Multimission Software Interface Specification (SIS)

SPICE
Omnibus SIS

NAIF Document No. 419
Version 1.1

Prepared by: C. Acton

Navigation and Ancillary Information Facility (NAIF)
Jet Propulsion Laboratory
National Aeronautics and Space Administration

PURPOSE: This "SIS" tells what and where are the useable interface specifications for all SPICE kernels. Within SPICE these specifications replace the traditional SIS modules.

CHANGE LOG

Version	Date	Page Nos.	Reason
1.0	23 July 2013	All	Original
1.1	8 October 2013	Multiple	Corrected a few typos. Added in an appendix a list of kernels covered by this omnibus SIS.

List of Acronyms

API	Application Program Interface (as in module or subroutine)
ASCII	American Standard Code for Information Interchange
JPL	Caltech/Jet Propulsion Laboratory
NAIF	Navigation and Ancillary Information Facility
SIS	Software Interface Specification
SPICE	S-, P-, I-, C- and E-kernels; the principal logical data components of a particular NASA ancillary information system

Section 1
General Description

1.1 Purpose of Document

This omnibus Software Interface Specification (SIS) explains why traditional SISs are not useful within the SPICE domain, and what replaces them.

1.2 Scope

This is a multimission SIS, applicable for all flight projects.

1.3 Reference Documents

SPICE system description: http://naif.jpl.nasa.gov/naif/aboutspice.html
SPICE toolkit documentation: http://naif.jpl.nasa.gov/pub/naif/toolkit_docs/C/index.html
(Replace the "C" with "FORTRAN", "IDL" or "MATLAB" for those other languages.)
SPICE tutorials: http://naif.jpl.nasa.gov/naif/tutorials.html

1.4 Functional Description

Traditional software interface specifications exist to describe the content and format of a digital data product such that a user of that product can write software to read and use the data therein. Such a digital data product might be an image file returned from a camera, or it might be a SPICE data file, called a kernel, containing one or another type of space geometry data.

Within the SPICE system it is not necessary, and indeed strongly discouraged, that any SPICE data user write his/her own software to read any kind of SPICE kernel. This is because NAIF has instead provided the needed software as a part of the SPICE Toolkit. The Toolkit contains "kernel reader" Application Program Interfaces (APIs, also called modules or subroutines) , plus a great deal more software used to compute observation geometry derived from the data contained in SPICE kernels.

Instead of reading a SPICE SIS, the SPICE user reads and follows the instructions contained in one of the Toolkit APIs. Each SPICE API source code file begins with a very substantial "header" providing all the information a user needs to use that API. **For those APIs that read SPICE kernel files, the header replaces the corresponding SIS.** A list of the SPICE kernels for which this omnibus SIS is applicable is provided in Appendix A. An example of a SPICE API header is provided in Appendix B.

All SPICE API "headers" use the same template and writing style for this important user-focused documentation. The header documentation discusses inputs and outputs, restrictions on use, implementation details, and usually inlcudes working examples.

For the record, there is not a one-to-one correspondence of API modules to SPICE kernel files (and thus to SPICE SISs). This is because many SPICE APIs may access a particular type of SPICE kernel. And many SPICE APIs access multiple kinds of kernels.

Additional helpful information for SPICE kernel users is provided in SPICE documents, such as the "required reading" technical reference documents that exist for major SPICE subsystems, and in the large collection of SPICE tutorials. One of those tutorials, "Summary of Key Points," contains two charts that relate SPICE kernel files and the SPICE APIs most often used with those kernels. These are found on the two pages labeled "Primary Kernel Interfaces." (The SPICE tutorials are found here: http://naif.jpl.nasa.gov/naif/tutorials.html)

1.5 Toolkit Characteristics

The SPICE Toolkits are available in four languages–ANSI Fortran 77, ANSI C, Interactive Data Lanaguage (IDL), and MATLAB. (Java Native Interface will soon be added.) For each language the Toolkit is available for multiple combinations of platform, operating system, OS architecture (32-bit or 64-bit) and, where applicable, compiler model. In total there are over 40 such "environments" supported by NAIF staff.

The source code is provided for every environment. Each Toolkit environment is fully tested, fully documented and fully built–ready for immediate use.

The SPICE Toolkits are always 100% backwards compatible. Once a Toolkit capability has been provided it is never removed or revised (other than to fix bugs).

1.6 Toolkit Availability

The SPICE Toolkits are freely available to anyone worldwide. They are available here on the NAIF server: http://naif.jpl.nasa.gov/naif/toolkit.html. There are no ITAR or licensing restrictions. Neither fees nor registration are required.

Appendix A.
List of SPICE Kernels

All SPICE kernel types are contained in the list below.

SPK	"Ephemeris kernel, containing position and often velocity of as object relative to another object. May contain data for many objects. Might contain reconstruction (historical) data, or predictive data, or both types.
PCK	Planetary constants kernel, containing size, shape and orientation of solar system bodies. May contain additional physical or carographic data as well. Both text and binary forms exist. The binary form is available for just a few bodies and contains only orientation data. SPICE APIs made to read PCKs can read both types. Text PCKs are usually prepared using data endorsed by the International Astronomical Union.
IK	Instrument kernel, containing field-of-view size, shape and orientation for science instruments, and for any other spacecraft assembly for which field-of-view information given in the SPICE style could be useful (e.g. star tracker, high-gain antenna, heat radiator).
CK	Camera-matrix kernel, providing time-varying orientation (attitude) of a spacecraft bus or any attached struture that can be aritculated and for which oreintation data are available, such as a high-gain antenna, solar arrays, a rover's sampling arm, or an instrument's moveable scanning mirror. May contain rate data as well as position data, if available. Might contain reconstruction (historical) data, or predictive data, or both types.
EK	Events kernel, intended to logically encompas three kinds of information: science observation plans (ESP), science observation sequence specifications (ESQ) and scientist' notebook comments (ENB). **The EK portion of SPICE was rarely used and should be considered depricated.**
FK	Frames kernel, provides specifications for the many mission-specific reference frames defined for a mission. Typically included are the spacecraft bus, antennas and solar arrays, and instruments (often reffered to as "instrument mounting alignment"). Note that a variety of generic reference frames–ones not uniquely associated with a given mission–are also available to SPICE users: these specifications are hard-coded in SPICE Toolkit software. (Note that NAIF uses the term "reference frame" or simply "frame" where many people use the term "coordinate system." Within SPICE a "coordinate system" defines the method by which positions within a "reference frame" are measured: e.g. Cartesian coordinates, polar coordinates, etc.)
LSK	Leap seconds kernel, providing a tabulation of leap seconds declared by the International Earth Rotation Service (IERS) plus a few related terms, all needed for time conversions between Universal Time Coordinated (UTC, sometimes referred to as SCET) and Barycentric Dynamical Time (TDB), also reffered to within SPICE as Ephemeris Time (ET).
SCLK	Spacecraft clock kernel, a tabulation of spacecraft clock correlation parameters computed by others and used within SPICE, along with the LSK, for time conversions between spacecraft clock time (also called SCLK) and barycentric dynamical time (TDB). (On rare occasions a time system other than barycentric

	dynamical time is used as one of the two time systems.)
DSK	Digital shape kernel, providing high-precision shape information for (usuall) solar system bodies. The DSK allows for two kinds of shape information: a tessellated plate model and a digital elevation model. Where appropriate source data exist for making a DSK, the DSK can substitute for the very simple tri-axial shape model data contained in a text-style PCK. But note that unlike for a text PCK a DSK contains ONLY shape information: a text PCK must be used for the orientation of the object in question. The DSK design offers many features not usually found in other shape representations.

Appendix B.
Example of a SPICE Toolkit API Header

Below is an example of a Toolkit API "header." This is for the C language version of an API named SPKPOS, used to read an SPK (ephemeris) file and return the position of a "target" relative to an "observer."

```
-Procedure spkpos_c ( S/P Kernel, position )

-Abstract

   Return the position of a target body relative to an observing
   body, optionally corrected for light time (planetary aberration)
   and stellar aberration.

-Disclaimer

   THIS SOFTWARE AND ANY RELATED MATERIALS WERE CREATED BY THE
   CALIFORNIA INSTITUTE OF TECHNOLOGY (CALTECH) UNDER A U.S.
   GOVERNMENT CONTRACT WITH THE NATIONAL AERONAUTICS AND SPACE
   ADMINISTRATION (NASA). THE SOFTWARE IS TECHNOLOGY AND SOFTWARE
   PUBLICLY AVAILABLE UNDER U.S. EXPORT LAWS AND IS PROVIDED "AS-IS"
   TO THE RECIPIENT WITHOUT WARRANTY OF ANY KIND, INCLUDING ANY
   WARRANTIES OF PERFORMANCE OR MERCHANTABILITY OR FITNESS FOR A
   PARTICULAR USE OR PURPOSE (AS SET FORTH IN UNITED STATES UCC
   SECTIONS 2312-2313) OR FOR ANY PURPOSE WHATSOEVER, FOR THE
   SOFTWARE AND RELATED MATERIALS, HOWEVER USED.

   IN NO EVENT SHALL CALTECH, ITS JET PROPULSION LABORATORY, OR NASA
   BE LIABLE FOR ANY DAMAGES AND/OR COSTS, INCLUDING, BUT NOT
   LIMITED TO, INCIDENTAL OR CONSEQUENTIAL DAMAGES OF ANY KIND,
   INCLUDING ECONOMIC DAMAGE OR INJURY TO PROPERTY AND LOST PROFITS,
   REGARDLESS OF WHETHER CALTECH, JPL, OR NASA BE ADVISED, HAVE
   REASON TO KNOW, OR, IN FACT, SHALL KNOW OF THE POSSIBILITY.

   RECIPIENT BEARS ALL RISK RELATING TO QUALITY AND PERFORMANCE OF
   THE SOFTWARE AND ANY RELATED MATERIALS, AND AGREES TO INDEMNIFY
   CALTECH AND NASA FOR ALL THIRD-PARTY CLAIMS RESULTING FROM THE
   ACTIONS OF RECIPIENT IN THE USE OF THE SOFTWARE.

-Required_Reading

   SPK
   NAIF_IDS
   FRAMES
   TIME

-Keywords

   EPHEMERIS
```

```
*/

   #include "SpiceUsr.h"
   #include "SpiceZfc.h"
   #include "SpiceZmc.h"


   void spkpos_c ( ConstSpiceChar   * targ,
                   SpiceDouble        et,
                   ConstSpiceChar   * ref,
                   ConstSpiceChar   * abcorr,
                   ConstSpiceChar   * obs,
                   SpiceDouble        ptarg[3],
                   SpiceDouble      * lt        )
/*

-Brief_I/O

   Variable  I/O  Description
   --------  ---  --------------------------------------------------
   targ       I   Target body name.
   et         I   Observer epoch.
   ref        I   Reference frame of output position vector.
   abcorr     I   Aberration correction flag.
   obs        I   Observing body name.
   ptarg      O   Position of target.
   lt         O   One way light time between observer and target.

-Detailed_Input

   targ        is the name of a target body.  Optionally, you may
               supply the integer ID code for the object as
               an integer string.  For example both "MOON" and
               "301" are legitimate strings that indicate the
               moon is the target body.

               The target and observer define a position vector
               which points from the observer to the target.

   et          is the ephemeris time, expressed as seconds past
               J2000 TDB, at which the position of the target body
               relative to the observer is to be computed.  `et'
               refers to time at the observer's location.

   ref         is the name of the reference frame relative to which
               the output position vector should be expressed. This
               may be any frame supported by the SPICE system,
               including built-in frames (documented in the Frames
               Required Reading) and frames defined by a loaded
               frame kernel (FK).

               When `ref' designates a non-inertial frame, the
```

orientation of the frame is evaluated at an epoch
dependent on the selected aberration correction. See
the description of the output position vector `ptarg'
for details.

abcorr indicates the aberration corrections to be applied to
the position of the target body to account for
one-way light time and stellar aberration. See the
discussion in the Particulars section for
recommendations on how to choose aberration
corrections.

'abcorr' may be any of the following:

"NONE" Apply no correction. Return the
geometric position of the target body
relative to the observer.

The following values of 'abcorr' apply to the
"reception" case in which photons depart from the
target's location at the light-time corrected epoch
et-lt and *arrive* at the observer's location at `et':

"LT" Correct for one-way light time (also
called "planetary aberration") using a
Newtonian formulation. This correction
yields the position of the target at
the moment it emitted photons arriving
at the observer at `et'.

The light time correction uses an
iterative solution of the light time
equation (see Particulars for details).
The solution invoked by the "LT" option
uses one iteration.

"LT+S" Correct for one-way light time and
stellar aberration using a Newtonian
formulation. This option modifies the
position obtained with the "LT" option
to account for the observer's velocity
relative to the solar system
barycenter. The result is the apparent
position of the target---the position
as seen by the observer.

"CN" Converged Newtonian light time
correction. In solving the light time
equation, the "CN" correction iterates
until the solution converges (three
iterations on all supported platforms).

The "CN" correction typically does not substantially improve accuracy because the errors made by ignoring relativistic effects may be larger than the improvement afforded by obtaining convergence of the light time solution. The "CN" correction computation also requires a significantly greater number of CPU cycles than does the one-iteration light time correction.

"CN+S" Converged Newtonian light time and stellar aberration corrections.

The following values of 'abcorr' apply to the "transmission" case in which photons *depart* from the observer's location at `et' and arrive at the target's location at the light-time corrected epoch et+lt:

"XLT" "Transmission" case: correct for one-way light time using a Newtonian formulation. This correction yields the position of the target at the moment it receives photons emitted from the observer's location at `et'.

"XLT+S" "Transmission" case: correct for one-way light time and stellar aberration using a Newtonian formulation. This option modifies the position obtained with the "XLT" option to account for the observer's velocity relative to the solar system barycenter. The computed target position indicates the direction that photons emitted from the observer's location must be "aimed" to hit the target.

"XCN" "Transmission" case: converged Newtonian light time correction.

"XCN+S" "Transmission" case: converged Newtonian light time and stellar aberration corrections.

Neither special nor general relativistic effects are accounted for in the aberration corrections applied by this routine.

Case and blanks are not significant in the string

'abcorr'.

obs is the name of an observing body. Optionally, you may supply the ID code of the object as an integer string. For example, both "EARTH" and "399" are legitimate strings to supply to indicate the observer is Earth.

-Detailed_Output

ptarg is a Cartesian 3-vector representing the position of the target body relative to the specified observer. `ptarg' is corrected for the specified aberrations, and is expressed with respect to the reference frame specified by `ref'. The three components of `ptarg' represent the x-, y- and z-components of the target's position.

Units are always km.

`ptarg' points from the observer's location at `et' to the aberration-corrected location of the target. Note that the sense of this position vector is independent of the direction of radiation travel implied by the aberration correction.

Non-inertial frames are treated as follows: letting ltcent be the one-way light time between the observer and the central body associated with the frame, the orientation of the frame is evaluated at et-ltcent, et+ltcent, or `et' depending on whether the requested aberration correction is, respectively, for received radiation, transmitted radiation, or is omitted. ltcent is computed using the method indicated by 'abcorr'.

lt is the one-way light time between the observer and target in seconds. If the target position is corrected for aberrations, then `lt' is the one-way light time between the observer and the light time corrected target location.

-Parameters

None.

-Exceptions

1) If name of target or observer cannot be translated to its NAIF ID code, the error SPICE(IDCODENOTFOUND) is signaled.

2) If the reference frame `ref' is not a recognized reference frame the error SPICE(UNKNOWNFRAME) is signaled.

3) If the loaded kernels provide insufficient data to
compute the requested position vector, the deficiency will
be diagnosed by a routine in the call tree of this routine.

4) If an error occurs while reading an SPK or other kernel file,
the error will be diagnosed by a routine in the call tree
of this routine.

-Files

This routine computes positions using SPK files that have been loaded into the SPICE system, normally via the kernel loading interface routine furnsh_c. See the routine furnsh_c and the SPK and KERNEL Required Reading for further information on loading (and unloading) kernels.

If the output position `ptarg' is to be expressed relative to a non-inertial frame, or if any of the ephemeris data used to compute `ptarg' are expressed relative to a non-inertial frame in the SPK files providing those data, additional kernels may be needed to enable the reference frame transformations required to compute the position. These additional kernels may be C-kernels, PCK files or frame kernels. Any such kernels must already be loaded at the time this routine is called.

-Particulars

This routine is part of the user interface to the SPICE ephemeris system. It allows you to retrieve position information for any ephemeris object relative to any other in a reference frame that is convenient for further computations.

This routine is identical in function to the routine SPKEZP except that it allows you to refer to ephemeris objects by name (via a character string).

Aberration corrections
==========

In space science or engineering applications one frequently wishes to know where to point a remote sensing instrument, such as an optical camera or radio antenna, in order to observe or otherwise receive radiation from a target. This pointing problem is complicated by the finite speed of light: one needs to point to where the target appears to be as opposed to where it actually is at the epoch of observation. We use the adjectives "geometric," "uncorrected," or "true" to refer to an actual position or state of a target at a specified epoch. When a geometric position or state vector is modified to reflect how it appears to an observer, we describe that vector by any of the

terms "apparent," "corrected," "aberration corrected," or "light time and stellar aberration corrected." The SPICE Toolkit can correct for two phenomena affecting the apparent location of an object: one-way light time (also called "planetary aberration") and stellar aberration.

One-way light time

Correcting for one-way light time is done by computing, given an observer and observation epoch, where a target was when the observed photons departed the target's location. The vector from the observer to this computed target location is called a "light time corrected" vector. The light time correction depends on the motion of the target relative to the solar system barycenter, but it is independent of the velocity of the observer relative to the solar system barycenter. Relativistic effects such as light bending and gravitational delay are not accounted for in the light time correction performed by this routine.

Stellar aberration

The velocity of the observer also affects the apparent location of a target: photons arriving at the observer are subject to a "raindrop effect" whereby their velocity relative to the observer is, using a Newtonian approximation, the photons' velocity relative to the solar system barycenter minus the velocity of the observer relative to the solar system barycenter. This effect is called "stellar aberration." Stellar aberration is independent of the velocity of the target. The stellar aberration formula used by this routine does not include (the much smaller) relativistic effects.

Stellar aberration corrections are applied after light time corrections: the light time corrected target position vector is used as an input to the stellar aberration correction.

When light time and stellar aberration corrections are both applied to a geometric position vector, the resulting position vector indicates where the target "appears to be" from the observer's location.

As opposed to computing the apparent position of a target, one may wish to compute the pointing direction required for transmission of photons to the target. This also requires correction of the geometric target position for the effects of light time and stellar aberration, but in this case the corrections are computed for radiation traveling *from* the observer to the target. We will refer to this situation as the "transmission" case.

The "transmission" light time correction yields the target's

location as it will be when photons emitted from the observer's location at `et' arrive at the target. The transmission stellar aberration correction is the inverse of the traditional stellar aberration correction: it indicates the direction in which radiation should be emitted so that, using a Newtonian approximation, the sum of the velocity of the radiation relative to the observer and of the observer's velocity, relative to the solar system barycenter, yields a velocity vector that points in the direction of the light time corrected position of the target.

One may object to using the term "observer" in the transmission case, in which radiation is emitted from the observer's location. The terminology was retained for consistency with earlier documentation.

Below, we indicate the aberration corrections to use for some common applications:

1) Find the apparent direction of a target. This is the most common case for a remote-sensing observation.

   Use "LT+S": apply both light time and stellar aberration corrections.

   Note that using light time corrections alone ("LT") is generally not a good way to obtain an approximation to an apparent target vector: since light time and stellar aberration corrections often partially cancel each other, it may be more accurate to use no correction at all than to use light time alone.

2) Find the corrected pointing direction to radiate a signal to a target. This computation is often applicable for implementing communications sessions.

   Use "XLT+S": apply both light time and stellar aberration corrections for transmission.

3) Compute the apparent position of a target body relative to a star or other distant object.

   Use "LT" or "LT+S" as needed to match the correction applied to the position of the distant object. For example, if a star position is obtained from a catalog, the position vector may not be corrected for stellar aberration. In this case, to find the angular separation of the star and the limb of a planet, the vector from the observer to the planet should be corrected for light time but not stellar aberration.

4) Obtain an uncorrected position vector derived directly from
   data in an SPK file.

      Use "NONE".

5) Use a geometric position vector as a low-accuracy estimate
   of the apparent position for an application where execution
   speed is critical:

      Use "NONE".

6) While this routine cannot perform the relativistic
   aberration corrections required to compute positions
   with the highest possible accuracy, it can supply the
   geometric positions required as inputs to these computations:

      Use "NONE", then apply relativistic aberration
      corrections (not available in the SPICE Toolkit).

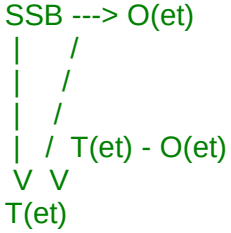
Below, we discuss in more detail how the aberration corrections
applied by this routine are computed.

Geometric case
==============

spkpos_c begins by computing the geometric position T(et) of the
target body relative to the solar system barycenter (SSB).
Subtracting the geometric position of the observer O(et) gives
the geometric position of the target body relative to the
observer. The one-way light time, 'lt', is given by

          | T(et) - O(et) |
   lt = --------------------
                  c

The geometric relationship between the observer, target, and
solar system barycenter is as shown:



The returned position is

   T(et) - O(et)

Reception case
==============

When any of the options "LT", "CN", "LT+S", "CN+S" is selected
for `abcorr', spkpos_c computes the position of the target body at
epoch et-lt, where 'lt' is the one-way light time. Let T(t) and
O(t) represent the positions of the target and observer
relative to the solar system barycenter at time t; then 'lt' is
the solution of the light-time equation

          | T(et-lt) - O(et) |
   lt = ------------------------   (1)
                    c

The ratio

     | T(et) - O(et) |
   ---------------------   (2)
             c

is used as a first approximation to 'lt'; inserting (2) into the
right hand side of the light-time equation (1) yields the
"one-iteration" estimate of the one-way light time ("LT").
Repeating the process until the estimates of 'lt' converge yields
the "converged Newtonian" light time estimate ("CN").

Subtracting the geometric position of the observer O(et) gives
the position of the target body relative to the observer:
T(et-lt) - O(et).

   SSB ---> O(et)
    | \ |
    | \ |
    | \ | T(et-lt) - O(et)
    | \ |
    V   V V
   T(et) T(et-lt)

The light time corrected position vector is

   T(et-lt) - O(et)

If correction for stellar aberration is requested, the target
position is rotated toward the solar system
barycenter-relative velocity vector of the observer. The
rotation is computed as follows:

Let r be the light time corrected vector from the observer to the object, and v be the velocity of the observer with respect to the solar system barycenter. Let w be the angle between them. The aberration angle phi is given by

sin(phi) = v sin(w) / c

Let h be the vector given by the cross product

h = r X v

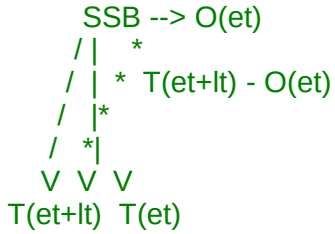
Rotate r by phi radians about h to obtain the apparent position of the object.

Transmission case
==========

When any of the options "XLT", "XCN", "XLT+S", "XCN+S" is selected, spkpos_c computes the position of the target body T at epoch et+lt, where 'lt' is the one-way light time. 'lt' is the solution of the light-time equation

$$lt = \frac{| T(et+lt) - O(et) |}{c} \qquad (3)$$

Subtracting the geometric position of the observer, O(et), gives the position of the target body relative to the observer: T(et-lt) - O(et).



The position component of the light-time corrected position is the vector

T(et+lt) - O(et)

If correction for stellar aberration is requested, the target position is rotated away from the solar system barycenter-relative velocity vector of the observer. The rotation is computed as in the reception case, but the sign of the rotation angle is negated.

Precision of light time corrections
==========

Corrections using one iteration of the light time solution

When the requested aberration correction is "LT", "LT+S", "XLT", or "XLT+S", only one iteration is performed in the algorithm used to compute 'lt'.

The relative error in this computation

```
| LT_ACTUAL - LT_COMPUTED |  /  LT_ACTUAL
```

is at most

```
   (V/C)**2
  ----------
   1 - (V/C)
```

which is well approximated by (V/C)**2, where V is the velocity of the target relative to an inertial frame and C is the speed of light.

For nearly all objects in the solar system V is less than 60 km/sec. The value of C is 300000 km/sec. Thus the one iteration solution for 'lt' has a potential relative error of not more than 4*10**-8. This is a potential light time error of approximately 2*10**-5 seconds per astronomical unit of distance separating the observer and target. Given the bound on V cited above:

> As long as the observer and target are
> separated by less than 50 astronomical units,
> the error in the light time returned using
> the one-iteration light time corrections
> is less than 1 millisecond.

Converged corrections

When the requested aberration correction is "CN", "CN+S", "XCN", or "XCN+S", three iterations are performed in the computation of 'lt'. The relative error present in this solution is at most

```
   (V/C)**4
  ----------
   1 - (V/C)
```

which is well approximated by (V/C)**4. Mathematically the precision of this computation is better than a nanosecond for any pair of objects in the solar system.

However, to model the actual light time between target and observer one must take into account effects due to general relativity. These may be as high as a few hundredths of a millisecond for some objects.

When one considers the extra time required to compute the converged Newtonian light time (the state of the target relative to the solar system barycenter is looked up three times instead of once) together with the real gain in accuracy, it seems unlikely that you will want to request either the "CN" or "CN+S" light time corrections. However, these corrections can be useful for testing situations where high precision (as opposed to accuracy) is required.

Relativistic Corrections
==========

This routine does not attempt to perform either general or special relativistic corrections in computing the various aberration corrections. For many applications relativistic corrections are not worth the expense of added computation cycles. If however, your application requires these additional corrections we suggest you consult the astronomical almanac (page B36) for a discussion of how to carry out these corrections.

-Examples

1) Load a planetary ephemeris SPK, then look up a series of geometric positions of the moon relative to the earth, referenced to the J2000 frame.

```
#include <stdio.h>
#include "SpiceUsr.h"

void main()
{

  #define       ABCORR       "NONE"
  #define       FRAME        "J2000"

  /.
  The name of the SPK file shown here is fictitious;
  you must supply the name of an SPK file available
  on your own computer system.
  ./
  #define       SPK          "planetary_spk.bsp"

  /.
  ET0 represents the date 2000 Jan 1 12:00:00 TDB.
```

```
      ./
      #define       ET0          0.0

      /.
      Use a time step of 1 hour; look up 100 states.
      ./
      #define       STEP         3600.0
      #define       MAXITR       100

      #define       OBSERVER     "earth"
      #define       TARGET       "moon"


      /.
      Local variables
      ./
      SpiceInt      i;

      SpiceDouble   et;
      SpiceDouble   lt;
      SpiceDouble   pos [3];


      /.
      Load the spk file.
      ./
      furnsh_c ( SPK );

      /.
      Step through a series of epochs, looking up a position vector
      at each one.
      ./
      for ( i = 0;  i < MAXITR;  i++ )
      {
        et  =  ET0 + i*STEP;

        spkpos_c ( TARGET,    et,   FRAME,  ABCORR,
                   OBSERVER,  pos,  &lt             );

        printf( "\net = %20.10f\n\n",                et     );
        printf( "J2000 x-position (km):   %20.10f\n", pos[0] );
        printf( "J2000 y-position (km):   %20.10f\n", pos[1] );
        printf( "J2000 z-position (km):   %20.10f\n", pos[2] );
      }
    }
```

-Restrictions

None.

-Literature_References

SPK Required Reading.

-Author_and_Institution

C.H. Acton (JPL)
B.V. Semenov (JPL)
N.J. Bachman (JPL)
W.L. Taber (JPL)

-Version

-CSPICE Version 2.0.4, 04-APR-2008 (NJB)

Corrected minor error in description of XLT+S aberration correction.

-CSPICE Version 2.0.3, 17-APR-2005 (NJB)

Error was corrected in example program: variable name `state' was changed to `pos' in printf calls.

-CSPICE Version 2.0.2, 13-OCT-2003 (EDW)

Various minor header changes were made to improve clarity. Added mention that 'lt' returns a value in seconds.

-CSPICE Version 2.0.1, 27-JUL-2003 (NJB) (CHA)

Various header corrections were made.

-CSPICE Version 2.0.0, 31-DEC-2001 (NJB)

Updated to handle aberration corrections for transmission of radiation. Formerly, only the reception case was supported. The header was revised and expanded to explain the functionality of this routine in more detail.

-CSPICE Version 1.0.0, 29-MAY-1999 (NJB) (WLT)

-Index_Entries

using names get target position relative to an observer
position relative to observer corrected for aberrations
read ephemeris data
read trajectory data

-&