ellipsoid using xp_gen_dem_altitudes_from_ellipsoid function. In this case, when the DEM is used, the geoid computation shall be disabled in the DEM configuration file.

7.71.6 Calling Interface

The calling interface of the **xp_dem_init** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
     long mode, model;
     char dem_file[XL_MAX_STR];
     xp_dem_id dem_id = {NULL};
     long ierr[XP_NUM_ERR_DEM_INIT], status;

     status = xp_dem_init(&mode, &model, dem_file, &dem_id, ierr);
}
```

The `XP_NUM_ERR_DEM_INIT` constant is defined in the file *explorer_pointing.h.*

7.71.7 Input Parameters

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

Table 175: Input parameters of xp_dem_init function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
mode	long *	-	Digital Elevation Model initialization mode. This parameter has no effect in current implementation (the DEM type is taken from configuration file), but a warning will be raised if the value does not coincide with the one in configuration file.	-	Complete
model	long *	-	Digital Elevation Model initialization model (dummy in current implementation)	-	Complete
dem_file	char[]	-	File used for DEM initialization (See DEM Configuration file in [D_H_SUM])	-	Complete

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

- DEM Initilization Mode: initialization mode (according to XD_Dem_model_enum in [D_H_SUM])

7.71.8 Output Parameters

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

Table 176: Output parameters of xp_dem_init

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
dem_id	xp_dem_id*	-	Structure that contains the DEM initialization.	-	-
ierr	long	-	Error vector	-	-

7.71.9 Warnings and Errors

Next table lists the possible error messages that can be returned by the **xp_dem_init** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_POINTING software library **xp_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 by translating the error vector returned by the **xp_dem_init** function by calling the function of the EO_POINTING software library **xp_get_code** (see [GEN_SUM])

Table 177: Error messages of xp_dem_init function

Error type	Error message	Cause and impact	Error code	Error No
ERR	DEM Mode ID is not correct	No calculation performed	XP_CFI_DEM_INIT_MODE_ID_ERR	0
ERR	DEM Model ID is not correct	No calculation performed	XP_CFI_DEM_INIT_MODEL_ID_ERR	1
ERR	DEM initialization file could not be opened	No calculation performed	XP_CFI_DEM_INIT_FILE_NOT_OPEN_ERR	2
ERR	Unable to store DEM initialization file (not enough memory)	No calculation performed	XP_CFI_DEM_INIT_MEMORY_ERR	3
ERR	Error while reading DEM initialization file. In case of using a Generic Raster DEM, this error message is used also to indicate problems in 'dem_raster_configuration.xml'.	No calculation performed	XP_CFI_DEM_INIT_FILE_READING_ERR	4
WARN	Default DEM values at Poles will be taken	Calculation performed. If required, default altitude values at the poles will be used.	XP_CFI_DEM_INIT_FILE_READING_WARN	5
WARN	DEM file mode and input mode are not the same	No calculation performed	XP_CFI_DEM_INIT_WRONG_MODEL_WARN	6
WARN	Input DEM configuration file version is deprecated	Calculation performed	XP_CFI_DEM_INIT_DEPRECATED_WARN	7
WARN	DEM Cache Type not supplied, assuming FIFO_CACHE with maximum size of 2 GB	Calculation performed	XP_CFI_DEM_INIT_CACHE_WARN	8
ERR	Error initializing TILE Database	No calculation performed	XP_CFI_DEM_INIT_TILE_DB_ERR	9
ERR	Error computing altitude at the poles	No calculation performed	XP_CFI_DEM_INIT_READ_POLES_ERR	10
WARN	DEM files at the poles not found. Default altitude will be	Calculation performed	XP_CFI_DEM_INIT_READ_POLES_WARN	11

	used			
WARN	Mini tile longitude size adjusted to %lf deg	Calculation performed	XP_CFI_DEM_INIT_MINI_TILE_LON_SIZE_WARN	12
WARN	Mini tile latitude size adjusted to %lf deg	Calculation performed	XP_CFI_DEM_INIT_MINI_TILE_LAT_SIZE_WARN	13
ERR	Error opening maximum altitude file %s	No calculation performed	XP_CFI_DEM_INIT_OPEN_MAX_ALT_FILE_ERR	14
ERR	Error reading maximum altitude file %s	No calculation performed	XP_CFI_DEM_INIT_READ_MAX_ALT_FILE_ERR	15

7.72 xp_dem_close

7.72.1 Overview

The **xp_dem_close** CFI function cleans up any memory allocation performed by the **xp_dem_init** functions.

7.72.2 Calling Interface

The calling interface of the **xp_dem_close** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
     xp_dem_id dem_id = {NULL};
     long ierr[XP_NUM_ERR_DEM_CLOSE], status;

     status = xp_dem_close(&dem_id, ierr);
}
```

The `XP_NUM_ERR_DEM_CLOSE` constant is defined in the file *explorer_pointing.h.*

7.72.3 Input Parameters

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

Table 178: Input parameters of xp_dem_close function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
dem_id	xp_dem_id*	-	Structure that contains the DEM initialization.	-	-

7.72.4 Output Parameters

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

Table 179: Output parameters of xp_dem_close

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

7.72.5 Warnings and Errors

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

Table 180: Error messages of xp_dem_close function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Could not close the Dem. Id. as it is not initialized or it is being used	No calculation performed	XP_CFI_DEM_CLOSE_WRONG_ID_ERR	0

7.73 xp_dem_compute

7.73.1 Overview

The **xp_dem_compute** CFI function compute the altitude over the see level for a point in the Earth. The altitude is calculated from the altitudes read from a digital elevation model (DEM).

7.73.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
    xl_model_id model_id = {NULL};
    xp_dem_id   dem_id   = {NULL};
    long ierr[XP_NUM_ERR_DEM_COMPUTE], status;
    double lon, lat, alt;
    status = xp_dem_compute(&model_id, &dem_id,
                            &lon, &lat,
                            &alt, ierr);
}
```

The `XP_NUM_ERR_DEM_COMPUTE` constant is defined in the file *explorer_pointing.h*.

7.73.3 Input Parameters

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

Table 181: Input parameters of xp_dem_compute function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
model_id	xl_model_id*	-	Model ID	-	-
dem_id	xp_dem_id*	-	Structure that contains the DEM initialization.	-	-
lon	double	-	Input longitude	degrees	[0, 360)
lat	double	-	Input latitude	degrees	[-90, 90]

7.73.4 Output Parameters

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

Table 182: Output parameters of xp_dem_compute

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
alt	double	-	Altitude	meters	-
ierr	long	-	Error vector	-	-

7.73.5 Warnings and Errors

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

Table 183: Error messages of xp_dem_compute function

Error type	Error message	Cause and impact	Error code	Error No

ERR	Error getting cell altitude	No calculation performed	XP_CFI_DEM_COMPUTE_GET_CELL_ERR	0
ERR	Error allocating memory	No calculation performed	XP_CFI_DEM_COMPUTE_MEMORY_ERR	1
WARN	Void value detected. Altitude computation based on the ellipsoid.	Calculation performed. A message informs the user.	XP_CFI_DEM_COMPUTE_VOID_VALUE_DETECTED_WARN	2

7.74 xp_dem_get_info

7.74.1 Overview

The **xp_dem_get_info** CFI function reads DEM information for a given geodetic point.

7.74.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
    xl_model_id model_id = {NULL};
    xp_dem_id   dem_id   = {NULL};
    long ierr[XP_NUM_ERR_DEM_GET_INFO], status;
    double lon, lat;
    xp_dem_info dem_info
    status = xp_dem_get_info(&model_id, &dem_id,
                             &lon, &lat,
                             &dem_info,
                             ierr);
}
```

The `XP_NUM_ERR_DEM_GET_INFO` constant is defined in the file *explorer_pointing.h*.

7.74.3 Input Parameters

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

Table 184: Input parameters of xp_dem_get_info function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
model_id	xl_model_id*	-	Model ID	-	-
dem_id	xp_dem_id*	-	Structure that contains the DEM initialization.	-	-
lon	double	-	Input longitude	degrees	[0, 360)
lat	double	-	Input latitude	degrees	[-90, 90]

7.74.4 Output Parameters

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

Table 185: Output parameters of xp_dem_get_info

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
dem_info	xp_dem_info	-	Structure containing DEM information	-	-
ierr	long	-	Error vector	-	-

7.74.5 Warnings and Errors

Next table lists the possible error messages that can be returned by the **xp_dem_get_info** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_POINTING software library **xp_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 by translating the error vector returned by the **xp_dem_ get_info** function by calling the function of the EO_POINTING software library **xp_get_code** (see [GEN_SUM]).

Table 186: Error messages of xp_dem_get_info function

Error	Error message	Cause and impact	Error code	Error

type				No
ERR	DEM Id. is not initalized	No calculation performed	XP_CFI_DEM_GET_INFO_STATUS_ERR	0
ERR	Input longitude is out of allowed range [0, 360]	No calculation performed	XP_CFI_DEM_GET_INFO_WRONG_LONGITUDE_ERR	1
ERR	Input latitude is out of allowed range [90, -90]	No calculation performed	XP_CFI_DEM_GET_INFO_WRONG_LATITUDE_ERR	2
ERR	Could not open DEM file: %s	No calculation performed	XP_CFI_DEM_GET_INFO_OPEN_FILE_ERR	3
ERR	Could not read DEM file: %s	No calculation performed	XP_CFI_DEM_GET_INFO_READ_FILE_ERR	4
ERR	Could not read dem_raster_configuration.xml	No calculation performed	XP_DEM_GET_INFO_READ_RASTER_ERR	5

7.75 xp_dem_get_id_data

7.75.1 Overview

The **xp_dem_get_id_data** CFI function returns DEM initialization data.

7.75.2 Calling interface

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

```
#include <explorer_lib.h>
{
        xp_dem_id dem_id;
        long status;
        xp_dem_id_data data;
        status = xp_dem_get_id_data (&dem_id, &data);
}
```

7.75.3 Input parameters

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

Table 187: Input parameters of xp_dem_get_id_data function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
dem_id	xp_dem_id *	-	Structure that contains the DEM initialization.	-	-

7.75.4 Output parameters

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

Table 188: Output parameters of xp_dem_get_id_data function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_dem_get_id_data	long	-	Status flag	-	-
data	xp_dem_id_data	-	DEM initialization data	-	-

7.75.5 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 dem_id was not initialised.

7.76 xp_dem_id_configure

7.76.1 Overview

The **xp_dem_id_configure** CFI function performs configuration operations on DEM cache. The following operations can be performed:

- CLEAR CACHE: all the tiles in the cache are unloaded but cache memory is not freed.
- FREE CACHE: all the tiles in the cache are unloaded and cache memory is freed.
- SET MAXIMUM CACHE SIZE: this operation can only be performed for FIFO cache. A new maximum size for cache is set. If there are more tiles loaded in cache than new maximum size, the tiles are unloaded in a FIFO (First in- First out) order till new maximum size is reached.
- LOAD TILE SET: this operation can only be performed for PRELOAD cache. A set of tiles corresponding to an input rectangular longitude-latitude area is loaded in cache.

7.76.2 Calling Interface

The calling interface of the **xp_dem_id_configure** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
    xp_dem_id   dem_id   = {NULL};
    long ierr[XP_NUM_ERR_DEM_ID_CONFIGURE], status;
    xp_dem_id_config config;

    status = xp_dem_id_configure(&dem_id, &config,
                                 ierr);
}
```

The `XP_NUM_ERR_DEM_ID_CONFIGURE` constant is defined in the file *explorer_pointing.h*.

7.76.3 Input Parameters

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

Table 189: Input parameters of xp_dem_id_configure function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
dem_id	xp_dem_id*	-	Structure that contains the DEM initialization.	-	-
config	xp_dem_id_config	-	Input operation on cache	-	- Possible values for command field: XP_LOAD_TILE_SET XP_CLEAR_CACHE XP_FREE_CACHE XP_SET_MAX_SIZE - For rectangular area: 0. ≤ longitude ≤ 360. -90. ≤ latitude ≤ 90.

7.76.4 Output Parameters

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

Table 190: Output parameters of xp_dem_id_configure

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_dem_id_configure	long	-	Status flag	-	-
ierr	long	-	Error vector	-	-

7.76.5 Warnings and Errors

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

Table 191: Error messages of xp_dem_id_configure function

Error type	Error message	Cause and impact	Error code	Error No
ERR	DEM id configured without cache	No calculation performed	XP_CFI_DEM_ID_CONFIGURE_NO_CACHE_ERR	0
ERR	Wrong configuration command provided	No calculation performed	XP_CFI_DEM_ID_CONFIGURE_WRONG_COMMAND_ERR	1
ERR	Error allocating memory	No calculation performed	XP_CFI_DEM_ID_CONFIGURE_MEMORY_ERR	2
ERR	Error in input longitudes	No calculation performed	XP_CFI_DEM_ID_CONFIGURE_LON_ERR	3
ERR	Error in input latitudes	No calculation performed	XP_CFI_DEM_ID_CONFIGURE_LAT_ERR	4
ERR	Requested area needs more memory than maximum cache size	No calculation performed	XP_CFI_DEM_ID_CONFIGURE_MAX_CACHE_ERR	5
ERR	Error loading tile	No calculation performed	XP_CFI_DEM_ID_CONFIGURE_LOAD_TILE_ERR	6
ERR	Error locking thread	No calculation performed	XP_CFI_DEM_ID_CONFIGURE_LOCK_THREAD_ERR	7

7.77 xp_dem_get_cell_value

7.77.1 Overview

The **xp_dem_get_cell_value** CFI function retrives the altitude value for the corresponding DEM and the given row and column.

The altitude value returned by the function is the value stored in the corresponding DEM file (without any processing of the value). Note that some DEM's can give this value as the altitude over the ellipsoid while others give the altitude over the geoid.

The row/column value refers to the number of row/column considering a DEM covering the whole Earth.

This way, row 0 corresponds to the first row in DEM that gives the altitudes at latitude 90deg south and the last row will contain the altitudes at latitude 90deg north.

The column 0 corresponds to the altitudes for longitude 0 deg while the last column refers to the altitudes at longitude 360 deg.

The total number of rows/columns can be get with the function **xp_dem_get_id_data (t**he returned structure contains these numbers: **xp_dem_id_data.dem_metadata.n_rows** and **xp_dem_id_data.dem_metadata.n_cols)**. Note that the total number of rows/columns of the DEM is related to the DEM resolution as follows:

number of rows = 180deg / (resolution along latitude axis)

number of columns = 360deg / (resolution along latitude axis)

For instance, a DEM with a resolution of 30 arcsecond:

number of rows = 180deg / (30/3600) = 21600

number of columns = 360deg / (30/3600) = 43200

7.77.2 Calling Interface

The calling interface of the **xp_dem_get_cell_value** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
    xp_dem_id   dem_id   = {NULL};
    long ierr[XP_NUM_ERR_DEM_GET_CELL_VALUE], status;
    long row;
    long column;
    double value;

    status = xp_dem_get_cell_value(&dem_id, row, column,
                                   &value, ierr);
}
```

The `XP_NUM_ERR_DEM_GET_CELL_VALUE` constant is defined in the file *explorer_pointing.h.*

7.77.3 Input Parameters

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

Table 192: Input parameters of xp_dem_get_cell_value function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
dem_id	xp_dem_id*	-	Structure that contains the DEM initialization.	-	-
row	long	-	DEM row number	-	.>=0 < total number of rows in DEM (see total number of rows/ columns)
column	long	-	DEM column number	-	>=0 < total number of columns in DEM (see total number of rows/ columns)

7.77.4 Output Parameters

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

Table 193: Output parameters of xp_dem_get_cell_value

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
value	double*	-	Altitude in DEM	m	-
ierr	long*	-	Error vector	-	-

7.77.5 Warnings and Errors

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

Table 194: Error messages of xp_dem_get_cell_value function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Error reading DEM point from file: %s	No calculation performed	XP_CFI_DEM_GET_CELL_VALUE_READ_POINT_ERR	0
ERR	Requested row is out of DEM	No calculation performed	XP_CFI_DEM_GET_CELL_VALUE_WRONG_ABS_ROW_ERR	1
ERR	Requested column is out of DEM	No calculation performed	XP_CFI_DEM_GET_CELL_VALUE_WRONG_ABS_COL_ERR	2
ERR	Requested row is out of preloaded DEM	No calculation performed	XP_CFI_DEM_GET_CELL_VALUE_WRONG_ROW_ERR	3
ERR	Requested column is out of preloaded DEM	No calculation performed	XP_CFI_DEM_GET_CELL_VALUE_WRONG_COLUMN_ERR	4
ERR	Error computing the geoid undulation	No calculation performed	XP_CFI_DEM_GET_CELL_VALUE_GET_GEOID_UNDU_ERR	5
ERR	Error getting DEM value from cache	No calculation performed	XP_CFI_DEM_GET_CELL_VALUE_READ_CACHE_POINT_ERR	6

7.78 xp_dem_get_cell_geod

7.78.1 Overview

The **xp_dem_get_cell_geod** CFI function retrives the geodetic point (latitude/longitude) for the corresponding DEM for the given row and column.

The row/column value refers to the number of row/column considering a DEM covering the whole Earth (see details in section 7.77.1)

7.78.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     xp_dem_id   dem_id   = {NULL};
     long ierr[XP_NUM_ERR_DEM_GET_CELL_GEOD], status;
     long row;
     long column;
     double lat;
     double lon;

     status = xp_dem_get_cell_geod(&dem_id, row, column,
                                   &lat, &lon, ierr);
}
```

The `XP_NUM_ERR_DEM_GET_CELL_GEOD` constant is defined in the file *explorer_pointing.h.*

7.78.3 Input Parameters

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

Table 195: Input parameters of xp_dem_get_cell_geod function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
dem_id	xp_dem_id*	-	Structure that contains the DEM initialization.	-	-
row	long	-	DEM row number	-	.>=0 < total number of rows in DEM (see total number of rows/columns)
column	long	-	DEM column number	-	>=0 < total number of columns in DEM (see total number of rows/columns)

7.78.4 Output Parameters

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

Table 196: Output parameters of xp_dem_get_cell_geod

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
lat	double*	-	Latitude corresponding to the input row/column	deg	[-90, 90]
lon	double*	-	Longitude corresponding to the input row/column	deg	[0, 360]
ierr	long*	-	Error vector	-	-

7.78.5 Warnings and Errors

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

Table 197: Error messages of xp_dem_get_cell_geod function

Error type	Error message	Cause and impact	Error code	Error No
ERR	DEM Id. is not initalized	No calculation performed	XP_CFI_DEM_GET_CELL_GEOD_DEM_STATUS_ERR	0

7.79 xp_target_inter

7.79.1 Overview

The **xp_target_inter** CFI function computes the first or the second intersection point of the line of sight from the satellite (defined by an elevation and an azimuth angle expressed in the selected Attitude Frame) with a surface located at a certain geodetic altitude over the Earth.

The light travel time (from the satellite to the target or vice versa) can be taken into account by the computations. For details about light propagation mode see the section 4.1.2.3.

7.79.2 Calling Interface

The calling interface of the **xp_target_inter** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
     long            sat_id;
     xp_attitude_id attitude_id = {NULL};
     xp_atmos_id    atmos_id = {NULL};
     xp_dem_id      dem_id = {NULL};
     xp_target_id   target_id = {NULL};
     long deriv, inter_flag, iray;
     double los_az, los_el, geod_alt, los_az_rate, los_el_rate, freq;
     long ierr[XP_NUM_ERR_TARGET_INTER], status, num_user_target,
         num_los_target;

     status = xp_target_inter(&sat_id,
                &attitude_id,
                &atmos_id,
                &dem_id,
                &deriv, &inter_flag, &los_az, &los_el, &geod_alt,
                &los_az_rate, &los_el_rate, &iray, &freq,
                &num_user_target, &num_los_target,
                &target_id, ierr);

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

```
      status = xp_target_inter_run(&run_id,
               &attitude_id,
               &deriv, &inter_flag, &los_az, &los_el, &geod_alt,
               &los_az_rate, &los_el_rate, &iray, &freq,
               &num_user_target, &num_los_target,
               &target_id, ierr);

  }
```

The `XP_NUM_ERR_TARGET_INTER` constant is defined in the file *explorer_pointing.h.*

7.79.3 Input Parameters

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

Table 198: Input parameters of xp_target_inter function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude. (input/output)	-	-
atmos_id	xp_atmos_id*	-	Structure that contains the atmosphere initialization.	-	The atmos_id has to be initialized with any of these modes: - XP_NO_REF_INIT - XP_STD_INIT - XP_USER_INIT
dem_id	xp_dem_id*	-	Structure that contains the DEM initialization.	-	-
deriv	long *	-	Derivative ID	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
inter_flag	long *	-	Flag for first or second inter section point selection	-	Allowed values: (1) XP_INTER_1ST (2) XP_INTER_2ND
los_az	double *	-	Azimuth of the LOS (Attitude Frame)	deg	>= 0 < 360
los_el	double *	-	Elevation of the LOS (Attitude Frame)	deg	>= -90 <= 90
geod_alt	double *	-	Geodetic altitude over the Earth	m	>= -bWGS
los_az_rate	double *	-	Azimuth-rate of the LOS (Attitude Frame)	deg/s	-
los_el_rate	double *	-	Elevation-rate of the LOS (Attitude Frame)	deg/s	-
iray	long *	-	Not used. The atmosphere refraction model can be defined via atmos_id initialization.	-	-

			If iray is different of XP_NO_REF_INIT, a warning is raised.		
freq	double *	-	Frequency of the signal	Hz	>= 0

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

- Derivative switch: deriv. See current document, Table 3.
- Intersection flag: inter_flag. See current document, Table 3 .
-

7.79.4 Output Parameters

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

Table 199: Output parameters of xp_target_inter

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
num_user_target	long*	-	Number of user defined targets calculated	-	>= 0 (Set to 1 for non multi-target routines)
num_los_target	long*	-	Number of LOS targets calculated	-	>= 0
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
ierr	long	-	Error vector	-	-

7.79.5 Warnings and Errors

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

Table 200: Error messages of xp_target_inter function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Attitude Id. is not initialized	No calculation performed	XP_CFI_TARGET_INTER_ATTITUDE_STATUS_ERR	0

ERR	Intersection flag is not correct	No calculation performed	XP_CFI_TARGET_INTER_INTER_FLAG_ERR	1
ERR	Invalid Frequency	No calculation performed	XP_CFI_TARGET_INTER_FREQ_ERR	2
ERR	Time reference ID is not correct	No calculation performed	XP_CFI_TARGET_INTER_TIME_REF_ERR	3
ERR	Deriv flag is not correct	No calculation performed	XP_CFI_TARGET_INTER_DERIV_FLAG_ERR	4
ERR	Ray Tracing Model ID is not correct	No calculation performed	XP_CFI_TARGET_INTER_IRAY_ID_ERR	5
ERR	Invalid LOS Azimuth	No calculation performed	XP_CFI_TARGET_INTER_LOS_AZIMUTH_ERR	6
ERR	Invalid LOS Elevation	No calculation performed	XP_CFI_TARGET_INTER_LOS_ELEVATION_ERR	7
ERR	Invalid Geodetic Altitude	No calculation performed	XP_CFI_TARGET_INTER_GEODETIC_ALT_ERR	8
ERR	Memory allocation error	No calculation performed	XP_CFI_TARGET_INTER_MEMORY_ERR	9
ERR	Internal computation error # 3	No calculation performed	XP_CFI_TARGET_INTER_INITIAL_LOOK_DIR_OR_PLANE_ERR	10
ERR	Time Reference not initialised	No calculation performed	XP_CFI_TARGET_INTER_TIME_REF_INIT_ERR	11
ERR	No target was found	No calculation performed	XP_CFI_TARGET_INTER_TARGET_NOT_FOUND_ERR	12
ERR	Internal computation error # 4	No calculation performed	XP_CFI_TARGET_INTER_RANGE_OR_POINTING_CALC_ERR	13
WARN	Path from satellite to target occulted by the Earth	Calculation performed. A message informs the user.	XP_CFI_TARGET_INTER_NEGATIVE_ALTITUDE_WARN	14
WARN	The ray tracing flag (iray) is ignored	Calculation performed. A message informs the user that this paremeter is not used. If the variable iray is equal to XP_NO_REF_INIT (=0), the warning is avoided.	XP_CFI_TARGET_INTER_IRAY_ID_WARN	15

7.80 xp_target_ground_range

7.80.1 Overview

The **xp_target_ground_range** CFI function computes the location of a point that is placed on a surface at a certain geodetic altitude over the Earth, that lays on the plane defined by the satellite position, the nadir and a reference point, and that is at a certain distance or ground range measured along that surface from that reference point.

This reference point is calculated being the intersection of the previous surface with the line of sight defined by an elevation and azimuth angle in the selected Attitude Frame.

The light travel time (from the satellite to the target or vice versa) can be taken into account by the computations. For details about light propagation mode see the section 4.1.2.3.

7.80.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     long            sat_id;
     xp_attitude_id attitude_id = {NULL};
     xp_dem_id      dem_id = {NULL};
     xp_target_id   target_id = {NULL};
     long deriv;
     double los_az, los_el, geod_alt, distance;
     double los_az_rate, los_el_rate;
     long ierr[XP_NUM_ERR_TARGET_GROUND_RANGE], status,
             num_user_target, num_los_target;

     status = xp_target_ground_range(&sat_id,
              &attitude_id,
              &dem_id,
              &deriv, &los_az,
              &los_el, &geod_alt, &distance, &los_az_rate,
              &los_el_rate, &num_user_target, &num_los_target,
              &target_id, ierr);

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

```
      status = xp_target_ground_range_run(&run_id,
               &attitude_id,
               &deriv, &los_az,
               &los_el, &geod_alt, &distance, &los_az_rate,
               &los_el_rate, &num_user_target, &num_los_target,
               &target_id, ierr);
  }
```

The XP_NUM_ERR_TARGET_GROUND_RANGE constant is defined in the file *explorer_pointing.h.*

7.80.3 Input Parameters

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

Table 201: Input parameters of xp_target_ground_range function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude. (input/output)	-	-
dem_id	xp_dem_id*	-	Structure that contains the DEM initialization.	-	-
deriv	long *	-	Derivative ID	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
los_az	double *	-	Azimuth of the LOS (Attitude Frame)	deg	>= 0 < 360
los_el	double *	-	Elevation of the LOS (Attitude Frame)	deg	>= -90 <= 90
geod_alt	double *	-	Geodetic altitude over the Earth (Earth fixed CS)	m	>= -bWGS
distance	double *	-	Distance or ground range to the reference point, positive from nadir in the azimuth direction (Earth Fixed CS)	m	-
los_az_rate	double *	-	Azimuth-rate of the LOS (Attitude Frame)	deg/s	-
los_el_rate	double *	-	Elevation-rate of the LOS (Attitude Frame)	deg/s	-

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

- Derivative switch: deriv. See current document, Table 3.

7.80.4 Output Parameters

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

Table 202: Output parameters of xp_target_ground_range

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
num_user_target	long*	-	Number of user defined targets calculated	-	>= 0 (Set to 1 for non multi-target routines)
num_los_target	long*	-	Number of LOS targets calculated	-	>= 0
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
ierr	long	-	Error vector	-	-

7.80.5 Warnings and Errors

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

Table 203: Error messages of xp_target_ground_range function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Attitude Id. is not initialized	No calculation performed	XP_CFI_TARGET_GR_RANGE_ATTITUDE_STATUS_ERR	0
ERR	Time reference ID is not correct	No calculation performed	XP_CFI_TARGET_GR_RANGE_TIME_REF_ERR	1
ERR	Deriv flag is not correct	No calculation performed	XP_CFI_TARGET_GR_RANGE_DERIV_FLAG_ERR	2
ERR	Invalid LOS Azimuth	No calculation performed	XP_CFI_TARGET_GR_RANGE_LOS_AZIMUTH_ERR	3
ERR	Invalid LOS Elevation	No calculation performed	XP_CFI_TARGET_GR_RANGE_LOS_ELEVATION_E RR	4

ERR	Invalid Geodetic Altitude	No calculation performed	XP_CFI_TARGET_GR_RANGE_GEODETIC_ALT_ERR	5
ERR	Memory allocation error	No calculation performed	XP_CFI_TARGET_GR_RANGE_MEMORY_ERR	6
ERR	Internal computation error # 3	No calculation performed	XP_CFI_TARGET_GR_RANGE_INITIAL_LOOK_DIR_OR_PLANE_ERR	7
ERR	Time reference is not initialized	No calculation performed	XP_CFI_TARGET_GR_RANGE_TIME_REF_INIT_ERR	8
ERR	No target was found	No calculation performed	XP_CFI_TARGET_GR_RANGE_TARGET_NOT_FOUND_ERR	9
ERR	Internal computation error #4	No calculation performed	XP_CFI_TARGET_GR_RANGE_RANGE_OR_POINTING_CALC_ERR	10

7.81 xp_target_incidence_angle

7.81.1 Overview

The **xp_target_incidence_angle** CFI function computes the location of a point that is placed on a surface at a certain geodetic altitude over the Earth and that is seen from the satellite on a line of sight that forms a certain azimuth angle in the selected Attitude Frame and that intersects that surface with a certain incidence angle.

The light travel time (from the satellite to the target or vice versa) can be taken into account by the computations. For details about light propagation mode see the section 4.1.2.3.

7.81.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     long            sat_id, deriv;
     xp_attitude_id attitude_id = {NULL};
     xp_dem_id       dem_id = {NULL};
     xp_target_id    target_id = {NULL};
     double los_az, inc_angle, geod_alt, los_az_rate;
     long ierr[XP_NUM_ERR_TARGET_INCIDENCE_ANGLE], status,
              num_user_target, num_los_target;

     status = xp_target_incidence_angle(&sat_id,
                    &attitude_id, &dem_id,
                    &deriv, &los_az,
                    &inc_angle, &geod_alt, &los_az_rate,
                    &num_user_target, &num_los_target,
                    &target_id, ierr);

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

     status = xp_target_incidence_angle_run(&run_id, &attitude_id,
                    &deriv, &los_az,
                    &inc_angle, &geod_alt, &los_az_rate,
                    &num_user_target, &num_los_target,
```

```
                        &target_id, ierr);
  }
```

The XP_NUM_ERR_TARGET_INCIDENCE_ANGLE constant is defined in the file *explorer_pointing.h.*

7.81.3 Input Parameters

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

Table 204: Input parameters of xp_target_incidence_angle function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude. (input/output)	-	-
dem_id	xp_dem_id *	-	Structure that contains the DEM initialization.	-	-
deriv	long *	-	Derivative ID	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
los_az	double *	-	Azimuth of the LOS (Attitude Frame)	deg	>= 0 < 360
inc_angle	double *	-	Incidence angle of the LOS (Earth fixed CS)	deg	>= 0 <= 90
geod_alt	double *	-	Geodetic altitude over the Earth (Earth fixed CS)	m	>= -bWGS
los_az_rate	double *	-	Azimuth-rate of the LOS (Attitude Frame)	deg/s	-

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

- Derivative switch: deriv. See current document, Table 3 .

7.81.4 Output Parameters

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

Table 205: Output parameters of xp_target_incidence_angle

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
num_user_target	long*	-	Number of user defined targets calculated	-	>= 0 (Set to 1 for non multi-target routines)
num_los_target	long*	-	Number of LOS targets calculated	-	>= 0
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
ierr	long	-	Error vector	-	-

7.81.5 Warnings and Errors

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

Table 206: Error messages of xp_target_incidence_angle function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Attitude Id. is not initialized	No calculation performed	XP_CFI_TARGET_INC_ANGLE_ATTITUDE_STATUS_ERR	0
ERR	Deriv flag is not correct	No calculation performed	XP_CFI_TARGET_INC_ANGLE_DERIV_FLAG_ERR	1
ERR	Invalid LOS Azimuth	No calculation performed	XP_CFI_TARGET_INC_ANGLE_LOS_AZIMUTH_ERR	2
ERR	Invalid Incidence Angle	No calculation performed	XP_CFI_TARGET_INC_ANGLE_INC_ANGLE_ERR	3
ERR	Invalid Geodetic Altitude	No calculation performed	XP_CFI_TARGET_INC_ANGLE_GEODETIC_ALT_ER R	4

ERR	Memory allocation error	No calculation performed	XP_CFI_TARGET_INC_ANGLE_MEMORY_ERR	5
ERR	Internal computation error #3	No calculation performed	XP_CFI_TARGET_INC_ANGLE_INITIAL_LOOK_DIR_OR_PLANE_ERR	6
ERR	Time Reference not initialised	No calculation performed	XP_CFI_TARGET_INC_ANGLE_TIME_REF_INIT_ERR	7
ERR	No target was found	No calculation performed	XP_CFI_TARGET_INC_ANGLE_TARGET_NOT_FOUND_ERR	8

7.82 xp_target_range

7.82.1 Overview

The **xp_target_range** CFI function computes the location of a point that is placed on a surface at a certain geodetic altitude over the Earth, that is seen from the satellite on a line of sight that forms a certain azimuth angle in the selected Attitude Frame, and that is at a certain range or slant-range from the satellite. If more than one target is found, the first one will be the one with an elevation closer to 90 degrees (targets will have the same azimuth, equal to input azimuth).

The light travel time (from the satellite to the target or vice versa) can be taken into account by the computations. For details about light propagation mode see the section 4.1.2.3.

7.82.2 Calling Interface

The calling interface of the **xp_target_range** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
     long            sat_id;
     xp_attitude_id attitude_id = {NULL};
     xp_dem_id       dem_id = {NULL};
     xp_target_id    target_id = {NULL};
     long deriv;
     double los_az, range, geod_alt, los_az_rate, range_rate;
     long ierr[XP_NUM_ERR_TARGET_RANGE], status, num_user_target,
              num_los_target;

     status = xp_target_range(&sat_id, &attitude_id, &dem_id,
                    &deriv, &los_az, &range,
                    &geod_alt, &los_az_rate, &range_rate,
                    &num_user_target, &num_los_target,
                    &target_id, ierr);

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

     status = xp_target_range_run(&run_id, &attitude_id,
                    &deriv, &los_az, &range,
                    &geod_alt, &los_az_rate, &range_rate,
                    &num_user_target, &num_los_target,
```

```
                     &target_id, ierr);
  }
```

The `XP_NUM_ERR_TARGET_RANGE` constant is defined in the file *explorer_pointing.h.*

7.82.3 Input Parameters

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

Table 207: Input parameters of xp_target_range function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude. (input/output)	-	-
dem_id	xp_dem_id *	-	Structure that contains the DEM initialization.	-	-
deriv	long *	-	Derivative ID	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
los_az	double *	-	Azimuth of the LOS (Attitude Frame)	deg	>= 0 < 360
range	double *	-	Range to the satellite (Earth fixed CS)	m	> 0
geod_alt	double *	-	Geodetic altitude over the Earth	m	>= -bWGS
los_az_rate	double *	-	Azimuth-rate of the LOS (Attitude Frame)	deg/s	-
range_rate	double *	-	Range-rate to the satellite (Earth fixed CS)	m/s	-

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

- Derivative switch: deriv. See current document, Table 3 .

7.82.4 Output Parameters

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

Table 208: Output parameters of xp_target_range

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
num_user_target	long*	-	Number of user defined targets calculated	-	>= 0 (Set to 1 for non multi-target routines)
num_los_target	long*	-	Number of LOS targets calculated (per user target)	-	>= 0
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
ierr	long	-	Error vector	-	-

7.82.5 Warnings and Errors

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

Table 209: Error messages of xp_target_range function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Attitude Id. is not initialized	No calculation performed	XP_CFI_TARGET_RANGE_ATTITUDE_STATUS_E RR	0
ERR	Deriv flag is not correct	No calculation performed	XP_CFI_TARGET_RANGE_DERIV_FLAG_ERR	1
ERR	Invalid LOS Azimuth	No calculation performed	XP_CFI_TARGET_RANGE_LOS_AZIMUTH_ERR	2
ERR	Invalid Range	No calculation performed	XP_CFI_TARGET_RANGE_RANGE_ERR	3
ERR	Invalid Geodetic Altitude	No calculation performed	XP_CFI_TARGET_RANGE_GEODETIC_ALT_ERR	4

ERR	Memory allocation error	No calculation performed	XP_CFI_TARGET_RANGE_MEMORY_ERR	5
ERR	Internal computation error #3	No calculation performed	XP_CFI_TARGET_RANGE_INITIAL_LOOK_DIR_OR_PLANE_ERR	6
ERR	Could not perform a time transformation	No calculation performed	XP_CFI_TARGET_RANGE_TIME_TRANSFORMATION_ERR	7
ERR	No target was found	No calculation performed	XP_CFI_TARGET_RANGE_TARGET_NOT_FOUND_ERR	8

7.83 xp_target_range_rate

7.83.1 Overview

The **xp_target_range_rate** CFI function computes the location of a point that is placed on a surface at a certain geodetic altitude over the Earth, that is at a certain range from the satellite, and this range has a certain change rate (range_rate). Associated Earth-fixed target is supposed to have zero range-rate value.

The light travel time (from the satellite to the target or vice versa) can be taken into account by the computations. For details about light propagation mode see the section 4.1.2.3.

7.83.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     long            sat_id;
     xp_attitude_id attitude_id = {NULL};
     xp_dem_id      dem_id = {NULL};
     xp_target_id   target_id = {NULL};
     long deriv;
     double ef_range_rate, range, geod_alt;
     double ef_range_rate_rate, range_rate;
     long ierr[XP_NUM_ERR_TARGET_RANGE_RATE], status, num_user_target,
              num_los_target;

     status = xp_target_range_rate(&sat_id,
               &attitude_id,
               &dem_id,
               &deriv, &ef_range_rate, &range,
               &geod_alt, &ef_range_rate_rate, &range_rate,
               &num_user_target, &num_los_target, &target_id, ierr);

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

     status = xp_target_range_rate_run(&run_id,
               &attitude_id,
```

```
                &deriv, &ef_range_rate, &range,
                &geod_alt, &ef_range_rate_rate, &range_rate,
                &num_user_target, &num_los_target, &target_id, ierr);
  }
```

The XP_NUM_ERR_TARGET_RANGE_RATE constant is defined in the file *explorer_pointing.h.*

7.83.3 Input Parameters

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

Table 210: Input parameters of xp_target_range_rate function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude. (input/output)	-	-
dem_id	xp_dem_id *	-	Structure that contains the DEM initialization.	-	-
attitude_frame_id	long *	-	Attitude Frame ID	-	Complete
deriv	long *	-	Derivative ID	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
ef_range_rate	double *	-	Range-rate of the related Earth-fixed target. (Earth fixed CS) Dummy parameter.	m/s	-
range	double *	-	Range or slant-range from tar get to satellite (Earth fixed CS)	m	> 0
geod_alt	double *	-	Geodetic altitude over the Earth	m	>= -bWGS
ef_range_rate_rate	double *	-	Range-rate-rate of the related Earth-fixed target (Earth fixed CS) Dummy parameter.	m/s2	-
range_rate	double *	-	Range-rate from target to sat ellite (Earth fixed CS) Dummy parameter.	m/s	-

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

- Derivative switch: deriv. See current document, Table 3 .

7.83.4 Output Parameters

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

Table 211: Output parameters of xp_target_range_rate

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
num_user_target	long*	-	Number of user defined targets calculated	-	>= 0 (Set to 1 for non multi-target routines)
num_los_target	long*	-	Number of LOS targets calculated	-	>= 0
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
ierr	long	-	Error vector	-	-

7.83.5 Warnings and Errors

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

Table 212: Error messages of xp_target_range_rate function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Attitude Id. is not initialized	No calculation performed	XP_CFI_TARGET_RANGE_RATE_ATTITUDE_STATUS_ERR	0
ERR	Deriv flag is not correct	No calculation performed	XP_CFI_TARGET_RANGE_RATE_DERIV_FLAG_E RR	1
ERR	Invalid Range	No calculation performed	XP_CFI_TARGET_RANGE_RATE_RANGE_ERR	2
ERR	Invalid Geodetic Altitude	No calculation performed	XP_CFI_TARGET_RANGE_RATE_GEODETIC_ALT_ERR	3
ERR	Memory allocation error	No calculation performed	XP_CFI_TARGET_RANG	4

			E_RATE_MEMORY_ERR	
ERR	Time Reference not initialised	No calculation performed	XP_CFI_TARGET_RANGE_RATE_TIME_REF_INIT_ERR	5
ERR	Internal computation error #3	No calculation performed	XP_CFI_TARGET_RANGE_RATE_INITIAL_LOOK_DIR_OR_PLANE_ERR	6
ERR	No target was found	No calculation performed	XP_CFI_TARGET_RANGE_RATE_TARGET_NOT_FOUND_ERR	7
WARN	Range rate algo only returns results without derivatives	Calculation performed. A message informs the user.	XP_CFI_TARGET_RANGE_RATE_DERIV_FLAG_WARN	8

7.84 xp_target_tangent

7.84.1 Overview

The **xp_target_tangent** CFI function computes the location of the tangent point over the Earth that is located on the line of sight defined by an elevation and azimuth angles expressed in the selected Attitude Frame.

The light travel time (from the satellite to the target or vice versa) can be taken into account by the computations. For details about light propagation mode see the section 4.1.2.3.

7.84.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     long            sat_id;
     xp_attitude_id attitude_id = {NULL};
     xp_atmos_id    atmos_id = {NULL};
     xp_dem_id      dem_id = {NULL};
     xp_target_id   target_id = {NULL};
     long deriv,      iray;
     double los_az, los_el, los_az_rate, los_el_rate, freq;
     long ierr[XP_NUM_ERR_TARGET_TANGENT], status, num_user_target,
          num_los_target;

     status = xp_target_tangent(&sat_id, &attitude_id,
                   &atmos_id,  &dem_id,
                   &deriv, &los_az, &los_el,
                   &los_az_rate, &los_el_rate, &iray, &freq,
                   &num_user_target, &num_los_target,
                   &target_id, ierr);

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

     status = xp_target_tangent_run(&run_id, &attitude_id,
                   &deriv, &los_az, &los_el,
```

```
                    &los_az_rate, &los_el_rate, &iray, &freq,
                    &num_user_target, &num_los_target,
                    &target_id, ierr);
  }
```

The XP_NUM_ERR_TARGET_TANGENT constant is defined in the file *explorer_pointing.h.*

7.84.3 Input Parameters

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

Table 213: Input parameters of xp_target_tangent function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude. (input/output)	-	-
atmos_id	xp_atmos_id*	-	Structure that contains the atmosphere initialization.	-	-
dem_id	xp_dem_id *	-	Structure that contains the DEM initialization.	-	-
deriv	long *	-	Derivative ID	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
los_az	double *	-	Azimuth of the LOS (Attitude Frame)	deg	>= 0 < 360
los_el	double *	-	Elevation of the LOS (Attitude Frame)	deg	>= -90 <= 90
los_az_rate	double *	-	Azimuth-rate of the LOS (Attitude Frame)	deg/s	-
los_el_rate	double *	-	Elevation-rate of the LOS (Attitude Frame)	deg/s	-
iray	long *	-	Not used. The atmosphere refraction model can be defined via atmos_id initialization.	-	-
freq	double *	-	Frequency of the signal	Hz	>= 0

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

- Derivative switch: deriv. See current document, Table 3 .

7.84.4 Output Parameters

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

Table 214: Output parameters of xp_target_tangent

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
num_user_target	long*	-	Number of user defined targets calculated	-	>= 0 (Set to 1 for non multi-target routines)
num_los_target	long*	-	Number of LOS targets calculated	-	>= 0
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
ierr	long	-	Error vector	-	-

7.84.5 Warnings and Errors

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

Table 215: Error messages of xp_target_tangent function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Attitude Id. is not initialized	No calculation performed	XP_CFI_TARGET_TANGENT_ATTITUDE_STATUS_ERR	0
ERR	Deriv flag is not correct	No calculation performed	XP_CFI_TARGET_TANGENT_DERIV_FLAG_ERR	1
ERR	Invalid LOS Azimuth	No calculation performed	XP_CFI_TARGET_TANGENT_LOS_AZIMUTH_ERR	2
ERR	Invalid LOS Elevation	No calculation performed	XP_CFI_TARGET_TANGENT_LOS_ELEVATION_ERR	3
ERR	Ray Tracing Model ID is not correct	No calculation performed	XP_CFI_TARGET_TANGENT_IRAY_ID_ERR	4

ERR	Invalid Frequency	No calculation performed	XP_CFI_TARGET_TANGENT_FREQ_ERR	5
ERR	Memory allocation error	No calculation performed	XP_CFI_TARGET_TANGENT_MEMORY_ERR	6
ERR	Internal computation error # 3	No calculation performed	XP_CFI_TARGET_TANGENT_INITIAL_LOOK_DIR_OR_PLANE_ERR	7
ERR	Time Reference not initialised	No calculation performed	XP_CFI_TARGET_TANGENT_TIME_REF_INIT_ERR	8
ERR	No target was found	No calculation performed	XP_CFI_TARGET_TANGENT_TARGET_NOT_FOUND_ERR	9
ERR	Error computing target when it is behind looking direction	No calculation performed	XP_CFI_TARGET_TANGENT_TG_PT_BEHIND_LOOK_DIR_ERR	10
ERR	Internal computation error # 4	No calculation performed	XP_CFI_TARGET_TANGENT_RANGE_OR_POINTING_CALC_ERR	11
WARN	Path from satellite to target occulted by the Earth	Calculation performed. A message informs the user.	XP_CFI_TARGET_TANGENT_NEGATIVE_ALTITUDE_WARN	12
WARN	Tangent point latitude is outside the selected corrective function latitude band	Calculation performed. A message informs the user.	XP_CFI_TARGET_TANGENT_PRED_WRONG_LAT_WARN, WARN	13
WARN	Tangent point is behind looking direction	Calculation performed. A message informs the user.	XP_CFI_TARGET_TANGENT_TG_PT_BEHIND_LOOK_DIR_WARN	14
WARN	The ray tracing flag (iray) is ignored	Calculation performed. A message informs the user that this paremeter is not used. If the variable iray is equal to XP_NO_REF_INIT (=0), the warning is avoided	XP_CFI_TARGET_TANGENT_IRAY_ID_WARN	15

7.85 xp_target_altitude

7.85.1 Overview

The **xp_target_altitude** CFI function computes the location of the tangent point over the Earth that is located on a surface at a certain geodetic altitude over the Earth and that is on a line of sight that forms a certain azimuth angle in the selected Attitude Frame.

The light travel time (from the satellite to the target or vice versa) can be taken into account by the computations. For details about light propagation mode see the section 4.1.2.3.

7.85.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
    long            sat_id;
    xp_attitude_id attitude_id = {NULL};
    xp_atmos_id    atmos_id = {NULL};
    xp_dem_id      dem_id = {NULL};
    xp_target_id   target_id = {NULL};
    long deriv,      iray;
    double los_az, geod_alt, los_az_rate, freq;
    long ierr[XP_NUM_ERR_TARGET_ALTITUDE], status, num_user_target,
            num_los_target;

    status = xp_target_altitude(sat_id, &attitude_id,
             &atmos_id, &dem_id,
             &deriv, &los_az, &geod_alt,
             &los_az_rate, &iray, &freq,
             &num_user_target, &num_los_target,
             &target_id, ierr);

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

    status = xp_target_altitude_run(run_id, &attitude_id,
             &deriv, &los_az, &geod_alt,
```

```
                &los_az_rate, &iray, &freq,
                &num_user_target, &num_los_target,
                &target_id, ierr);
  }
```

The XP_NUM_ERR_TARGET_ALTITUDE constant is defined in the file *explorer_pointing.h.*

7.85.3 Input Parameters

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

Table 216: Input parameters of xp_target_altitude function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude. (input/output)	-	-
atmos_id	xp_atmos_id*	-	Structure that contains the atmosphere initialization.	-	-
dem_id	xp_dem_id *	-	Structure that contains the DEM initialization.	-	-
deriv	long *	-	Derivative ID	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
los_az	double *	-	Azimuth of the LOS (Attitude Frame)	deg	>= 0 < 360
geod_alt	double *	-	Geodetic altitude over the Earth	m	>= -bWGS
los_az_rate	double *	-	Azimuth-rate of the LOS (Attitude Frame)	deg/s	-
iray	long *	-	Not used. The atmosphere refraction model can be defined via atmos_id initialization.	-	-
freq	double *	-	Frequency of the signal	Hz	>= 0

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

- Derivative switch: deriv. See current document, Table 3 .

7.85.4 Output Parameters

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

Table 217: Output parameters of xp_target_altitude

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
num_user_target	long*	-	Number of user defined targets calculated	-	>= 0 (Set to 1 for non multi-target routines)
num_los_target	long*	-	Number of LOS targets calculated	-	>= 0
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
ierr	long	-	Error vector	-	-

7.85.5 Warnings and Errors

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

Table 218: Error messages of xp_target_altitude function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Attitude Id. is not initialized	No calculation performed	XP_CFI_TARGET_ALT_ATTITUDE_STATUS_ERR	0
ERR	Deriv flag is not correct	No calculation performed	XP_CFI_TARGET_ALT_DERIV_FLAG_ERR	1
ERR	Invalid LOS Azimuth	No calculation performed	XP_CFI_TARGET_ALT_LOS_AZIMUTH_ERR	2
ERR	Invalid Geodetic Altitude	No calculation performed	XP_CFI_TARGET_ALT_GEODETIC_ALT_ERR	3
ERR	Ray Tracing Model ID is not correct	No calculation performed	XP_CFI_TARGET_ALT_IRAY_ID_ERR	4

ERR	Invalid Frequency	No calculation performed	XP_CFI_TARGET_ALT_FREQ_ERR	5
ERR	Memory allocation error	No calculation performed	XP_CFI_TARGET_ALT_MEMORY_ERR	6
ERR	Time Reference not initialised	No calculation performed	XP_CFI_TARGET_ALT_TIME_REF_INIT_ERR	7
ERR	Internal computation error #3	No calculation performed	XP_CFI_TARGET_ALT_INITIAL_LOOK_DIR_OR_PLANE_ERR	8
ERR	No target was found	No calculation performed	XP_CFI_TARGET_ALT_TARGET_NOT_FOUND_ERR	9
ERR	Internal computation error #4	No calculation performed	XP_CFI_TARGET_ALT_RANGE_OR_POINTING_CALC_ERR	10
WARN	Path from satellite to target occulted by the Earth	Calculation performed. A message informs the user.	XP_CFI_TARGET_ALT_NEGATIVE_ALTITUDE_WAR N	11
WARN	Tangent point latitude is outside the selected corrective function latitude band	Calculation performed. A message informs the user.	XP_CFI_TARGET_ALT_PRED_WRONG_LAT_WARN	12
WARN	The ray tracing flag (iray) is ignored	Calculation performed. A message informs the user that this parameter is not used. If the variable iray is equal to XP_NO_REF_INIT (=0), the warning is avoided.	XP_CFI_TARGET_ALT_IRAY_ID_WARN	13

7.86 xp_target_star

7.86.1 Overview

The **xp_target_star** CFI function computes the location of the tangent point over the Earth that is located on the line of sight that points to a star defined by its right ascension and declination coordinates.

The light travel time (from the satellite to the target or vice versa) can be taken into account by the computations. For details about light propagation mode see the section 4.1.2.3.

7.86.2 Calling Interface

The calling interface of the **xp_target_star** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
     long            sat_id;
     xp_attitude_id attitude_id = {NULL};
     xp_atmos_id    atmos_id = {NULL};
     xp_dem_id      dem_id = {NULL};
     xp_target_id   target_id = {NULL};
     long deriv, iray;
     double star_ra, star_dec, star_ra_rate, star_dec_rate, freq;
     long ierr[XP_NUM_ERR_TARGET_STAR], status, num_user_target,
              num_los_target;

     status = xp_target_star(&sat_id, &attitude_id,
               &atmos_id, &dem_id,
               &deriv, &star_ra, star_dec,
               &star_ra_rate, &star_dec_rate, &iray, &freq,
               &num_user_target, &num_los_target,
               &target_id, ierr);

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

     status = xp_target_star_run(&run_id, &attitude_id,
               &deriv, &star_ra, star_dec,
               &&star_ra_rate, &star_dec_rate, &iray, &freq,
               &num_user_target, &num_los_target,
```

```
                &target_id, ierr);
  }
```

The XP_NUM_ERR_TARGET_STAR constant is defined in the file *explorer_pointing.h.*

7.86.3 Input Parameters

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

Table 219: Input parameters of xp_target_star function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude. (input/output)	-	-
atmos_id	xp_atmos_id*	-	Structure that contains the atmosphere initialization.	-	-
dem_id	xp_dem_id *	-	Structure that contains the DEM initialization.	-	-
deriv	long *	-	Derivative ID	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
star_ra	double *	-	Right ascension of the star (True of Date CS)	deg	>= 0 < 360
star_dec	double *	-	Declination of the star (True of Date CS)	deg	>= -90 <= +90
star_ra_rate	double *	-	Right ascension rate of the star (True of Date CS)	deg/s	-
star_dec_rate	double *	-	Declination rate of the star (True of Date CS)	deg/s	-
iray	long *	-	Not used. The atmosphere refraction model can be defined via atmos_id initialization.	-	-
freq	double *	-	Frequency of the signal	Hz	>= 0

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

- Derivative switch: deriv. See current document, Table 3 .

7.86.4 Output Parameters

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

Table 220: Output parameters of xp_target_star

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
num_user_target	long*	-	Number of user defined targets calculated	-	>= 0 (Set to 1 for non multi-target routines)
num_los_target	long*	-	Number of LOS targets calculated	-	>= 0
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
ierr	long	-	Error vector	-	-

7.86.5 Warnings and Errors

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

Table 221: Error messages of xp_target_star function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Attitude Id. is not initialized	No calculation performed	XP_CFI_TARGET_STAR_ATTITUDE_STATUS_ERR	0
ERR	Deriv flag is not correct	No calculation performed	XP_CFI_TARGET_STAR_DERIV_FLAG_ERR	1
ERR	Invalid Right Ascension of the star	No calculation performed	XP_CFI_TARGET_STAR_RA_ERR	2
ERR	Invalid Declination of the star	No calculation performed	XP_CFI_TARGET_STAR_DEC_ERR	3
ERR	Ray Tracing Model ID is not correct	No calculation performed	XP_CFI_TARGET_STAR_IRAY_ID_ERR	4

ERR	Invalid Frequency	No calculation performed	XP_CFI_TARGET_STAR_FREQ_ERR	5
ERR	Memory allocation error	No calculation performed	XP_CFI_TARGET_STAR_MEMORY_ERR	6
ERR	Internal computation error # 3	No calculation performed	XP_CFI_TARGET_STAR_INITIAL_LOOK_DIR_OR_PLANE_ERR	7
ERR	Time reference ID is not correct	No calculation performed	XP_CFI_TARGET_STAR_TIME_REF_INIT_ERR	8
ERR	No target was found	No calculation performed	XP_CFI_TARGET_STAR_TARGET_NOT_FOUND_ER R	9
ERR	Tangent point is behind looking direction	No calculation performed	XP_CFI_TARGET_STAR_TG_PT_BEHIND_LOOK_DIR_ERR	10
ERR	Internal computation error # 4	No calculation performed	XP_CFI_TARGET_STAR_RANGE_OR_POINTING_CALC_ERR	11
WARN	Path from satellite to target occulted by the Earth	Calculation performed. A message informs the user.	XP_CFI_TARGET_STAR_NEGATIVE_ALTITUDE_WA RN	12
WARN	Tangent point latitude is outside the selected corrective function latitude band	Calculation performed. A message informs the user.	XP_CFI_TARGET_STAR_PRED_WRONG_LAT_WAR N	13
WARN	Tangent point is behind looking direction		XP_CFI_TARGET_STAR_TG_PT_BEHIND_LOOK_DIR_WARN	14
WARN	The ray tracing flag (iray) is ignored	Calculation performed. A message informs the user that this paremeter is not used. If the variable iray is equal to XP_NO_REF_INIT (=0), the warning is avoided	XP_CFI_TARGET_STAR_IRAY_ID_WARN	15

7.87 xp_target_station

7.87.1 Overview

The **xp_target_station** CFI function computes the most relevant observation parameters of the link between the satellite and a ground station.

The light travel time (from the satellite to the target or vice versa) can be taken into account by the computations. For details about light propagation mode see the section 4.1.2.3.

7.87.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     long           sat_id;
     xp_attitude_id attitude_id = {NULL};
     xp_dem_id      dem_id = {NULL};
     xp_target_id   target_id = {NULL};
     long deriv;
     double geoc_long, geod_lat, geod_alt, min_link_el;
     long ierr[XP_NUM_ERR_TARGET_STATION], status, num_user_target,
              num_los_target;

     status = xp_target_station(&sat_id,
               &attitude_id, &dem_id,
               &deriv, &geoc_long, &geod_lat,
               &geod_alt, &min_link_el,
               &num_user_target, &num_los_target,
               &target_id, ierr);

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

     status = xp_target_station_run(&run_id,
               &attitude_id,
               &deriv, &geoc_long, &geod_lat,
               &geod_alt, &min_link_el,
```

```
                &num_user_target, &num_los_target,
                &target_id, ierr);
  }
```

The XP_NUM_ERR_TARGET_STATION constant is defined in the file *explorer_pointing.h.*

7.87.3 Input Parameters

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

Table 222: Input parameters of xp_target_station function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude. (input/output)	-	-
dem_id	xp_dem_id *	-	Structure that contains the DEM initialization.	-	-
deriv	long *	-	Derivative ID	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
geoc_long	double *	-	GS geocentric longitude (Earth fixed CS)	deg	>= 0 < 360
geod_lat	double *	-	GS geodetic latitude (Earth fixed CS)	deg	>= -90 < = 90
geod_alt	double *	-	GS geodetic altitude (Earth fixed CS)	m	>= -bWGS
min_link_el	double *	-	GS minimum link elevation (Topocentric CS)	deg	>= -90 <= 90

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

- Derivative switch: deriv. See current document, Table 3 .

7.87.4 Output Parameters

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

Table 223: Output parameters of xp_target_station

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
num_user_target	long*	-	Number of user defined targets calculated	-	>= 0 (Set to 1 for non multi-target routines)
num_los_target	long*	-	Number of LOS targets calculated	-	>= 0
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
ierr	long	-	Error vector	-	-

7.87.5 Warnings and Errors

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

Table 224: Error messages of xp_target_station function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Attitude Id. is not initialized	No calculation performed	XP_CFI_TARGET_STATION_ATTITUDE_STATUS_ERR	0
ERR	Deriv flag is not correct	No calculation performed	XP_CFI_TARGET_STATION_DERIV_FLAG_ERR	1
ERR	Invalid GS Geocentric Longitude	No calculation performed	XP_CFI_TARGET_STATION_GEOC_LONG_ERR	2
ERR	Invalid GS Geodetic Latitude	No calculation performed	XP_CFI_TARGET_STATION_GEOD_LAT_ERR	3
ERR	Invalid GS Geodetic Altitude	No calculation performed	XP_CFI_TARGET_STATION_GEODETIC_ALT_ERR	4

ERR	Invalid GS Minimum Link Elevation	No calculation performed	XP_CFI_TARGET_STATION_MIN_LINK_EL_ERR	5
ERR	Memory allocation error	No calculation performed	XP_CFI_TARGET_STATION_MEMORY_ERR	6
ERR	Internal computation error #3	No calculation performed	XP_CFI_TARGET_STATION_STAVIS_COMP_FAILED_ERR	7

7.88 xp_target_generic

7.88.1 Overview

The **xp_target_generic** CFI function allows the user to provide the target location (position and velocity) and later calculate extra results from it.

The light travel time (from the satellite to the target or vice versa) can be taken into account by the computations. For details about light propagation mode see the section 4.1.2.3.

7.88.2 Calling Interface

The calling interface of the **xp_target_generic** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
    long           sat_id;
    xp_attitude_id attitude_id = {NULL};
    xp_dem_id      dem_id = {NULL};
    xp_target_id   target_id = {NULL};
    long deriv;
    double targ_pos[3], targ_vel[3], targ_acc[3];
    long ierr[XP_NUM_ERR_TARGET_GENERIC], status, num_user_target,
             num_los_target;

    status = xp_target_generic(&sat_id,
              &attitude_id,
              &dem_id,
              &deriv, targ_pos, targ_vel, targ_acc,
              &num_user_target, &num_los_target,
              &target_id, ierr);

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

    status = xp_target_generic_run(&run_id,
              &attitude_id,
              &deriv, targ_pos, targ_vel, targ_acc,
              &num_user_target, &num_los_target,
```

```
                &target_id, ierr);
}
```

The `XP_NUM_ERR_TARGET_GENERIC` constant is defined in the file *explorer_pointing.h.*

7.88.3 Input Parameters

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

Table 225: Input parameters of xp_target_generic function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude. (input/output)	-	-
dem_id	xp_dem_id *	-	Structure that contains the DEM initialization.	-	-
deriv	long *	-	Derivative ID	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
targ_pos	double[3]	[0-2]	Target position vector (Earth Fixed CS)	m	-
targ_vel	double[3]	[0-2]	Target velocity vector (Earth Fixed CS)	m/s	-
targ_acc	double[3]	[0-2]	Target acceleration vector (Earth Fixed CS)	m/s2	-

It is possible to use enumeration values rather than integer values for some of the input arguments:
- Derivative switch: deriv. See current document, Table 3 .

7.88.4 Output Parameters

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

Table 226: Output parameters of xp_target_generic

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
num_user_target	long*	-	Number of user defined targets calculated	-	>= 0 (Set to 1 for non multi-target

					routines)
num_los_target	long*	-	Number of LOS targets calculated	-	>= 0
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
ierr	long	-	Error vector	-	-

7.88.5 Warnings and Errors

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

Table 227: Error messages of xp_target_generic function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Attitude Id. is not initialized	No calculation performed	XP_CFI_TARGET_GENERIC_ATTITUDE_STATUS_ERR	0
ERR	Deriv flag is not correct	No calculation performed	XP_CFI_TARGET_GENERIC_DERIV_FLAG_ERR	1
ERR	Memory allocation error	No calculation performed	XP_CFI_TARGET_GENERIC_MEMORY_ERR	2
ERR	Internal computation error # 3	No calculation performed	XP_CFI_TARGET_GENERIC_INITIAL_LOOK_DIR_OR_PLANE_ERR	3

7.89 xp_target_reflected

7.89.1 Overview

The **xp_target_reflected** CFI function allows the user to compute, from S/C position and attitude, and emitting source position, the point of reflection from the source towards the SC at a certain geodetic altitude.

Note: in some limit configurations the function will return a degraded solution (returning also a warning of type XP_CFI_TARGET_REFLECTED_DEGRADED_SOL_WARN), where a maximum difference between the incidence angle and the reflected angle can be up to 5 milidegrees.

7.89.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     long           sat_id, deriv, source_type;
     long           status, num_user_target, num_los_target;
     xp_attitude_id attitude_id = {NULL};
     xp_target_id   target_id = {NULL};
     double         geod_alt, deflection_north, deflection_east, 
source_param[XP_NUM_SOURCE_PARAM];
     long          ierr[XP_NUM_ERR_TARGET_REFLECTED]

     status = xp_target_reflected( &sat_id, &attitude_id,
                         &deriv, &geod_alt,
                         &deflection_north, &deflection_east,
                         &source_type, source_param,
                         /* outputs */
                       &num_user_target, &num_los_target,
                       &target_id, ierr);

     /* Or, using the run_id */
     long run_id;
     status = xp_target_reflected_run( &run_id,
                        &attitude_id, &deriv, &geod_alt,
                          &deflection_north, &deflection_east,
                          &source_type, source_param,
```

```
                /* outputs */
              &num_user_target, &num_los_target,
              &target_id, ierr);
}
```

The `XP_NUM_ERR_TARGET_GENERIC` constant is defined in the file *explorer_pointing.h.*

7.89.3 Input Parameters

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

Table 228: Input parameters of xp_target_reflected function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude. (input/output)	-	-
deriv	long *	-	Derivative ID	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
geod_lat	double *	-	GS geodetic latitude (Earth fixed CS)	deg	>= -90 < = 90
deflection_north	double *	-	North-South component of the vertical deflection	deg	>= -90 < = 90
deflection_east	double *	-	East-West component of the vertical deflection	deg	>= -90 < = 90
source_type	long *	-	The type of source	-	Allowed values: XP_SOURCE_STAR XP_SOURCE_SUN XP_SOURCE_MOON XP_SOURCE_GENERIC
source param	double[XP_NUM_SOURCE_PARAM]	[0-5]	• if source_type is XP_SRC_STAR is selected, source_param = [ra (deg), dec (deg), paral lax, 0.0, 0.0, 0.0], i.e. star coordinates in BM2000 CS. • if source_type is XP_SOURCE_GENERI	Right ascension and decli nation in degrees. Position vector in meters,	-

			C source_param = [x, y, z, vx,vy,vz] position and velocity in EF • else dummy values.	velocity in meters/ second	

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

- Derivative switch: deriv. See current document, Table 3 .
- Source Identification: source_type. See current document, Table 3 .

7.89.4 Output Parameters

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

Table 229: Output parameters of xp_target_reflected

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
num_user_target	long*	-	Number of user defined targets calculated	-	>= 0 (Set to 1 for non multi-target routines)
num_los_target	long*	-	Number of LOS targets calculated	-	>= 0
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
ierr	long	-	Error vector	-	-

7.89.5 Warnings and Errors

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

Table 230: Error messages of xp_target_reflected function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Attitude Id. is not initialized	No calculation performed	XP_CFI_TARGET_REFLECT	0

			ED_ATTITUDE_STATUS_ER R	
ERR	Error when calling xp_target_star	No calculation performed	XP_CFI_TARGET_REFLECT ED_TARGET_STAR_ERR	1
ERR	Error when calling xp_target_tangent_sun	No calculation performed	XP_CFI_TARGET_REFLECT ED_TARGET_TG_SUN_ERR	2
ERR	Error when calling xp_target_tangent_moon	No calculation performed	XP_CFI_TARGET_REFLECT ED_TARGET_TG_MOON_ER R	3
ERR	Error when calling xl_change_cart_cs	No calculation performed	XP_CFI_TARGET_REFLECT ED_CHANGE_CS_ERR	4
ERR	Could not get direction pointing	No calculation performed	XP_CFI_TARGET_REFLECT ED_DIR_POINT_ERR	5
ERR	Error when calling xp_target_tangent	No calculation performed	XP_CFI_TARGET_REFLECT ED_TARGET_TG_ERR	6
ERR	Wrong input parameter "source_type"	No calculation performed	XP_CFI_TARGET_REFLECT ED_WRONG_SRC_TYPE_ER R	7
ERR	Error when calling xp_target_extra_main	No calculation performed	XP_CFI_TARGET_REFLECT ED_TGT_EXTRA_MAIN_ER R	8
ERR	Target tangent altitude is less than the requested altitude	No calculation performed	XP_CFI_TARGET_REFLECT ED_WRONG_ALTITUDE_ER R	9
ERR	Could not get cartesian coordinates for the star	No calculation performed	XP_CFI_TARGET_REFLECT ED_RADEC_TO_CART_ERR	10
ERR	Could not get the Sun coordinates	No calculation performed	XP_CFI_TARGET_REFLECT ED_SUN_ERR	11
ERR	Could not get the Moon coordinates	No calculation performed	XP_CFI_TARGET_REFLECT ED_MOON_ERR	12
ERR	Iteration did not converge. Impossible to find the deflection point	No calculation performed	XP_CFI_TARGET_REFLECT ED_ITER_ERR	13
ERR	Could not compute GPM attitude frame	No calculation performed	XP_CFI_TARGET_REFLECT ED_SAT_NOM_ATT_INIT_E RR	14
ERR	Could not initialise the attitude frame	No calculation performed	XP_CFI_TARGET_REFLECT ED_ATT_INIT_ERR	15
ERR	Could not compute the attitude frame	No calculation performed	XP_CFI_TARGET_REFLECT ED_ATT_COMPUTE_ERR	16
ERR	Error when calling xp_target_extra_specular_re flection	No calculation performed	XP_CFI_TARGET_REFLECT ED_TGT_EXTRA_REF_ERR	17
ERR	Error when calling	No calculation performed	XP_CFI_TARGET_REFLECT	18

	xp_target_inter		ED_TARGET_INTER_ERR	
ERR	Error when calling xp_target_extra_vector	No calculation performed	XP_CFI_TARGET_REFLECT ED_TARGET_EXTRA_ERR	19
ERR	Error when calling xp_target_generic	No calculation performed	XP_CFI_TARGET_REFLECT ED_TARGET_GENERIC_ERR	20
ERR	Could not change EF coordinates to topocentric	No calculation performed	XP_CFI_TARGET_REFLECT ED_EF_TO_TOP_ERR	21
ERR	Could not change topocentric coordinates to EF	No calculation performed	XP_CFI_TARGET_REFLECT ED_TOP_TO_EF_ERR	22
ERR	Could not close target	No calculation performed	XP_CFI_TARGET_REFLECT ED_TGT_CLOSE_ERR	23
ERR	Could not close attitude	No calculation performed	XP_CFI_TARGET_REFLECT ED_ATT_CLOSE_ERR	24
ERR	Memory allocation error	No calculation performed	XP_CFI_TARGET_REFLECT ED_MEMORY_ERR	25
ERR	Could not compute the satellite to target range	No calculation performed	XP_CFI_TARGET_REFLECT ED_RANGE_CALC_ERR	26
WARN	Degraded solution returned	Calculation performed	XP_CFI_ARGET_REFLECTED_DEGRADED_SOL_WARN	27

7.90 xp_target_travel_time

7.90.1 Overview

The **xp_target_travel_time** CFI function computes the point of the line or sight from the satellite (defined by an elevation and an azimuth angle expressed in the selected Attitude Frame) at a given travel time along the (curved) line of sight.

The light travel time (from the satellite to the target or vice versa) can be taken into account by the computations. For details about light propagation mode see the section 4.1.2.3.

7.90.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     long            sat_id;
     xp_attitude_id attitude_id = {NULL};
     xp_atmos_id    atmos_id = {NULL};
     xp_dem_id      dem_id = {NULL};
     xp_target_id   target_id = {NULL};
     long deriv, iray;
     double los_az, los_el, travel_time
     double los_az_rate, los_el_rate, travel_time_rate, freq;
     long ierr[XP_NUM_ERR_TARGET_TRAVEL_TIME], status,
              num_user_target, num_los_target;

     status = xp_target_travel_time(&sat_id,
               &attitude_id,
               &atmos_id,
               &dem_id,
               &deriv, &los_az,
               &los_el, &travel_time, &los_az_rate, &los_el_rate,
               &travel_time_rate, &iray, &freq,
               &num_user_target, &num_los_target,
               &target_id, ierr);

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

```
      long run_id;

      status = xp_target_travel_time_run(&run_id,
                 &attitude_id,
                 &deriv, &los_az,
                 &los_el, &travel_time, &los_az_rate, &los_el_rate,
                 &travel_time_rate, &iray, &freq,
                 &num_user_target, &num_los_target,
                 &target_id, ierr);
  }
```

The `XP_NUM_ERR_TARGET_TRAVEL_TIME` constant is defined in the file *explorer_pointing.h.*

7.90.3 Input Parameters

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

Table 231: Input parameters of xp_target_travel_time function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude. (input/output)	-	-
atmos_id	xp_atmos_id*	-	Structure that contains the atmosphere initialization.	-	The atmos_id has to be initialized with any of these modes: - XP_NO_REF_INIT - XP_STD_INIT - XP_USER_INIT
dem_id	xp_dem_id *	-	Structure that contains the DEM initialization.	-	-
deriv	long *	-	Derivative ID	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
los_az	double *	-	Azimuth of the LOS (Attitude Frame)	deg	>= 0 < 360
los_el	double *	-	Elevation of the LOS (Attitude Frame)	deg	>= -90 <= 90
travel_time	double *	-	Travel time along the (curved) line of sight	s	> 0
los_az_rate	double *	-	Azimuth-rate of the LOS (Attitude Frame)	deg/s	-
los_el_rate	double *	-	Elevation-rate of the LOS (Attitude Frame)	deg/s	-
travel_time_rate	double *	-	Travel time-rate along the (curved) line of sight (currently not used)	s/s	-
iray	long *	-	Not used. The atmosphere refraction model can be defined via atmos_id initialization.	-	-
freq	double *	-	Frequency of the signal	Hz	>= 0

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

- Derivative switch: deriv. See current document, Table 3 .

7.90.4 Output Parameters

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

Table 232: Output parameters of xp_target_travel_time

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
num_user_target	long*	-	Number of user defined targets calculated	-	>= 0 (Set to 1 for non multi-target routines)
num_los_target	long*	-	Number of LOS targets calculated	-	>= 0
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
ierr	long	-	Error vector	-	-

7.90.5 Warnings and Errors

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

Table 233: Error messages of xp_target_travel_time function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Attitude Id. is not initialized	No calculation performed	XP_CFI_TARGET_TRAVEL_TIME_ATTITUDE_STATUS_ERR	0
ERR	Intersection flag is not correct	No calculation performed	XP_CFI_TARGET_TRAVEL_TIME_INTER_FLAG_ER R	1
ERR	Invalid Frequency	No calculation performed	XP_CFI_TARGET_TRAVEL_TIME_FREQ_ERR	2

ERR	Time reference ID is not correct	No calculation performed	XP_CFI_TARGET_TRAVEL_TIME_TIME_REF_ERR	3
ERR	Deriv flag is not correct	No calculation performed	XP_CFI_TARGET_TRAVEL_TIME_DERIV_FLAG_ER R	4
ERR	Ray Tracing Model ID is not correct	No calculation performed	XP_CFI_TARGET_TRAVEL_TIME_IRAY_ID_ERR	5
ERR	Invalid LOS Azimuth	No calculation performed	XP_CFI_TARGET_TRAVEL_TIME_LOS_AZIMUTH_E RR	6
ERR	Invalid LOS Elevation	No calculation performed	XP_CFI_TARGET_TRAVEL_TIME_LOS_ELEVATION_ERR	7
ERR	Invalid Geodetic Altitude	No calculation performed	XP_CFI_TARGET_TRAVEL_TIME_GEODETIC_ALT_ERR	8
ERR	Memory allocation error	No calculation performed	XP_CFI_TARGET_TRAVEL_TIME_MEMORY_ERR	9
ERR	Internal computation error # 3	No calculation performed	XP_CFI_TARGET_TRAVEL_TIME_INITIAL_LOOK_DIR_OR_PLANE_ERR	10
ERR	Time Reference not initialised	No calculation performed	XP_CFI_TARGET_TRAVEL_TIME_TIME_REF_INIT_ERR	11
ERR	No target was found	No calculation performed	XP_CFI_TARGET_TRAVEL_TIME_TARGET_NOT_FOUND_ERR	12
ERR	Internal computation error # 4	No calculation performed	XP_CFI_TARGET_TRAVEL_TIME_RANGE_OR_POINTING_CALC_ERR	13
WARN	Path from satellite to target occulted by the Earth	Calculation performed. A message informs the user.	XP_CFI_TARGET_TRAVEL_TIME_NEGATIVE_ALTITUDE_WARN	14
WARN	The ray tracing flag (iray) is ignored	Calculation performed. A message informs the user that this paremeter is not used. If the variable iray is equal to XP_NO_REF_INIT (=0), the warning is avoided	XP_CFI_TARGET_TRAVEL_TIME_IRAY_ID_WARN	15

7.91 xp_target_tangent_sun

7.91.1 Overview

The **xp_target_tangent_sun** CFI function computes the location of the tangent point over the Earth that is located on the line of sight that points to the Sun.

Note: a correction can be applied in order to compensate the travel time of Sun light travel time. This correction is not applied with default model. To activate this correction, the Sun model in xl_model_id must be initialized with the enum XL_MODEL_SUN_TRAVEL_TIME using the function xl_model_init (see [LIB_SUM]).

The light travel time (from the satellite to the target or vice versa) can be taken into account by the computations. For details about light propagation mode see the section 4.1.2.3.

7.91.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     long             sat_id;
     xp_attitude_id attitude_id = {NULL};
     xp_atmos_id    atmos_id = {NULL};
     xp_dem_id      dem_id = {NULL};
     xp_target_id   target_id = {NULL};
     long deriv,       iray;
     double freq;
     long ierr[XP_NUM_ERR_TARGET_TANGENT_SUN], status,
          num_user_target, num_los_target;

     status = xp_target_tangent_sun(&sat_id,
                &attitude_id, &atmos_id, &dem_id,
                &deriv, &iray, &freq,
                &num_user_target, &num_los_target,
                &target_id, ierr);

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

     status = xp_target_tangent_sun_run(&run_id,
```

```
                &attitude_id,
                &deriv, &iray, &freq,
                &num_user_target, &num_los_target,
                &target_id, ierr);
}
```

The XP_NUM_ERR_TARGET_TANGENT_SUN constant is defined in the file *explorer_pointing.h.*

7.91.3 Input Parameters

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

Table 234: Input parameters of xp_target_tangent_sun function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude. (input/output)	-	-
atmos_id	xp_atmos_id*	-	Structure that contains the atmosphere initialization.	-	-
dem_id	xp_dem_id *	-	Structure that contains the DEM initialization.	-	-
deriv	long *	-	Derivative ID	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
iray	long *	-	Not used. The atmosphere refraction model can be defined via atmos_id initialization.	-	-
freq	double *	-	Frequency of the signal	Hz	>= 0

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

- Derivative switch: deriv. See current document, Table 3 .

7.91.4 Output Parameters

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

Table 235: Output parameters of xp_target_tangent_sun

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
num_user_target	long*	-	Number of user defined targets calculated	-	>= 0 (Set to 1 for non multi-target routines)
num_los_target	long*	-	Number of LOS targets calculated	-	>= 0
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
ierr	long	-	Error vector	-	-

7.91.5 Warnings and Errors

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

Table 236: Error messages of xp_target_tangent_sun function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Attitude ID is not initialized	No calculation performed	XP_CFI_SUN_ATTITUDE_ STATUS_ERR	0
ERR	Deriv flag is not correct	No calculation performed	XP_CFI_SUN_DERIV_FLA G_ERR	1
ERR	Ray Tracing Model ID is not correct	No calculation performed	XP_CFI_SUN_IRAY_ID_ER R	2
ERR	Invalid Frequency	No calculation performed	XP_CFI_SUN_FREQ_ERR	3
ERR	Input state vector does not satisfy loose tolerance requirement	No calculation performed	XP_CFI_SUN_INVALID_S V_ERR	4
ERR	Time Reference not initialised	No calculation performed	XP_CFI_SUN_TIME_REF_I NIT_ERR	5
ERR	Internal computation error # 1	No calculation performed	XP_CFI_SUN_SUN_POSITI ON_CALC_ERR	6
ERR	Internal computation error # 2	No calculation performed	XP_CFI_SUN_SUN_CS_CA LC_ERR	7
ERR	Internal computation error # 3	No calculation performed	XP_CFI_SUN_SUN_POINTI NG_CALC_ERR	8
ERR	Internal computation error # 4	No calculation performed	XP_CFI_SUN_TARGET_ST AR_ERR	9
ERR	Internal computation error # 5	No calculation performed	XP_CFI_SUN_TG_PT_BEHI ND_LOOK_DIR_ERR	10

WARN	Input state vector does not satisfy tight tolerance requirement	Calculation performed. A message informs the user.	XP_CFI_SUN_INVALID_SV_WARN	11
WARN	Tangent point is behind looking direction	Calculation performed. A message informs the user.	XP_CFI_SUN_TG_PT_BEHIND_LOOK_DIR_WARN	12
WARN	The ray tracing flag (iray) is ignored	Calculation performed. A message informs the user that this paremeter is not used. If the variable iray is equal to XP_NO_REF_INIT (=0), the warning is avoided.	XP_CFI_SUN_IRAY_ID_WARN	13

7.92 xp_target_tangent_moon

7.92.1 Overview

The **xp_target_tangent_moon** CFI function computes the location of the tangent point over the Earth that is located on the line of sight that points to the Moon.

7.92.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
{
     long             sat_id;
     xp_attitude_id attitude_id = {NULL};
     xp_atmos_id    atmos_id = {NULL};
     xp_dem_id      dem_id = {NULL};
     xp_target_id   target_id = {NULL};
     long deriv,       iray;
     double freq;
     long ierr[XP_NUM_ERR_TARGET_TANGENT_MOON], status,
         num_user_target, num_los_target;

     status = xp_target_tangent_moon(&sat_id,
              &attitude_id, &atmos_id, &dem_id,
              &deriv, &iray, &freq,
              &num_user_target, &num_los_target,
              &target_id, ierr);

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

     status = xp_target_tangent_moon_run(&run_id,
              &attitude_id,
              &deriv, &iray, &freq,
              &num_user_target, &num_los_target,
              &target_id, ierr);
```

```
}
```

The `XP_NUM_ERR_TARGET_TANGENT_MOON` constant is defined in the file *explorer_pointing.h*.

7.92.3 Input Parameters

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

Table 237: Input parameters of xp_tangent_target_moon function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude. (input/output)	-	-
atmos_id	xp_atmos_id*	-	Structure that contains the atmosphere initialization.	-	-
dem_id	xp_dem_id *	-	Structure that contains the DEM initialization.	-	-
deriv	long *	-	Derivative ID	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
iray	long *	-	Not used. The atmosphere refraction model can be defined via atmos_id initialization.	-	-
freq	double *	-	Frequency of the signal	Hz	>= 0

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

- Derivative switch: deriv. See current document, Table 3 .

7.92.4 Output Parameters

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

Table 238: Output parameters of xp_tangent_target_moon

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
num_user_target	long*	-	Number of user defined targets calculated	-	>= 0 (Set to 1 for non multi-target routines)

num_los_target	long*	-	Number of LOS targets calculated	-	>= 0
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
ierr	long	-	Error vector	-	-

7.92.5 Warnings and Errors

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

Table 239: Error messages of xp_target_tangent_moon function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Attitude ID is not initialized	No calculation performed	XP_CFI_MOON_ATTITUDE_STATUS_ERR	0
ERR	Deriv flag is not correct	No calculation performed	XP_CFI_MOON_DERIV_FLAG_ERR	1
ERR	Ray Tracing Model ID is not correct	No calculation performed	XP_CFI_MOON_IRAY_ID_ERR	2
ERR	Invalid Frequency	No calculation performed	XP_CFI_MOON_FREQ_ER R	3
ERR	Input state vector does not satisfy loose tolerance requirement	No calculation performed	XP_CFI_MOON_INVALID_SV_ERR	4
ERR	Time Reference not initialised	No calculation performed	XP_CFI_MOON_TIME_REF_INIT_ERR	5
ERR	Internal computation error #1	No calculation performed	XP_CFI_MOON_MOON_POSITION_CALC_ERR	6
ERR	Internal computation error #2	No calculation performed	XP_CFI_MOON_MOON_CS_CALC_ERR	7
ERR	Internal computation error #3	No calculation performed	XP_CFI_MOON_MOON_POINTING_CALC_ERR	8

ERR	Internal computation error #4	No calculation performed	XP_CFI_MOON_TARGET_STAR_ERR	9
ERR	Internal computation error #5	No calculation performed	XP_CFI_MOON_TG_PT_BEHIND_LOOK_DIR_ERR	10
WARN	Input state vector does not satisfy tight tolerance requirement	Calculation performed. A message informs the user.	XP_CFI_MOON_INVALID_SV_WARN	11
WARN	Tangent point is behind looking direction	Calculation performed. A message informs the user.	XP_CFI_MOON_TG_PT_BEHIND_LOOK_DIR_WARN	12
WARN	The ray tracing flag (iray) is ignored	Calculation performed. A message informs the user that this paremeter is not used. If the variable iray is equal to XP_NO_REF_INIT (=0), the warning is avoided.	XP_CFI_MOON_IRAY_ID_WARN	13

7.93 xp_target_sc

7.93.1 Overview

The **xp_target_sc** CFI function computes the pointing from one satellite to another satellite.

The light travel time (from the satellite to the target or vice versa) can be taken into account by the computations. For details about light propagation mode see the section 4.1.2.3.

7.93.2 Calling Interface

The calling interface of the **xp_target_sc** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
     long sat_id1, sat_id2;
     xp_attitude_id attitude_id1 = {NULL};
     xp_attitude_id attitude_id2 = {NULL};
     xp_target_id target_id = {NULL};
     long deriv;
     long ierr[XP_NUM_ERR_TARGET_SC], status, num_user_target,
              num_los_target;

     status = xp_target_sc(&sat_id1, &attitude_id1,
                           &sat_id2, &attitude_id2,
                           &deriv,
                           &num_user_target, &num_los_target,
                           &target_id, ierr);
```

The `XP_NUM_ERR_TARGET_SC` constant is defined in the file *explorer_pointing.h.*

7.93.3 Input Parameters

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

Table 240: Input parameters of xp_target_sc function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id1	long *	-	Satellite ID of source satellite	-	Complete
attitude_id1	xp_attitude _id*	-	Structure that contains the Attitude (input/output) of	-	-

			source satellite		
sat_id2	long *	-	Satellite ID of target satellite	-	Complete
attitude_id2	xp_attitude_id*	-	Structure that contains the Attitude (input/output) of target satellite	-	-
deriv	long *	-	Derivative ID	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND

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

- Derivative switch: deriv. See current document, Table 3.

7.93.4 Output Parameters

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

Table 241: Output parameters of xp_target_sc

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
num_user_target	long*	-	Number of user defined targets calculated	-	>= 0 (Set to 1 for non multi-target routines)
num_los_target	long*	-	Number of LOS targets calculated	-	>= 0
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
ierr	long	-	Error vector	-	-

7.93.5 Warnings and Errors

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

Table 242: Error messages of xp_target_sc function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Attitude Id. is not initialized	No calculation performed	XP_CFI_TARGET_SC_ATTITUDE_STATUS_ERR	0
ERR	Deriv flag is not correct	No calculation performed	XP_CFI_TARGET_SC_DERIV_FLAG_ERR	1
ERR	Error in call to xp_target_generic	No calculation performed	XP_CFI_TARGET_SC_TARGET_GENERIC_ERR	2
ERR	Error changing reference frame	No calculation performed.	XP_CFI_TARGET_SC_CHANGE_FRAME_ERR	3
ERR	Error computing pointing parameters	No calculation performed	XP_CFI_TARGET_SC_POINT_PARAM_COMPUTE_ERR	4

7.94 xp_multi_target_inter

7.94.1 Overview

The **xp_multi_target_inter** CFI function computes the first or the second intersection points of the line of sight from the satellite (defined by an elevation and an azimuth angle expressed in the selected Attitude Frame) with surfaces located at certain geodetic altitudes over the Earth.

The light travel time (from the satellite to the target or vice versa) can be taken into account by the computations. For details about light propagation mode see the section 4.1.2.3.

7.94.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     long             sat_id;
     xp_attitude_id attitude_id = {NULL};
     xp_atmos_id      atmos_id = {NULL};
     xp_dem_id        dem_id = {NULL};
     xp_target_id     target_id = {NULL};
     long deriv, inter_flag, iray;
     double los_az, los_el, geod_alt[XP_MAX_NUM_MULTI_TARGET],
              los_az_rate, los_el_rate, freq;
     long ierr[XP_NUM_ERR_MULTI_TARGET_INTER], num_target, status
              num_user_target, num_los_target;

     status = xp_multi_target_inter(&sat_id,
               &attitude_id,
               &atmos_id,
               &dem_id,
               &deriv, &inter_flag, &los_az,
               &los_el, &num_target, geod_alt, &los_az_rate,
               &los_el_rate, &iray, &freq,
               &num_user_target, &num_los_target,
               &target_id, ierr);

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

```
      long run_id;

      status = xp_multi_target_inter_run(&run_id,
                 &attitude_id,
                 &deriv, &inter_flag, &los_az,
                 &los_el, &num_target, geod_alt, &los_az_rate,
                 &los_el_rate, &iray, &freq,
                 &num_user_target, &num_los_target,
                 &target_id, ierr);
  }
```

The XP_NUM_ERR_MULTI_TARGET_INTER constant is defined in the file *explorer_pointing.h.*

7.94.3 Input Parameters

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

Table 243: Input parameters of xp_multi_target_inter function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude. (input/output)	-	-
atmos_id	xp_atmos_id*	-	Structure that contains the atmosphere initialization.	-	The atmos_id has to be initialized with any of these modes: - XP_NO_REF_INIT - XP_STD_INIT - XP_USER_INIT
dem_id	xp_dem_id *	-	Structure that contains the DEM initialization.	-	-
deriv	long *	-	Derivative ID	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
inter_flag	long *	-	Flag for first or second inter section point selection	-	Allowed values: (1) XP_INTER_1ST (2) XP_INTER_2ND
los_az	double *	-	Azimuth of the LOS (Attitude Frame)	deg	>= 0 < 360
los_el	double *	-	Elevation of the LOS (Attitude Frame)	deg	>= -90 <= 90
num_target	long *	-	Number of user defined altitudes	-	> 0
geod_alt	double [XP_MAX_NUM_MULTI_TARGET]	-	Geodetic altitude over the Earth, sorted vector, strict monotonic decreasing	m	>= -bWGS
los_az_rate	double *	-	Azimuth-rate of the LOS (Attitude Frame)	deg/s	-
los_el_rate	double *	-	Elevation-rate of the LOS (Attitude Frame)	deg/s	-

iray	long *	-	Not used. The atmosphere refraction model can be defined via atmos_id initialization.	-	-
freq	double *	-	Frequency of the signal	Hz	>= 0

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

- Derivative switch: deriv. See current document, Table 3 .
- Intersection flag: inter_flag. See current document, Table 3 .

7.94.4 Output Parameters

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

Table 244: Output parameters of *xp_multi_target_inter*

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
num_user_target	long*		Number of user defined targets calculated		>= 0 <= num_target
num_los_target	long*		Number of LOS targets calculated		>= 0
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
ierr	long	-	Error vector	-	-

7.94.5 Warnings and Errors

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

Table 245: Error messages of xp_multi_target_inter function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Attitude Id. is not initialized	No calculation performed	XP_CFI_MULTI_TARGET_INTER_ATTITUDE_STATUS_ERR	0
ERR	Intersection flag is not correct	No calculation performed	XP_CFI_MULTI_TARGET_INTER_INTER_FLAG_ERR	1
ERR	Invalid Frequency	No calculation performed	XP_CFI_MULTI_TARGET_INTER_FREQ_ERR	2
ERR	Time reference ID is not correct	No calculation performed	XP_CFI_MULTI_TARGET_INTER_TIME_REF_ERR	3
ERR	Deriv flag is not correct	No calculation performed	XP_CFI_MULTI_TARGET_INTER_DERIV_FLAG_ERR	4
ERR	Ray Tracing Model ID is not correct	No calculation performed	XP_CFI_MULTI_TARGET_INTER_IRAY_ID_ERR	5
ERR	Invalid LOS Azimuth	No calculation performed	XP_CFI_MULTI_TARGET_INTER_LOS_AZIMUTH_E RR	6
ERR	Invalid LOS Elevation	No calculation performed	XP_CFI_MULTI_TARGET_INTER_LOS_ELEVATION_ERR	7
ERR	Invalid Geodetic Altitude	No calculation performed	XP_CFI_MULTI_TARGET_INTER_GEODETIC_ALT_E RR	8
ERR	Memory allocation error	No calculation performed	XP_CFI_MULTI_TARGET_INTER_MEMORY_ERR	9
ERR	Internal computation error #3	No calculation performed	XP_CFI_MULTI_TARGET_INTER_INITIAL_LOOK_DIR_OR_PLANE_ERR	10
ERR	Time Reference not initialised	No calculation performed	XP_CFI_MULTI_TARGET_INTER_TIME_REF_INIT_E RR	11
ERR	No target was found	No calculation performed	XP_CFI_MULTI_TARGET_INTER_TARGET_NOT_FOUND_ERR	12
ERR	Internal computation error #4	No calculation performed	XP_CFI_MULTI_TARGET_INTER_RANGE_OR_POINTING_CALC_ERR	13
WARN	Path from satellite to target occulted by the Earth	Calculation performed. A message informs the user.	XP_CFI_MULTI_TARGET_INTER_NEGATIVE_ALTITUDE_WARN	14
WARN	The ray tracing flag (iray) is	Calculation performed.	XP_CFI_MULTI_TARGET_INT	15

	ignored	A message informs the user that this paremeter is not used. If the variable iray is equal to XP_NO_REF_INIT (=0), the warning is avoided.	ER_IRAY_ID_WARN	

7.95 xp_multi_target_travel_time

7.95.1 Overview

The **xp_multi_target_travel_time** CFI function computes the points of the line or sight from the satellite (defined by an elevation and an azimuth angle expressed in the selected Attitude Frame) at given travel times along the (curved) line of sight.

The light travel time (from the satellite to the target or vice versa) can be taken into account by the computations. For details about light propagation mode see the section 4.1.2.3.

7.95.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     long            sat_id;
     xp_attitude_id attitude_id = {NULL};
     xp_atmos_id    atmos_id = {NULL};
     xp_dem_id      dem_id = {NULL};
     xp_target_id   target_id = {NULL};
     long deriv, iray;
     double los_az, los_el, travel_time[XP_MAX_NUM_MULTI_TARGET];
     double los_az_rate, los_el_rate, travel_time_rate, freq;
     long num_target, num_user_target, num_los_target;
     long ierr[XP_NUM_ERR_MULTI_TARGET_TRAVEL_TIME], status;

     status = xp_multi_target_travel_time(&sat_id,
                &attitude_id,
                &atmos_id,
                &dem_id,
                &deriv, &los_az, &los_el,
                &num_target, travel_time, &los_az_rate,
                &los_el_rate,    &travel_time_rate, &iray, &freq,
                &num_user_target, &num_los_target,
                &target_id, ierr);

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

```
      long run_id;

      status = xp_multi_target_travel_time_run(&run_id,
                 &attitude_id,
                 &deriv, &los_az, &los_el,
                 &num_target, travel_time, &los_az_rate,
                 &los_el_rate,    &travel_time_rate, &iray, &freq,
                 &num_user_target, &num_los_target,
                 &target_id, ierr);
  }
```

The XP_NUM_ERR_MULTI_TARGET_TRAVEL_TIME constant is defined in the file *explorer_pointing.h.*

7.95.3 Input Parameters

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

Table 246: Input parameters of xp_multi_target_travel_time function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude. (input/output)	-	-
atmos_id	xp_atmos_id*	-	Structure that contains the atmosphere initialization.	-	The atmos_id has to be initialized with any of these modes: - XP_NO_REF_INIT - XP_STD_INIT - XP_USER_INIT
dem_id	xp_dem_id *	-	Structure that contains the DEM initialization.	-	-
deriv	long *	-	Derivative ID	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
los_az	double *	-	Azimuth of the LOS (Attitude Frame)	deg	>= 0 < 360
los_el	double *	-	Elevation of the LOS (Attitude Frame)	deg	>= -90 <= 90
num_target	long *	-	Number of user defined times	-	> 0
travel_time	double [XP_MAX_NUM_MULTI_TARGET]	-	Travel time along the (curved) line of sight,sorted vector, strict monotonic increasing	s	> 0
los_az_rate	double *	-	Azimuth-rate of the LOS (Attitude Frame)	deg/s	-
los_el_rate	double *	-	Elevation-rate of the LOS (Attitude Frame)	deg/s	-
travel_time_rate	double *	-	Travel time-rate along the (curved) line of sight. Constant number (currently not used).	s/s	-
iray	long *	-	Not used. The atmosphere	-	-

			refraction model can be defined via atmos_id initialization.		
freq	double *	-	Frequency of the signal	Hz	>= 0

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

- Derivative switch: deriv. See current document, Table 3 .

7.95.4 Output Parameters

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

Table 247: Output parameters of xp_multi_target_travel_time

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
num_user_target	long*	-	Number of user defined targets calculated	-	>= 0 <= num_target
num_los_target	long*	-	Number of LOS targets calculated	-	>= 0
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
ierr	long	-	Error vector	-	-

7.95.5 Warnings and Errors

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

Table 248: Error messages of xp_multi_target_travel_time function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Attitude Id. is not initialized	No calculation performed	XP_CFI_MULTI_TARGET_TRAVEL_TIME_ATTITUDE_STATUS_ERR	0
ERR	Intersection flag is not correct	No calculation performed	XP_CFI_MULTI_TARGET_TRAVEL_TIME_INTER_FLAG_ERR	1
ERR	Invalid Frequency	No calculation performed	XP_CFI_MULTI_TARGET_TRAVEL_TIME_FREQ_ER R	2
ERR	Time reference ID is not correct	No calculation performed	XP_CFI_MULTI_TARGET_TRAVEL_TIME_TIME_REF_ERR	3
ERR	Deriv flag is not correct	No calculation performed	XP_CFI_MULTI_TARGET_TRAVEL_TIME_DERIV_FL	4

			AG_ERR	
ERR	Ray Tracing Model ID is not correct	No calculation performed	XP_CFI_MULTI_TARGET_ TRAVEL_TIME_IRAY_ID_ ERR	5
ERR	Input state vector does not satisfy loose tolerance requirement	No calculation performed	XP_CFI_MULTI_TARGET_ TRAVEL_TIME_INVALID_ SV_ERR	6
ERR	Invalid LOS Azimuth	No calculation performed	XP_CFI_MULTI_TARGET_ TRAVEL_TIME_LOS_AZI MUTH_ERR	7
ERR	Invalid LOS Elevation	No calculation performed	XP_CFI_MULTI_TARGET_ TRAVEL_TIME_LOS_ELE VATION_ERR	8
ERR	Invalid Geodetic Altitude	No calculation performed	XP_CFI_MULTI_TARGET_ TRAVEL_TIME_GEODETI C_ALT_ERR	9
ERR	Memory allocation error	No calculation performed	XP_CFI_MULTI_TARGET_ TRAVEL_TIME_MEMORY _ERR	10
ERR	Internal computation error #3	No calculation performed	XP_CFI_MULTI_TARGET_ TRAVEL_TIME_INITIAL_L OOK_DIR_OR_PLANE_ER R	11
ERR	Time Reference not initialised	No calculation performed	XP_CFI_MULTI_TARGET_ TRAVEL_TIME_TIME_REF _INIT_ERR	12
ERR	No target was found	No calculation performed	XP_CFI_MULTI_TARGET_ TRAVEL_TIME_TARGET_ NOT_FOUND_ERR	13
ERR	Internal computation error #4	No calculation performed	XP_CFI_MULTI_TARGET_ TRAVEL_TIME_RANGE_O R_POINTING_CALC_ERR	14
WARN	Input state vector does not satisfy tight tolerance requirement	Calculation performed. A message informs the user.	XP_CFI_MULTI_TARGET_ TRAVEL_TIME_INVALID_ SV_WARN	15
WARN	Path from satellite to target occulted by the Earth	Calculation performed. A message informs the user.	XP_CFI_MULTI_TARGET_ TRAVEL_TIME_NEGATIV E_ALTITUDE_WARN	16
WARN	The ray tracing flag (iray) is ignored	Calculation performed. A message informs the user that this paremeter is not used. If the variable iray is equal to XP_NO_REF_INIT (=0), the warning is avoided.	XP_MULTI_TARGET_TRAVEL _TIME_IRAY_ID_WARN	17

7.96 xp_target_list_inter

7.96.1 Overview

The **xp_target_list_inter** CFI function computes the first or the second intersection point of the line of sight from the satellite (expressed as pairs of azimuth and elevation angles in the selected Attitude Frame) with a surface located at a certain geodetic altitude over the Earth.

The sets of azimuth and elevation points can be defined in 3 different ways:

- A list of azimuth and elevation pairs.
- A strip of lines of sight, with a fixed azimuth and elevation angles changing with a given step.
- A grid of lines of sight, with both azimuth and elevation angles changing with a given step.

For each pair a user target is computed. To obtain the extra values for all the targets, the functions xp_target_list_extra_xxx can be used. The position of the target in the output array of these extra functions has the following criterion (note also that xp_target_extra_xxx functions can also be used to obtain the results of only one target with the same index criterion, but xp_target_list_extra_xxx are optimized to obtain the results for all the targets):

1) In case of a list, the index of the list.
2) In case of a strip: being n_el the number of elevation values (note that minimum and maximum elevation values are always included in the list):

   n_el=TRUNC((max_elevation-min_elevation)/step_elevation+1)

   The target number is computed in increasing elevation order, from lower to upper elevation:
   - For 0 ≤ i < n_el-1: target number i corresponds to pair (azimuth, min_elevation + i*step_elevation).
   - For i = n_el-1: target number n_el-1 corresponds to pair (azimuth, max_elevation).
3) In case of a grid: being n_el the number of elevation values (note that minimum and maximum elevation values are always included in the list):

   n_el=TRUNC((max_elevation-min_elevation)/step_elevation+1)

   being n_az the number of azimuth values (note that minimum and maximum azimuth values are always included in the list):

   n_az=TRUNC((max_azimuth-min_azimuth)/step_azimuth+1)

   The target number is computed by increasing azimuth and elevation order: from minimum azimuth to maximum azimuth and, for every azimuth value, from minimum elevation to maximum elevation. That is:

   - For 0 ≤ i < n_el-1: target number i corresponds to pair (min_azimuth, min_elevation + i*step_elevation).
   - For i = n_el-1: target number n_el-1 corresponds to pair (min_azimuth, max_elevation).
   - For n_el ≤ i < 2*n_el-1 target number i corresponds to pair (min_azimuth + step_azimuth, min_elevation + i*step_elevation).

...
(for 0 ≤ j < n_az-1 and defining k = i-j*n_el)

- For j*n_el ≤ i < (j+1)*n_el-1 target number i corresponds to pair (min_azimuth + j*step_azimuth, min_elevation + k*step_elevation).

- For i = (j+1)*n_el-1 target number i corresponds to pair (min_azimuth + j*step_azimuth, max_elevation).

(for j = n_az-1 and defining k=i-(n_az-1)*n_el)

- For (n_az-1)*n_el ≤ i < n_az*n_el-1: target number i corresponds to pair (max_azimuth , min_elevation + k*step_elevation).

- For i = n_az*n_el-1 target number i corresponds to pair (max_azimuth , max_elevation)

The light travel time (from the satellite to the target or vice versa) can be taken into account by the computations. For details about light propagation mode see the section 4.1.2.3.

7.96.2 Calling Interface

The calling interface of the **xp_target_list_inter** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
    long            sat_id;
    xp_attitude_id attitude_id = {NULL};
    xp_atmos_id    atmos_id = {NULL};
    xp_dem_id      dem_id = {NULL};
    xp_target_id   target_id = {NULL};
    long deriv, inter_flag;
    xp_instrument_data instrument_data;
    double geod_alt;
    long ierr[XP_NUM_ERR_TARGET_LIST_INTER], status;
    xp_target_output target_out;

    status = xp_target_list_inter(&sat_id,
             &attitude_id,
             &atmos_id,
             &dem_id,
             &deriv, &inter_flag,
             &instrument_data, &geod_alt,
```

```
                &target_out,
                &target_id, ierr);
}
```

The `XP_NUM_ERR_TARGET_INTER` constant is defined in the file *explorer_pointing.h.*

7.96.3 Input Parameters

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

Table 249: Input parameters of xp_target_lists_inter function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sat_id	long *	-	Satellite ID	-	Complete
attitude_id	xp_attitude_id*	-	Structure that contains the Attitude. (input/output)	-	-
atmos_id	xp_atmos_id*	-	Structure that contains the atmosphere initialization.	-	The atmos_id has to be initialized with any of these modes: - XP_NO_REF_INIT - XP_STD_INIT - XP_USER_INIT
dem_id	xp_dem_id*	-	Structure that contains the DEM initialization.	-	-
deriv	long *	-	Derivative ID	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
inter_flag	long *	-	Flag for first or second inter section point selection	-	Allowed values: (1) XP_INTER_1ST (2) XP_INTER_2ND
instrument_data	xp_instrument_data	-	Azimuth/elevation input data and frequency	deg	$0 \leq$ azimuth < 360 $-90 \leq$ elevation <= 90
geod_alt	double *	-	Geodetic altitude over the Earth	m	>= -bWGS

It is possible to use enumeration values rather than integer values for some of the input arguments:
- Derivative switch: deriv. See current document, Table 3.
- Intersection flag: inter_flag. See current document, Table 3 .
- Azimuth elevation input type. See current document, Table 3 .

7.96.4 Output Parameters

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

Table 250: Output parameters of xp_target_list_inter

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
target_out	xp_target_output*	-	Number of user and LOS defined targets calculated. Note: the memory allocated in this struct must be freed by the user: **free**(target_out.num_los_target);	-	≥ 0
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
ierr	long	-	Error vector	-	-

7.96.5 Warnings and Errors

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

Table 251: Error messages of xp_target_list_inter function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Attitude Id. is not initialized	No calculation performed	XP_CFI_TARGET_LIST_INTER_ATTITUDE_STATUS_ERR	0
ERR	Intersection flag is not correct	No calculation performed	XP_CFI_TARGET_LIST_INTER_I NTER_FLAG_ERR	1
ERR	Invalid Frequency	No calculation performed	XP_CFI_TARGET_LIST_INTER_ FREQ_ERR	2
ERR	Time reference ID is not correct	No calculation performed	XP_CFI_TARGET_LIST_INTER_ TIME_REF_ERR	3
ERR	Deriv flag is not correct	No calculation performed	XP_CFI_TARGET_LIST_INTER_ DERIV_FLAG_ERR	4
ERR	Ray Tracing Model ID is not	No calculation performed	XP_CFI_TARGET_LIST_INTE	5

	correct		R_I RAY_ID_ERR	
ERR	Invalid LOS Azimuth	No calculation performed	XP_CFI_TARGET_LIST_INTER_ LOS_AZIMUTH_ERR	6
ERR	Invalid LOS Elevation	No calculation performed	XP_CFI_TARGET_LIST_INTER_ LOS_ELEVATION_ERR	7
ERR	Invalid Geodetic Altitude	No calculation performed	XP_CFI_TARGET_LIST_INTER_ GEODETIC_ALT_ERR	8
ERR	Memory allocation error	No calculation performed	XP_CFI_TARGET_LIST_INTER_ MEMORY_ERR	9
ERR	Internal computation error # 3	No calculation performed	XP_CFI_TARGET_LIST_INTER_I NITIAL_LOOK_DIR_OR_PLANE_ERR	10
ERR	Time Reference not initialized	No calculation performed	XP_CFI_TARGET_LIST_INTER_ TIME_REF_INIT_ERR	11
ERR	No target was found	No calculation performed	XP_CFI_TARGET_LIST_INTER_ TARGET_NOT_FOUND_ERR	12
ERR	Internal computation error # 4	No calculation performed	XP_CFI_TARGET_LIST_INTER_ RANGE_OR_POINTING_CALC_ERR	13
WARN	Path from satellite to target occulted by the Earth	Calculation performed. A message informs the user.	XP_CFI_TARGET_LIST_INTER_ NEGATIVE_ALTITUDE_WARN	14
ERR	Wrong instrument data type	No calculation performed	XP_CFI_TARGET_LIST_INTER_INSTRUMENT_TYPE_ERR	15
ERR	Error linking IDs	No calculation performed	XP_CFI_TARGET_LIST_INTER_LINK_IDS_ERR	16
ERR	Maximum number of targets exceeded	No calculation performed	XP_CFI_TARGET_LIST_INTER_MAX_TARGETS_ERR	17

7.97 xp_target_extra_vector

7.97.1 Overview

The **xp_target_extra_vector** CFI function provides the following output parameters for the target(s) in input data structure.: target position, velocity and acceleration vectors, line of sight direction, range, travel time and their corresponding derivatives.

Note on target_number with targets computed with xp_target_list_inter or xp_target_range:

the target_number to be used to get a specific LOS target is an incremental number. That is, if there are N user targets US1, US2, ... USN and a number of LOS targets for every user target NLOS1, NLOS2, ..., NLOSN, if we want to get LOS target with index 1 corresponding to user target US3, the target_number to be used is NLOS1+NLOS2+1.

The target_number can also be got with the array returned by xp_target_get_id_data.

7.97.2 Calling Interface

The calling interface of the **xp_target_extra_vector** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
     long choice, target_type, target_number;
     xp_target_id   target_id = {NULL};
     double vector_results[XP_SIZE_TARGET_RESULT_VECTOR],
          vector_results_rate[XP_SIZE_TARGET_RESULT_VECTOR],
          vector_results_rate_rate[XP_SIZE_TARGET_RESULT_VECTOR];
     long ierr[XP_NUM_ERR_TARGET_EXTRA_VECTOR], status;

     status = xp_target_extra_vector (&target_id, &choice,
                                      &target_type, &target_number,
                                      vector_results,
                                      vector_results_rate,
                                      vector_results_rate_rate, ierr);
}
```

The `XP_SIZE_TARGET_RESULT_VECTOR` and `XP_NUM_ERR_TARGET_EXTRA_VECTOR` constants aredefined in the file *explorer_pointing.h.*

7.97.3 Input Parameters

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

Table 252: Input parameters of xp_target_extra_vector function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
choice	long *	-	Flag to select the extra results to be computed. Even though derivatives could be requested by user, they will not be calculated if the target was computed without derivatives (a warning is raised in this case).	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
target_type	long *		Flag to select the type of target		XP_USER_TARGET_TYPE XP_LOS_TARGET_TYPE XP_DEM_TARGET_TYPE
target_number	long *	-	Target number		>= 0

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

- Choice. (See Table 3).

7.97.4 Output Parameters

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

Table 253: Output parameters of xp_target_extra_vector

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
vector_results	double[XP_SIZE_TARGET_ RESULT_VECTOR]	[0]	Target Position X (Earth-Fixed)	m	
		[1]	Target Position Y (Earth-Fixed)	m	
		[2]	Target Position Z (Earth-Fixed)	m	
		[3]	Direction LOS X (Earth-Fixed)	-	
		[4]	Direction LOS Y (Earth-Fixed)	-	
		[5]	Direction LOS Z (Earth-Fixed)	-	
		[6]	Range to Attitude Frame Origin	m	
		[7]	Travel Time to Attitude Frame Origin	s	
		[8]	Target time UTC	days	
		[9:XP_SIZE_TARGET_RESULT_VECTOR]	(dummy)	-	-
vector_results_rate	double[XP_SIZE_TARGET_ RESULT_VECTOR]	[0]	Target Velocity X (Earth-Fixed)	m/ s	
		[1]	Target Velocity Y (Earth-Fixed)	m/ s	
		[2]	Target Velocity Z (Earth-Fixed)	m/ s	
		[3]	Direction Rate LOS X (Earth-Fixed)	1/s	
		[4]	Direction Rate LOS Y (Earth-Fixed)	1/s	
		[5]	Direction Rate LOS Z (Earth-Fixed)	1/s	
		[6]	Range Rate to Attitude Frame Origin	m/s	
		[7]	Travel Time Rate to Attitude Frame Origin	s/s	
		[8]	Dummy		
		[9:XP_SIZE_TARGET_RESULT_VECTOR]	(dummy)	-	-
vector_results_rate_rate	double[XP_SIZE_TARGET_ RESULT_VECTOR]	[0]	Target Acceleration X (Earth-Fixed)	m/s2	
		[1]	Target Acceleration Y (Earth-Fixed)	m/s2	
		[2]	Target Acceleration Z (Earth-Fixed)	m/s2	
		[3]	Direction Rate Rate LOS X (Earth-	1/s2	

			Fixed)		
		[4]	Direction Rate Rate LOS Y (Earth-Fixed)	1/s2	
		[5]	Direction Rate Rate LOS Z (Earth-Fixed)	1/s2	
		[6]	Range Rate Rate to Attitude Frame Origin	m/s2	
		[7]	Travel Time Rate Rate to Attitude Frame Origin	s/s2	
		[8]	Dummy		
		[9:XP_SIZE_TARGET_RESULT_VECTOR]	(dummy)	-	-
ierr	long	-	Error vector	-	-

Note that:

- first derivative parameters (vector_results_rate) are returned as zeros if derivative flag (deriv) was set to NO_DER when the target was computed and that second derivative parameters (vector_results_rate_rate) are returned as zeros if derivative flag (deriv) was set to NO_DER or 1ST_DER.
- when a refraction mode is selected, the second derivative parameters (vector_results_rate_rate) are returned as zeros.
- when light propagation mode is used the target position is at **target time** (see section 4.1.2.3).

7.97.5 Warnings and Errors

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

Table 254: Error messages of xp_target_extra_vector function

Error type	Error message	Cause and impact	Error code	Error No
ERR	The Target ID does not contain any data	No calculation performed	XP_CFI_TARGET_EXTRA_VECTOR_NO_DATA_ERR	0

ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_EXTRA_VECTOR_NO_SUCH_USER_TARGET_ERR	1
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_EXTRA_VECTOR_NO_SUCH_LOS_TARGET_ERR	2
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_EXTRA_VECTOR_NO_SUCH_EARTH_TARGET_ERR	3
ERR	Could not compute the DEM target	No calculation performed	XP_CFI_TARGET_EXTRA_VECTOR_EARTH_TARGET_COMPUT_ERR	4
ERR	Wrong target type	No calculation performed	XP_CFI_TARGET_EXTRA_VECTOR_WRONG_TARGET_TYPE_ERR	5
ERR	Wrong deriv input flag	No calculation performed	XP_CFI_TARGET_EXTRA_VECTOR_DERIV_FLAG_ERR	6
WARN	1st. Derivatives are not available	Calculation performed. A message informs the user.	XP_CFI_TARGET_EXTRA_VECTOR_DER_1ST_NOT_AVAIL_WARN	7
WARN	2nd. Derivatives are not available	Calculation performed. A message informs the user.	XP_CFI_TARGET_EXTRA_VECTOR_DER_2ND_NOT_AVAIL_WARN	8
WARN	DEM files were not found	Calculation performed.	XP_CFI_TARGET_EXTRA_VECTOR_ELLIPSOID_WARN	9
WARN	Warning in XP_DEM_inter	Calculation performed.	XP_CFI_TARGET_EXTRA_VECTOR_DEM_INTER_WARN	10
ERR	Error converting time to UTC	No calculation performed	XP_CFI_TARGET_EXTRA_VECTOR_CONVERT_TO_UTC_ERR	11
WARN	No precise intersection found with DEM. Degraded solution returned	Calculation performed	XP_CFI_TARGET_EXTRA_VECTOR_DEM_DEGRADED_SOLUTION_WARN	12

7.98 xp_target_list_extra_vector

7.98.1 Overview

The **xp_target_list_extra_vector** CFI function provides the same results as xp_target_extra_vector function but for all the targets computed with xp_target_list_inter function.

This function has been optimized to improve the run-time performance of the target computation of all the targets and runs in multithreading (Remark: multithreading is not enabled on MacOS platforms, see section 6).

7.98.1.1 Note on multithreading:

Improvement in performance due to mutithread parallelization depends on many factors, including hardware set-up (i.e. multicore processor) and number of targets computed. In some cases (e.g. low number of targets), due to the high overhead of starting threads, parallelization may even degrade performances. In this case, it is recommended to disable multithreading or reduce the number of threads by using omp_set_num_threads openmp function.

NOTE for MACIN64 platform, Xcode 5 users:

As of version 5, llvm-gcc has been removed from Xcode and the default compiler is clang.

clang can build an application linking against the EOCFI C / C++ libraries.

However openmp is not supported by clang. Therefore, the –fopenmp shall not be used.

Functions using parallelized computations, e.g. xp_target_list... functions will work in single-thread mode.

7.98.2 Calling Interface

The calling interface of the **xp_target_list_extra_vector** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
     long choice, target_type, target_number;
     xp_target_id target_id = {NULL};
       xp_target_extra_vector_results_list list;
       long ierr[XP_NUM_ERR_TARGET_LIST_EXTRA_VECTOR], status;

     status = xp_target_list_extra_vector (&target_id, &choice,
                                            &target_type,
                                            &list, ierr);
}
```

The `XP_NUM_ERR_TARGET_LIST_EXTRA_VECTOR` constant is defined in the file *explorer_pointing.h.*

7.98.3 Input Parameters

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

Table 255: Input parameters of xp_target_list_extra_vector function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
choice	long *	-	Flag to select the extra results to be computed. Even though derivatives could be requested by user, they will not be calculated if the target was computed without derivatives (a warning is raised in this case).	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
target_type	long *		Flag to select the type of target		XP_USER_TARGET_TYPE XP_LOS_TARGET_TYPE XP_DEM_TARGET_TYPE

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

- Choice. (See Table 3).

7.98.4 Output Parameters

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

Table 256: Output parameters of xp_target_list_extra_vector

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
list	xp_target_extra_vector_results_list	-	List of extra results. Note: the memory allocated in this struct must be freed by the user with: **free**(list.extra_vector_results);	-	-
ierr	long	-	Error vector	-	-

The values corresponding to returned arrays are the same as in the case of xp_target_extra_vector (see 7.97.4).

7.98.5 Warnings and Errors

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

Table 257: Error messages of xp_target_list_extra_vector function

Error type	Error message	Cause and impact	Error code	Error No
ERR	The Target ID does not contain any data	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_V ECTOR_NO_DATA_ERR	0
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_V ECTOR_NO_SUCH_USER_TARGET_ERR	1
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_V ECTOR_NO_SUCH_EARTH_TARGET_ERR	2
ERR	Could not compute the DEM target	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_V ECTOR_EARTH_TARGET_COMPUT_ERR	3
ERR	Wrong target type	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_	4

			V ECTOR_WRONG_TARGET_T YPE_ERR	
ERR	Wrong deriv input flag	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_ V ECTOR_DERIV_FLAG_ERR	5
WARN	1st. Derivatives are not available	Calculation performed. A message informs the user.	XP_CFI_TARGET_LIST_EXTRA_ V ECTOR_DER_1ST_NOT_AVA IL_WARN	6
WARN	2nd. Derivatives are not available	Calculation performed. A message informs the user.	XP_CFI_TARGET_LIST_EXTRA_ V ECTOR_DER_2ND_NOT_AV AIL_WARN	7
WARN	DEM files were not found	Calculation performed.	XP_CFI_TARGET_LIST_EXTRA_ VECTOR_ELLIPSOID_WARN	8
WARN	Warning in XP_DEM_inter	Calculation performed.	XP_CFI_TARGET_LIST_EXTRA_ VECTOR_DEM_INTER_WARN	9
ERR	Error allocating memory	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_ VECTOR_MEMORY_ERR	10
WARN	No precise intersection found with DEM. Degraded solution returned	Calculation performed	XP_CFI_TARGET_LIST_EXTRA_ VECTOR_DEM_DEGRADED_SO LUTION_WARN	11
ERR	Void value detected in DEM	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_ VECTOR_DEM_VOID_VALUE_D ETECTED_ERR	12

7.99 xp_target_extra_main

7.99.1 Overview

The **xp_target_extra_main** CFI function computes the extra parameter for the target(s) in input data structure.

Note on target_number with targets computed with xp_target_list_inter or xp_target_range:

the target_number to be used to get a specific LOS target is an incremental number. That is, if there are N user targets US1, US2, ... USN and a number of LOS targets for every user target NLOS1, NLOS2, ..., NLOSN, if we want to get LOS target with index 1 corresponding to user target US3, the target_number to be used is NLOS1+NLOS2+1.

The target_number can also be got with the array returned by xp_target_get_id_data.

7.99.2 Calling Interface

The calling interface of the **xp_target_extra_main** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
     long choice, target_type, target_number;
     double main_results[XP_SIZE_TARGET_RESULT_MAIN],
         main_results_rate[XP_SIZE_TARGET_RESULT_MAIN],
         main_results_rate_rate[XP_SIZE_TARGET_RESULT_MAIN];
     xp_target_id   target_id = {NULL};
     long ierr[XP_NUM_ERR_TARGET_EXTRA_MAIN], status;

     status = xp_target_extra_main (&target_id, &choice, &target_type,
                                    &target_number,
                                    main_results, main_results_rate,
                                    main_results_rate_rate, ierr);
}
```

The `XP_SIZE_TARGET_EXTRA_MAIN` and `XP_NUM_ERR_TARGET_RESULT_MAIN` constants aredefined in the file *explorer_pointing.h*.

7.99.3 Input Parameters

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

Table 258: Input parameters of xp_target_extra_main function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
choice	long *	-	Flag to select the extra results to be computed. Even though derivatives could be requested by user, they will not be calculated if the target was computed without derivatives (a warning is raised in this case).	-	Complete
target_type	long *		Flag to select the type of target		XP_USER_TARGET_TYPE XP_LOS_TARGET_TYPE XP_DEM_TARGET_TYPE
target_number	long *	-	Target number		>= 0

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

- Choice. (See Table 3).

7.99.4 Output Parameters

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

Table 259: Output parameters of xp_target_extra_main

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
main_results	double[XP_SIZE_TARGET_ RESULT_MAIN]	[0]	Target geocentric longitude (Earth Fixed CS)	deg	>= 0 < 360
		[1]	Target geocentric latitude (Earth Fixed CS)	deg	>= -90 <= +90
		[2]	Target geodetic latitude (Earth Fixed CS)	deg	>= -90 <= +90
		[3]	Target geodetic altitude (Earth Fixed CS)	m	-
		[4]	Satellite to target azimuth (Topocentric CS)	deg	>= 0 < 360
		[5]	Satellite to target elevation (Topocentric CS)	deg	>= -90 <= +90
		[6]	Satellite to target pointing: Azimuth (attitude frame)	deg	>= 0 < 360
		[7]	Satellite to target pointing: Elevation (attitude frame)	deg	>= -90 <= +90
		[8]	Target to satellite pointing: Azimuth (Topocentric)	deg	>= 0 < 360
		[9]	Target to satellite pointing: Elevation (Topocentric)	deg	>= -90 <= +90
		[10]	Target to source Satellite Pointing: Azimuth (attitude frame). Note: this value is only meaningful when target has been computed with xp_target_sc	deg	>= 0 < 360
		[11]	Target to source Satellite Pointing: Elevation (attitude frame) Note: this value is only meaningful when target has been computed with xp_target_sc	deg	>= -90 <= 90
		[12:XP_SIZE_TARGET_RESU	(dummy)	-	-

	LT_MAIN]			
main_results_rate double[XP_SIZE_TARGET_ RESULT_MAIN]	[0]	Target geocentric longitude rate (Earth Fixed CS)	deg/s	-
	[1]	Target geocentric latitude rate (Earth Fixed CS)	deg/s	-
	[2]	Target geodetic latitude rate (Earth Fixed CS)	deg/s	-
	[3]	Target geodetic altitude rate (Earth Fixed CS)	m/s	-
	[4]	Satellite to target azimuth rate (Topocentric CS)	deg/s	-
	[5]	Satellite to target elevation rate (Topocentric CS)	deg/s	-
	[6]	Satellite to target pointing: Azimuth rate (attitude frame)	deg/s	-
	[7]	Satellite to target pointing: Elevation rate (attitude frame)	deg/s	-
	[8]	Target to satellite pointing: Azimuth rate (Topocentric)	deg/s	-
	[9]	Target to satellite pointing: Elevation rate (Topocentric)	deg/s	-
	[10]	Target to source Satellite Pointing: Azimuth rate (attitude frame). Note: this value is only meaningful when target has been computed with xp_target_sc	deg	-
	[11]	Target to source Satellite Pointing: Elevation rate (attitude frame) Note: this value is only meaningful when target has been computed with xp_target_sc	deg	-
	[12:XP_SIZE_TARGET_RESULT_MAIN]	(dummy)	-	-
main_results_rate_rate double[XP_SIZE_TARGET_ RESULT_MAIN]	[0]	Target geocentric longitude rate-rate (Earth Fixed CS)	deg/s2	-
	[1]	Target geocentric latitude rate-rate	deg/s2	-

			(Earth Fixed CS)		
		[2]	Target geodetic latitude rate-rate (Earth Fixed CS)	deg/s2	-
		[3]	Target geodetic altitude rate-rate (Earth Fixed CS)	m/s2	-
		[4]	Satellite to target azimuth rate-rate (Topocentric CS)	deg/s2	-
		[5]	Satellite to target elevation rate-rate (Topocentric CS)	deg/s2	-
		[6]	Satellite to target pointing: Azimuth rate-rate (attitude frame)	deg/s2	-
		[7]	Satellite to target pointing: Elevation rate-rate (attitude frame)	deg/s2	-
		[8]	Target to satellite pointing: Azimuth rate-rate (Topocentric)	deg/s2	-
		[9]	Target to satellite pointing: Elevation rate-rate (Topocentric)	deg/s2	-
		[10]	Target to source Satellite Pointing: Azimuth rate rate (attitude frame). Note: this value is only meaningful when target has been computed with xp_target_sc	deg	-
		[11]	Target to source Satellite Pointing: Elevation rate rate (attitude frame) Note: this value is only meaningful when target has been computed with xp_target_sc	deg	-
		[12:XP_SIZE_TARGET_RESULT_MAIN]	(dummy)	-	-
ierr	long	-	Error vector	-	-

Note that first derivative parameters (vector_results_rate) are returned as zeros if derivative flag (deriv) was set to NO_DER when the target was computed and that second derivative parameters (vector_results_rate_rate) are returned as zeros if derivative flag (deriv) was set to NO_DER or 1ST_DER.

Note also that when a refraction mode is selected, the second derivative parameters (vector_results_rate_rate) are returned as zeros.

7.99.5 Warnings and Errors

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

Table 260: Error messages of xp_target_extra_main function

Error type	Error message	Cause and impact	Error code	Error No
ERR	No target data available	No calculation performed	XP_CFI_TARGET_EXTRA_MAIN_NO_DATA_ERR	0
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_EXTRA_MAIN_NO_SUCH_USER_TARGET_ERR	1
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_EXTRA_MAIN_NO_SUCH_LOS_TARGET_ERR	2
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_EXTRA_MAIN_NO_SUCH_EARTH_TARGET_ERR	3
ERR	Could not compute the DEM target	No calculation performed	XP_CFI_TARGET_EXTRA_MAIN_EARTH_TARGET_COMPUT_ERR	4
ERR	Wrong target type	No calculation performed	XP_CFI_TARGET_EXTRA_MAIN_WRONG_TARGET_TYPE_ERR	5
ERR	Invalid time reference in target data	No calculation performed	XP_CFI_TARGET_EXTRA_MAIN_INVALID_TIME_REF_E RR	6
ERR	Error calling to XL_Car_Geo CFI function	No calculation performed	XP_CFI_TARGET_EXTRA_MAIN_CAR_TO_GEO_ERR	7
ERR	Error getting tranformation matrix to Topocentric CS	No calculation performed	XP_CFI_TARGET_EXTRA_MAIN_TOPO_ERR	8
ERR	Error getting direction angles	No calculation performed	XP_CFI_TARGET_EXTRA_MAIN_DIR_POINTING_ERR	9
ERR	Error while changing coordinate system	No calculation performed	XP_CFI_TARGET_EXTRA_MAIN_CS_CHANGE_ERR	10
WARN	Warning: Derivatives cannot	Calculation performed	XP_CFI_TARGET_EXTRA_M	11

	be calculated		AIN_DERIV_WARN	
WARN	Warning calling to XL_Car_Geo CFI function	Calculation performed, but derivatives will not be computed	XP_CFI_TARGET_EXTRA_MAIN_AMBIGUOUS_SINGULAR_WARN	12
WARN	DEM files were not found. Intersection with the ellipsoid is returned	Calculation performed	XP_CFI_TARGET_EXTRA_MAIN_ELLIPSOID_WARN	13
WARN	Warnging in XP_DEM_inter	Calculation performed	XP_CFI_TARGET_EXTRA_MAIN_DEM_INTER_WARN	14
WARN	No precise intersection found with DEM. Degraded solution returned	Calculation performed	XP_CFI_TARGET_EXTRA_MAIN_DEM_DEGRADED_SOLUTION_WARN	15
ERR	Void value detected in DEM	No calculation performed	XP_CFI_TARGET_EXTRA_MAIN_DEM_VOID_VALUE_DETECTED_ERR	16

7.100 xp_target_list_extra_main

7.100.1 Overview

The **xp_target_list_extra_main** CFI function provides the same results as xp_target_extra_main function but for all the targets computed with xp_target_list_inter function.

This function has been optimized to improve the run-time performance of the target computation of all the targets and runs in multithreading (Remark: multithreading is not enabled on MacOS platforms, see section 6).

See note on mutithreading in section 7.98.1.1.

7.100.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     long choice, target_type, target_number;
     xp_target_id target_id = {NULL};
       xp_target_extra_main_results_list list;
       long ierr[XP_NUM_ERR_TARGET_LIST_EXTRA_MAIN], status;

     status = xp_target_list_extra_main (&target_id, &choice,
                                          &target_type,
                                          &list, ierr);
}
```

The XP_NUM_ERR_TARGET_LIST_EXTRA_MAIN constant is defined in the file *explorer_pointing.h.*

7.100.3 Input Parameters

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

Table 261: Input parameters of xp_target_list_extra_main function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
choice	long *	-	Flag to select the extra results to be computed. Even though derivatives could be requested by user, they will not be calculated if the target was computed without derivatives (a warning is raised in this case).	-	Complete
target_type	long *		Flag to select the type of target		XP_USER_TARGET_TYPE XP_LOS_TARGET_TYPE XP_DEM_TARGET_TYPE

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

- Choice. (See Table 3).

7.100.4 Output Parameters

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

Table 262: Output parameters of xp_target_list_extra_main

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
list	xp_target_extra_main_results_list	-	List of extra results. Note: the memory allocated in this struct must be freed by the user with: **free**(list.extra_main_results);	-	-
ierr	long	-	Error vector	-	-

The values corresponding to returned arrays are the same as in the case of xp_target_extra_main (see section 7.99.4).

7.100.5 Warnings and Errors

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

Table 263: Error messages of xp_target_list_extra_main function

Error type	Error message	Cause and impact	Error code	Error No
ERR	No target data available	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_M AIN_NO_DATA_ERR	0
ERR	Invalid time reference in target data	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_M AIN_INVALID_TIME_REF_ERR	1
ERR	Error calling to XL_Car_Geo CFI function	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_M AIN_CAR_TO_GEO_ERR	2
ERR	Error getting tranformation matrix to Topocentric CS	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_M AIN_TOPO_ERR	3
ERR	Error getting direction angles	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_M AIN_DIR_POINTING_ERR	4

ERR	Error while changing coordinate system	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_M AIN_CS_CHANGE_ERR	5
WARN	Warning: Derivatives cannot be calculated	Calculation performed	XP_CFI_TARGET_LIST_EXTRA_M AIN_DERIV_WARN	6
WARN	Warning calling to XL_Car_Geo CFI function	Calculation performed, but derivatives will not be computed	XP_CFI_TARGET_LIST_EXTRA_M AIN_AMBIGUOUS_SINGUL AR_WARN	7
ERR	Error allocating memory	Calculation performed	XP_CFI_TARGET_LIST_EXTRA_MAIN_MEMORY_ERR	8
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_M AIN_NO_SUCH_EARTH_TA RGET_ERR	9
ERR	Could not compute the DEM target	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_M AIN_EARTH_TARGET_COM PUT_ERR	10
WARN	DEM files were not found. Intersection with the ellipsoid is returned	Calculation performed	XP_CFI_TARGET_LIST_EXTRA_MAIN_ELLIPSOID_WARN	11
WARN	Warning in XP_DEM_inter	Calculation performed	XP_CFI_TARGET_LIST_EXTRA_MAIN_DEM_INTER_WARN	12
ERR	Wrong target type	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_M AIN_WRONG_TARGET_TYP E_ERR	13
WARN	No precise intersection found with DEM. Degraded solution returned	Calculation performed	XP_CFI_TARGET_LIST_EXTRA_MAIN_DEM_DEGRADED_SOLU TION_WARN	14
ERR	Void value detected in DEM	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_MAIN_DEM_VOID_VALUE_DETE CTED_ERR	15

7.101 xp_target_extra_aux

7.101.1 Overview

The **xp_target_extra_aux** CFI function computes auxiliary parameters for the target in input data structure.

Note on target_number with targets computed with xp_target_list_inter or xp_target_range:

the target_number to be used to get a specific LOS target is an incremental number. That is, if there are N user targets US1, US2, ... USN and a number of LOS targets for every user target NLOS1, NLOS2, ..., NLOSN, if we want to get LOS target with index 1 corresponding to user target US3, the target_number to be used is NLOS1+NLOS2+1.

The target_number can also be got with the array returned by xp_target_get_id_data.

7.101.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     long choice, target_type, target_number;
     double aux_results[XP_SIZE_TARGET_RESULT_AUX],
         aux_results_rate[XP_SIZE_TARGET_RESULT_AUX],
         aux_results_rate_rate[XP_SIZE_TARGET_RESULT_AUX];
     xp_target_id   target_id = {NULL};
     long ierr[XP_NUM_ERR_TARGET_EXTRA_AUX], status;

     status = xp_target_extra_aux(&target_id, &choice, &target_type,
                                  &target_number,
                                   aux_results, aux_results_rate,
                                   aux_results_rate_rate, ierr);
}
```

The `XP_SIZE_TARGET_RESULT_AUX` and `XP_NUM_ERR_TARGET_EXTRA_AUX` constants are defined in the file *explorer_pointing.h.*

7.101.3 Input Parameters

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

Table 264: Input parameters of xp_target_extra_aux

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
choice	long *	-	Flag to select the extra results to be computed. Even though derivatives could be requested by user, they will not be calculated if the target was computed without derivatives (a warning is raised in this case).	-	Complete
target_type	long *		Flag to select the type of target		XP_USER_TARGET_TYPE XP_LOS_TARGET_TYPE XP_DEM_TARGET_TYPE
target_number	long *	-	Target number		>= 0

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

- Choice. (See Table 3).

7.101.4 Output Parameters

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

Table 265: Output parameters of xp_target_extra_aux

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
aux_results	double[XP_SIZE_TARGET_ RESULT_AUX]	[0]	Radius of curvature in the look direction at the nadir of the target (Earth fixed CS)	m	>= 0
		[1]	Distance from the nadir of the target to the satellite nadir. (Earth fixed CS)	m	>= 0
		[2]	Minimum distance from the nadir of the tar get to the ground track (Earth Fixed CS). It is regarded as positive distance when the nadir of the target is located on the left hand side of the ground track.	m	-
		[3]	Distance from the SSP to the point located on the ground track that is at a minimum distance from the nadir of the target (Earth fixed CS) It is regarded as positive distance when that point is located on the ground track ahead the SSP (in the direction of the motion of the SSP)	m	-
		[4]	Mean Local Solar Time at target.	decimal hour	>= 0 < 24
		[5]	True Local Solar Time at target.	decimal hour	>= 0 < 24
		[6]	Right ascension at which the look direction from the satellite to the target points at tar get point. (True of Date CS)	deg	>= 0 < 360
		[7]	Declination at which the look direction from the satellite to the target points at target point. (True of Date CS)	deg	>= -90 < 90
		[8:XP_SIZE_TARGET_RESULT_AUX]	(dummy)	-	-
aux_results_rate		[0]	Radius of curvature-rate in the look direction at the nadir of the target (Earth fixed CS)	m/s	-

double[XP_SIZE_TARGET_ RESULT_AUX]					
		[1]	Distance-rate from the nadir of the target to the satellite nadir. (Earth fixed CS)	m	>= 0
		[2]	Distance-rate from the nadir of the target to the ground track (Earth fixed CS)	m/s	-
		[3]	Distance-rate from the SSP to the point located on the ground track that is at a minimum distance from the nadir of the target (Earth fixed CS)		
		[4:7]	(dummy)	-	-
		[8]	Northward component of the velocity relative to the Earth of the nadir of the target (Topocentric CS)	m/s	-
		[9]	Eastward component of the velocity relative to the Earth of the nadir of the target (Topocentric CS)	m/s	-
		[10]	Azimuth of the velocity relative to the Earth of the nadir of the target. (Topocentric CS)	deg	>= 0 < 360
		[11]	Magnitude of the velocity relative to the Earth of the nadir of the target. (Topocentric CS)	m/s	>= 0
		[12:XP_SIZE_TARGET_RESULT_AUX]	(dummy)	-	-
aux_results_rate_rate	double[XP_SIZE_TARGET_RESULT_AUX]	[0]	Radius of curvature-rate-rate in the look direction at the nadir of the target (Earth fixed CS)	m/s	-
		[1]	Distance-rate-rate from the nadir of the target to the satellite nadir. (Earth fixed CS)	m	>= 0
		[2]	Distance-rate-rate from the nadir of the target to the ground track (Earth fixed CS)	m/s	-
		[3]	Distance-rate-rate from the SSP to the point located on the ground track that is at a minimum distance from the nadir of the target (Earth fixed CS)		

		[4:XP_SIZE_TARGET_RESULT_AUX]	(dummy)	-	-
ierr	long	-	Error vector	-	-

7.101.5 Warnings and Errors

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

Table 266: Error messages of xp_target_extra_aux function

Error type	Error message	Cause and impact	Error code	Error No
ERR	No target data available	No calculation performed	XP_CFI_TARGET_EXTRA_AUX_NO_DATA_ERR	0
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_EXTRA_AUX_NO_SUCH_USER_TARGET_ERR	1
ERR	The target does not exist	No calculation performed.	XP_CFI_TARGET_EXTRA_AUX_NO_SUCH_LOS_TARGET_ERR	2
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_EXTRA_AUX_NO_SUCH_EARTH_TARGET_ERR	3
ERR	Could not compute the DEM target	No calculation performed	XP_CFI_TARGET_EXTRA_AUX_EARTH_TARGET_COMPUT_ERR	4
ERR	Wrong target type	No calculation performed	XP_CFI_TARGET_EXTRA_AUX_WRONG_TARGET_TYPE_ERR	5
ERR	Invalid time reference in target data	No calculation performed	XP_CFI_TARGET_EXTRA_AUX_INVALID_TIME_REF_ER R	6
ERR	Error calling to XL_Car_Geo CFI function	No calculation performed	XP_CFI_TARGET_EXTRA_AUX_CAR_TO_GEO_ERR	7
ERR	Error getting tranformation matrix to Topocentric CS.	No calculation performed	XP_CFI_TARGET_EXTRA_AUX_TOPO_ERR	8

ERR	Error getting direction angles	No calculation performed	XP_CFI_TARGET_EXTRA_AUX_DIR_POINTING_ERR	9
ERR	Error computing radius of curvature	No calculation performed	XP_CFI_TARGET_EXTRA_AUX_RADII_CURVATURE_CALC_ERR	10
ERR	Error computing pointing after crossing the Earth atmosphere	No calculation performed	XP_CFI_TARGET_EXTRA_AUX_POINTING_AFTER_ATM_CALC_ERR	11
ERR	Error computing distance	No calculation performed	XP_CFI_TARGET_EXTRA_AUX_DISTANCE_CALC_ERR	12
ERR	Error computing velocity of the target's nadir	No calculation performed	XP_CFI_TARGET_EXTRA_AUX_TOP_VEL_CALC_ERR	13
WARN	Error computing MLST of TLST	Calculation performed	XP_CFI_TARGET_EXTRA_AUX_MLST_OR_TLST_CALC_ERR	14
WARN	Warning: Path from satellite to target occulted by the Earth	Calculation performed	XP_CFI_TARGET_EXTRA_AUX_NEGATIVE_ALTITUDE_WARN	15
WARN	Warning calling to XL_Car_Geo CFI function	Calculation performed	XP_CFI_TARGET_EXTRA_AUX_AMBIGUOUS_SINGULAR_WARN	16
WARN	Warning: Derivatives cannot be calculated	Calculation performed	XP_CFI_TARGET_EXTRA_AUX_DERIV_WARN	17
WARN	DEM files were not found. Intersection with the ellipsoid is returned	Calculation performed	XP_CFI_TARGET_EXTRA_AUX_ELLIPSOID_WARN, WARN	18
WARN	Warnging in XP_DEM_inter	Calculation performed	XP_CFI_TARGET_EXTRA_AUX_DEM_INTER_WARN	19
ERR	This function cannot be used with a target id computed with target S/C	Calculation not performed	XP_CFI_TARGET_EXTRA_AUX_SC_ERR	20
WARN	No precise intersection found with DEM. Degraded solution returned	Calculation performed	XP_CFI_TARGET_EXTRA_AUX_DEM_DEGRADED_SOLUTION_WARN	21
ERR	Void value detected in DEM	Calculation not performed	XP_CFI_TARGET_EXTRA_AUX_DEM_VOID_VALUE_DETECTED_ERR	22

7.102 xp_target_list_extra_aux

7.102.1 Overview

The **xp_target_list_extra_aux** CFI function provides the same results as xp_target_extra_aux function but for all the targets computed with xp_target_list_inter function.

This function has been optimized to improve the run-time performance of the target computation of all the targets and runs in multithreading (Remark: multithreading is not enabled on MacOS platforms, see section 6).

See note on mutithreading in section 7.98.1.1.

7.102.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     long choice, target_type, target_number;
     xp_target_id target_id = {NULL};
       xp_target_extra_aux_results_list list;
       long ierr[XP_NUM_ERR_TARGET_LIST_EXTRA_AUX], status;

     status = xp_target_list_extra_aux (&target_id, &choice,
                                          &target_type,
                                          &list, ierr);
}
```

The `XP_NUM_ERR_TARGET_LIST_EXTRA_AUX` constant is defined in the file *explorer_pointing.h.*

7.102.3 Input Parameters

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

Table 267: Input parameters of xp_target_list_extra_aux function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
choice	long *	-	Flag to select the extra results to be computed. Even though derivatives could be requested by user, they will not be calculated if the target was computed without derivatives (a warning is raised in this case).	-	Complete
target_type	long *		Flag to select the type of target		XP_USER_TARGET_TYPE XP_LOS_TARGET_TYPE XP_DEM_TARGET_TYPE

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

- Choice. (See Table 3).

7.102.4 Output Parameters

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

Table 268: Output parameters of xp_target_list_extra_aux

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
list	xp_target_extra_aux_results_list	-	List of extra results. Note: the memory allocated in this struct must be freed by the user with: **free**(list.extra_aux_results);	-	-
ierr	long	-	Error vector	-	-

The values corresponding to returned arrays are the same as in the case of xp_target_extra_aux (see section 7.101.4).

7.102.5 Warnings and Errors

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

Table 269: Error messages of xp_target_list_extra_aux function

Error type	Error message	Cause and impact	Error code	Error No
ERR	No target data available	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_AUX_NO_DATA_ERR	0
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_AUX_NO_SUCH_EARTH_TARGET_ERR	1
ERR	Could not compute the DEM target	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_AUX_EARTH_TARGET_COMPUT_ERR	2
ERR	Wrong target type	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_AUX_WRONG_TARGET_TYPE_ERR	3
ERR	Invalid time reference in target	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_	4

	data		A UX_INVALID_TIME_REF_ER R	
ERR	Error calling to XL_Car_Geo CFI function	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_ A UX_CAR_TO_GEO_ERR	5
ERR	Error getting tranformation matrix to Topocentric CS.	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_ AUX_ TOPO_ERR	6
ERR	Error getting direction angles	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_ A UX_DIR_POINTING_ERR	7
ERR	Error computing radius of curvature	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_ A UX_RADII_CURVATURE_C ALC_ERR	8
ERR	Error computing pointing after crossing the Earth atmosphere	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_ A UX_POINTING_AFTER_ATM _CALC_ERR	9
ERR	Error computing distance	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_ A UX_DISTANCE_CALC_ERR	10
ERR	Error computing velocity of the target's nadir	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_ A UX_TOP_VEL_CALC_ERR	11
WARN	Error computing MLST of TLST	Calculation performed	XP_CFI_TARGET_LIST_EXTRA_ A UX_MLST_OR_TLST_CALC_ ERR	12
WARN	Warning: Path from satellite to target occulted by the Earth	Calculation performed	XP_CFI_TARGET_LIST_EXTRA_ A UX_NEGATIVE_ALTITUDE_ WARN	13
WARN	Warning calling to XL_Car_Geo CFI function	Calculation performed	XP_CFI_TARGET_LIST_EXTRA_ A UX_AMBIGUOUS_SINGULA R_WARN	14
WARN	Warning: Derivatives cannot be calculated	Calculation performed	XP_CFI_TARGET_LIST_EXTRA_ A UX_DERIV_WARN	15
WARN	DEM files were not found. Intersection with the ellipsoid is returned	Calculation performed	XP_CFI_TARGET_LIST_EXTRA_ AUX_ELLIPSOID_WARN, WARN	16
WARN	Warnging in XP_DEM_inter	Calculation performed	XP_CFI_TARGET_LIST_EXTRA_ AUX_DEM_INTER_WARN	17
ERR	This function cannot be used with a target id computed with target S/C	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_ AUX_SC_ERR	18
ERR	Memory allocation error	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_ AUX_MEMORY_ERR	19
WARN	No precise intersection found with DEM. Degraded solution returned	Calculation performed	XP_CFI_TARGET_LIST_EXTRA_ AUX_DEM_DEGRADED_SOLUTI ON_WARN	20
ERR	Void value detected in DEM	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_ AUX_DEM_VOID_VALUE_DETE	21

			CTED_ERR	

7.103 xp_target_extra_ef_target

7.103.1 Overview

The **xp_target_extra_ef_target** CFI function computes the parameter for an Earth fixed target related to the target in input data structure.

Note on target_number with targets computed with xp_target_list_inter or xp_target_range:

the target_number to be used to get a specific LOS target is an incremental number. That is, if there are N user targets US1, US2, ... USN and a number of LOS targets for every user target NLOS1, NLOS2, ..., NLOSN, if we want to get LOS target with index 1 corresponding to user target US3, the target_number to be used is NLOS1+NLOS2+1.

The target_number can also be got with the array returned by xp_target_get_id_data.

7.103.2 Calling Interface

The calling interface of the **xp_target_extra_ef_target** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
     long target_type, target_number, choice;
     double freq;
     double ef_target_results_rate[XP_SIZE_EF_TARGET_RESULT],
     ef_target_results_rate_rate[XP_SIZE_EF_TARGET_RESULT];
     xp_target_id   target_id = {NULL};
     long ierr[XP_NUM_ERR_TARGET_EXTRA_EF_TARGET], status;

     status = xp_target_extra_ef_target(&target_id, &choice,
                              &target_type, &target_number, &freq,
                              ef_target_results_rate,
                              ef_target_results_rate_rate, ierr);
}
```

The `XP_SIZE_TARGET_RESULT_EF_TARGET` and `XP_NUM_ERR_TARGET_EXTRA_EF_TARGET` constants are defined in the file *explorer_pointing.h.*

7.103.3 Input Parameters

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

Table 270: Input parameters of xp_target_extra_ef_target function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
choice	long *	-	Flag to select the extra results to be computed. Even though derivatives could be requested by user, they will not be calculated if the target was computed without derivatives (a warning is raised in this case).	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
target_type	long *		Flag to select the type of target		XP_USER_TARGET_TYPE XP_LOS_TARGET_TYPE XP_DEM_TARGET_TYPE
target_number	long *	-	Target number		>= 0
freq	double *	-	Frequency of the signal	Hz	>=0

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

- Choice. (See Table 3).

7.103.4 Output Parameters

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

Table 271: Output parameters of xp_target_extra_ef_target

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
ef_target_results_rate	double [XP_SIZE_EF_TARGET_RESULT]	[0]	2-way Doppler shift of the signal (Earth Fixed CS)	Hz	-
		[1]	Earthfixed target to satellite range-rate. (Earth Fixed CS)	m/s	-
		[2]	Earthfixed target to satellite azimuth-rate. (Topocentric CS)	deg/s	-
		[3]	Earthfixed target to satellite elevation-rate. (Topocentric CS)	deg/s	-
		[4]	Satellite to earthfixed target azimuth-rate. (Topocentric CS)	deg/s	-
		[5]	Satellite to earthfixed target elevation-rate. (Topocentric CS)	deg/s	-
		[6]	Satellite to earthfixed target azimuth-rate. (Attitude Frame)	deg/s	-
		[7]	Satellite to earthfixed target elevation-rate. (Attitude Frame)	deg/s	-
ef_target_results_rate_rate	double [XP_SIZE_EF_TARGET_RESULT]	[0]	(dummy)	-	-
		[1]	Earthfixed target to satellite range-rate- rate. (Earth Fixed CS)	m/s2	-
		[2]	Earthfixed target to satellite azimuth-rate- rate. (Topocentric CS)	deg/s2	-
		[3]	Earthfixed target to satellite elevation-rate-rate. (Topocentric CS)	deg/s2	-
		[4]	Satellite to earthfixed target azimuth-rate- rate. (Topocentric CS)	deg/s2	-
		[5]	Satellite to earthfixed target elevation-rate-rate. (Topocentric CS)	deg/s2	-
		[6]	Satellite to earthfixed target azimuth-rate- rate-rate. (Attitude Frame)	deg/s2	-

		[7]	Satellite to earthfixed target elevation-rate-rate. (Attitude Frame)	deg/s2	-
ierr	long	-	Error vector	-	-

7.103.5 Warnings and Errors

Next table lists the possible error messages that can be returned by the **xp_target_extra_ef_target** CFI function after translating the returned error vector into the equivalent list of error messages by calling the function of the EO_POINTING software library **xp_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 by translating the error vector returned by the **xp_target_extra_ef_target** function by calling the function of the EO_POINTING software library **xp_get_code** (see [GEN_SUM])

Table 272: Error messages of xp_target_extra_ef_target function

Error type	Error message	Cause and impact	Error code	Error No
ERR	No target data available	No calculation performed	XP_CFI_TARGET_EXTRA_EF_TARGET_NO_DATA_ERR	0
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_EXTRA_EF_TARGET_NO_SUCH_USER_TARGET_ERR	1
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_EXTRA_EF_TARGET_NO_SUCH_LOS_TARGET_ERR	2
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_EXTRA_EF_TARGET_NO_SUCH_EARTH_TARGET_ERR	3
ERR	Could not compute the DEM target	No calculation performed	XP_CFI_TARGET_EXTRA_EF_TARGET_EARTH_TARGET_COMPUT_ERR	4
ERR	Wrong target type	No calculation performed	XP_CFI_TARGET_EXTRA_EF_TARGET_WRONG_TARGET_TYPE_ERR	5
ERR	Wrong input deriv flag	No calculation performed	XP_CFI_TARGET_EXTRA_EF_TARGET_DERIV_FLAG_ERR	6
ERR	Error getting target geodetic coordinates	No calculation performed	XP_CFI_TARGET_EXTRA_EF_TARGET_GEO_COORD_ERR	7
ERR	Invalid time reference in target data	No calculation performed	XP_CFI_TARGET_EXTRA_EF_TARGET_INVALID_TIME_	8

			REF_ERR	
ERR	Internal computation error	No calculation performed	XP_CFI_TARGET_EXTRA_EF_TARGET_RANGE_OR_POINTING_CALC_ERR	9
ERR	Wrong Atmospheric model in target data	No calculation performed	XP_CFI_TARGET_EXTRA_EF_TARGET_MODE_COMBINATION_SWITCHES_ERR	10
ERR	Error calling to XL_Car_Geo CFI function	No calculation performed	XP_CFI_TARGET_EXTRA_EF_TARGET_CAR_GEO_ERR	11
WARN	2nd. Derivatives are not available	Calculation performed	XP_CFI_TARGET_EXTRA_EF_TARGET_DER_2ND_NOT_AVAIL_WARN	12
WARN	Warning calling to XL_Car_Geo CFI function	Calculation performed	XP_CFI_TARGET_EXTRA_EF_TARGET_AMBIGUOUS_SINGULAR_WARN	13
WARN	1ST Derivative not computed for target. Satellite to target azimuth and elevation rates (SRAR CS) cannot be calculated	Calculation performed, except for azimuth and elevation rates in SRAR coordinate system.	XP_CFI_TARGET_EXTRA_EF_TARGET_DERIV_FLAG_WARN	14
WARN	DEM files were not found. Intersection with the ellipsoid is returned	Calculation performed	XP_CFI_TARGET_EXTRA_EF_ELLIPSOID_WARN	15
WARN	Warning in XP_DEM_inter	Calculation performed	XP_CFI_TARGET_EXTRA_EF_DEM_INTER_WARN	16
ERR	This function cannot be used with a target id computed with target S/C	No calculation performed	XP_CFI_TARGET_EXTRA_EF_TARGET_SC_ERR	17
WARN	No precise intersection found with DEM. Degraded solution returned	Calculation performed	XP_CFI_TARGET_EXTRA_EF_TARGET_DEM_DEGRADED_SOLUTION_WARN	18
ERR	Void value detected in DEM	No calculation performed	XP_CFI_TARGET_EXTRA_EF_TARGET_DEM_VOID_VALUE_DETECTED_ERR	19

7.104 xp_target_list_extra_ef_target

7.104.1 Overview

The **xp_target_list_extra_ef_target** CFI function provides the same results as xp_target_extra_ef_target function but for all the targets computed with xp_target_list_inter function.

This function has been optimized to improve the run-time performance of the target computation of all the targets and runs in multithreading (Remark: multithreading is not enabled on MacOS platforms, see section 6).

See note on mutithreading in section 7.98.1.1.

7.104.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     long choice, target_type, target_number;
     double freq;
     xp_target_id target_id = {NULL};
       xp_target_extra_ef_target_results_list list;
       long ierr[XP_NUM_ERR_TARGET_LIST_EXTRA_EF_TARGET], status;

     status = xp_target_list_extra_ef_target (&target_id, &choice,
                                              &target_type, &freq
                                              &list, ierr);
}
```

The `XP_NUM_ERR_TARGET_LIST_EXTRA_EF_TARGET` constant is defined in the file *explorer_pointing.h.*

7.104.3 Input Parameters

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

Table 273: Input parameters of xp_target_list_extra_ef_target function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
choice	long *	-	Flag to select the extra results to be computed. Even though derivatives could be requested by user, they will not be calculated if the target was computed without derivatives (a warning is raised in this case).	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
target_type	long *	-	Flag to select the type of target		XP_USER_TARGET_TYPE XP_LOS_TARGET_TYPE XP_DEM_TARGET_TYPE
freq	double *	-	Frequency of the signal	Hz	>=0

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

- Choice. (See Table 3).

7.104.4 Output Parameters

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

Table 274: Output parameters of xp_target_list_extra_ef_target

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
list	xp_target_extra_ef_target_results_list	-	List of extra results. Note: the memory allocated in this struct must be freed by the user with: **free**(list.extra_ef_target_results);	-	-
ierr	long	-	Error vector	-	-

The values corresponding to returned arrays are the same as in the case of xp_target_extra_ef_target (see section 7.103.4).

7.104.5 Warnings and Errors

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

Table 275: Error messages of xp_target_list_extra_ef_target function

Error type	Error message	Cause and impact	Error code	Error No
ERR	No target data available	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_E F_TARGET_NO_DATA_ERR	0
ERR	Wrong target type	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_E F_TARGET_WRONG_TARGET_TYPE_ERR	1
ERR	Wrong input deriv flag	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_E F_TARGET_DERIV_FLAG_ERR	2
ERR	Error getting target geodetic coordinates	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_E F_TARGET_GEO_COORD_ERR	3

ERR	Invalid time reference in target data	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_E F_TARGET_INVALID_TIME_REF_ERR	4
ERR	Internal computation error	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_E F_TARGET_RANGE_OR_POINTING_CALC_ERR	5
ERR	Wrong Atmospheric model in target data	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_E F_TARGET_MODE_COMBINATION_SWITCHES_ERR	6
ERR	Error calling to XL_Car_Geo CFI function	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_E F_TARGET_CAR_GEO_ERR	7
WARN	2nd. Derivatives are not available	Calculation performed	XP_CFI_TARGET_LIST_EXTRA_E F_TARGET_DER_2ND_NOT_AVAIL_WARN	8
WARN	Warning calling to XL_Car_Geo CFI function	Calculation performed	XP_CFI_TARGET_LIST_EXTRA_E F_TARGET_AMBIGUOUS_SINGULAR_WARN	9
WARN	1ST Derivative not computed for target. Satellite to target azimuth and elevation rates (SRAR CS) cannot be calculated	Calculation performed, except for azimuth and elevation rates in SRAR coordinate system.	XP_CFI_TARGET_LIST_EXTRA_E F_TARGET_DERIV_FLAG_WARN	10
WARN	DEM files were not found. Intersection with the ellipsoid is returned	Calculation performed	XP_CFI_TARGET_LIST_EXTRA_EF_ELLIPSOID_WARN	11
WARN	Warning in XP_DEM_inter	Calculation performed	XP_CFI_TARGET_LIST_EXTRA_EF_DEM_INTER_WARN	12
ERR	This function cannot be used with a target id computed with target S/C	No calculation performed	XP_CFI_TARGET_EXTRA_EF_TARGET_SC_ERR	13
ERR	Memory allocation error	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_AUX_MEMORY_ERR	14
WARN	No precise intersection found with DEM. Degraded solution returned	Calculation performed	XP_CFI_TARGET_LIST_EXTRA_EF_TARGET_DEM_DEGRADED_SOLUTION_WARN	15
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_EF_TARGET_NO_SUCH_EARTH_TARGET_ERR	16
ERR	Could not compute the DEM target	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_EF_TARGET_EARTH_TARGET_COMPUT_ERR	17
ERR	Void value detected in DEM	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_EF_TARGET_DEM_VOID_VALUE_DETECTED_ERR	18

7.105 xp_target_extra_target_to_sun

7.105.1 Overview

The **xp_target_extra_target_to_sun** CFI function computes extra parameters related to the pointing from the target in input data structure to the sun.

Notes:

1) On target_number with targets computed with xp_target_list_inter or xp_target_range:

the target_number to be used to get a specific LOS target is an incremental number. That is, if there are N user targets US1, US2, ... USN and a number of LOS targets for every user target NLOS1, NLOS2, ..., NLOSN, if we want to get LOS target with index 1 corresponding to user target US3, the target_number to be used is NLOS1+NLOS2+1.

The target_number can also be got with the array returned by xp_target_get_id_data.

2) A correction can be applied in order to compensate the travel time of light. This correction is not applied with default model. To activate this correction, the Sun model in xl_model_id must be initialized with the enum XL_MODEL_SUN_TRAVEL_TIME using the function xl_model_init (see [LIB_SUM]).

7.105.2 Calling Interface

The calling interface of the **xp_target_extra_target_to_sun** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
     long target_type, target_number, choice, iray;
     double freq;
     double sun_results[XP_SIZE_SUN_RESULT],
             sun_results_rate[XP_SIZE_SUN_RESULT],
             sun_results_rate_rate[XP_SIZE_SUN_RESULT];
     xp_target_id   target_id = {NULL};
     long ierr[XP_NUM_ERR_TARGET_EXTRA_TARGET_TO_SUN], status;

     status = xp_target_extra_target_to_sun
                               (&target_id, &choice, &target_type,
                                &target_number, &iray, &freq,
                                sun_results, sun_results_rate,
                                sun_results_rate_rate, ierr);
}
```

The XP_SIZE_TARGET_RESULT_TARGET_TO_SUN and XP_NUM_ERR_TARGET_EXTRA_TARGET_TO_SUN constants are defined in the file *explorer_pointing.h.*

7.105.3 Input Parameters

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

Table 276: Input parameters of xp_target_extra_target_to_sun function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
choice	long *	-	Flag to select the extra results to be computed. Even though derivatives could be requested by user, they will not be calculated if the target was computed without derivatives (a warning is raised in this case).	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
target_type	long *		Flag to select the type of target		XP_USER_TARGET_TYPE XP_LOS_TARGET_TYPE XP_DEM_TARGET_TYPE
target_number	long *	-	Target number		>= 0
iray	long *	-	Not used. The atmosphere refraction model can be defined via atmos_id initialization.	-	-
freq	double *	-	Frequency of the signal	Hz	>=0

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

- Choice. (See Table 3).

7.105.4 Output Parameters

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

Table 277: Output parameters of xp_target_extra_target_to_sun

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
sun_results	double [XP_SIZE_SUN_RESULT]	[0]	Target to Sun (centre) azimuth. (Topocentric CS)	deg	>= 0 < 360
		[1]	Target to Sun (centre) elevation. (Topocentric CS)	deg	>= -90 <= +90
		[2]	Tangent altitude over the Earth in the target to Sun (centre) look direction. (Earth fixed CS)	m	-
		[3]	Target to Sun visibility flag: - 1: Sun eclipsed by the Earth. +1: Sun in sight.	-	+1, -1
		[4:XP_SIZE_SUN_RESULT]	(dummy)	-	-
sun_results_rate	double [XP_SIZE_SUN_RESULT]	[0]	Target to Sun (centre) azimuth-rate. (Topocentric CS)	deg/s	-
		[1]	Target to Sun (centre) elevation-rate. (Topocentric CS)	deg/s	-
		[2:XP_SIZE_SUN_RESULT]	(dummy)	-	-
sun_results_rate_rate	double [XP_SIZE_SUN_RESULT]	[0]	Target to Sun (centre) azimuth-rate. (Topocentric CS)	deg/s2	-
		[1]	Target to Sun (centre) elevation-rate. (Topocentric CS)	deg/s2	-
		[2:XP_SIZE_SUN_RESULT]	(dummy)	-	-
ierr	long	-	Error vector	-	-

7.105.5 Warnings and Errors

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

Table 278: Error messages of xp_target_extra_target_to_sun function

Error type	Error message	Cause and impact	Error code	Error No
ERR	No target data available	No calculation performed	XP_CFI_TARGET_TO_SUN_NO_DATA_ERR	0
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_TO_SUN_NO_SUCH_USER_TARGET_ERR	1
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_TO_SUN_NO_SUCH_LOS_TARGET_E RR	2
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_TO_SUN_NO_SUCH_EARTH_TARGET_ERR	3
ERR	Could not compute the DEM target	No calculation performed	XP_CFI_TARGET_TO_SUN_EARTH_TARGET_COMPUT_ERR	4
ERR	Wrong target type	No calculation performed	XP_CFI_TARGET_TO_SUN_WRONG_TARGET_TYPE_ER R	5
ERR	Wrong input deriv flag	No calculation performed	XP_CFI_TARGET_TO_SUN_DERIV_FLAG_ERR	6
ERR	Error getting Sun position	No calculation performed	XP_CFI_TARGET_TO_SUN_SUN_POS_ERR	7
ERR	Invalid time reference in target data.	No calculation performed	XP_CFI_TARGET_TO_SUN_INVALID_TIME_REF_ERR	8
ERR	Error changing from TOD to EF.	No calculation performed	XP_CFI_TARGET_TO_SUN_TOD_TO_EF_ERR	9
ERR	Error getting direction vector from target to Sun.	No calculation performed	XP_CFI_TARGET_TO_SUN_DIR_VECTOR_ERR	10
ERR	Error getting geodetic	No calculation performed	XP_CFI_TARGET_TO_SUN_	11

	coordinates of the target		CAR_GEO_ERR	
ERR	Internal Computation Error. Target not Found.	No calculation performed	XP_CFI_TARGET_TO_SUN_ TARGET_NOT_FOUND_ERR	12
ERR	Wrong Atmospheric model in target data.	No calculation performed	XP_CFI_TARGET_TO_SUN_ MODE_COMBINATION_SWI TCHES_ERR	13
ERR	Error getting tranformation matrix to Topocentric CS.	No calculation performed	XP_CFI_TARGET_TO_SUN_ TOPO_ERR	14
ERR	Error getting Azimut/Elevation	No calculation performed	XP_CFI_TARGET_TO_SUN_ DIR_POINTING_ERR	15
WARN	Input Derivative flag level is too high. Derivative flag set to the value used in the main target function	Calculation performed	XP_CFI_TARGET_TO_SUN_ DERIV_FLAG_WARN	16
WARN	Precission not reached while calculating Sun pointing parameters	Calculation performed	XP_CFI_TARGET_TO_SUN_ MAX_ALLOWED_ITERATIO NS_WARN	17
WARN	DEM files were not found. Intersection with the ellipsoid is returned"	Calculation performed	XP_CFI_TARGET_TO_SUN_ELLI PSOID_WARN	18
WARN	Warning in XP_DEM_inter	Calculation performed	XP_CFI_TARGET_TO_SUN_DE M_INTER_WARN	19
WARN	The ray tracing flag (iray) is ignored	Calculation performed. A message informs the user that this paremeter is not used. If the variable iray is equal to XP_NO_REF_INIT (=0), the warning is avoided	XP_CFI_TARGET_TO_SUN_IRA Y_ID_WARN	20
WARN	No precise intersection found with DEM. Degraded solution returned	Calculation performed	XP_CFI_TARGET_TO_SUN_DE M_DEGRADED_SOLUTION_WA RN	21
ERR	Void value detected in DEM	No calculation performed	XP_CFI_TARGET_TO_SUN_DE M_VOID_VALUE_DETECTED_E RR	22

7.106 xp_target_list_extra_target_to_sun

7.106.1 Overview

The **xp_target_list_extra_target_to_sun** CFI function provides the same results as xp_target_extra_target_to_sun function but for all the targets computed with xp_target_list_inter function.

This function has been optimized to improve the run-time performance of the target computation of all the targets and runs in multithreading (Remark: multithreading is not enabled on MacOS platforms, see section 6).

See note on mutithreading in section 7.98.1.1.

Note: a correction can be applied in order to compensate the travel time of Sun light travel time. This correction is not applied with default model. To activate this correction, the Sun model in xl_model_id must be initialized with the enum XL_MODEL_SUN_TRAVEL_TIME using the function xl_model_init (see [LIB_SUM]).

7.106.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     long choice, target_type, target_number;
     double freq;
     long iray;
     xp_target_id target_id = {NULL};
       xp_target_extra_sun_target_results_list list;
       long ierr[XP_NUM_ERR_TARGET_LIST_EXTRA_TARGET_TO_SUN], status;

     status = xp_target_list_extra_target_to_sun (&target_id, &choice,
                                                  &target_type, &iray, &freq
                                                  &list, ierr);
}
```

The XP_NUM_ERR_TARGET_LIST_EXTRA_TARGET_TO_SUN constant is defined in the file *explorer_pointing.h.*

7.106.3 Input Parameters

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

Table 279: Input parameters of xp_target_list_extra_target_to_sun function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
choice	long *	-	Flag to select the extra results to be computed. Even though derivatives could be requested by user, they will not be calculated if the target was computed without derivatives (a warning is raised in this case).	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
target_type	long *	-	Flag to select the type of target		XP_USER_TARGET_TYPE XP_LOS_TARGET_TYPE XP_DEM_TARGET_TYPE
iray	long *	-	Not used. The atmosphere refraction model can be defined via atmos_id initialization.	-	-
freq	double *	-	Frequency of the signal	Hz	>=0

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

- Choice. (See Table 3).

7.106.4 Output Parameters

The output parameters of the **xp_target_list_extra_target** _to_sunCFI function are:

Table 280: Output parameters of xp_target_list_extra_target_to_sun

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
list	xp_target_extra_sun target_results_list	-	List of extra results. Note: the memory allocated in this struct must be freed by the user with: **free**(list.extra_sun_target_results);	-	-
ierr	long	-	Error vector	-	-

The values corresponding to returned arrays are the same as in the case of xp_target_extra_target_to_sun (see section 7.105.4).

7.106.5 Warnings and Errors

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

Table 281: Error messages of xp_target_list_extra_target_to_sun function

Error type	Error message	Cause and impact	Error code	Error No
ERR	No target data available	No calculation performed	XP_CFI_TARGET_LIST_TO_SUN_ NO_DATA_ERR	0
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_LIST_TO_SUN_ NO_SUCH_EARTH_TARGET_ERR	1
ERR	Could not compute the DEM target	No calculation performed	XP_CFI_TARGET_LIST_TO_SUN_ EARTH_TARGET_COMPUT_ERR	2
ERR	Wrong target type	No calculation performed	XP_CFI_TARGET_LIST_TO_SUN_ WRONG_TARGET_TYPE_ERR	3

ERR	Wrong input deriv flag	No calculation performed	XP_CFI_TARGET_LIST_TO_SUN_ DERIV_FLAG_ERR	4
ERR	Error getting Sun position	No calculation performed	XP_CFI_TARGET_LIST_TO_SUN_S UN_POS_ERR	5
ERR	Invalid time reference in target data.	No calculation performed	XP_CFI_TARGET_LIST_TO_SUN_I NVALID_TIME_REF_ERR	6
ERR	Error changing from TOD to EF.	No calculation performed	XP_CFI_TARGET_LIST_TO_SUN_ TOD_TO_EF_ERR	7
ERR	Error getting direction vector from target to Sun.	No calculation performed	XP_CFI_TARGET_LIST_TO_SUN_ DIR_VECTOR_ERR	8
ERR	Error getting geodetic coordinates of the target	No calculation performed	XP_CFI_TARGET_LIST_TO_SUN_ CAR_GEO_ERR	9
ERR	Internal Computation Error. Target not Found.	No calculation performed	XP_CFI_TARGET_LIST_TO_SUN_ TARGET_NOT_FOUND_ERR	10
ERR	Wrong Atmospheric model in target data.	No calculation performed	XP_CFI_TARGET_LIST_TO_SUN_ MODE_COMBINATION_SWITCHES_ERR	11
ERR	Error getting tranformation matrix to Topocentric CS.	No calculation performed	XP_CFI_TARGET_LIST_TO_SUN_ TOPO_ERR	12
ERR	Error getting Azimut/Elevation	No calculation performed	XP_CFI_TARGET_LIST_TO_SUN_ DIR_POINTING_ERR	13
WARN	Input Derivative flag level is too high. Derivative flag set to the value used in the main target function	Calculation performed	XP_CFI_TARGET_LIST_TO_SUN_ DERIV_FLAG_WARN	14
WARN	Precission not reached while calculating Sun pointing parameters	Calculation performed	XP_CFI_TARGET_LIST_TO_SUN_ MAX_ALLOWED_ITERATIONS_WARN	15
WARN	DEM files were not found. Intersection with the ellipsoid is returned"	Calculation performed	XP_CFI_TARGET_LIST_TO_SUN_ELLIPSOID_WARN	16
WARN	Warning in XP_DEM_inter	Calculation performed	XP_CFI_TARGET_LIST_TO_SUN_DEM_INTER_WARN	17
WARN	The ray tracing flag (iray) is ignored	Calculation performed. A message informs the user that this paremeter is not used. If the variable iray is equal to XP_NO_REF_INIT (=0), the warning is avoided	XP_CFI_TARGET_LIST_TO_SUN_IRAY_ID_WARN	18

ERR	Error allocating memory	No calculation performed	XP_CFI_TARGET_LIST_LIST_TO_SUN_MEMORY_ERR	19
WARN	No precise intersection found with DEM. Degraded solution returned	Calculation performed	XP_CFI_TARGET_LIST_TO_SUN_DEM_DEGRADED_SOLUTION_WARN	20
ERR	Void value detected in DEM	No calculation performed	XP_CFI_TARGET_LIST_TO_SUN_DEM_VOID_VALUE_DETECTED_ERR	21

7.107 xp_target_extra_target_to_moon

7.107.1 Overview

The **xp_target_extra_target_to_moon** CFI function computes extra parameters related to the pointing from the target in input data structure to the moon.

Note on target_number with targets computed with xp_target_list_inter or xp_target_range:

the target_number to be used to get a specific LOS target is an incremental number. That is, if there are N user targets US1, US2, ... USN and a number of LOS targets for every user target NLOS1, NLOS2, ..., NLOSN, if we want to get LOS target with index 1 corresponding to user target US3, the target_number to be used is NLOS1+NLOS2+1.

The target_number can also be got with the array returned by xp_target_get_id_data.

7.107.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     long target_type, target_number, choice, iray;
     double freq;
     double moon_results[XP_SIZE_moon_RESULT],
          moon_results_rate[XP_SIZE_MOON_RESULT],
          moon_results_rate_rate[XP_SIZE_MOON_RESULT];
     xp_target_id   target_id = {NULL};
     long ierr[XP_NUM_ERR_TARGET_EXTRA_TARGET_TO_MOON], status;

     status = xp_target_extra_target_to_moon
                              (&target_id, &choice, &target_type,
                               &target_number, &iray, &freq,
                               moon_results, moon_results_rate,
                               moon_results_rate_rate, ierr);
}
```

The `XP_SIZE_TARGET_RESULT_TARGET_TO_MOON` and `XP_NUM_ERR_TARGET_EXTRA_TARGET_TO_MOON` constants are defined in the file *explorer_pointing.h.*

7.107.3 Input Parameters

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

Table 282: Input parameters of xp_target_extra_target_to_moon function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
choice	long *	-	Flag to select the extra results to be computed. Even though derivatives could be requested by user, they will not be calculated if the target was computed without derivatives (a warning is raised in this case).	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
target_type	long *		Flag to select the type of target		XP_USER_TARGET_TYPE XP_LOS_TARGET_TYPE XP_DEM_TARGET_TYPE
target_number	long *	-	Target number		>= 0
iray	long *	-	Not used. The atmosphere refraction model can be defined via atmos_id initialization.	-	-
freq	double *	-	Frequency of the signal	Hz	>=0

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

- Choice. (See Table 3).

7.107.4 Output Parameters

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

Table 283: Output parameters of xp_target_extra_target_to_moon

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
moon_resul ts	double [XP_SIZE_M OON_RESUL T]	[0]	Target to Moon (centre) azimuth. (Topocentric CS)	deg	>= 0 < 360
		[1]	Target to Moon (centre) el. (Topocentric CS)	deg	>= -90 <= +90
		[2]	Tangent altitude over theEarth in the target to Moon (centre) look direction. (Earth fixed CS)	m	-
		[3]	Target to Moon visibility flag: - 1: Moon eclipsed by the Earth. +1: Moon in sight.	-	+1, -1
		[4:XP_SIZ E_MOON _RESULT]	(dummy)	-	-
moon_resul ts_rate	double [XP_SIZE_M OON_RESUL T]	[0]	Target to Moon (centre) azimuth-rate. (Topocentric CS)	deg/s	-
		[1]	Target to Moon (centre) eleva tion-rate. (Topocentric CS)	deg/s	-
		[2:XP_SIZ E_MOON _RESULT]	(dummy)	-	-
moon_resul ts_rate_rate	double [XP_SIZE_M OON_RESUL T]	[0]	Target to Moon (centre) azimuth-rate. (Topocentric CS)	deg/s2	-
		[1]	Target to Moon (centre) eleva tion-rate. (Topocentric CS)	deg/s2	-
		[2:XP_SIZ E_MOON _RESULT]	(dummy)	-	-
ierr	long	-	Error vector	-	-

7.107.5 Warnings and Errors

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

Table 284: Error messages of xp_target_extra_target_to_moon function

Error type	Error message	Cause and impact	Error code	Error No
ERR	No target data available	No calculation performed	XP_CFI_TARGET_TO_MOON_NO_DATA_ERR	0
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_TO_MOON_NO_SUCH_USER_TARGET_ERR	1
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_TO_MOON_NO_SUCH_LOS_TARGET_ERR	2
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_TO_MOON_NO_SUCH_EARTH_TARGET_ERR	3
ERR	Could not compute the DEM target	No calculation performed	XP_CFI_TARGET_TO_MOON_EARTH_TARGET_COMPUT_ERR	4
ERR	Wrong target type	No calculation performed	XP_CFI_TARGET_TO_MOON_WRONG_TARGET_TYPE_E RR	5
ERR	Wrong input deriv flag	No calculation performed	XP_CFI_TARGET_TO_MOON_DERIV_FLAG_ERR	6
ERR	Error getting Moon position	No calculation performed	XP_CFI_TARGET_TO_MOON_MOON_POS_ERR	7
ERR	Invalid time reference in target data.	No calculation performed	XP_CFI_TARGET_TO_MOON_INVALID_TIME_REF_ERR	8
ERR	Error changing from TOD to EF.	No calculation performed	XP_CFI_TARGET_TO_MOON_TOD_TO_EF_ERR	9
ERR	Error getting direction vector from target to Moon.	No calculation performed	XP_CFI_TARGET_TO_MOON_DIR_VECTOR_ERR	10
ERR	Error getting geodetic	No calculation performed	XP_CFI_TARGET_TO_MOON	11

	coordinates of the target		_CAR_GEO_ERR	
ERR	Internal Computation Error. Target not Found.	No calculation performed	XP_CFI_TARGET_TO_MOON _TARGET_NOT_FOUND_ER R	12
ERR	Wrong Atmospheric model in target data.	No calculation performed	XP_CFI_TARGET_TO_MOON _MODE_COMBINATION_SW ITCHES_ERR	13
ERR	Error getting tranformation matrix to Topocentric CS.	No calculation performed	XP_CFI_TARGET_TO_MOON _TOPO_ERR	14
ERR	Error getting Azimut/Elevation	No calculation performed	XP_CFI_TARGET_TO_MOON _DIR_POINTING_ERR	15
WARN	Input Derivative flag level is too high. Derivative flag set to the value used in the main target function	Calculation performed	XP_CFI_TARGET_TO_MOON _DERIV_FLAG_WARN	16
WARN	Precission not reached while calculating Moon pointing parameters	Calculation performed	XP_CFI_TARGET_TO_MOON _MAX_ALLOWED_ITERATI ONS_WARN	17
WARN	DEM files were not found. Intersection with the ellipsoid is returned	Calculation performed	XP_CFI_TARGET_TO_MOON_E LLIPSOID_WARN	18
WARN	Warning in XP_DEM_inter	Calculation performed	XP_CFI_TARGET_TO_MOON_D EM_INTER_WARN	19
WARN	The ray tracing flag (iray) is ignored	Calculation performed A message informs the user that this paremeter is not used. If the variable iray is equal to XP_NO_REF_INIT (=0), the warning is avoided	XP_CFI_TARGET_TO_MOON_IR AY_ID_WARN	20
WARN	No precise intersection found with DEM. Degraded solution returned	Calculation performed	XP_CFI_TARGET_TO_MOON_D EM_DEGRADED_SOLUTION_W ARN	21
ERR	Void value detected in DEM	No calculation performed	XP_CFI_TARGET_TO_MOON_D EM_VOID_VALUE_DETECTED_ ERR	22

7.108 xp_target_list_extra_target_to_moon

7.108.1 Overview

The **xp_target_list_extra_target_to_moon** CFI function provides the same results as xp_target_extra_target_to_moon function but for all the targets computed with xp_target_list_inter function.

This function has been optimized to improve the run-time performance of the target computation of all the targets and runs in multithreading (Remark: multithreading is not enabled on MacOS platforms, see section 6).

See note on mutithreading in section 7.98.1.1.

7.108.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     long choice, target_type, target_number;
     double freq;
     long iray;
     xp_target_id target_id = {NULL};
       xp_target_extra_moon_target_results_list list;
       long ierr[XP_NUM_ERR_TARGET_LIST_EXTRA_TARGET_TO_MOON], status;

     status = xp_target_list_extra_target_to_moon (&target_id, &choice,
                                                   &target_type, &iray, &freq
                                                   &list, ierr);
}
```

The `XP_NUM_ERR_TARGET_LIST_EXTRA_TARGET_TO_MOON` constant is defined in the file *explorer_pointing.h.*

7.108.3 Input Parameters

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

Table 285: Input parameters of xp_target_list_extra_target_to_moon function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
choice	long *	-	Flag to select the extra results to be computed. Even though derivatives could be requested by user, they will not be calculated if the target was computed without derivatives (a warning is raised in this case).	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
target_type	long *	-	Flag to select the type of target		XP_USER_TARGET_TYPE XP_LOS_TARGET_TYPE XP_DEM_TARGET_TYPE
iray	long *	-	Not used. The atmosphere refraction model can be defined via atmos_id initialization.	-	-
freq	double *	-	Frequency of the signal	Hz	>=0

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

- Choice. (See Table 3).

7.108.4 Output Parameters

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

Table 286: Output parameters of xp_target_list_extra_target_to_moon

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
list	xp_target_extra_moon_target_results_list	-	List of extra results. Note: the memory allocated in this struct must be freed by the user with: **free**(list.extra_moon_target_results);	-	-
ierr	long	-	Error vector	-	-

The values corresponding to returned arrays are the same as in the case of xp_target_extra_target_to_moon (see section 7.107.4).

7.108.5 Warnings and Errors

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

Table 287: Error messages of xp_target_list_extra_target_to_moon function

Error type	Error message	Cause and impact	Error code	Error No
ERR	No target data available	No calculation performed	XP_CFI_TARGET_LIST_TO_MOON _NO_DATA_ERR	0
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_LIST_TO_MOON _NO_SUCH_EARTH_TARGET_ERR	1
ERR	Could not compute the DEM target	No calculation performed	XP_CFI_TARGET_LIST_TO_MOON _EARTH_TARGET_COMPUT_ERR	2
ERR	Wrong target type	No calculation performed	XP_CFI_TARGET_LIST_TO_MOON _WRONG_TARGET_TYPE_E	3

			RR	
ERR	Wrong input deriv flag	No calculation performed	XP_CFI_TARGET_LIST_TO_MOON _DERIV_FLAG_ERR	4
ERR	Error getting Moon position	No calculation performed	XP_CFI_TARGET_LIST_TO_MOON _MOON_POS_ERR	5
ERR	Invalid time reference in target data.	No calculation performed	XP_CFI_TARGET_LIST_TO_MOON _INVALID_TIME_REF_ERR	6
ERR	Error changing from TOD to EF.	No calculation performed	XP_CFI_TARGET_LIST_TO_MOON _TOD_TO_EF_ERR	7
ERR	Error getting direction vector from target to Moon.	No calculation performed	XP_CFI_TARGET_LIST_TO_MOON _DIR_VECTOR_ERR	8
ERR	Error getting geodetic coordinates of the target	No calculation performed	XP_CFI_TARGET_LIST_TO_MOON _CAR_GEO_ERR	9
ERR	Internal Computation Error. Target not Found.	No calculation performed	XP_CFI_TARGET_LIST_TO_MOON _TARGET_NOT_FOUND_ERR	10
ERR	Wrong Atmospheric model in target data.	No calculation performed	XP_CFI_TARGET_LIST_TO_MOON _MODE_COMBINATION_SWITCHES_ERR	11
ERR	Error getting tranformation matrix to Topocentric CS.	No calculation performed	XP_CFI_TARGET_LIST_TO_MOON _TOPO_ERR	12
ERR	Error getting Azimut/Elevation	No calculation performed	XP_CFI_TARGET_LIST_TO_MOON _DIR_POINTING_ERR	13
WARN	Input Derivative flag level is too high. Derivative flag set to the value used in the main target function	Calculation performed	XP_CFI_TARGET_LIST_TO_MOON _DERIV_FLAG_WARN	14
WARN	Precission not reached while calculating Moon pointing parameters	Calculation performed	XP_CFI_TARGET_LIST_TO_MOON _MAX_ALLOWED_ITERATIONS_WARN	15
WARN	DEM files were not found. Intersection with the ellipsoid is returned	Calculation performed	XP_CFI_TARGET_LIST_TO_MOON_ELLIPSOID_WARN	16
WARN	Warning in XP_DEM_inter	Calculation performed	XP_CFI_TARGET_LIST_TO_MOON_DEM_INTER_WARN	17
WARN	The ray tracing flag (iray) is ignored	Calculation performed A message informs the user that this paremeter is not used. If the variable iray is equal to	XP_CFI_TARGET_LIST_TO_MOON_IRAY_ID_WARN	18

		XP_NO_REF_INIT (=0), the warning is avoided		
ERR	Error allocating memory	No calculation performed	XP_CFI_TARGET_LIST_TO_MOON_MEMORY_ERR	19
WARN	No precise intersection found with DEM. Degraded solution returned	Calculation performed	XP_CFI_TARGET_LIST_TO_MOON_DEM_DEGRADED_SOLUTION_WARN	20
ERR	Void value detected in DEM	No calculation performed	XP_CFI_TARGET_LIST_TO_MOON_DEM_VOID_VALUE_DETECTED_ERR	21

7.109 xp_target_extra_specular_reflection

7.109.1 Overview

The **xp_target_extra_specular_reflection** CFI function calculates the direction of the specular reflection associated to a given target.

Note on target_number with targets computed with xp_target_list_inter or xp_target_range:

the target_number to be used to get a specific LOS target is an incremental number. That is, if there are N user targets US1, US2, ... USN and a number of LOS targets for every user target NLOS1, NLOS2, ..., NLOSN, if we want to get LOS target with index 1 corresponding to user target US3, the target_number to be used is NLOS1+NLOS2+1.

The target_number can also be got with the array returned by xp_target_get_id_data.

7.109.2 Calling Interface

The calling interface of the **xp_target_extra_specular_reflection** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
    long target_type, target_number, choice, iray;
    double freq;
    double spec_reflec_results[XP_SIZE_TARGET_RESULT_SPEC_REFL],
     spec_reflec_results_rate[XP_SIZE_TARGET_RESULT_SPEC_REFL],
     spec_reflec_results_rate_rate[XP_SIZE_TARGET_RESULT_SPEC_REFL];
    xp_target_id   target_id = {NULL};
    long ierr[XP_NUM_ERR_TARGET_EXTRA_SPEC_REFL], status;

    status = xp_target_extra_specular_reflection
                            (&target_id, &choice, &target_type,
                             &target_number,
                              &deflection_north, &deflection_east,
                              spec_reflec_results,
                              spec_reflec_results_rate,
                             spec_reflec_results_rate_rate, ierr);
}
```

The `XP_SIZE_TARGET_RESULT_SPEC_REFL` and `XP_NUM_ERR_TARGET_EXTRA_SPEC_REFL` constants are defined in the file *explorer_pointing.h*.

7.109.3 Input Parameters

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

Table 288: Input parameters of xp_target_extra_specular_reflection function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
choice	long *	-	Flag to select the extra results to be computed. Even though derivatives could be requested by user, they will not be calculated if the target was computed without derivatives (a warning is raised in this case).	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
target_type	long *		Flag to select the type of target		XP_USER_TARGET_TYPE XP_LOS_TARGET_TYPE XP_DEM_TARGET_TYPE
target_number	long *	-	Target number		>= 0
deflection_north	double *	-	North-South component of the vertical deflection	deg	>= -90 < = 90
deflection_east	double *	-	East-West component of the vertical deflection	deg	>= -90 < = 90

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

- Choice. (See Table 3).

7.109.4 Output Parameters

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

Table 289: Output parameters of xp_target_extra_specular_reflection

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
spec_reflec _results	double [XP_SIZE_TA RGET_ RESULT_SP EC_REFL]	[0]	X coordinate of reflected pointing direction. (EF CS)	m	-
		[1]	Y coordinate of reflected pointing direction. (EF CS)	m	-
		[2]	Z coordinate of reflected pointing direction. (EF CS)	m	-
		[3]	Azimuth of the reflected pointing direction. (Topocentric CS)	deg	[0, 360)
		[4]	Elevation of the reflected pointing direction. (Topocentric CS)	deg	[-90, 90]
		[5]	Right ascension at which the reflected pointing direction points at target point. (True of Date CS)	deg	[0, 360)
		[6]	Declination at which the reflected pointing direction points at target point. (True of Date CS)	deg	[-90, 90]
spec_reflec _results_rat e	double [XP_SIZE_TA RGET_ RESULT_SP EC_REFL]	[0]	X velocity of reflected pointing direction. (EF CS)	m/s	-
		[1]	Y velocity of reflected pointing direction. (EF CS)	m/s	-
		[2]	Z velocity of reflected pointing direction. (EF CS)	m/s	-
		[3]	Azimuth rate of the reflected pointing direction (Topocentric CS)	deg/s	-
		[4]	Elevation rate of the reflected pointing direction (Topocentric CS)	deg/s	-
		[5]	Right ascension rate at which the reflected pointing direction points at target point. (True of Date CS)	deg/s	-
		[6]	Declination rate at which the reflected pointing direction points at target point. (True of Date CS)	deg/s	-

spec_reflec_results_rate_rate	double [XP_SIZE_TARGET_RESULT_SPEC_REFL]	[0]	X acceleration of reflected point ing direction. (EF CS)	m/s2	-
		[1]	Y acceleration of reflected point ing direction. (EF CS)	m/s2	-
		[2]	Z acceleration of reflected point ing direction. (EF CS)	m/s2	-
		[3]	Azimuth rate rate of the reflected pointing direction (Topocentric CS)	deg/s2	-
		[4]	Elevation rate rate of the reflected pointing direction (Topocentric CS)	deg/s2	-
		[5]	Right ascension rate rate at which the reflected pointing direction points at target point. (True of Date CS)	deg/s2	-
		[6]	Declination rate rate at which the reflected pointing direction points at target point. (True of Date CS)	deg/s2	-
ierr	long	-	Error vector	-	-

7.109.5 Warnings and Errors

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

Table 290: Error messages of xp_target_extra_specular_reflection function

Error type	Error message	Cause and impact	Error code	Error No
ERR	No target data available	No calculation performed	XP_CFI_TARGET_EXTRA_SPECULAR_REFLECT_NO_DATA_ERR	0
ERR	Input deflection angle is out of range	No calculation performed	XP_CFI_TARGET_EXTRA_SPECULAR_REFLECT_WRON	1

			G_DEF_ANGLE_ERR	
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_EXTRA_SPECULAR_REFLECT_NO_SUCH_USER_TARGET_ERR	2
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_EXTRA_SPECULAR_REFLECT_NO_SUCH_LOS_TARGET_ERR	3
ERR	Could not compute the DEM target	No calculation performed	XP_CFI_TARGET_EXTRA_SPECULAR_REFLECT_EARTH_TARGET_COMPUT_ERR	4
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_EXTRA_SPECULAR_REFLECT_NO_SUCH_EARTH_TARGET_ERR	5
ERR	Wrong target type	No calculation performed	XP_CFI_TARGET_EXTRA_SPECULAR_REFLECT_WRONG_TARGET_TYPE_ERR	6
ERR	Error getting geodetic coordinates of the target	No calculation performed	XP_CFI_TARGET_EXTRA_SPECULAR_REFLECT_CAR_TO_GEO_ERR	7
ERR	Error getting tranformation matrix to Topocentric CS	No calculation performed	XP_CFI_TARGET_EXTRA_SPECULAR_REFLECT_TOPO_CS_ERR	8
ERR	Error getting direction angles	No calculation performed	XP_CFI_TARGET_EXTRA_SPECULAR_REFLECT_DIR_POINTING_ERR	9
ERR	Error while changing coordinate system	No calculation performed	XP_CFI_TARGET_EXTRA_SPECULAR_REFLECT_CHANGE_CS_ERR	10
WARN	Input Derivative flag level is too high. Derivative flag set to the value used in the main target function	Calculation performed	XP_CFI_TARGET_EXTRA_SPECULAR_REFLECT_DERIV_WARN	11
WARN	DEM files were not found. Intersection with the ellipsoid is returned	Calculation performed	XP_CFI_TARGET_EXTRA_SPECULAR_REFLECT_WARN	12
WARN	Warning in XP_DEM_inter	Calculation performed	XP_CFI_TARGET_EXTRA_SPECULAR_REFLECT_DEM_INTER_WARN	13
ERR	This function cannot be used with a target id computed with target S/C	No calculation performed	XP_CFI_TARGET_EXTRA_SPECULAR_REFLECT_SC_ERR	14
WARN	No precise intersection found with DEM. Degraded solution returned	Calculation performed	XP_CFI_TARGET_EXTRA_SPECULAR_REFLECT_DEM_DEGRADED_SOLUTION_WARN	15
ERR	Void value detected in DEM	No calculation performed	XP_CFI_TARGET_EXTRA_SPECULAR_REFLECT_DEM_VOID_V	16

			ALUE_DETECTED_ERR	

7.110 xp_target_list_extra_specular_reflection

7.110.1 Overview

The **xp_target_list_extra_specular_reflection** CFI function provides the same results as xp_target_extra_specular_reflection function but for all the targets computed with xp_target_list_inter function.

This function has been optimized to improve the run-time performance of the target computation of all the targets and runs in multithreading (Remark: multithreading is not enabled on MacOS platforms, see section 6).

See note on mutithreading in section 7.98.1.1.

7.110.2 Calling Interface

The calling interface of the **xp_target_list_extra_specular_reflection** CFI function is the following (input parameters are <u>underlined</u>):

```
#include <explorer_pointing.h>
{
     long choice, target_type, target_number;
     double deflection_north, deflection_east;
     long iray;
     xp_target_id target_id = {NULL};
       xp_target_extra_spec_reflec_target_results_list list;
       long ierr[XP_NUM_ERR_TARGET_LIST_EXTRA_SPEC_REFL], status;

     status = xp_target_list_extra_specular_reflection (&target_id,
                                              &choice, &target_type,
                                              &deflection_north,
                                              &deflection_east,
                                              &list, ierr);
}
```

The `XP_NUM_ERR_TARGET_LIST_EXTRA_SPEC_REFL` constant is defined in the file *explorer_pointing.h.*

7.110.3 Input Parameters

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

Table 291: Input parameters of xp_target_list_extra_specular_reflection function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
target_id	xp_target_id*	-	Structure that contains the Target results	-	-
choice	long *	-	Flag to select the extra results to be computed. Even though derivatives could be requested by user, they will not be calculated if the target was computed without derivatives (a warning is raised in this case).	-	Allowed values: (0) XP_NO_DER (1) XP_DER_1ST (2) XP_DER_2ND
target_type	long *	-	Flag to select the type of target		XP_USER_TARGET_TYPE XP_LOS_TARGET_TYPE XP_DEM_TARGET_TYPE
deflection_north	double *	-	North-South component of the vertical deflection	deg	>= -90 < = 90
deflection_east	double *	-	East-West component of the vertical deflection	deg	>= -90 < = 90

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

- Choice. (See Table 3).

7.110.4 Output Parameters

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

Table 292: Output parameters of xp_target_list_extra_specular_reflection

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
list	xp_target_extra_spec_reflec_target_results_list	-	List of extra results. Note: the memory allocated in this struct must be freed by the user with: **free**(list.extra_spec_reflect_target_results);	-	-
ierr	long	-	Error vector	-	-

The values corresponding to returned arrays are the same as in the case of xp_target_extra_specular_reflection (see section 7.109.4).

7.110.5 Warnings and Errors

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

Table 293: Error messages of xp_target_list_extra_specular_reflection function

Error type	Error message	Cause and impact	Error code	Error No
ERR	No target data available	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_S PECULAR_REFLECT_NO_DATA_ERR	0
ERR	Input deflection angle is out of range	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_S PECULAR_REFLECT_WRONG_DEF_ANGLE_ERR	1
ERR	Could not compute the DEM target	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_S PECULAR_REFLECT_EARTH_TARGET_COMPUT_ERR	2
ERR	Wrong target type	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_S PECULAR_REFLECT_WRON	3

			G_TARGET_TYPE_ERR	
ERR	The target does not exist	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_S PECULAR_REFLECT_NO_SUCH_EARTH_TARGET_ERR	4
ERR	Error getting geodetic coordinates of the target	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_S PECULAR_REFLECT_CAR_TO_GEO_ERR	5
ERR	Error getting tranformation matrix to Topocentric CS	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_S PECULAR_REFLECT_TOPO_CS_ERR	6
ERR	Error getting direction angles	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_S PECULAR_REFLECT_DIR_POINTING_ERR	7
ERR	Error while changing coordinate system	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_S PECULAR_REFLECT_CHANGE_CS_ERR	8
WARN	Input Derivative flag level is too high. Derivative flag set to the value used in the main target function	Calculation performed	XP_CFI_TARGET_LIST_EXTRA_S PECULAR_REFLECT_DERIV_WARN	9
WARN	DEM files were not found. Intersection with the ellipsoid is returned	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_SPECULAR_REFLECT_ELLIPSOID_WARN	10
WARN	Warning in XP_DEM_inter	Calculation performed	XP_CFI_TARGET_LIST_EXTRA_SPECULAR_REFLECT_DEM_INTER_WARN	11
ERR	This function cannot be used with a target id computed with target S/C	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_SPECULAR_REFLECT_SC_ERR	12
ERR	Error allocating memory	No calculation performed	XP_CFI_TARGET_LIST_EXTRA_SPECULAR_REFLECT_MEMORY_ERR	13
WARN	No precise intersection found with DEM. Degraded solution returned	Calculation performed	XP_CFI_TARGET_LIST_EXTRA_SPECULAR_REFLECT_DEM_DEGRADED_SOLUTION_WARN	14

7.111 xp_target_close

7.111.1 Overview

The **xp_target_close** CFI function cleans up any memory allocation performed by the Target functions.

7.111.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     xp_target_id target_id = {NULL};
     long ierr[XP_NUM_ERR_TARGET_CLOSE], status;

     status =     xp_target_close(&target_id, ierr);
}
```

The XP_NUM_ERR_TARGET_CLOSE constant is defined in the file *explorer_pointing.h.*

7.111.3 Input Parameters

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

Table 294: Input parameters of xp_target_close function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
target_id	xp_target_id*	-	Structure that contains the Target results	-	-

7.111.4 Output Parameters

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

Table 295: Output parameters of xp_target_close

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

7.111.5 Warnings and Errors

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

Table 296: Error messages of xp_target_close function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Target ID is not initialized or it is being used	No calculation performed	XP_CFI_TARGET_CLOSE_WRONG_ID_ERR	11

7.112 xp_target_get_id_data

7.112.1 Overview

The **xp_target_get_id_data** CFI function returns the target initialization data.

If the target_id has been computed with xp_target_list_inter or xp_target_range function, this function returns an array with as many elements as num_user_target. For every element, the list of LOS targets corresponding to user target are provided.

If the target_id has been computed with any other function, the returned array has only one position with the list of user targets and the list of LOS targets.

Note on usage: the user must reserve the input-output array to the function, no internal allocation is done.

7.112.2 Calling interface

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

```
#include <explorer_lib.h>
{
        xp_target_id target_id;
        long status;
        xp_target_id_data *data;
        status =  xp_target_get_id_data (&target_id, data);
}
```

7.112.3 Input parameters

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

Table 297: Input parameters of xp_target_get_id_data function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
target_id	xp_target_id *	-	Target Id.	-	-

7.112.4 Output parameters

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

Table 298: Output parameters of xp_target_get_id_data function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
xp_target_get_id_data	long	-	Status flag	-	-
data	xp_target_id_data*	-	Target initialization data	-	-

7.112.5 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 target_id was not computed.

7.113 xp_gen_dem_max_altitude_file

7.113.1 Overview

The **xp_gen_dem_max_altitude_file** CFI function generates a binary file with the maximum altitudes corresponding to every mini-tile, as described in input DEM configuration file. This file is needed to the maximum altitudes algo as described in section 7.71.4.

7.113.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     char *dem_config_file;
       long ierr[XP_NUM_ERR_DEM_MAX_ALT_FILE], status;

     status = xp_gen_dem_max_altitude_file(dem_config_file,
                                           ierr);
}
```

The `XP_NUM_ERR_DEM_MAX_ALT_FILE` constant is defined in the file *explorer_pointing.h*.

7.113.3 Input Parameters

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

Table 299: Input parameters of xp_gen_dem_max_altitude_file function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
dem_config_file	Char*	-	DEM configuration file with description of mini tiles	-	-

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

- Choice. (See Table 3).

7.113.4 Output Parameters

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

Table 300: Output parameters of xp_gen_dem_max_altitude_file

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

7.113.5 Warnings and Errors

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

Table 301: Error messages of xp_gen_dem_max_altitude function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Error reading DEM configuration file	No computation performed	XP_GEN_DEM_MAX_ALTITUDE_READ_DEM_CFG_ERR	0
ERR	Error allocating memory	No computation performed	XP_GEN_DEM_MAX_ALTITUDE_MEMORY_ERR	1
WARN	Some DEM tiles are not present. Corresponding altitudes set to zero.	Computation performed	XP_GEN_DEM_MAX_ALTITUDE_OPEN_TILE_WARN	2
ERR	Error reading DEM tile %s	No computation performed	XP_GEN_DEM_MAX_ALTITUDE_READ_TILE_ERR	3
ERR	Error initializing DEM ID	No computation performed	XP_GEN_DEM_MAX_ALTITUDE_DEM_INIT_ERR	4
ERR	Error closing DEM ID	No computation performed	XP_GEN_DEM_MAX_ALTITUDE_DEM_CLOSE_ERR	5
ERR	Error opening output file %s	No computation performed	XP_GEN_DEM_MAX_ALTITUDE_OPEN_OUTPUT_FILE_ERR	6
ERR	Error writing to output file %s	No computation performed	XP_GEN_DEM_MAX_ALTITUDE_WRITE_OUTPUT_ERR	7
ERR	No output file name provided	No computation performed	XP_GEN_DEM_MAX_ALTITUDE_CONFIG_ERR	8

7.113.6 Executable Program

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

gen_dem_max_altitude_file **-dem_cfg_file** dem_configuration_file
[**-v**]
[**-xl_v**]
[**-xo_v**]
[**-xp_v**]
[**-help**]
[**-show**]

Note that:

- Order of parameters does not matter.
- Bracketed parameters are not mandatory.
- The input DEM configuration file must have the "MiniTiles_Configuration" tag in "DEM_User_Parameters" section.
- [**-xl_v**] option for EO_LIB Verbose mode.
- [**-xo_v**] option for EO_ORBIT Verbose mode.
- [**-xp_v**] option for EO_POINTING Verbose mode.
- [**-v**] option for Verbose mode for all libraries (default is Silent).
- [**-show**] displays the inputs of the function and the results.

Example:

```
gen_dem_max_altitude_file -dem_config_file
S1A_TEST_INT_DEMCFG_00000000T000000_99999999T999999_0003.EOF
```

7.114 xp_gen_dem_altitudes_from_ellipsoid

7.114.1 Overview

The **xp_gen_dem_altitudes_from_ellipsoid** CFI function generates, for an input DEM ACE2 or GDEM V2 dataset, whose altitudes are expressed w.r.t the geoid, an equivalent DEM but with the heights referenced to the ellipsoid, not to the geoid. This way the geoid undulation computation can be avoided at runtime and performance can be improved.

It can be computed the whole DEM or only a set of tiles, depending on the inputs to the function. The field “set_type” of xp_gen_dem_alt_from_ellipsoid_inputs struct can take the following values to select which DEM set to compute:

- XP_ALL_DEM: all DEM tiles will be computed.
- XP_DEM_SET: only the tiles of the DEM fully inside the interval provided by “lon_min”, “lon_max”, “lat_min” and “lat_max” fields of xp_gen_dem_alt_from_ellipsoid_inputs struct are computed.

7.114.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
     char *dem_config_file, *output_dir;
     long num_harmonics;
     xp_gen_dem_alt_from_ellipsoid_inputs inputs;
       long ierr[XP_NUM_ERR_DEM_ALT_FROM_ELLIPSOID], status;

     status = xp_gen_dem_altitudes_from_ellipsoid(dem_config_file,
                                  &num_harmonics, output_dir,
                                  &inputs, ierr);
}
```

The `XP_NUM_ERR_DEM_ALT_FROM_ELLIPSOID` constant is defined in the file *explorer_pointing.h.*

7.114.3 Input Parameters

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

Table 302: Input parameters of xp_gen_dem_altitudes_from_ellipsoid function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
dem_config_file	char*	-	DEM configuration file with description of mini tiles	-	-
num_harmonics	long*	-	Number of harmonics to be used in geoid computation.	-	>0
output_dir	char*	-	Output directory where generated DEM will be placed	-	-
inputs	xp_gen_dem_alt_from_ellipsoid_inputs*	-	DEM set to be processed	For range limits: deg	Longitude: [-180., 180.] Latitude: [-90., 90.]

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

- Choice. (See Table 3).

7.114.4 Output Parameters

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

Table 303: Output parameters of xp_gen_dem_altitudes_from_ellipsoid

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

7.114.5 Warnings and Errors

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

Table 304: Error messages of xp_gen_dem_max_altitude function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Error reading DEM configuration file	No computation performed	XP_CFI_GEN_DEM_ALTITUDE_FROM_ELLIPSOID_READ_DEM_CFG_ERR	0
ERR	Wrong input DEM type	No computation performed	XP_CFI_GEN_DEM_ALTITUDE_FROM_ELLIPSOID_WRONG_DEM_TYPE_ERR	1
ERR	Error allocating memory	No computation performed	XP_CFI_GEN_DEM_ALTITUDE_FROM_ELLIPSOID_MEMORY_ERR	2
ERR	Error reading DEM tile %s	No computation performed	XP_CFI_GEN_DEM_ALTITUDE_FROM_ELLIPSOID_READ_TILE_ERR	3
ERR	Error computing geoid undulation	No computation performed	XP_CFI_GEN_DEM_ALTITUDE_FROM_ELLIPSOID_GEOID_UNDU_ERR	4
ERR	Error opening output file %s	No computation performed	XP_CFI_GEN_DEM_ALTITUDE_FROM_ELLIPSOID_OPEN_OUTPUT_ERR	5
ERR	Error writing to output file %s	No computation performed	XP_CFI_GEN_DEM_ALTITUDE_FROM_ELLIPSOID_WRITE_OUT	6

			PUT_ERR	
WARN	Some DEM tiles are not present. Corresponding altitudes set to zero.	Computation performed	XP_CFI_GEN_DEM_ALTITUDE_FROM_ELLIPSOID_OPEN_TILE_WARN	7
ERR	Wrong input DEM set type	No computation performed	XP_CFI_GEN_DEM_ALTITUDE_FROM_ELLIPSOID_WRONG_DEM_SET_TYPE_ERR	8
ERR	Wrong input longitude interval (must be in interval [-180, 180] deg	No computation performed	XP_CFI_GEN_DEM_ALTITUDE_FROM_ELLIPSOID_WRONG_SET_LONGITUDE_ERR	9
ERR	Wrong input latitude interval (must be in interval [-90, 90] deg	No computation performed	XP_CFI_GEN_DEM_ALTITUDE_FROM_ELLIPSOID_WRONG_SET_LATITUDE_ERR	10

7.115 xp_attitude_transform

7.115.1 Overview

The **xp_attitude_transform** CFI function allows the user to change the reference frame in which the internal data of the attitude ids are expressed.

Remark: all attitude related computations are performed in the True of Date Coordinate System. This means that, if one attitude id has been initialized with data expressed in another reference frame (e.g. attitude file with quaternions expressed in EF), at each call of a function using such attitude id (e.g. xp_attitude_compute) one or more conversions to True of Date will be performed. If the reference frame is changed to True of Date (using xp_attitude_transform), such conversions will not be executed and this will result in a run-time performance improvement.

Note: transformation of attitude ids that are initialized with Start Tracker files is not supported.

7.115.2 Calling Interface

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

```
#include <explorer_pointing.h>
{
    xp_transform_cfg transform_cfg;
  xp_attitude_def attitude_def;
  long ierr[XP_NUM_ERR_ATTITUDE_TRANSFORM], status;

    status = xp_attitude_transform(&transform_cfg, &attitude_def
                                   ierr);
}
```

The `XP_NUM_ERR_ATTITUDE_TRANSFORM` constant is defined in the file *explorer_pointing.h.*

7.115.3 Input Parameters

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

Table 305: Input parameters of xp_attitude_transform function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
transform_cfg	xp_transform_cfg*	-	Structure containing parameters used for the transformation	-	-
attitude_def	xp_attitude_def*	-	Structure containing: satellite nominal attitude, satellite attitude target, instrument attitude target and attitude type		

7.115.4 Output Parameters

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

Table 306: Output parameters of xp_attitude_transform

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
attitude_def	xp_attitude_def*	-	Structure containing: satellite nominal attitude, satellite attitude target, instrument attitude target and attitude type (Input/Output parameter)	-	-
ierr	long	-	Error vector	-	-

7.115.5 Warnings and Errors

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

Table 307: Error messages of xp_attitude_transform function

Error type	Error message	Cause and impact	Error code	Error No
ERR	Error transforming quaternions to vectors	No computation performed	XP_ATTITUDE_TRANSFORM_QUAT_2_VEC_ERR	0
ERR	Error computing rotation matrix	No computation performed	XP_ATTITUDE_TRANSFORM_GET_ROTATION_ERR	1
ERR	Error transforming vectors to quaternions	No computation performed	XP_ATTITUDE_TRANSFORM_VEC_2_QUAT_ERR	2
ERR	Uninitialized attitude	No computation performed	XP_ATTITUDE_TRANSFORM_UNINIT_ATTITUDE_ERR	3
ERR	Only XP_SAT_NOMINAL_ATT, XP_SAT_ATT and XP_INSTR_ATT allowed	No computation performed	XP_ATTITUDE_TRANSFORM_UNALLOWED_ATT_TYPE_ERR	4
ERR	Transformation not supported for Star Tracker data	No computation performed	XP_ATTITUDE_TRANSFORM_STAR_TRACKER_NOT_SUPPORTED_ERR	5

7.116 xp_free_target_id_data

7.116.1 Overview

The **xp_free_target_id_data** CFI function allows the user to free the memory allocated by the xp_target_id_data structure.

7.116.2 Calling interfaces

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

```
#include <explorer_pointing.h>
{
      xp_target_id_data *data;
      long num_user_target;
      long status;

      status = xp_free_target_id_data(data, num_user_target);
}
```

7.116.3 Input Parameters

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

Table 308: Input parameters of xp_free_target_id_data function

C name	C type	Array Element	Description (Reference)	Unit (Format)	Allowed Range
data	xp_target_id_data	-	Structure containing the target parameters	-	-
num_user_target	long	-	Number of the targets defined by user	-	-

7.116.4 Output Parameters

The **xp_free_target_id_data** CFI function has no output parameters.

7.116.5 Warnings and Errors

The **xp_free_target_id_data** CFI function has no warnings and errors defined.

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) Xeon(R)CPU ES-2630 @ 2.40GHz 2.40GHz	Microsoft Windows 7	16 GB

The table below shows the time (in miliseconds - ms) each function takes to be run under each platform:

Function ID	WINDOWS64	LINUX64	LINUX64 LEGACY	MACIN64
xp_attitude_init	0.002300	0.001000	0.000000	0.002000
xp_sat_nominal_att_init	0.001505	0.000200	0.000160	0.000220
xp_sat_nominal_att_init_file * 5 Quaternions	0.536000	0.320000	0.270000	0.360000
xp_sat_nominal_att_get_file	0.000258	0.000200	0.000150	0.000210
xp_sat_nominal_att_set_file	0.000292	0.000170	0.000120	0.000120
xp_sat_nominal_att_init_model	0.001540	0.000200	0.000200	0.000300
xp_sat_nominal_att_get_param	0.000048	0.000010	0.000010	0.000000
xp_sat_nominal_att_set_param	0.000049	0.000012	0.000012	0.000008
xp_sat_att_matrix_init	0.001800	0.001000	0.001000	0.001000
xp_sat_att_init_harmonic	0.001600	0.000400	0.000200	0.000200
xp_sat_att_init_file	0.533000	0.330000	0.280000	0.360000
xp_sat_att_angle_init	0.001498	0.000350	0.000230	0.000200
xp_sat_att_get_angles	0.000013	0.000011	0.000009	0.000005
xp_sat_att_set_angles	0.000014	0.000010	0.000010	0.000010
xp_sat_att_quat_plus_matrix_init	0.001720	0.000600	0.000300	0.000500

xp_sat_att_quat_plus_angle_init	0.001790	0.000600	0.000500	0.000500
xp_sat_att_set_quat_plus_matrix	0.000180	0.000100	0.000100	0.000100
xp_sat_att_set_quat_plus_angle	0.000280	0.000100	0.000800	0.000100
xp_instr_att_init_harmonic	0.001620	0.000400	0.000200	0.000200
xp_instr_att_get_harmonic	0.000101	0.000050	0.000030	0.000030
xp_instr_att_set_harmonic	0.000100	0.000000	0.000000	0.000000
xp_instr_att_init_file	0.516000	0.300000	0.240000	0.320000
xp_instr_att_matrix_init	0.001552	0.000680	0.000500	0.000720
xp_instr_att_get_matrix	0.000042	0.000012	0.000012	0.000008
xp_instr_att_angle_init	0.001519	0.000200	0.000160	0.000240
xp_instr_att_get_angles	0.000013	0.000020	0.000010	0.000010
xp_instr_att_set_angles	0.000014	0.000010	0.000010	0.000000
xp_attitude_compute * target frame: XP_INSTR_ATT	0.029200	0.016000	0.018000	0.014000
xp_attitude_get_id_data	0.000223	0.000070	0.000050	0.000040
xp_atmos_init	0.620800	0.328000	0.224000	0.524000
xp_atmos_get_id_data	0.000008	0.000000	0.000000	0.000010
xp_dem_init	1.447000	0.280000	0.270000	0.320000
xp_dem_get_info	0.070100	0.005000	0.010000	0.009000
xp_dem_compute	0.010000	0.010000	0.080000	0.010000
xp_dem_id_configure * load tiles->free cache	35.570000	11.400000	11.100000	9.100000
xp_get_attitude_data	0.009200	0.004000	0.006000	0.004000
xp_target_inter	0.004000	0.004000	0.002000	0.002000
xp_target_get_id_data	0.000980	0.000600	0.000400	0.000600
xp_target_extra_vector * No derivates	0.000445	0.000420	0.000350	0.000220
xp_target_ground_range	0.026200	0.028000	0.026000	0.008000
xp_target_incidence_angle	0.044400	0.032000	0.030000	0.022000
xp_target_range	0.012000	0.008000	0.008000	0.008000
xp_target_range_rate	0.013000	0.010000	0.010000	0.010000
xp_target_tangent	0.012400	0.008000	0.008000	0.002000
xp_target_altitude	0.009900	0.006000	0.007000	0.006000
xp_target_star	0.000000	0.000000	0.000000	0.000000

xp_target_extra_vector * 1st Derivates	0.000458	0.000426	0.000355	0.000217
xp_target_tangent_moon	0.025000	0.020000	0.030000	0.010000
xp_target_generic	0.002830	0.001100	0.000700	0.001100
xp_target_station	0.005370	0.002600	0.002100	0.001800
xp_target_extra_main	0.014620	0.011800	0.012500	0.006700
xp_target_extra_aux	0.070000	0.060000	0.060000	0.040000
xp_target_extra_target_to_sun	0.021440	0.016400	0.019700	0.008000
xp_target_extra_target_to_moon	0.022040	0.020200	0.021800	0.008700
xp_target_extra_ef_target	0.013000	0.010000	0.010000	0.000000
xp_target_extra_specular_reflection	0.008020	0.005900	0.006800	0.003600
xp_target_sc	0.159000	0.100000	0.120000	0.070000
xp_target_reflected	12.634000	8.200000	8.810000	7.760000
xp_target_travel_time	0.017940	0.013400	0.012000	0.006800
xp_multi_target_inter	0.042000	0.030000	0.030000	0.010000
xp_multi_target_travel_time	0.031500	0.022000	0.020000	0.012000
xp_change_frame	0.028500	0.016000	0.018000	0.013000
xp_target_list_inter * (time per user target)	0.002059	0.001961	0.005882	0.001961
xp_target_inter + xp_target_extra_main * (DEM target)	0.027800	0.024000	0.058000	0.014000
xp_target_list_extra_vector * (time per user target - DEM target)	0.010685	0.008823	0.020586	0.007842
xp_target_list_extra_main * (time per user target - DEM target)	0.011567	0.007842	0.017645	0.013724
xp_target_list_extra_aux * (time per user target - DEM target)	0.013234	0.009803	0.021567	0.042153
xp_target_list_extra_ef_target * (time per user target - DEM target)	0.011469	0.007842	0.016665	0.012744
xp_target_list_extra_specular_reflection * (time per user target - DEM target)	0.011469	0.008823	0.013724	0.011764
xp_target_list_extra_target_to_moon * (time per user target - DEM target)	0.011371	0.007842	0.013724	0.011764
xp_target_list_extra_target_to_sun * (time per user target - DEM target)	0.011371	0.008823	0.013724	0.010783
xp_gen_dem_max_altitude_file * 3x3 minitiles (9 minitiles/tile)	27934.000000	31330.000000	30890.000000	17220.000000

xp_gen_dem_altitudes_from_ellipsoid * 1 tile, 30 harmonics	71806.000000	27040.000000	750.000000	17320.000000
xp_attitude_transform	0.000000	0.000000	0.000000	0.000000
xp_gen_attitude_data * (sat nominal, 2 orbits, time_step=10 sec)	63.779999	43.400002	45.000000	25.900000
xp_gen_attitude_file * (sat nominal, 2 orbits, time_step=10 sec)	93.830002	93.300003	79.800003	71.800003

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 nano-second and so it can be considered as “0”.

9 LIBRARY PRECAUTIONS

The following precaution shall be taking into account when using EO_POINTING library:

- When a message like
  <LIBRARY NAME> >>> ERROR in *xp_function*: Internal computation error # *n*
  or
  <LIBRARY NAME> >>> WARNING in *xp_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.

10 USER REFRACTION FILE

The refraction table is needed to initialize the atmosphere id with an user initialization mode.

The file must contain the co-index of refraction at different geometric altitudes, starting from 0 Km. The altitude should be strict monotonic increasing.

The format of that file must be as follows (a text file without headers):

Table 309: User refraction file format

1st column Geometric altitude [m]	**2nd column** Co-index of refraction N
0.000	262.049
1000.000	238.630
2000.000	216928
3000.000	195.392
...	...
90000.000	0.001
95000.000	0.000
100000.000	0.000

Note in this table that:

- the relative index of refraction $m = 1 + N \times 10^{-6}$, where N is the co-index of refraction.
- The fields of each row must be separated by blanks (at least one).