Summary of Key Points

January 2020
Which Pieces of SPICE Must I Use?

- There’s not a simple answer
  - Depends on what activity or mission you are working on
  - Depends on what computation(s) you wish to make

- Don’t feel overwhelmed
  - Many seemingly complex computations can be made using just a few SPICE APIs

- The next several charts highlight some key points
  - We assume you have already looked at the major SPICE tutorials, or already have some familiarity with SPICE
  - We assume you have successfully downloaded and installed the SPICE Toolkit

- Consider printing this tutorial and keeping it near your workstation
Reminder of Key Subsystems

SPK: Position (and velocity) of things ("ephemeris objects")
PCK: Size/shape/orientation of solar system bodies
    For binary PCKs, only orientation is provided; use a text PCK to obtain size/shape
    See also DSK below
IK: Instrument field-of-view geometry (see also FK below)
CK: Orientation of spacecraft or spacecraft structures that rotate
FK: Definition and specification details for many reference frames; also includes instrument mounting alignments
DSK: High fidelity shape data, better than what’s in a text PCK
    (But limited availability)
LSK: Time conversion: UTC (SCET) ↔ ET (TDB)
SCLK and LSK: Time conversion: SCLK ↔ ET (TDB)
### Primary Kernel Interfaces - 1

#### Navigation and Ancillary Information Facility

**Which SPICE APIs are most commonly used with a given kernel type?**

<table>
<thead>
<tr>
<th>Kernel Type</th>
<th>APIs Used</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPK</td>
<td>SPKEZR, SPKPOS, SPKCOV, SPKOBJ</td>
<td></td>
</tr>
<tr>
<td>PCK</td>
<td>SXFORM, PXFORM, SPKEZR, SPKPOS, BODVRD</td>
<td></td>
</tr>
<tr>
<td>IK</td>
<td>GETFOV, G*POOL</td>
<td></td>
</tr>
<tr>
<td>CK</td>
<td>SXFORM, PXFORM SPKEZR, SPKPOS, CKCOV, CKOBJ</td>
<td></td>
</tr>
</tbody>
</table>

* Partial implementation starting with N66 Toolkits

---

Notes: FURNSH is used to load (provide access to) all SPICE kernels.

API names shown are for FORTRAN versions:
- use lower case and add an “_c” when using C
- use lower case and prepend “cspice_” when using Icy (IDL) and Mice (MATLAB)
For a given high-level Toolkit API, which kinds of kernels will or may be needed?

### Kernel Type(s) Needed

<table>
<thead>
<tr>
<th>API Name</th>
<th>SPK</th>
<th>PCK</th>
<th>IK</th>
<th>CK</th>
<th>FK</th>
<th>LSK</th>
<th>SCLK</th>
<th>DSK</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPKEZR, SPKPOS</td>
<td>Y</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>L</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>SXFORM, PXFORM</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>L</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>CKGP, CKGPAV</td>
<td>M</td>
<td>M</td>
<td>Y</td>
<td>M</td>
<td>L</td>
<td>L</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GETFOV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G*POOL</td>
<td></td>
<td>M</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STR2ET, TIMEOUT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCS2E, SCE2S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHRONOS (time conversion app.)</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>Y</td>
<td>M</td>
<td></td>
</tr>
</tbody>
</table>

- **Yes** = is needed
- **Likely** = very likely needed
- **Maybe** = may be needed
More: for a given high-level API, which kinds of kernels will or may be needed?

**Kernel Type(s) Needed**

<table>
<thead>
<tr>
<th>API Name</th>
<th>SPK</th>
<th>PCK</th>
<th>IK</th>
<th>CK</th>
<th>FK</th>
<th>LSK</th>
<th>SCLK</th>
<th>DSK*</th>
</tr>
</thead>
<tbody>
<tr>
<td>SINCPT</td>
<td>Y</td>
<td>L</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>L</td>
<td>L</td>
<td>M</td>
</tr>
<tr>
<td>DSKXV, DSKXSI</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>Y</td>
</tr>
<tr>
<td>LATSRF</td>
<td>M</td>
<td></td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>ILUMIN, ILLUMG, ILLUMF</td>
<td>Y</td>
<td>L</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>L</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>SUBPNT, SUBSLR</td>
<td>Y</td>
<td>L</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>L</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>GFOCLT, OCCULT</td>
<td>Y</td>
<td>L</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>L</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>SRFNRM</td>
<td></td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>LIMBPT</td>
<td>Y</td>
<td>L</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>L</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>TERMPT</td>
<td>Y</td>
<td>Y</td>
<td>M</td>
<td>M</td>
<td>L</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
</tbody>
</table>

* Partial implementation starting with N66 Toolkits

Yes = is needed  
Likely = likely needed  
Maybe = may be needed
Kernel “Coverage” Cautions

• Your set of kernels must:
  – contain data for all “objects” of interest
    » Sometimes you must include intermediary objects that provide a connection (recall the chaining discussion in the SPK tutorial)
  – contain data covering the time span(s) of interest to you
    » Watch out for data gaps within that time span
    » Watch out for the difference between ET and UTC
      • The difference as of 2017 January 01 is ~69.182 seconds (ET > UTC)
  – contain all the kernel types needed by SPICE to answer your question
    » As the previous charts show, you may need one or more kernels that are not obvious
  – be managed (loaded) properly if there are overlapping (competing) data within the set of files you are using
What Kernels are Available?

It depends on the mission or task you are working on.

There are typically three categories of kernel data available.

- **Mission operations** kernels – those used by the flight teams to fly the mission and prepare the archival science products
  
  » These are the most up to date, but it could be a challenge to select the ones you need

- **PDS Archived** kernels – those that have been selected from (or made from) the mission ops kernels, and then are well organized and documented for the PDS archive.
  
  » These data sets are well organized, well documented, and contain helpful “furnsh” kernels (meta-kernels).

- **Generic** kernels – those that are used by many missions and are not tied to any one mission
  
  » Relevant generic kernels are usually included in the PDS Archived and the Mission Operations kernels data sets mentioned above

- All three types can be found here: https://naif.jpl.nasa.gov/naif/data.html

The situation might be similar for non-JPL missions, but this is up to whatever institution is producing the kernels.
How Can I Find Possibly Useful Toolkit APIs?

• Review the previous charts
• Look at the appropriate SPICE tutorial(s)
• Look at the “Most Used xxx APIs” document
  …/doc/html/info/mostused.html
• Search the permuted index:
  – spicelib_idx for the FORTRAN toolkits  …/doc/html/info/spicelib_idx.html
    » This index also correlates entry point names with source code files.
  – cspice_idx for the C toolkits  …/doc/html/info/cspice_idx.html
  – icy_idx for the IDL toolkits  …/doc/html/info/icy_idx.html
  – mice_idx for the MATLAB toolkits  …/doc/html/info/mice_idx.html
• Read relevant portions of a SPICE “required reading” technical reference document (e.g. “spk.req”)
  – …/doc/html/req/spk.html  for the hyperlinked html version (best)
  – …/doc/spk.req  for the plain text version
How Can I Understand How To Use Those APIs?

The primary user-oriented documentation about each API is found in the “header” located at the top of each source code file and also in the API's HTML page in the API reference guide.

- You can “Google” an API name to see its header
  » For example: spkezr, spkezr_c, or cspice_spkezr (for Icy or Mice)
  » (More documentation is found at the additional entry points for those FORTRAN APIs that have multiple entry points.)

Reference documentation for major subsystems is found in like-named “required reading” documents (e.g. spk.req, ck.req, etc.)

The SPICE tutorials contain much helpful information.

NAIF’s self-training materials provide an orderly approach to learning about SPICE:
Does NAIF Provide Any Examples?

- Nearly all API headers contain one or more working examples
- "Most Useful SPICELIB Subroutines" has code fragments …/doc/html/info/mostused.html
- The "required reading" reference documents often contain examples …/doc/html/req/index.html
- The "Program_<language>" tutorial contains a substantial working example
- Some simple "cookbook" programs are found in the Toolkit …/src/cookbook/…
- Make use of the SPICE Programming Lessons available from the NAIF server
  - ftp://naif.jpl.nasa.gov/pub/naif/toolkit_docs/Lessons/