



---

Navigation and Ancillary Information Facility

# Frames Kernel FK

September 2009



# Introduction

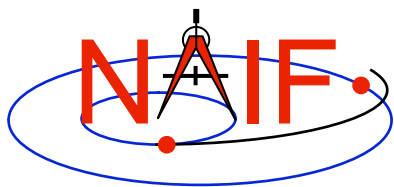
---

Navigation and Ancillary Information Facility

## What does the FRAMES subsystem do?

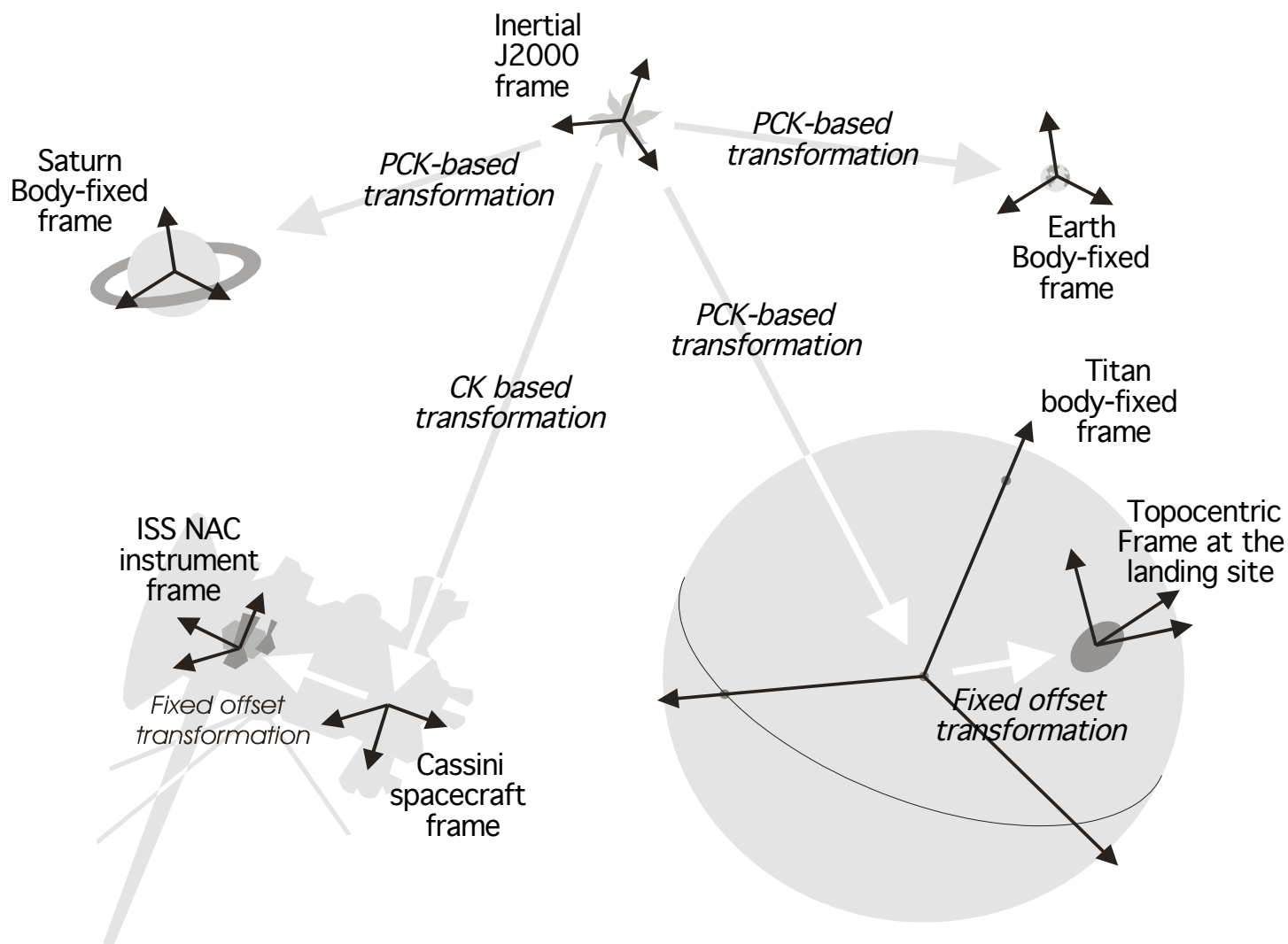
- It establishes relationships between reference frames used in geometry computations -- it "chains frames together."
- It connects frames with sources of their orientation specifications.
- Based on these relationships and orientation source information, it allows SPICE software to compute transformations between neighboring frames in the "chain," and to combine these transformations in the right order, thus providing an ability to compute orientation of any frame in the chain with respect to any other frame in the chain at any time. (\*)

(\*) If the complete set of underlying SPICE data needed to compute the transformation is available.



# Sample Frame Tree and Chains

Navigation and Ancillary Information Facility





# Frame Classes

---

Navigation and Ancillary Information Facility

## Frame class

## Examples

### **Inertial**

- Earth Equator/Equinox of Epoch (J2000, ...)
- Planet Equator/Equinox of Epoch (MARSIAU, ...)
- Ecliptic of Epoch (ECLIPJ2000, ...)

### **Body-fixed**

- Solar system body IAU frames (IAU\_SATURN, ...)
- High accuracy Earth frames (ITRF93, ...)
- High accuracy Moon frames (MOON\_PA, MOON\_ME)

### **CK-based**

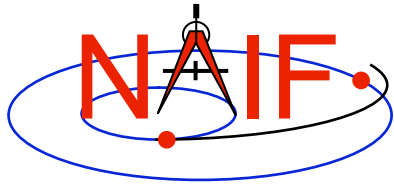
- Spacecraft (CASSINI\_SC\_BUS, ...)
- Moving parts of an instrument (MPL\_RA\_JOINT1, ...)

### **Fixed Offset**

- Instrument mounting alignment (CASSINI\_ISS\_NAC, ...)
- Topocentric (DSS-14\_TOPO, ...)

### **Dynamic**

- See the Dynamic Frames tutorial

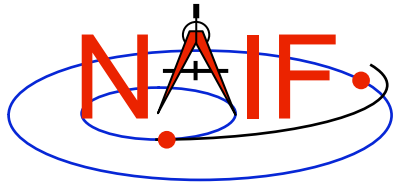


# Frames Class Specifications

---

Navigation and Ancillary Information Facility

<b><i><u>Frame class</u></i></b>	<b><i><u>Frame Defined in</u></i></b>	<b><i><u>Orientation provided in</u></i></b>
<b>Inertial</b>	<b>Toolkit</b>	<b>Toolkit</b>
<b>Bodyfixed</b>	<b>Toolkit or FK</b>	<b>PCK</b>
<b>CK based</b>	<b>FK</b>	<b>CK</b>
<b>Fixed offset</b>	<b>FK</b>	<b>FK</b>
<b>Dynamic</b>	<b>FK</b>	<b>Toolkit, or computed using FK, SPK, CK, and/or PCK</b>



# FRAMES Subsystem Interfaces

---

Navigation and Ancillary Information Facility

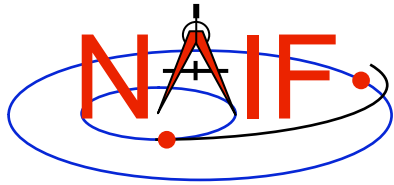
**SXFORM/PXFORM** returns state or position transformation matrix

```
CALL SXFORM ( 'FROM_FRAME_NAME', 'TO_FRAME_NAME', ET, MAT6x6 )  
CALL PXFORM ( 'FROM_FRAME_NAME', 'TO_FRAME_NAME', ET, MAT3x3 )
```

**SPKEZR/SPKPOS** returns state or position vector in specified frame

```
CALL SPKEZR ( BOD, ET, 'FRAME_NAME', CORR, OBS, STATE, LT )  
CALL SPKPOS ( BOD, ET, 'FRAME_NAME', CORR, OBS, POSITN, LT )
```

The above are FORTRAN examples, using SPICELIB modules.  
The same interfaces exist for C, using CSPICE modules, and for Icy and Mice.



# What are the Names of Frames?

---

Navigation and Ancillary Information Facility

- Refer to “NAIF IDs” Tutorial for an introduction to reference frame names and IDs
- Refer to **FRAMES.REQ** for the list of NAIF “built in” (hard coded) inertial and body-fixed frames
- Refer to a project’s **Frames Kernel (FK)** file for a list of frames defined for the spacecraft, its subsystems and instruments
- Refer to an earth stations **FK** for a list of frames defined for the DSN and other stations
- Refer to the moon **FKs** for descriptions of the body-fixed frames defined for the moon



# Frames Kernel File

Navigation and Ancillary Information Facility

- Uses the SPICE text kernel file standards
- Loaded using the FURNISH routine
- Usually contains comprehensive information about the defined frames in the text section(s) of the file
- Contains frame definition information consisting of a set of keywords in the data sections of the file. Below are examples of a CK-based frame and a fixed-offset frame definitions:

## CK-based Frame Example

```
FRAME_DAWN_SPACECRAFT = -203000
FRAME_-203000_NAME     = 'DAWN_SPACECRAFT'
FRAME_-203000_CLASS    = 3
FRAME_-203000_CLASS_ID = -203000
FRAME_-203000_CENTER   = -203
CK_-203000_SCLK        = -203
CK_-203000_SPK         = -203
```

## Fixed-offset Frame Example

```
FRAME_DAWN_FC1         = -203110
FRAME_-203110_NAME     = 'DAWN_FC1'
FRAME_-203110_CLASS    = 4
FRAME_-203110_CLASS_ID = -203110
FRAME_-203110_CENTER   = -203
TKFRAME_-203110_RELATIVE = 'DAWN_SPACECRAFT'
TKFRAME_-203110_SPEC    = 'ANGLES'
TKFRAME_-203110_UNITS   = 'DEGREES'
TKFRAME_-203110_ANGLES  = ( 0.0, 0.0, 0.0 )
TKFRAME_-203110_AXES    = ( 1, 2, 3 )
```

- These examples are discussed in detail on the next few slides





# Frame Definition Details - 1

## Navigation and Ancillary Information Facility

```
FRAME_DAWN_SPACECRAFT = -203000
FRAME_-203000_NAME     = 'DAWN_SPACECRAFT'
FRAME_-203000_CLASS    = 3
FRAME_-203000_CLASS_ID = -203000
FRAME_-203000_CENTER   = -203
CK_-203000_SCLK        = -203
CK_-203000_SPK         = -203
```

```
FRAME_DAWN_FC1         = -203110
FRAME_-203110_NAME     = 'DAWN_FC1'
FRAME_-203110_CLASS    = 4
FRAME_-203110_CLASS_ID = -203110
FRAME_-203110_CENTER   = -203
TKFRAME_-203110_RELATIVE = 'DAWN_SPACECRAFT'
TKFRAME_-203110_SPEC    = 'ANGLES'
TKFRAME_-203110_UNITS  = 'DEGREES'
TKFRAME_-203110_ANGLES  = ( 0.0, 0.0, 0.0 )
TKFRAME_-203110_AXES   = ( 1, 2, 3 )
```

- The Frame ID is an integer number used by the SPICE system as a “handle” in buffering and retrieving various parameters associated with a frame. In an FK it “glues” together the keywords defining the frame.



## Frame Definition Details - 2

### Navigation and Ancillary Information Facility

```
FRAME_DAWN_SPACECRAFT = -203000
FRAME_-203000_NAME     = 'DAWN_SPACECRAFT'
FRAME_-203000_CLASS    = 3
FRAME_-203000_CLASS_ID = -203000
FRAME_-203000_CENTER   = -203
CK_-203000_SCLK        = -203
CK_-203000_SPK         = -203
```

```
FRAME_DAWN_FC1         = -203110
FRAME_-203110_NAME     = 'DAWN_FC1'
FRAME_-203110_CLASS    = 4
FRAME_-203110_CLASS_ID = -203110
FRAME_-203110_CENTER   = -203
TKFRAME_-203110_RELATIVE = 'DAWN_SPACECRAFT'
TKFRAME_-203110_SPEC    = 'ANGLES'
TKFRAME_-203110_UNITS   = 'DEGREES'
TKFRAME_-203110_ANGLES  = ( 0.0, 0.0, 0.0 )
TKFRAME_-203110_AXES    = ( 1, 2, 3 )
```

- The “FRAME\_<name> = <id>” and “FRAME\_<id>\_NAME = <name>” keywords establish the association between the name and ID of the frame



## Frame Definition Details - 3

### Navigation and Ancillary Information Facility

```
FRAME_DAWN_SPACECRAFT = -203000
FRAME_-203000_NAME    = 'DAWN_SPACECRAFT'
FRAME_-203000_CLASS   = 3
FRAME_-203000_CLASS_ID = -203000
FRAME_-203000_CENTER  = -203
CK_-203000_SCLK       = -203
CK_-203000_SPK        = -203
```

```
FRAME_DAWN_FC1        = -203110
FRAME_-203110_NAME    = 'DAWN_FC1'
FRAME_-203110_CLASS   = 4
FRAME_-203110_CLASS_ID = -203110
FRAME_-203110_CENTER  = -203
TKFRAME_-203110_RELATIVE = 'DAWN_SPACECRAFT'
TKFRAME_-203110_SPEC   = 'ANGLES'
TKFRAME_-203110_UNITS  = 'DEGREES'
TKFRAME_-203110_ANGLES = ( 0.0, 0.0, 0.0 )
TKFRAME_-203110_AXES   = ( 1, 2, 3 )
```

- The **FRAME...CLASS** keyword specifies the method by which the frame is related to its base frame
- This keyword is set to:
  - 2 for PCK-based frames
  - 3 for CK-based frames
  - 4 for fixed-offset frames
  - 5 for dynamic frames



# Frame Definition Details - 4

## Navigation and Ancillary Information Facility

```
FRAME_DAWN_SPACECRAFT = -203000
FRAME_-203000_NAME     = 'DAWN_SPACECRAFT'
FRAME_-203000_CLASS    = 3
FRAME_-203000_CLASS_ID = -203000
FRAME_-203000_CENTER   = -203
CK_-203000_SCLK        = -203
CK_-203000_SPK         = -203
```

```
FRAME_DAWN_FC1         = -203110
FRAME_-203110_NAME     = 'DAWN_FC1'
FRAME_-203110_CLASS    = 4
FRAME_-203110_CLASS_ID = -203110
FRAME_-203110_CENTER   = -203
TKFRAME_-203110_RELATIVE = 'DAWN_SPACECRAFT'
TKFRAME_-203110_SPEC    = 'ANGLES'
TKFRAME_-203110_UNITS   = 'DEGREES'
TKFRAME_-203110_ANGLES  = ( 0.0, 0.0, 0.0 )
TKFRAME_-203110_AXES    = ( 1, 2, 3 )
```

- The **FRAME...CLASS\_ID** is the number that connects a frame with the orientation data for it.
  - For body-fixed frames the **CLASS\_ID** is the ID of the natural body. It is used as input to PCK routines called by the Frame subsystem to compute orientation of the frame.
    - » The Frame ID and **CLASS\_ID** are not the same for the body-fixed frames defined in the Toolkit but they can be the same for frames defined in FK files.
  - For CK-based frames the **CLASS\_ID** is the CK structure ID. It is used as input to CK routines called by the Frame subsystem to compute orientation of the frame.
    - » Usually the **CLASS\_ID** of a CK-based frame is the same as the frame ID, but this is not required.
  - For fixed offset and dynamic frames the **CLASS\_ID** is the ID that is used to retrieve the frame definition keywords.
    - » The **CLASS\_ID** of a fixed offset or dynamic frame is the same as the frame ID.



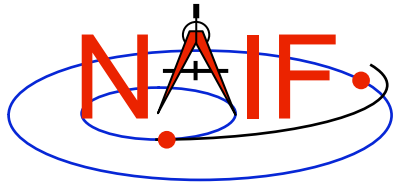
# Frame Definition Details - 5

## Navigation and Ancillary Information Facility

```
FRAME_DAWN_SPACECRAFT = -203000
FRAME_-203000_NAME     = 'DAWN_SPACECRAFT'
FRAME_-203000_CLASS    = 3
FRAME_-203000_CLASS_ID = -203000
FRAME_-203000_CENTER   = -203
CK_-203000_SCLK        = -203
CK_-203000_SPK         = -203
```

```
FRAME_DAWN_FC1         = -203110
FRAME_-203110_NAME     = 'DAWN_FC1'
FRAME_-203110_CLASS    = 4
FRAME_-203110_CLASS_ID = -203110
FRAME_-203110_CENTER   = -203
TKFRAME_-203110_RELATIVE = 'DAWN_SPACECRAFT'
TKFRAME_-203110_SPEC    = 'ANGLES'
TKFRAME_-203110_UNITS  = 'DEGREES'
TKFRAME_-203110_ANGLES = ( 0.0, 0.0, 0.0 )
TKFRAME_-203110_AXES   = ( 1, 2, 3 )
```

- **The FRAME...CENTER specifies the ephemeris object at which the frame origin is located**
  - It is used **ONLY** to compute the light-time corrected orientation of the frame



# Frame Definition Details - 6

## Navigation and Ancillary Information Facility

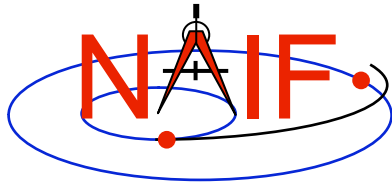
```

FRAME_DAWN_SPACECRAFT = -203000
FRAME_-203000_NAME     = 'DAWN_SPACECRAFT'
FRAME_-203000_CLASS    = 3
FRAME_-203000_CLASS_ID = -203000
FRAME_-203000_CENTER   = -203
CK_-203000_SCLK        = -203
CK_-203000_SPK         = -203
  
```

```

FRAME_DAWN_FC1         = -203110
FRAME_-203110_NAME     = 'DAWN_FC1'
FRAME_-203110_CLASS    = 4
FRAME_-203110_CLASS_ID = -203110
FRAME_-203110_CENTER   = -203
TKFRAME_-203110_RELATIVE = 'DAWN_SPACECRAFT'
TKFRAME_-203110_SPEC    = 'ANGLES'
TKFRAME_-203110_UNITS   = 'DEGREES'
TKFRAME_-203110_ANGLES  = ( 0.0, 0.0, 0.0 )
TKFRAME_-203110_AXES    = ( 1, 2, 3 )
  
```

- **Additional keywords are included depending on the frame class**
  - For CK frames, CK...SCLK and CK...SPK keywords identify the spacecraft clock ID and physical object ID associated with the CK structure ID
  - For fixed-offset frames, TKFRAME \* keywords specify the base frame and the fixed orientation with respect to this frame
  - For dynamic frames, additional keywords depend on the dynamic frame family



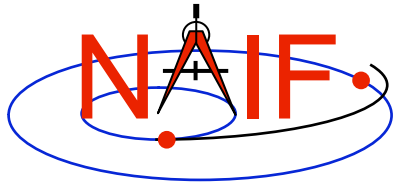
# CK-Based Frames “Must Know”

---

Navigation and Ancillary Information Facility

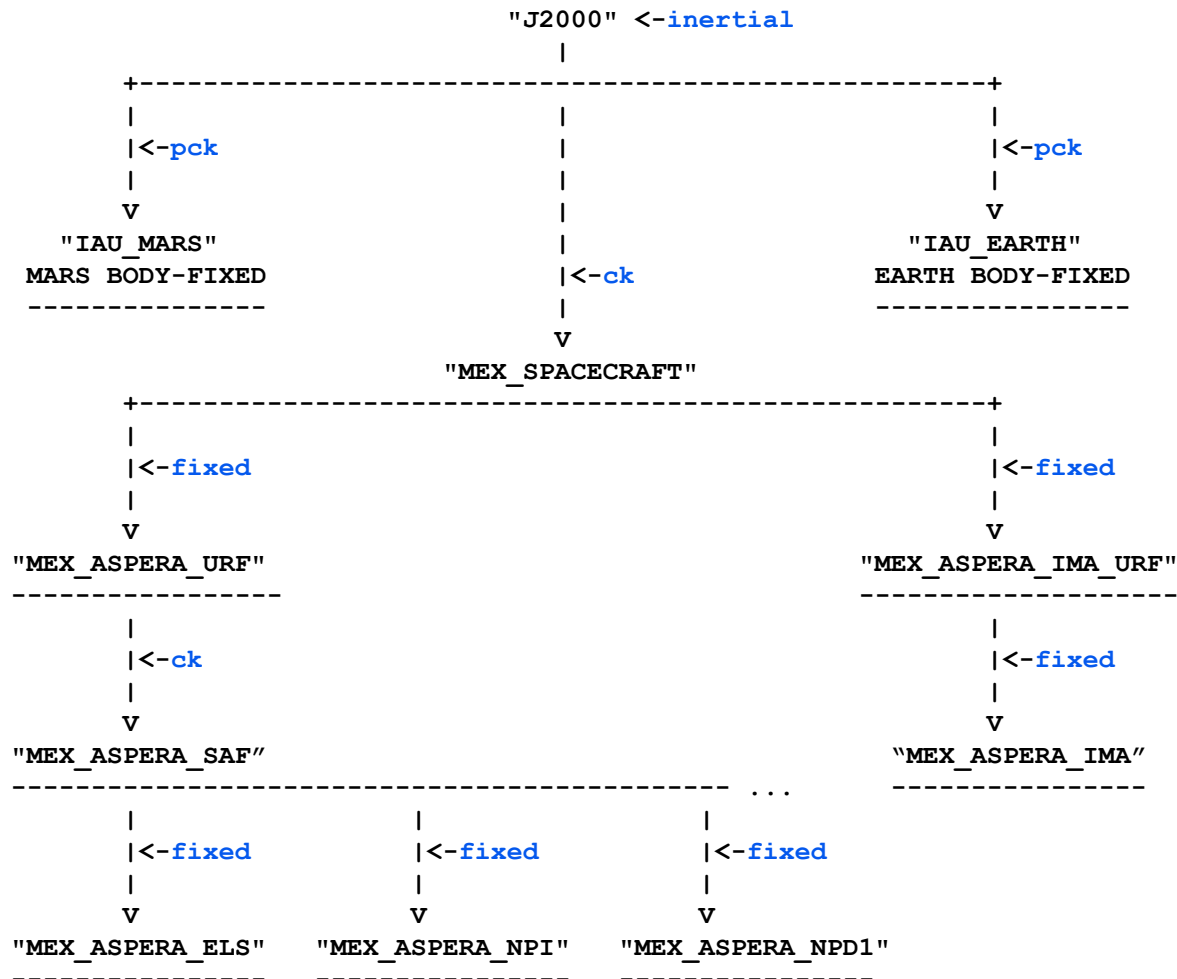
**These are VERY IMPORTANT points you must understand!**

- The frames routines (SPKEZR, SPKPOS, SXFORM, PXFORM) all read CK files using tolerance = 0
  - For **discrete** CKs the orientation of a CK-based frame will be computed only if the time provided to a Frames routine exactly matches one of the times stored in the CK file; otherwise an error will be signaled.
  - For **continuous** CKs the orientation of a CK-based frame will be computed only if the time provided to a Frames routine falls within one of the interpolation intervals defined by the CK file; otherwise an error will be signaled.
- Using SPKEZR or SXFORM requires CKs with angular rates
  - Since these routines return a state vector (6x1) or state transformation matrix (6x6), angular rates must be present in the CK in order to compute vectors and matrices; if rates are not present, an error will be signaled.
  - SPKPOS and PXFORM, which return a position vector (3x1) and a position transformation matrix (3x3) respectively, can be used instead because they require only orientation data to be present in the CK.
- Ephemeris time input to Frames routines is converted to SCLK to access CKs
  - SCLK and LSK kernels must be loaded to support this conversion.
  - SCLK ID is specified in one of the CK frame definition keywords; if not, it's assumed to be the Frame ID divided by a 1000.



# Frame Tree Example: ASPERA Instrument on Mars Express

## Navigation and Ancillary Information Facility



Blue text indicates frame class





# FK Utility Programs

---

Navigation and Ancillary Information Facility

- **The following FK and frames utility programs are included in the Toolkit:**
  - FRMDIFF**      samples orientation of a frame or compares orientation of two frames
  - CKBRIEF**      summarizes coverage for one or more CK files
  - BRIEF**          summarizes coverage for one or more binary PCK files
- **These additional FK and frames utility programs are provided on the NAIF Web site (<http://naif.jpl.nasa.gov/naif/utilities.html>)**
  - PINPOINT**      creates SPK and topocentric frames FK files for fixed locations (ground stations, etc)
  - BINGO**          converts FK files between UNIX and DOS text formats



# Additional Information on FK

---

Navigation and Ancillary Information Facility

- **For more information about FK and frames, look at the following documents**
  - Frames Required Reading
  - Using Frames Tutorial
  - Dynamic Frames Tutorial
  - NAIF IDs Tutorial
  - headers for the routines mentioned in this tutorial
  - Most Useful SPICELIB Routines
  - FRMDIFF User's Guide
  - Porting\_kernels tutorial
- **Related documents**
  - CK Required Reading
  - PCK Required Reading
  - SPK Required Reading
  - Rotations Required Reading