API Manual

Documentation for the Strand7 Application Programming Interface

FINITE ELEMENT ANALYSIS

API Manual

Documentation for the Strand7 Application Programming Interface

Release 3.1.2

Strand7 Pty Limited:

Suite 1, Level 5, 65 York Street
Sydney NSW 2000 Australia

info@strand7.com
www.strand7.com

Contents

Introduction

The Strand7 Application Programming Interface (API) allows programmers to develop applications that interact directly with Strand7, bypassing the Strand7 graphical user interface (GUI). This makes it possible to create programs that

1. generate and modify Strand7 models;
2. execute the Strand7 solvers; and
3. extract results and other information from the Strand7 model.

Data so obtained can then be used by the application for display or further processing.

The Strand7 API consists of a Dynamic Link Library (DLL) file (St7API.dll) and a number of header and include files.

The DLL contains the functions that are documented in this manual.

The header files allow external programs to communicate with St7API.dll. They define all the constants used and the function interfaces for each language supported. A different set of header files is needed for each language (e.g. Delphi, C++, Fortran, etc.). Note that in some cases, header files are even compiler product dependent – e.g., the header files for Intel Visual Fortran will be different to the header files for GNU Fortran. The languages supported are documented in the following sections. Header files for new languages/compilers are added when required – please contact us if you need header files for a different language.

The majority of this manual is devoted to describing each of the functions in the Strand7 API. The C syntax is given in the descriptions along with the input and output parameters.

The manual also lists error codes, conventions and types for property information, attributes and results, amongst other quantities.

For compiler specific information, see the *Using the Strand7 API* section.

Using the Strand7 API

The steps needed to use the Strand7 API can be summarised as follows:

1. To enable the Strand7 API for operation, it must be licensed with the Strand7 keycode. You can check if your Strand7 installation includes the API licence by investigating the **Licence Information** option on the Strand7 main menu.

2. The Strand7 API file `St7API.dll` must be located in a directory where it can be found by the calling program. This means that `St7API.dll` must be in a directory that is within the Windows search path. Alternatively, it is possible to specify where the DLL is located via the Windows API function `LOADLIBRARY`. See the Win32 API for additional information about this.

   Note that since `St7API.dll` itself loads other libraries and DLLs that come with Strand7, applications will generally not work correctly if `St7API.dll` is copied into the calling program's directory.

3. To call the functions in the API, an interface file that declares the exported function calls in `St7API.dll` is needed. This file is provided in the Strand7 API Toolkit and its name is dependent on the compiler. Examples include the following:

`St7APICall.pas`	for Delphi
`St7APICall.h`	for C/C++ and Matlab
`St7APICall.vb`	for Microsoft Visual Basic
`St7APICall.bas`	for Microsoft Visual Basic 6 and VBA
`St7API.cs`	for Microsoft Visual C#
`St7APICall.f90`	for Fortran
`St7API.py`	for Python

4. For some languages, explicit loading of `St7API.dll` is required via the Windows API call `LOADLIBRARY`. The code to do this is also provided in the Strand7 API Toolkit for the languages where it is needed. Examples include the following:

   `St7APILoad.cpp` for C++
   `St7APILoad.f90` for Compaq/Intel Visual Fortran

5. Most of the API functions employ pre-defined constants. These constants are conveniently defined within an external file in the Strand7 API Toolkit. It is not essential that you use this file, especially if you prefer to declare your arrays as 1-based instead of the 0-based approach used. The name of the constants file is dependent on the compiler. Examples include the following:

`St7APIConst.pas`	for Delphi
`St7APIConst.h`	for C/C++
`St7APIConst.vb`	for Microsoft Visual Basic
`St7APIConst.bas`	for Microsoft Visual Basic 6 and VBA
`St7API.cs`	for Microsoft Visual C#
`St7APIConst.f90`	for Fortran
`St7APIConst.m`	for Matlab
`St7API.py`	for Python

The following sections describe how each compiler can use the source/include files supplied with the Strand7 API Toolkit to create programs that use the Strand7 API.

Using the Strand7 API with Delphi

There are two Delphi include files in the API toolkit – these are `St7APICall.pas` and `St7ApiConst.pas`. An example of a declaration in `St7APICall.pas` is:

```
function St7Init():Longint; stdcall external 'St7API.dll';
```

Linking to the include files involves adding "compiler include" statements, as follows:

```
unit MainForm;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics,
  Controls, Forms, Dialogs, StdCtrls;

  {$i St7APIConst.pas}
  {$i St7APICall.pas}
.
.
```

API Strings and Delphi

The Strand7 API uses null-terminated strings. This is different to the so-called Delphi short string. You should not pass short strings to Strand7 API functions. A null-terminated string can be passed as either a packed array of `AnsiChar` or as a `PAnsiChar`. As shipped, `St7APICall.pas` uses the type `CharString = packed array[0..kMaxStrLen] of AnsiChar`, defined in `St7APIConst.pas`. However, you could replace this with `CharString = PAnsiChar` if you prefer.

API Arrays and Delphi

Many Strand7 API functions use arrays of longint or double as parameters. These are always passed by reference. In most cases, when an array is passed to a Strand7 API function using the Delphi interface, the array type is specified. For example, the array for node coordinates is defined as `Array3Double = array[0..2]` of double. As Delphi allows you to bypass Pascal's strong type rules, you could redefine the function:

```
St7GetNodeXYZ(uID:Longint; NodeNum:Longint; var XYZ:Array3Doubles):Longint;
```

as:

```
St7GetNodeXYZ(uID:Longint; NodeNum:Longint; var XYZ):Longint;
```

This would then allow you to pass anything to the function for the XYZ variable. However, this would increase the possibility of programming errors because the compiler can no longer detect type conflicts.

As mentioned above, most of the function definitions in St7APICall.pas are typed. There are some exceptions, e.g. the function:

```
St7SetResFileBeamResult(uID:Longint; CaseNum,Beam,Quantity:Longint; var
Doubles):Longint;
```

This is generally done for functions that require arrays of variable lengths. Of course you can change this declaration if you prefer the full type checking offered by Pascal.

Using the Strand7 API with C++

There are two header/include files and one source file included in the Strand7 API Toolkit – these are `St7APICall.h`, `St7APIConst.h` and `St7APILoad.cpp`. To use these files include the two header files and add `St7APILoad.cpp` to your project.

```
#include “St7APIConst.h”
#include “St7APICall.h”
```

`St7APILoad.cpp` includes two functions. These functions are `LoadSt7API` and `FreeSt7API`, to load and free the DLL respectively. These must be called by your program to load the DLL for use and then to unload it after use. `LoadSt7API` must be called before the call to `St7Init`. An example of part of `LoadSt7API` is as follows:

```
HMODULE hDLL;

bool LoadSt7API()
{
  hSt7API=LoadLibrary("St7API.dll");
  if (hSt7API!=NULL)
  {
    St7Init=(St7InitType)GetProcAddress(hSt7API,"St7Init");
    .
    .
```

API Strings and C++

The Strand7 API uses null-terminated strings. These are always declared as `char*` in the normal C++ convention.

API Arrays and C++

Many Strand7 API functions use arrays of longint or double as parameters. These are always passed by reference and declared as `double*` or `long*`.

Using the Strand7 API with Visual Basic 6 and VBA

There are two source files included in the API Toolkit – these are `St7APICall.bas` and `St7APIConst.bas`. To use these files add them to your project.

API Strings

The Strand7 API uses null-terminated strings. These are always declared as `ByVal StringName As String`. Note that as Visual Basic strings will be declared as a fixed length array, e.g. `Dim FileName As String * 255`, an API call returning the string will null-terminate the string via the CHAR=`0` at some point. All character values beyond this point will be undefined.

API Arrays

Many Strand7 API functions use arrays of longint or double as parameters. These are always passed by reference and declared as `ByRef LongArray As Long` or `ByRef DoubleArray As Double`. The array passing syntax `LongArray() As Long` or `DoubleArray() As Double` should not be used with the Strand7 API. The arrays to be passed should be declared as `Dim LongArray(n) As Long` or `Dim DoubleArray(n) As Double`, where `n` is some integer value. When passing these arrays to a Strand7 API function it is essential that the first index of the array be passed. The following example further illustrates the correct procedure:

```
function declaration:
Declare Function St7GetNodeXYZ& Lib "St7API.DLL"(ByVal uID As Long, ByVal NodeNum
As Long, ByRef XYZ As Double)

variable declaration:
Dim XYZ(2) As Double

function call:
ErrorCode = St7GetNodeXYZ(1, NodeNumber, XYZ(0))
```

API Boolean

Many Strand7 API functions use `boolean` or arrays of `boolean` as parameters. These should always be passed as `Byte`, (both by value and by reference). This is necessary because the Strand7 API uses single byte boolean representation, which is compatible with the `Byte` type. The `Boolean` type is two bytes long, therefore not compatible. True boolean values will therefore be represented by `Byte=1` and False boolean values will be represented by `Byte=0`.

Using the Strand7 API with Visual Basic

There are two source files included in the API Toolkit – these are `St7APICall.vb` and `St7APIConst.vb`. To use these files add them to your project.

API Strings and Visual Basic

The Strand7 API uses null-terminated strings. Strings that are passed to the API are declared as `ByVal StringName As String` and strings that are returned from the API are declared as `ByVal StringName As StringBuilder`. When you pass a string to the API, Visual Basic will ensure that the string is null-terminated. When you wish to retrieve a string from the API you will need to pass a StringBuilder object with a pre-allocated buffer. The returned string can be retrieved from the StringBuilder object using the

`StringBuilder.ToString()` method, which will copy the retrieved characters up until the terminating null character. The following example further illustrates the correct procedure for retrieving a string:

```
Dim sb As New StringBuilder(kMaxStrLen)
Dim errorstring as String

Call St7GetAPIErrorString(ERR7_FileNotFound, sb, sb.Capacity)
errorstring = sb.ToString()
```

API Arrays and Visual Basic

Many Strand7 API functions use arrays of longint or double as parameters. These are always passed by reference and declared as `ByRef LongArray As Long` or `ByRef DoubleArray As Double`. The array passing syntax `LongArray() As Long` or `DoubleArray() As Double` should not be used with the Strand7 API. The arrays to be passed should be declared as `Dim LongArray(n) As Long` or `Dim DoubleArray(n) As Double`, where `n` is some integer value. When passing these arrays to a Strand7 API function via Visual Basic, it is essential that the first index of the array be passed. The following example further illustrates the correct procedure:

```
function declaration:
Declare Function St7GetNodeXYZ& Lib "St7API.DLL"(ByVal uID As Long, ByVal NodeNum
As Long, ByRef XYZ As Double)

variable declaration:
Dim XYZ(2) As Double

function call:
ErrorCode = St7GetNodeXYZ(1, NodeNumber, XYZ(0))
```

API Boolean and Visual Basic

Many Strand7 API functions use `boolean` or arrays of `boolean` as parameters. These should always be passed as `Byte` in Visual Basic, (both by value and by reference). This is necessary because the Strand7 API uses single byte boolean representation, which is compatible with the Visual Basic `Byte` type. The Visual Basic `Boolean` type is two bytes long, therefore not compatible. True boolean values will therefore be represented by `Byte=1` and False boolean values will be represented by `Byte=0`.

Using the Strand7 API with Visual C#

There is one source file included in the API Toolkit – this is `St7API.cs`. To use this file add it to your project.

The API functions and constants are declared within a static class called `St7`. When calling the API functions and using the API constants it is necessary to prefix the function or constant name with the `St7` class name followed by a period character.

API Strings and Visual C#

The Strand7 API uses null-terminated strings. Strings that are passed to the API are declared as `string StringName` and strings that are returned from the API are declared as `StringBuilder StringName`. When you pass a string to the API, C# will ensure that the string is null-terminated. When you wish to retrieve a string from the API you will need to pass a `StringBuilder` object with a pre-allocated buffer. The returned string can be retrieved from the `StringBuilder` object using the `StringBuilder.ToString()` method, which will copy the retrieved characters up until the terminating null character. The following example further illustrates the correct procedure for retrieving a string:

```
StringBuilder sb = new StringBuilder(St7.kMaxStrLen);
string errorstring;

St7.St7GetAPIErrorString(St7.ERR7_FileNotFound, sb, sb.Capacity);
errorstring = sb.ToString();
```

API Arrays and Visual C#

Many Strand7 API functions use arrays of longint or double as parameters. These are always passed by reference and should be declared as `double[] DoubleArray = new double[n]` or `int[] IntegerArray = new int[n]`, where n is some integer value.

Using the Strand7 API with Visual Fortran

There are three source files included in the API Toolkit – these are `St7APICall.f90`, `St7APIConst.f90` and `St7APILoad.f90`. To use these files add them to your project and insert USE statements at the top of each subroutine that uses the API.

```
USE St7APICall
USE St7APIConst
```

API Strings and Visual Fortran

The Strand7 API uses null-terminated strings. These are always declared as `CHARACTER(LEN=*)` in the interface section (`St7APICall.f90`), and are passed by reference. Strings will be declared in your program as `CHARACTER(LEN=255)`. An API call returning the string will null-terminate the string with `CHAR=0` at some point. All character values beyond this point will be undefined.

API Arrays and Visual Fortran

Many Strand7 API functions use arrays of longint or double as parameters. These are always passed by reference and should be declared as

```
REAL(8) :: DOUBLEARRAY(n)
INTEGER(4) :: INTEGERARRAY(n)
```

where `n` is some integer value.

API Boolean and Visual Fortran

Many Strand7 API functions use boolean or arrays of boolean as parameters. These should always be declared as `LOGICAL(1)` in Visual Fortran. This is necessary because the Strand7 API uses single byte boolean representation, whereas the Visual Fortran `LOGICAL` type can be up to four bytes long and therefore not compatible.

Using the Strand7 API with GNU Fortran

There are three source files included in the API Toolkit – these are `St7APICall.f90`, `St7APIConst.f90` and `St7APILoad.f90`. To use these files insert USE statements at the top of each subroutine that uses the API.

```
USE St7APICall
USE St7APIConst
USE St7APILoad
```

API Strings and GNU Fortran

The Strand7 API uses null-terminated strings. These are always declared as

```
CHARACTER(KIND=C_CHAR) :: CHARARRAY(*)
```

in the interface section (`St7APICall.f90`), and are passed by reference. Strings will be declared in your program as `CHARACTER(LEN=255)`. An API call returning the string will null-terminate the string with `CHAR=0` at some point. All character values beyond this point will be undefined.

API Arrays and GNU Fortran

Many Strand7 API functions use arrays of longint or double as parameters. These are always passed by reference and should be declared as;

```
REAL(8) :: DOUBLEARRAY(n)
INTEGER(4) :: INTEGERARRAY(n)
```

where n is some integer value.

API Boolean and GNU Fortran

Many Strand7 API functions use boolean or arrays of boolean as parameters. These should always be declared as `LOGICAL(1)` in GNU Fortran. This is necessary because the Strand7 API uses single byte boolean representation, which is compatible with the GNU Fortran `LOGICAL(1)` type. The GNU Fortran `LOGICAL` type can be up to four bytes long and therefore not compatible.

Using the Strand7 API with Matlab®

There are two header/include files included in the API toolkit – these are `St7APICall.h` and `St7APIConst.m`.

Matlab includes a series of built-in functions that can be used to load and manipulate the `St7API.dll`. See `loadlibrary`, `libisloaded` and `unloadlibrary` within Matlab for additional information.

Due to Matlab's interpreted operation, all API calls must be made via the `calllib` Matlab built-in function. The complete list of API function arguments is passed into `calllib` on the right-hand side, but just the API function's error return and list of pointer arguments is assigned on the left-hand side, for example:

```
XYZ = zeros(3, 1);
[iErr, XYZ] = calllib('St7API', 'St7GetNodeXYZ', uID,  NodeNum, XYZ);
```

The variable `iErr` is the integer error return from `St7GetNodeXYZ`, and the array XYZ is a pointer argument in the function's argument list (in C notation, this is denoted by an asterisk in the argument list – `long*`, `double*`, `bool*` or `char*`). The list of pointer arguments on the left-hand side must be in the same order as the right-hand side, and contains both input and output pointer arguments. If a pointer argument is assigned an output value by the Strand7 API, then it must be allocated before the call to `calllib`. Note that text strings are also pointer arguments, even when they are passed into the API function.

It is possible to pass in dummy variables for the output arguments listed on the right-hand side. These values are never actually referenced or assigned and exist only so that `calllib` can match the number of variables. The list of pointer arguments on the left-hand side may also be truncated, but it must be complete up to the last listed argument.

See `calllib` within Matlab for additional information.

Using the Strand7 API with Python

Both constants and function definitions are in the module `St7API.py`; the module should be placed in a directory in the Python path (for example `C:\Python35\Lib\site-packages`) so that it can be used without being copied to the directory of each new project. The module is for 64-bit Python versions and can be used with Python 2.6 upwards, including Python 3.

For Python 3.8 upwards, the Windows search path is no longer used to locate St7API.dll. To allow Python to locate the DLL, specify the full path to St7API.dll by modifying the LoadLibrary line in St7API.py. Alternatively, you can call os.add_dll_directory() to locate St7API.dll prior to importing the St7API module.

The module is loaded using

```
import St7API
```

after which functions and constants can be accessed using the prefix `St7API`, for example,

```
St7API.St7Init()
```

The prefix is omitted if the contents of the module is imported into the current namespace, for example,

```
from St7API import *
St7Init()
```

Types from `ctypes` are used for input and output with the Strand7 API. Input arguments of type integer, double, boolean and string (bytes in Python 3) are cast into the appropriate type, including conversion to pointers. Output arguments must be declared explicitly using `ctypes` constructors since the native python types are immutable. For example, if a function has output argument of type double it should be declared as `ctypes.c_double()`.

Arrays in Python

Python lists must be converted to `ctypes` arrays before passing as arguments to Strand7 API calls. A type for an array of length n can be created using the syntax `arrayType = singularType * n`.

For example,

```
unitsArray = ctypes.c_int * St7API.kLastUnit
units = unitsArray()
```

creates an array of integers 6 elements long suitable for passing to functions *St7GetUnits*, *St7SetUnits* and *St7ConvertUnits*.

Arrays from `ctypes` are indexed in the same manner as python lists, for example

```
units[St7API.ipSTRESSU] = St7API.suMEGAPASCAL
```

Using the Strand7 API with Grasshopper®

Grasshopper provides three different scripting components that can be used to interface with the Strand7 API. The components are for Python, C# and VB.NET languages. They can be found on the maths tab script panel and are called GhPython script, C# script and VB script, respectively.

GhPython

To use the Strand7 API in a GhPython script component, the St7API.py Python module needs to be added to the Rhino® Python library folder. The default location is the …\Plug-ins\IronPython\Lib folder of the Rhino installation. The Strand7 Python API headers are based on the Python language specification and make use of the ctypes library. However, there are different requirements in Rhino 6 compared with Rhino 7.

GhPython with Rhino 6

Since the type checking behaviour of ctypes as implemented by GhPython in Rhino 6 differs from the language specification, code modifications may be required. In particular, it is necessary to explicitly convert from types such as ‘Array of 12 Doubles’ to the more generic ‘Pointer to Double’ (for example).

A convenient way of performing the required conversions is to define pointer conversion functions once for each data type, then use these functions when invoking Strand7 API calls with arrays as arguments. For example,

```
# Define pointer conversion functions (only needs to be done once).
PI = ctypes.POINTER(ctypes.c_long)
PD = ctypes.POINTER(ctypes.c_double)
PB = ctypes.POINTER(ctypes.c_bool)

# Cast arrays into pointers at each call.
XYZ = (ctypes.c_double * 3)()
XYZ[:] = [1.2, 3.4, 5.6]
St7API.St7SetNodeXYZ(uID, iNode, PD(XYZ))
```

GhPython with Rhino 7

No explicit conversion of data types is required, and the Strand7 Python interface can be used as described in *Using the Strand7 API with Python*.

C# and VB.NET

For C# and VB.NET languages, a class library, contained in the file St7API_DotNet.dll, is provided to interface with the Strand7 API. This library is a replacement for include files, and needs to be added to the referenced assemblies of all .NET script components that use the Strand7 API. The library does not replace the Strand7 API; Strand7 still needs to be correctly configured and licensed, and St7API.dll needs to be in the Windows search path.

To add the library to a Grasshopper script component, right click the component and select Manage Assemblies... from the right-click menu. On the Referenced Assemblies page, click Add and select the St7API_DotNet.dll file.

The library is compiled from the C# St7API.cs file, and therefore Strand7 API functions and constants are called in the same way as they would be called from a regular C# program. The functions and constants are all members of a static class called St7, defined within a namespace called St7API.

Initialisation

St7SetIconSize

Sets the size of the icons to be used in the model and solver windows. This function must be called before *St7Init,* otherwise it will return an error code.

```
long St7SetIconSize(long IconSize)
```

Input Parameters

`IconSize`

0 – Strand7 selects a size automatically.

1..5 – User specified icon size; 1 is a 16x16 pixel icon with each successive size increasing the icon by 8 pixels (i.e. 24x24, 32x32, etc.)

St7GetIconSize

Returns the size of the icons used in the model and solver windows.

```
long St7GetIconSize(long* IconSize)
```

Output Parameters

`IconSize`

0 – Strand7 selects a size automatically.

1..5 – User specified icon size; 1 is a 16x16 pixel icon with each successive size increasing the icon by 8 pixels (i.e. 16x16, 24x24, 32x32, 40x40 and 48x48)

St7GetAPIPath

Returns the full path name to the directory that contains the Strand7 API that is currently loaded.

```
long St7GetAPIPath(char* St7Path, long MaxStringLen)
```

Input Parameters

`MaxStringLen`

Maximum number of characters allocated for St7Path.

Output Parameters

`St7Path`

Full path name to the ..\Strand7\Bin directory that contains the St7API.dll that is currently loaded.

St7Init

Initialises the Strand7 API DLL. This function must be called before calls are made to any API functions other than *St7SetIconSize*, *St7GetIconSize*, *St7GetAPIPath*, *St7SetLicenceOptions*, *St7GetLicenceOptions*, *St7Version, St7BuildString*, *St7GetListSeparatorCode, St7GetDecimalSeparatorCode* or *St7GetMaxModelFileID*.

```
long St7Init()
```

St7Release

Releases the Strand7 API DLL and unloads the Strand7 licence manager freeing any active licences. *St7Init* must again be called before API functions can be executed again (other than *St7SetIconSize*, *St7GetIconSize*, *St7GetAPIPath*, *St7SetLicenceOptions*, *St7GetLicenceOptions*, *St7Version, St7BuildString*, *St7GetListSeparatorCode, St7GetDecimalSeparatorCode* or *St7GetMaxModelFileID*).

```
long St7Release()
```

St7SetLicenceOptions

Sets how the API responds when it suffers a loss of licence.

```
long St7SetLicenceOptions(long Mode, long MaxRetry, long RetryPause)
```

Input Parameters

`Mode`

lmMessageBox – display a dialog with Retry and Abort buttons. The API will automatically retry every RetryPause seconds.
lmWaitRetry – silently attempt to obtain a licence MaxRetry times every RetryPause seconds. If the licence is not obtained, an error code is returned.
lmAbort – returns a licence related error code without retrying.

`MaxRetry`

Number of times to automatically retry to obtain a licence. This option applies to lmWaitRetry mode.

`RetryPause`

Duration in seconds to wait between automatic retry attempts. This option applies to lmMessageBox and lmWaitRetry modes.

St7GetLicenceOptions

Returns how the API responds when it suffers a loss of licence.

```
long St7GetLicenceOptions(long* Mode, long* MaxRetry, long* RetryPause)
```

Output Parameters

`Mode`

lmMessageBox – display a dialog with Retry and Abort buttons. The API will automatically retry every RetryPause seconds.
lmWaitRetry – silently attempt to obtain a licence MaxRetry times every RetryPause seconds. If the

licence is not obtained, an error code is returned.
lmAbort – returns a licence related error code without retrying.

`MaxRetry`

Number of times to automatically retry to obtain a licence. This option applies to lmWaitRetry mode.

`RetryPause`

Duration in seconds to wait between automatic retry attempts. This option applies to lmMessageBox and lmWaitRetry modes.

St7Version

Returns the version information for the Strand7 API DLL that is currently loaded.

```
long St7Version(long* Major, long* Minor, long* Point)
```

Output Parameters

`Major`

Major version number A in the A.B.C format.

`Minor`

Minor version number B in the A.B.C format.

`Point`

Point version number C in the A.B.C format.

St7BuildString

Returns the build information for the Strand7 API DLL that is currently loaded.

```
long St7BuildString(char* BuildString, long MaxStringLen)
```

Input Parameters

`MaxStringLen`

The maximum number of characters allocated for BuildString.

Output Parameters

`BuildString`

A text string identifying the Strand7 API DLL build.

St7GetMaxModelFileID

Returns the maximum model file ID. All of the functions that operate on a model file, reference the model file via the uID parameter in the function's parameter list. Multiple model files can be open simultaneously, each one referenced by a different model file ID (for example, see *St7OpenFile*).

```
long St7GetMaxModelFileID(long* MaxID)
```

Output Parameters

`MaxID`

Maximum model file ID. A valid uID is in the range 1 to MaxID.

St7GetListSeparatorCode

Returns the code point for the system's list delimiter character.

```
long St7GetListSeparatorCode(long* Code)
```

Output Parameters

`Code`

Locale dependent list delimiter character code point. For example, 44 for a comma.

St7GetDecimalSeparatorCode

Returns the code point for the system's decimal character.

```
long St7GetDecimalSeparatorCode(long* Code)
```

Output Parameters

`Code`

Locale dependent decimal character code point. For example, 46 for a period.

St7GetLastError

Returns the error code generated by the last Strand7 API call.

```
long St7GetLastError()
```

St7GetLastOpenFileCode

Returns the error code generated by the last call to *St7OpenFile* or *St7OpenFileReadOnly*.

```
long St7GetLastOpenFileCode()
```

The function returns a long integer packed with logical bit flags at the following addresses:

`[ibFileNameTooLongOrInvalid]` – File name is too long or contains invalid characters.

`[ibFileSharingError]` – File is locked by another process.

`[ibFileCantRead]` – File could not be read.

`[ibFileNotFound]` – File could not be found.

`[ibFileInvalidData]` – File contains invalid data.

`[ibFileTruncated]` – File length is less than expected.

`[ibFileIsBXS]` – File is a BXS file.

`[ibFileIsNotSt7]` – File is not an ST7 file.

`[ibInsufficientFreeSpace]` – Insufficient free space in ScratchPath location to create file cache.

St7GetLastSaveFileCode

Returns the error code generated by the last call to *St7SaveFile*, *St7SaveFileCopy*, *St7SaveDeformedCopy*, *St7SaveViewOnlyCopy*, *St7SaveSubModel* or *St7SaveBeamSectionMesh*.

```
long St7GetLastSaveFileCode()
```

The function returns a long integer packed with logical bit flags at the following addresses:

`[ibFileNameTooLongOrInvalid]` – File name is too long or contains invalid characters.

`[ibFileSharingError]` – File is locked by another process.

`[ibFileCantWrite]` – File could not be written.

`[ibInsufficientFreeSpace]` – Insufficient free space to save the file.

`[ibFileReadOnly]` – The file is a read-only file.

St7GetAPIErrorString

Returns the error message corresponding to the specified Strand7 API error code. Error codes corresponding to a Strand7 solver error should be processed using *St7GetSolverErrorString*.

```
long St7GetAPIErrorString(long iErr, char* ErrorString, long MaxStringLen)
```

Input Parameters

`iErr`

Strand7 API error code.

`MaxStringLen`

Maximum number of characters allocated for ErrorString.

Output Parameters

`ErrorString`

Error message string corresponding to iErr.

St7GetSolverErrorString

Returns the error message corresponding to the specified Strand7 solver error code. Error codes corresponding to a Strand7 API error should be processed using *St7GetLastOpenFileCode*.

```
long St7GetSolverErrorString(long iErr, char* ErrorString, long MaxStringLen)
```

Input Parameters

`iErr`

Strand7 solver error code.

`MaxStringLen`

Maximum number of characters allocated for ErrorString.

Output Parameters

`ErrorString`

Error message string corresponding to iErr.

St7GetDisplayOptionsPath

Returns the full path name of the display options file.

```
long St7GetDisplayOptionsPath(char* ConfigPath, long MaxStringLen)
```

Input Parameters

`MaxStringLen`

The maximum number of characters allocated for ConfigPath.

Output Parameters

`ConfigPath`

Full path name of the display options file.

St7SetDisplayOptionsPath

Sets the full path to the display options file. If only the directory is given, then Settings.cfg3 will be used to control display behaviour. The display options are only used to define settings for model files created subsequently to this call.

```
long St7SetDisplayOptionsPath(char* ConfigPath)
```

Input Parameters

`ConfigPath`

Full path name of the display options file.

File Management

St7FileVersion

Returns the version of Strand7 used to save the specified file.

```
long St7FileVersion(char* FileName, long* Major, long* Minor, long* Point)
```

Input Parameters

`FileName`

Full path and filename for the Strand7 model.

Output Parameters

`Major`

Major version number A in the A.B.C format.

`Minor`

Minor version number B in the A.B.C format.

`Point`

Point version number C in the A.B.C format.

St7OpenFile

Opens an existing Strand7 model file with an exclusive lock. This call is required before any data can be read from, or written to, a Strand7 model file. Multiple model files can be opened simultaneously by using a different model file ID for each file (see *St7GetMaxModelFileID*).

```
long St7OpenFile(long uID, char* FileName, char* ScratchPath)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and filename for the Strand7 model.

`ScratchPath`

A valid path to be used for temporary storage.

St7OpenFileReadOnly

Opens a Strand7 model file in read-only mode; the file cannot be modified while open in this mode. This call is required before any data can be read from a Strand7 model file. Multiple model files can be opened simultaneously by using a different model file ID for each file (see *St7GetMaxModelFileID*).

```
long St7OpenFileReadOnly(long uID, char* FileName, char* ScratchPath)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and filename for the Strand7 model.

`ScratchPath`

A valid path to be used for temporary storage.

St7CloseFile

Closes an open Strand7 model file. All associated scratch files that may have been created are automatically deleted.

```
long St7CloseFile(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7NewFile

Creates and opens a new Strand7 model file. Multiple model files can be opened simultaneously by using a different model file ID for each file (see *St7GetMaxModelFileID*). Note that if a file of the same name exists, the existing file will stay open and will not be overwritten until the new file is saved.

```
long St7NewFile(long uID, char* FileName, char* ScratchPath)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and filename for the Strand7 model.

`ScratchPath`

A valid path to be used for temporary storage.

St7SaveFile

Saves a Strand7 model file. The file remains open after the call. This function cannot be called if the file has open result files associated with it.

```
long St7SaveFile(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7SaveFileCopy

Saves a copy of a Strand7 model to a new file. The file remains open after the call. This function cannot be called if the file has open result files associated with it.

```
long St7SaveFileCopy(long uID, char* FileName)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and filename for the Strand7 model.

St7SaveDeformedCopy

Saves a copy of a Strand7 model to a new file with the current node displacements added to the original node coordinates to produce a new model with a deformed shape. The model results file must be open prior to calling this function.

```
long St7SaveDeformedCopy(long uID, char* FileName, long ResultCase,
     double DispScale, long ScaleType)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and filename for the new Strand7 model.

`ResultCase`

The result case from which the deformed shape is taken.

`DispScale`

The displacement scale of the deformed shape.

`ScaleType`

The type of scale to apply; either dsPercent or dsAbsolute.

Applicability

Applicable to stLinearStatic, stLinearBuckling, stLoadInfluence, stNonlinearStatic, stQuasiStatic, stNaturalFrequency, stHarmonicResponse, stSpectralResponse, stLinearTransientDynamic and stNonlinearTransientDynamic.

St7SaveViewOnlyCopy

Saves a copy of the Strand7 model as a new View-Only file (.St7V). The function offers controls to allow or restrict access to specific aspects of the model data when subsequently opening the View-Only file.

```
long St7SaveViewOnlyCopy(long uID, char* FileName, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and filename for the Strand7 model.

`Integers[0..15]`

`[ipVoShowCoordinates]` – btTrue to show node coordinate values in Entity Inspector, Whiteboard, TEXT tab, Peek Window and LISTINGS tab.

`[ipVoShowTEXT]` – btTrue to allow access to the TEXT tab.

`[ipVoShowCASES]` – btTrue to allow access to the CASES tab.

`[ipVoAllowSave]` – btTrue to allow the file to be saved to enable view changes and other display settings to be updated in the file.

`[ipVoShowTables]` – btTrue to allow access to the LAYOUTS/Tables tab.

`[ipVoShowPlies]` – btTrue to allow access to the LAYOUTS/Plies tab.

`[ipVoShowLaminates]` – btTrue to allow access to the LAYOUTS/Laminates tab.

`[ipVoShowPlateRC]` – btTrue to allow access to the LAYOUTS/Plate RC tab.

`[ipVoShowCreep]` – btTrue to allow access to the LAYOUTS/Creep tab.

`[ipVoShowPaths]` – btTrue to allow access to the LAYOUTS/Paths tab.

`[ipVoShowCavities]` – btTrue to allow access to the LAYOUTS/Cavities tab.

`[ipVoShowProperties]` – btTrue to allow access to VISUAL/Global/Properties.

`[ipVoShowLISTINGS]` – btTrue to allow access to the LISTINGS tab.

`[ipVoShowAttribSummary]` – btTrue to allow access to the SUMMARY/Attributes tab.

`[ipVoShowPropSummary]` – btTrue to allow access to the SUMMARY/Property tab.

`[ipVoShowModelSummary]` – btTrue to allow access to the SUMMARY/Model tab.

St7SaveSubModel

Saves selected entities as a new Strand7 file.

```
long St7SaveSubModel(long uID, char* FileName)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and filename for the Strand7 model.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

Usage

If a result file is open, a set of enforced displacements is applied to all nodes that lie on the boundary between selected and unselected elements. The displacement scale setting has no effect on the node coordinates stored in the sub-model.

St7ValidateResultFile

Checks that a result file associated with the specified Strand7 model may be opened. All supported result file types may be validated.

```
long St7ValidateResultFile(long uID, char* FileName, long* ValidationCode,
     long* Solver)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and filename for the Strand7 result file.

Output Parameters

`ValidationCode`

A long integer packed with logical bit flags at the following addresses:

`[ibResFileNotFound]` – FileName not found.

`[ibResFileCannotOpen]` – FileName could not be opened; check read access.

`[ibResFileNotResultFile]` – FileName is not a Strand7 result file.

`[ibResFileOldVersion]` – FileName was generated by an older version of Strand7.

`[ibResFileFutureVersion]` – FileName was generated by a newer version of Strand7.

`[ibResFileWrongNumNodes]` – FileName references more/less nodes than defined by the model opened as uID.

`[ibResFileWrongNumBeams]` – FileName references more/less beams than defined in model opened as uID.

`[ibResFileWrongNumPlates]` – FileName references more/less plates than defined in model opened as uID.

`[ibResFileWrongNumBricks]` – FileName references more/less bricks than defined in model opened as uID.

`[ibResFileWrongModelID]` – Internal FileName ID conflicts with the model internal ID; results may be incompatible.

`[ibResFileUnknownError]` – Unknown error encountered opening FileName.

`[ibResFileIsCombination]` – FileName is a combination file.

`[ibResFileIsMultiFile]` – FileName has not been created directly by a solver process.

`[ibResFileTruncated]` – FileName contains fewer result cases than its header stipulates.

Values are bit = 1 for true and bit = 0 for false.

`Solver`

One of the solver types listed in *Solver Types*.

St7SetResultFileOpenFlag

Shows or hides certain types of result cases when opening a result file.

```
long St7SetResultFileOpenFlag(long uID, long Index, bool State)
```

Input Parameters

`uID`

Strand7 model file ID.

`Index`

The types of result cases to show or hide; one of ipHideUnconvergedLBA, ipHideNegativeLBA, ipHideUnconvergedNFA, ipHideZeroNFA, ipHideModalSRA, ipHideUnconvergedNLA, ipHideSubStepNLA, ipHideUnconvergedNTA, ipHideSubStepNTA, ipHideUnconvergedQSA or ipHideSubStepQSA.

`State`

True sets the index and therefore hides the result cases of the specified type.

False clears the index and therefore shows the result cases of the specified type.

St7GetResultFileOpenFlag

Returns show/hide state of certain types of result cases when opening a result file.

```
long St7GetResultFileOpenFlag(long uID, long Index, bool* State)
```

Input Parameters

`uID`

Strand7 model file ID.

`Index`

The types of result cases to show or hide; one of ipHideUnconvergedLBA, ipHideNegativeLBA, ipHideUnconvergedNFA, ipHideZeroNFA, ipHideModalSRA, ipHideUnconvergedNLA, ipHideSubStepNLA, ipHideUnconvergedNTA, ipHideSubStepNTA, ipHideUnconvergedQSA or ipHideSubStepQSA.

Output Parameters

`State`

True if the index is set, and therefore the result cases of the specified type are hidden.

St7SetNFAFileOpenMinMass

Sets the mass participation threshold for result cases of Natrual Frequency results.

```
long St7SetNFAFileOpenMinMass(long uID, double Mass)
```

Input Parameters

`uID`

Strand7 model file ID.

`Mass`

Minimum mass participation (percentage between 0.0 and 50.0) for a result case to be available when natural frequency results are opened.

St7GetNFAFileOpenMinMass

Returns the mass participation threshold for result cases of Natrual Frequency results.

```
long St7GetNFAFileOpenMinMass(long uID, double* Mass)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Mass`

Minimum mass participation (percentage) for a result case to be available when natural frequency results are opened.

St7OpenResultFile

Opens a result file associated with the specified Strand7 model. All supported result file types may be opened.

```
long St7OpenResultFile(long uID, char* FileName, char* SpectralName,
     long CombinationCode, long* NumPrimary, long* NumSecondary)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and filename for the Strand7 result file.

`SpectralName`

Full path and filename for the spectral result file to be combined. A null string may be passed to combine with the default spectral file as defined by the user in the Strand7 model.

`CombinationCode`

kNoCombinations – no combinations.
kGenerateNewCombinations – generate new combinations.
kUseExistingCombinations – open previously saved combinations if an valid .LSC file exists, otherwise, generate the combinations.

Note that result envelopes are not calculated – use *St7GenerateEnvelopes* for these.

Output Parameters

`NumPrimary`

Number of primary result cases available.

`NumSecondary`

Number of secondary result cases (excluding envelopes). If they were not generated at the time the results were opened (that is, Combinations was set to False), they can be generated when they are required by calling *St7GenerateLSACombinations*.

Dependencies

`Hidden Results`

Result cases may be hidden according to the settings of *St7SetResultFileOpenFlag* and *St7SetNFAFileOpenMinMass*.

St7GenerateLSACombinations

Generate the secondary result cases for the linear load case combinations in a Strand7 model. The result file must be open.

```
long St7GenerateLSACombinations(long uID, long* NumSecondary, long* WarningCode)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumSecondary`

Number of secondary result cases available.

`WarningCode`

Either wcLSACombineNoWarning if the operation was successful, or wcLSACombineInvalidSRA if the combinations were generated but the specified spectral file was invalid.

St7GenerateEnvelopes

Generate the secondary result cases for the result envelopes specified in the Strand7 model. The result file must be open.

```
long St7GenerateEnvelopes(long uID, long* NumLimitEnvelopes,
     long* NumCombinationEnvelopes, long* NumFactorsEnvelopes)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumLimitEnvelopes`

Number of limit envelope results cases available.

`NumCombinationEnvelopes`

Number of combination envelope results cases available.

`NumFactorsEnvelopes`

Number of factors envelope results cases available.

St7CloseResultFile

Closes any open result file associated with the specified Strand7 model.

```
long St7CloseResultFile(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

Entity Selection

St7SetEntitySelectState

Sets the selected state of the specified entity.

```
long St7SetEntitySelectState(long uID, long Entity, long EntityNum,
     long EndEdgeFace, bool Selected)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyLINK, tyVERTEX, tyGEOMETRYEDGE, tyGEOMETRYCOEDGE, tyGEOMETRYLOOP, tyGEOMETRYFACE or tyLOADPATH.

`EntityNum`

Entity number.

`EndEdgeFace`

Local entity number; either 1 or 2 for tyBEAM; one of 1, 2, 3 or 4 for tyPLATE; or one of 1, 2, 3, 4, 5 or 6 for tyBRICK. Use 0 to select tyNODE or to select an entire tyBEAM, tyPLATE or tyBRICK.

`Selected`

Either True or False.

St7GetEntitySelectState

Returns the select state of the specified entity.

```
long St7GetEntitySelectState(long uID, long Entity, long EntityNum,
     long EndEdgeFace, bool* Selected)
```

Input Parameters

`UID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyLINK, tyVERTEX, tyGEOMETRYEDGE, tyGEOMETRYCOEDGE, tyGEOMETRYLOOP, tyGEOMETRYFACE or tyLOADPATH.

`EntityNum`

Entity number.

`EndEdgeFace`

Local entity number; either 1 or 2 for tyBEAM; one of 1, 2, 3 or 4 for tyPLATE; or one of 1, 2, 3, 4, 5 or 6 for tyBRICK. Use 0 to check the state of tyNODE or the state of and entire tyBEAM, tyPLATE or tyBRICK.

Output Parameters

`Selected`

Either True or False.

St7SetBrickSelectState

Sets the selected state of the specified brick, brick face or edge on a brick face.

```
long St7SetBrickSelectState(long uID, long BrickNum, long FaceNum, long EdgeNum,
     bool Selected)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

One of 0, 1, 2, 3, 4, 5 or 6.

`EdgeNum`

One of 0, 1, 2, 3 or 4.

`Selected`

Either True or False.

Usage

Setting FaceNum=0 selects or unselects the whole brick, irrespective of the value of EdgeNum.

Setting EdgeNum=0 selects or unselects the whole face when FaceNum is non-zero.

When edges of a brick face are selected, all selected edges must be on the same face. Selecting an edge on a different face will automatically unselect edges on any other currently selected faces.

St7GetBrickSelectState

Returns the selected state of the specified brick, brick face or edge on a brick face.

```
long St7GetBrickSelectState(long uID, long BrickNum, long FaceNum, long EdgeNum,
     bool* Selected)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

One of 0, 1, 2, 3, 4, 5 or 6.

`EdgeNum`

One of 0, 1, 2, 3 or 4.

Output Parameters

`Selected`

Either True or False.

St7SetAllEntitySelectState

Selects or deselects all entities of a given type.

```
long St7SetAllEntitySelectState(long uID, long Entity, bool Selected)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyLINK, tyVERTEX, tyGEOMETRYEDGE, tyGEOMETRYCOEDGE, tyGEOMETRYLOOP, tyGEOMETRYFACE or tyLOADPATH.

`Selected`

Either True or False.

St7SetEntitySelectStateByProperty

Selects or deselects all entities of a given type based on the property number, link type or load path template number.

```
long St7SetEntitySelectStateByProperty(long uID, long Entity, long PropertyNum,
    bool Selected)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK, tyLINK, tyGEOMETRYEDGE, tyGEOMETRYCOEDGE, tyGEOMETRYFACE or tyLOADPATH.

`PropertyNum`

For elements and geometric entities, the property number.

For links, one of the link types described in *Link Types*.

For Load Paths, the load path template number.

`Selected`

Either True or False.

St7SetEntitySelectStateByGroup

Selects or deselects all entities of a given type in a given group.

```
long St7SetEntitySelectStateByGroup(long uID, long Entity, long GroupID,
     bool Selected)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK, tyLINK, tyGEOMETRYEDGE, tyGEOMETRYCOEDGE, tyGEOMETRYFACE or tyLOADPATH.

`GroupID`

Group ID.

`Selected`

Either True or False.

St7SetEntitySelectStateByEntitySet

Selects or deselects all entities of a given type in a given entity set.

```
long St7SetEntitySelectStateByEntitySet(long uID, long Entity, long SetNum,
     bool Selected)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyLINK, tyVERTEX, tyGEOMETRYEDGE, tyGEOMETRYCOEDGE, tyGEOMETRYFACE or tyLOADPATH.

`SetNum`

Entity set number. Use zero to select or deselect enties contained in no entity set.

`Selected`

Either True or False.

St7GetEntitySelectCount

Returns the number entities of the specified type that are selected.

```
long St7GetEntitySelectCount(long uID, long Entity, long* NumSelected)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyLINK, tyVERTEX, tyGEOMETRYEDGE, tyGEOMETRYCOEDGE, tyGEOMETRYFACE or tyLOADPATH.

Output Parameters

`NumSelected`

Selected count.

Model Window

St7CreateModelWindow

Creates the window to display a Strand7 model graphically.

```
long St7CreateModelWindow(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7DestroyModelWindow

Destroys the model window for a Strand7 model.

```
long St7DestroyModelWindow(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7GetModelWindowState

Returns the state of the model window for a Strand7 model.

```
long St7GetModelWindowState(long uID, long* State)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`State`

One of wsModelWindowNotCreated, wsModelWindowVisible, wsModelWindowMaximised, wsModelWindowMinimised or wsModelWindowHidden.

St7GetModelWindowHandle

Returns the windows handle to the model window.

```
long St7GetModelWindowHandle(long uID, HWND* Handle)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Handle`

Handle to the model window.

St7SetModelWindowParent

Sets the parent control for the model window. This function can be used to dock the model window inside another control.

```
long St7SetModelWindowParent(long uID, HWND Handle)
```

Input Parameters

`uID`

Strand7 model file ID.

`Handle`

Handle to the parent control for the model window.

St7GetModelWindowParent

Returns the current parent control for the model window.

```
long St7GetModelWindowParent(long uID, HWND* Handle)
```

Input Parameters

`uID`

Strand7 model file ID.

`Handle`

Handle to the parent control for the model window.

St7ShowModelWindow

Shows the model window.

```
long St7ShowModelWindow(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7HideModelWindow

Hides the model window.

```
long St7HideModelWindow(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7SetWindowBackgroundMode

Sets the background mode of the model window.

```
long St7SetWindowBackgroundMode(long uID, long WindowMode, long BackgroundMode)
```

Input Parameters

`uID`

Strand7 model file ID.

`WindowMode`

Either wmPreProcessing or wmPostProcessing.

`BackgroundMode`

One of bgSolid, bgImage, bgGradient or bgImageGradient.

St7GetWindowBackgroundMode

Returns the background mode of the model window.

```
long St7GetWindowBackgroundMode(long uID, long WindowMode, long* BackgroundMode)
```

Input Parameters

`uID`

Strand7 model file ID.

`WindowMode`

Either wmPreProcessing or wmPostProcessing.

Output Parameters

`BackgroundMode`

One of bgSolid, bgImage, bgGradient or bgImageGradient.

St7SetWindowColours

Sets the background colour(s) of the model window.

```
long St7SetWindowColours(long uID, long WindowMode, long SolidColour,
     long GradientColour)
```

Input Parameters

`uID`

Strand7 model file ID.

`WindowMode`

Either wmPreProcessing or wmPostProcessing.

`SolidColour`

Solid background colour or upper colour for gradient backgrounds. See also *RGB Colours.*

`GradientColour`

Lower colour for gradient backgrounds. See also *RGB Colours*.

St7GetWindowColours

Returns the background colour(s) of the model window.

```
long St7GetWindowColours(long uID, long WindowMode, long* SolidColour,
     long* GradientColour)
```

Input Parameters

`uID`

Strand7 model file ID.

`WindowMode`

Either wmPreProcessing or wmPostProcessing.

Output Parameters

`SolidColour`

Solid background colour or upper colour for gradient backgrounds. See also *RGB Colours*.

`GradientColour`

Lower colour for gradient backgrounds. See also *RGB Colours*.

St7SetWindowImageFile

Sets the image file to be used for bgImage and bgImageGradient background modes.

```
long St7SetWindowImageFile(long uID, char* FileName)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

The name of the image file. The file name extension must be one of '.BMP', '.JPG', '.JPEG' or '.PNG', each representing the respective image file type. A blank string may be used to clear the current image file name.

St7GetWindowImageFile

Returns the name of the image file used for bgImage and bgImageGradient background modes.

```
long St7GetWindowImageFile(long uID, char* FileName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxStringLen`

Maximum number of characters allocated for FileName.

Output Parameters

`FileName`

The name of the image file.

St7SetWindowImageLocation

Sets the location of the image used for bgImage and bgImageGradient background modes.

```
long St7SetWindowImageLocation(long uID, long ImageLocation)
```

Input Parameters

`uID`

Strand7 model file ID.

`ImageLocation`

One of ilCentre, ilTile, ilStretch, ilTopLeft, ilTopRight, ilBottomLeft, ilBottomRight.

St7GetWindowImageLocation

Returns the location of the image used for bgImage and bgImageGradient background modes.

```
long St7GetWindowImageLocation(long uID, long* ImageLocation)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`ImageLocation`

One of ilCentre, ilTile, ilStretch, ilTopLeft, ilTopRight, ilBottomLeft, ilBottomRight.

St7SetWindowImageSize

Sets the size of the image used for bgImage and bgImageGradient background modes.

```
long St7SetWindowImageSize(long uID, long ImageSize)
```

Input Parameters

`uID`

Strand7 model file ID.

`ImageSize`

A value between 0 and 100 representing the size of the image as a percentage of the width of the model window. The height of the image is scaled proportionally. If ImageSize is 0, the image is rendered pixel by pixel, unscaled.

St7GetWindowImageSize

Returns the size of the image used for bgImage and bgImageGradient background modes.

```
long St7GetWindowImageSize(long uID, long* ImageSize)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`ImageSize`

A value between 0 and 100 representing the size of the image as a percentage of the width of the model window. The height of the image is scaled proportionally. If ImageSize is 0, the image is rendered pixel by pixel, unscaled.

St7UpdateElementPropertyData

Updates the display database used by the model window so that a redraw shows modifications made to entities when *St7RedrawModel* is used; e.g. changes to property colours, beam section dimensions, plate thickness, etc.

```
long St7UpdateElementPropertyData(long uID, long Entity, long PropNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

The property type; either ptBEAMPROP, ptPLATEPROP or ptBRICKPROP.

`PropNum`

The updated property number.

St7ShowWindowCombos

Shows the Load Case, Freedom Case and UCS combo boxes on the model window.

```
long St7ShowWindowCombos(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7HideWindowCombos

Hides the Load Case, Freedom Case and UCS combo boxes from the model window.

```
long St7HideWindowCombos(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7ShowWindowEntityPanel

Shows the entity panel on the model window.

```
long St7ShowWindowEntityPanel(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7HideWindowEntityPanel

Hides the entity panel from the model window.

```
long St7HideWindowEntityPanel(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7ShowWindowStatusBar

Shows the status bar at the bottom of the model window.

```
long St7ShowWindowStatusBar(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7HideWindowStatusBar

Hides the status bar from the model window.

```
long St7HideWindowStatusBar(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7EnableWindowStatusBar

Enables mouse input for the status bar shown by *St7ShowWindowStatusBar* in the model window.

```
long St7EnableWindowStatusBar(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7DisableWindowStatusBar

Disables mouse input for the status bar shown by *St7ShowWindowStatusBar* in the model window.

```
long St7DisableWindowStatusBar(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7SetWindowStatusBarRefreshMode

Sets the way the status bar responds to actions that invalidate it.

```
long St7SetWindowStatusBarRefreshMode(long uID, bool AutoRefresh)
```

Input Parameters

`uID`

Strand7 model file ID.

`AutoRefresh`

True to automatically update the entity counts in the status bar are as they change.

False to pause entity count updates in the status bar.

API calls will take longer to perform with AutoRefresh=True since more work is required to continually update the status bar in response to changes.

St7RefreshWindowStatusBar

Updates the contents of the status bar irrespective of the refresh mode set by *St7SetWindowStatusBarRefreshMode*.

```
long St7RefreshWindowStatusBar(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7EnableWindowEntityInspector

Enables the entity inspector (shift-mouse-over) in the model window.

```
long St7EnableWindowEntityInspector(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7DisableWindowEntityInspector

Disables the entity inspector (shift-mouse-over) in the model window.

```
long St7DisableWindowEntityInspector(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7ShowWindowSelectionToolbar

Shows the selection toolbar on the model window.

```
long St7ShowWindowSelectionToolbar(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7HideWindowSelectionToolbar

Hides the selection toolbar from the model window.

```
long St7HideWindowSelectionToolbar(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7ShowWindowCaption

Shows the model caption and window controls at the top of the model window.

```
long St7ShowWindowCaption(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7HideWindowCaption

Hides the model caption and window controls from the model window.

```
long St7HideWindowCaption(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7ShowWindowViewToolbar

Shows the view toolbar on the model window.

```
long St7ShowWindowViewToolbar(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7HideWindowViewToolbar

Hides the view toolbar from the model window.

```
long St7HideWindowViewToolbar(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7ShowWindowResultsToolbar

Shows the results toolbar on the model window, when a result file is open.

```
long St7ShowWindowResultsToolbar(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7HideWindowResultsToolbar

Hides the results toolbar from the model window, when a result file is open.

```
long St7HideWindowResultsToolbar(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7ShowWindowShowHideToolbar

Shows the toolbar, which enables the user to show and hide model entities by Type/Property, Group, Set or Viewport.

```
long St7ShowWindowShowHideToolbar(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7HideWindowShowHideToolbar

Hides the toolbar, which enables the user to show and hide model entities by Type/Property, Group, Set or Viewport.

```
long St7HideWindowShowHideToolbar(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7EnableWindowResize

Allows resizing of the graphical window by mouse.

```
long St7EnableWindowResize(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7DisableWindowResize

Prevents resizing of the graphical window by mouse.

```
long St7DisableWindowResize(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7EnableWindowViewChanges

Allows model window view changes by the application user (for example, clicking and dragging to rotate the model).

```
long St7EnableWindowViewChanges(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7DisableWindowViewChanges

Disallows model window view changes by the application user (for example, clicking and dragging to rotate the model). Note that when the view toolbar is showing, the model window view can be changed by the user irrespective of this setting.

```
long St7DisableWindowViewChanges(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7SetModelWindowRefresh

Sets the way the model window responds to actions that invalidate it.

```
long St7SetModelWindowRefresh(long uID, long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`Mode`

One of wrAutoRefresh, wrPauseClear or wrPauseNoClear.

In wrAutoRefresh mode, the model window remains synchronised with the model, automatically refreshing as changes are made. For example, if new elements are created, the model window will refresh automatically as each element is added.

In wrPauseClear and wrPauseNoClear modes, the model window remains responsive until a user action or API call invalidates it. While the model window is valid it can respond to view changes and/or window resizes. Once the model window is invalidated, the window is not updated until either

a) *St7RedrawModel* is called (which performs a one-off synchronisation); or
b) this function is called with the wrAutoRefresh mode (which performs and maintains synchronisation).

The difference between wrPauseClear and wrPauseNoClear is in what is displayed in an invalidated model window. An invalidated model window will be cleared with wrPauseClear, whereas it will continue to display the previous graphics with wrPauseNoClear, even if the previous graphics are out of date with the current state of the model. The advantage of wrPauseNoClear is that it avoids a screen flicker that can occur when the window is cleared between pausing it and refreshing it after the model has changed. Both modes will disable interaction with the model window (such as the Entity Inspector and selecting).

API calls will take longer to perform with the wrAutoRefresh mode since more work is required to continually redraw the model in response to changes.

St7ClearModelWindow

Clears the graphics within the model window and suspends graphics updates until either *St7RedrawModel* is called or *St7SetModelWindowRefresh* is called with wrAutoRefresh.

```
long St7ClearModelWindow(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7RedrawModel

Redraws the graphics within the model window.

```
long St7RedrawModel(long uID, bool Rescale)
```

Input Parameters

`uID`

Strand7 model file ID.

`Rescale`

True to rescale the view limits when the graphics are redrawn.

St7RotateModel

Sets the view angle for graphical display of the model in the model window.

```
long St7RotateModel(long uID, double RX, double RY, double RZ)
```

Input Parameters

`uID`

Strand7 model file ID.

`RX`

Rotation about the global X axis (degrees).

`RY`

Rotation about the global Y axis (degrees).

`RZ`

Rotation about the global Z axis (degrees).

St7ZoomModel

Zooms the graphical display of the model in the model window.

```
long St7ZoomModel(long uID, double CentreX, double CentreY, double ZoomScale)
```

Input Parameters

`uID`

Strand7 model file ID.

`CentreX`

The X coordinate of the zoom origin with respect to the model window. Any positive or negative real number may be used; 0.0 refers to the left edge of the window, 0.5 refers to the centre of the window and 1.0 refers to the right edge of the window. Values outside the range zero to one may be used.

`CentreY`

The Y coordinate of the zoom origin with respect to the model window. Any positive or negative real number may be used; 0.0 refers to the bottom edge of the window, 0.5 refers to the centre of the window and 1.0 refers to the top edge of the window. Values outside the range zero to one may be used.

`ZoomScale`

Values greater than one enlarge the model (i.e., zoom in); values less than one reduce the model (i.e., zoom out). ZoomScale must be greater than zero.

St7PanModel

Pans the graphical display of the model in the model window.

```
long St7PanModel(long uID, double PanX, double PanY)
```

Input Parameters

`uID`

Strand7 model file ID.

`PanX`

The amount by which to pan the model horizontally with respect to the model window. A positive value pans the model to the right; a negative value pans the model to the left; a magnitude of 1.0 pans the model by one full window width.

`PanY`

The amount by which to pan the model vertically with respect to the model window. A positive value pans the model upward; a negative value pans the model downward; a magnitude of 1.0 pans the model by one full window height.

St7ShowEntity

Shows all entities of the specified type within the model window.

```
long St7ShowEntity(long uID, long Entity)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyLINK, tyVERTEX, tyGEOMETRYFACE or tyLOADPATH.

St7HideEntity

Hides all entities of the specified type within the model window.

```
long St7HideEntity(long uID, long Entity)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyLINK, tyVERTEX, tyGEOMETRYFACE or tyLOADPATH.

St7GetEntityVisibility

Returns the visibility of the entities of the specified type within the model window.

```
long St7GetEntityVisibility(long uID, long Entity, bool* Visible)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyLINK, tyVERTEX, tyGEOMETRYFACE or tyLOADPATH.

Output Parameters

`Visible`

Visible state; either True or False.

St7ShowPointAttributes

Shows the node and vertex attributes within the model window.

```
long St7ShowPointAttributes(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7HidePointAttributes

Hides the node and vertex attributes within the model window.

```
long St7HidePointAttributes(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7GetPointAttributesVisibility

Returns the visibility of point attributes within the model window.

```
long St7GetPointAttributesVisibility(long uID, bool* Visible)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Visible`

Visible state; either True or False.

St7ShowEntityAttributes

Shows the element attributes within the model window.

```
long St7ShowEntityAttributes(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7HideEntityAttributes

Hides the element attributes within the model window.

```
long St7HideEntityAttributes(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7GetEntityAttributesVisibility

Returns the visibility of entity attributes within the model window.

```
long St7GetEntityAttributesVisibility(long uID, bool* Visible)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Visible`

Visible state; either True or False.

St7PositionModelWindow

Sets the screen position of the model window.

```
long St7PositionModelWindow(long uID, long Left, long Top, long Width,
     long Height)
```

Input Parameters

`uID`

Strand7 model file ID.

`Left`

Pixel position of the left edge of the model window.

`Top`

Pixel position of the top edge of the model window.

`Width`

Pixel width of the model window.

`Height`

Pixel height of the model window.

St7GetModelWindowPosition

Returns the screen position of the model window.

```
long St7GetModelWindowPosition(long uID, long* Left, long* Top, long* Width,
     long* Height)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Left`

Pixel position of the left edge of the model window.

`Top`

Pixel position of the top edge of the model window.

`Width`

Pixel width of the model window.

`Height`

Pixel height of the model window.

St7GetDrawAreaPosition

Returns the screen position for drawing the model graphics within the model window.

```
long St7GetDrawAreaPosition(long uID, long* Left, long* Top, long* Width,
    long* Height)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Left`

Pixel position of the left edge of the model graphics.

`Top`

Pixel position of the top edge of the model graphics.

`Width`

Pixel width of the model graphics.

`Height`

Pixel height of the model graphics.

St7GetDrawAreaSize

Returns the screen area available for drawing the model graphics within the model window.

```
long St7GetDrawAreaSize(long uID, long* Width, long* Height)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Width`

Pixel width of the drawing area.

`Height`

Pixel height of the drawing area.

St7ShowProperty

Shows all of the entities of the specified property within the model window.

```
long St7ShowProperty(long uID, long Entity, long PropNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK or tyGEOMETRYFACE.

`PropNum`

The ID number of the property to show.

St7HideProperty

Hides all of the entities of the specified property number within the model window.

```
long St7HideProperty(long uID, long Entity, long PropNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK or tyGEOMETRYFACE.

`PropNum`

The ID number of the property to hide.

St7GetPropertyVisibility

Determines the visibility of a property within the model window.

```
long St7GetPropertyVisibility(long uID, long Entity, long PropNum, bool* Visible)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK or tyGEOMETRYFACE.

`PropNum`

Property number.

Output Parameters

`Visible`

Visible state; either True or False.

St7ShowGroup

Shows all entities in the specified group within the model window.

```
long St7ShowGroup(long uID, long GroupID)
```

Input Parameters

`uID`

Strand7 model file ID.

`GroupID`

The ID number of the group to show.

St7HideGroup

Hides all of the entities in the specified group within the model window.

```
long St7HideGroup(long uID, long GroupID)
```

Input Parameters

`uID`

Strand7 model file ID.

`GroupID`

The ID number of the group to hide.

St7GetGroupVisibility

Determines the visibility of a group within the model window.

```
long St7GetGroupVisibility(long uID, long GroupID, bool* Visible)
```

Input Parameters

`uID`

Strand7 model file ID.

`GroupID`

The ID number of the group.

Output Parameters

`Visible`

Visible state; either True or False.

St7ShowEntitySet

Shows all entities in the specified entity set within the model window.

```
long St7ShowEntitySet(long uID, long SetNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`SetNum`

The entity set to show.

St7HideEntitySet

Hides all entities in the specified entity set within the model window.

```
long St7HideEntitySet(long uID, long SetNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`SetNum`

The entity set to hide.

St7GetEntitySetVisibility

Determines the visibility of an entity set within the model window.

```
long St7GetEntitySetVisibility(long uID, long SetNum, bool* Visible)
```

Input Parameters

`uID`

Strand7 model file ID.

`SetNum`

The entity set number.

Output Parameters

`Visible`

Visible state; either True or False.

St7SetAllEntitiesOn

Resets the display to show all entities and attributes in the model window.

```
long St7SetAllEntitiesOn(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7GetEntityNumVisibility

Determines whether or not an entity is visible in the model window.

```
long St7GetEntityNumVisibility(long uID, long Entity, long EntityNum,
     bool* Visible)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyLINK, tyVERTEX, tyGEOMETRYFACE or tyLOADPATH.

`EntityNum`

Entity number.

Output Parameters

`Visible`

Visible state; either True or False.

St7SetBeamResultDisplay

Sets the display options for beam results within the model window.

```
long St7SetBeamResultDisplay(long uID, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Integers[0..12]`

`[ipResultType]` – Beam result type; one of rtAsNone, rtAsContour, rtAsDiagram or rtAsVector.

`[ipResultQuantity]` – See *Result Display Options*. When result quantity is rqBeamUserC, the currently active user equation must be assigned via *St7SetResultUserEquation* or *St7SetStoredResultUserEquation* before calling *St7SetBeamResultDisplay*.

`[ipResultSystem]` – See *Result Display Options*.

`[ipResultComponent]` – See *Result Display Options*.

`[ipVectorStyle]` – Vector display style; one of vtVectorTranslationMag, vtVectorRotationMag, vtVectorTranslationComponents or vtVectorRotationComponents.

`[ipReferenceNode]` – Reference node number when showing displacement results, or 0 for total displacement.

`[ipAbsoluteValue]` – btTrue to contour the absolute value for rtAsContour.

`[ipDiagram1..ipDiagram6]` – An array of flags, either btTrue or btFalse, that are used to set up rtAsDiagram displays.

`[ipVector1..ipVector6]` – An array of flags, either btTrue or btFalse, that are used to set up rtAsVector displays.

St7SetPlateResultDisplay

Sets the display options for plate results within the model window.

```
long St7SetPlateResultDisplay(long uID, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Integers[0..12]`

`[ipResultType]` – Plate result type; one of rtAsNone, rtAsContour or rtAsVector.

`[ipResultQuantity]` – See *Result Display Options*. When result quantity is rqPlateUserC, the currently active user equation must be assigned via *St7SetResultUserEquation* or *St7SetStoredResultUserEquation* before calling *St7SetPlateResultDisplay*.

`[ipResultSystem]` – See *Result Display Options*.

`[ipResultComponent]` – See *Result Display Options*; ignored when result type is rtAsVector.

`[ipResultSurface]` – See *Result Display Options*.

`[ipVectorStyle]` – Vector display style; one of vtVectorTranslationMag, vtVectorRotationMag, vtVectorTranslationComponents or vtVectorRotationComponents; used when result quantity is a nodal quantity (e.g. displacement); ignored when result quantity is an element quantity (e.g. principal stress).

`[ipReferenceNode]` – Reference node number when showing displacement results, or 0 for total displacement.

`[ipAbsoluteValue]` – btTrue to contour the absolute value for rtAsContour.

`[ipVector1..ipVector6]` – An array of flags, either btTrue or btFalse, that are used to set up rtAsVector displays; applicable to nodal result quantities (e.g. displacement) and element result quantities (e.g. principal stress).

St7SetBrickResultDisplay

Sets the display options for the brick results within the model window.

```
long St7SetBrickResultDisplay(long uID, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Integers[0..12]`

`[ipResultType]` – Brick result type; one of rtAsNone, rtAsContour or rtAsVector.

`[ipResultQuantity]` – See *Result Display Options*. When result quantity is rqBrickUserC, the currently active user equation must be assigned via *St7SetResultUserEquation* or *St7SetStoredResultUserEquation* before calling *St7SetBrickResultDisplay*.

`[ipResultSystem]` – See *Result Display Options*.

`[ipResultComponent]` – See *Result Display Options*; ignored when result type is rtAsVector.

`[ipVectorStyle]` – Vector display style; one of vtVectorTranslationMag, vtVectorRotationMag, vtVectorTranslationComponents or vtVectorRotationComponents; used when result quantity is a nodal quantity (e.g. displacement); ignored when result quantity is an element quantity (e.g. principal stress).

`[ipReferenceNode]` – Reference node number when showing displacement results, or 0 for total displacement.

`[ipAbsoluteValue]` – btTrue to contour the absolute value for rtAsContour.

`[ipVector1..ipVector6]` – An array of flags, either btTrue or btFalse, that are used to set up rtAsVector displays; applicable to nodal result quantities (e.g. displacement) and element result quantities (e.g. principal stress).

St7SetLinkResultDisplay

Sets the display options for the link results within the model window.

```
long St7SetLinkResultDisplay(long uID, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Integers[0..12]`

`[ipResultType]` – Link result type; one of rtAsNone, rtAsContour or rtAsVector.

`[ipResultQuantity]` – See *Result Display Options*.

`[ipResultSystem]` – See *Result Display Options*.

`[ipResultComponent]` – See *Result Display Options*.

`[ipVectorStyle]` – Vector display style; one of vtVectorTranslationMag, vtVectorRotationMag, vtVectorTranslationComponents or vtVectorRotationComponents.

`[ipReferenceNode]` – Reference node number when showing displacement results, or 0 for total displacement.

`[ipAbsoluteValue]` – btTrue to contour the absolute value for rtAsContour.

`[ipVector1..ipVector6]` – An array of flags, either btTrue or btFalse, that are used to set up rtAsVector displays.

St7SetWindowResultCase

Sets the result case to be displayed within the model window.

```
long St7SetWindowResultCase(long uID, long ResultCase)
```

Input Parameters

`uID`

Strand7 model file ID.

`ResultCase`

The result case ID number to be displayed.

St7SetWindowLoadCase

Sets the load case to be displayed within the model window.

```
long St7SetWindowLoadCase(long uID, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

St7SetWindowFreedomCase

Sets the freedom case to be displayed within the model window.

```
long St7SetWindowFreedomCase(long uID, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Freedom case number.

St7SetWindowUCSCase

Sets the UCS case to be displayed within the model window.

```
long St7SetWindowUCSCase(long uID, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

The UCS ID number to be displayed.

St7SetNumericOptions

Sets the display mode of numbers in the model window.

```
long St7SetNumericOptions(long uID, long Mode, long Style, long Digits,
     long Exponent, double Zero)
```

Input Parameters

`uID`

Strand7 model file ID.

`Mode`

The model window mode for which the numeric options are being assigned. Either nmPreProcessing or nmPostProcessing.

`Style`

One of nsFixed, nsEngineering, nsScientific or nsAuto.

`Digits`

Number of digits to display.

`Exponent`

Either efLowered or efRaised.

`Zero`

Zero cut-off; numbers with a smaller magnitude than this are displayed as zero.

St7GetNumericOptions

Returns the display mode of numbers in the model window.

```
long St7GetNumericOptions(long uID, long Mode, long* Style, long* Digits,
     long* Exponent, double* Zero)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Mode`

The model window mode for which the numeric options are being retrieved. Either nmPreProcessing or nmPostProcessing.

`Style`

One of nsFixed, nsEngineering, nsScientific or nsAuto.

`Digits`

Number of digits to display.

`Exponent`

Either efLowered or efRaised.

`Zero`

Zero cut-off; numbers with a smaller magnitude than this are displayed as zero.

St7SetEntityFont

Sets the font for displaying entity labels.

```
long St7SetEntityFont(long uID, long Entity, char* FontName, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyLINK, tyVERTEX, tyGEOMETRYFACE or tyLOADPATH.

`FontName`

String containing the name of the font.

`Integers[0..4]`

`[ipFontSize]` – Font size.

`[ipFontColour]` – Font colour. See also *RGB Colours*.

`[ipFontStyleBold]` – btTrue for bold.

`[ipFontStyleItalic]` – btTrue for italic.

`[ipFontStyleUnderline]` – btTrue for underline.

St7GetEntityFont

Returns the font for displaying entity labels.

```
long St7GetEntityFont(long uID, long Entity, char* FontName, long MaxStringLen,
     long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyLINK, tyVERTEX, tyGEOMETRYFACE or tyLOADPATH.

`MaxStringLen`

Maximum number of characters allocated for FontName.

Output Parameters

`FontName`

String containing the name of the font.

`Integers[0..4]`

`[ipFontSize]` – Font size.

`[ipFontColour]` – Font colour. See also *RGB Colours*.

`[ipFontStyleBold]` – btTrue for bold.

`[ipFontStyleItalic]` – btTrue for italic.

`[ipFontStyleUnderline]` – btTrue for underline.

Model Window Results Settings

St7SetResultSettingsStyle

Sets the parameters accessible through **Results Settings/Settings.../Style** in the GUI.

```
long St7SetResultSettingsStyle(long uID, long Solver, long Entity, long Quantity,
     long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Solver`

One of the solver types listed in *Solver Types*.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK or tyLINK.

`Quantity`

One of the `QUANTITY` options listed in *Result Display Options*.

`Integers[0..14]`

`[ipContourStyle]` – Either csRainbow, csRainbowEnds, csMono, csLines or csBands.

`[ipReverse]` – btTrue if the contour is reversed.

`[ipSeparator]` – btTrue if a separator is included between ranges for the user defined contour.

`[ipBand1Colour]` – Colour for **Band 1**. See also *RGB Colours*.

`[ipBand2Colour]` – Colour for **Band 2**. See also *RGB Colours*.

`[ipSeparatorColour]` – Colour for **Separator**. See also *RGB Colours*.

`[ipLineBackColour]` – Colour for **Line Back**. See also *RGB Colours*.

`[ipMonoColour]` – Colour for **Mono**. See also *RGB Colours*.

`[ipMinColour]` – Colour for **< Min**. See also *RGB Colours*.

`[ipMaxColour]` – Colour for **> Max**. See also *RGB Colours*.

`[ipLimitMin]` – btTrue if the colour specified in ipMinColour is enabled for the out-of-bound contour.

`[ipLimitMax]` – btTrue if the colour specified in ipMaxColour is enabled for the out-of-bound contour.

`[ipVectorThickness]` – Vector Thickness.

`[ipVectorLength]` – Vector **Nominal Length**.

`[ipVectorAnchor]` – One of vaTail, vaHead or vaBoth.

St7GetResultSettingsStyle

Returns the parameters accessible through **Results Settings/Settings.../Style** in the GUI.

```
long St7GetResultSettingsStyle(long uID, long Solver, long Entity, long Quantity,
     long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Solver`

One of the solver types listed in *Solver Types*.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK or tyLINK.

`Quantity`

One of the `QUANTITY` options listed in *Result Display Options*.

Output Parameters

`Integers[0..14]`

`[ipContourStyle]` – Either csRainbow, csRainbowEnds, csMono, csLines or csBands.

`[ipReverse]` – btTrue if the contour is reversed.

`[ipSeparator]` – btTrue if a separator is included between ranges for the user defined contour.

`[ipBand1Colour]` – Colour for **Band 1**. See also *RGB Colours*.

`[ipBand2Colour]` – Colour for **Band 2**. See also *RGB Colours*.

`[ipSeparatorColour]` – Colour for **Separator**. See also *RGB Colours*.

`[ipLineBackColour]` – Colour for **Line Back**. See also *RGB Colours*.

`[ipMonoColour]` – Colour for **Mono**. See also *RGB Colours*.

`[ipMinColour]` – Colour for **< Min**. See also *RGB Colours*.

`[ipMaxColour]` – Colour for **> Max**. See also *RGB Colours*.

`[ipLimitMin]` – btTrue if the colour specified in ipMinColour is enabled for the out-of-bound contour.

`[ipLimitMax]` – btTrue if the colour specified in ipMaxColour is enabled for the out-of-bound contour.

`[ipVectorThickness]` – Vector Thickness.

`[ipVectorLength]` – Vector **Nominal Length**.

`[ipVectorAnchor]` – One of vaTail, vaHead or vaBoth.

St7SetResultSettingsLimits

Sets the parameters accessible through **Results Settings/Settings.../Limits** in the GUI.

```
long St7SetResultSettingsLimits(long uID, long Solver, long Entity,
     long Quantity, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`Solver`

One of the solver types listed in *Solver Types*.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK or tyLINK.

`Quantity`

One of the `QUANTITY` options listed in *Result Display Options*.

`Integers[0..4]`

`[ipContourLimit]` – Either clDefault, clUserRange, clRounded or clUserSpecified.

`[ipContourMode]` – Either cmContinuous or cmDiscrete.

`[ipNumContours]` – Number of contours with the discrete setting.

`[ipSetMinLimit]` – btTrue if the lower limit is enabled for the user range.

`[ipSetMaxLimit]` – btTrue if the upper limit is enabled for the user range.

`Doubles[0..1]`

`[ipMinLimit]` – Lower limit for the user range.

`[ipMaxLimit]` – Upper limit for the user range.

St7GetResultSettingsLimits

Returns the parameters accessible through **Results Settings/Settings.../Limits** in the GUI.

```
long St7GetResultSettingsLimits(long uID, long Solver, long Entity,
     long Quantity, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`Solver`

One of the solver types listed in *Solver Types*.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK or tyLINK.

`Quantity`

One of the `QUANTITY` options listed in *Result Display Options*.

Output Parameters

`Integers[0..4]`

`[ipContourLimit]` – Either clDefault, clUserRange, clRounded or clUserSpecified.

`[ipContourMode]` – Either cmContinuous or cmDiscrete.

`[ipNumContours]` – Number of contours with the discrete setting.

`[ipSetMinLimit]` – btTrue if the lower limit is enabled for the user range.

`[ipSetMaxLimit]` – btTrue if the upper limit is enabled for the user range.

`Doubles[0..1]`

`[ipMinLimit]` – Lower limit for the user range.

`[ipMaxLimit]` – Upper limit for the user range.

St7SetResultSettingsLimitsString

Sets the parameters accessible through **Results Settings/Settings.../Limits** in the GUI.

```
long St7SetResultSettingsLimitsString(long uID, long Solver, long Entity,
     long Quantity, char* LimitsString)
```

Input Parameters

`uID`

Strand7 model file ID.

`Solver`

One of the solver types listed in *Solver Types*.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK or tyLINK.

`Quantity`

One of the `QUANTITY` options listed in *Result Display Options*.

`LimitsString`

String containing the user defined contour boundaries. Use *St7GetListSeparatorCode* to get the system's list separator character.

St7GetResultSettingsLimitsString

Returns the parameters accessible through **Results Settings/Settings.../Limits** in the GUI.

```
long St7GetResultSettingsLimitsString(long uID, long Solver, long Entity,
     long Quantity, char* LimitsString, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`Solver`

One of the solver types listed in *Solver Types*.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK or tyLINK.

`Quantity`

One of the `QUANTITY` options listed in *Result Display Options*.

`MaxStringLen`

Maximum number of characters allocated for LimitsString.

Output Parameters

`LimitsString`

String containing the user defined contour boundaries. Use *St7GetListSeparatorCode* to get the system's list separator character.

St7SetResultSettingsLegend

Sets the parameters accessible through **Results Settings/Settings.../Legend** in the GUI.

```
long St7SetResultSettingsLegend(long uID, long Solver, long Entity,
     long Quantity, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Solver`

One of the solver types listed in *Solver Types*.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK or tyLINK.

`Quantity`

One of the `QUANTITY` options listed in *Result Display Options*.

`Integers[0..5]`

`[ipLegendPosition]` – Either lpNone, lpTopLeft, lpTopRight, lpBottomLeft, lpBottomRight or lpFloating.

`[ipOpaqueLegend]` – btTrue for **Opaque**.

`[ipShowMinMax]` – btTrue for **Show Min/Max**.

`[ipHistogram]` – btTrue for **Histogram**.

`[ipLegendWidth]` – Legend width.

`[ipLegendHeight]` – Legend height.

St7GetResultSettingsLegend

Returns the parameters accessible through **Results Settings/Settings.../Legend** in the GUI.

```
long St7GetResultSettingsLegend(long uID, long Solver, long Entity,
     long Quantity, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Solver`

One of the solver types listed in *Solver Types*.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK or tyLINK.

`Quantity`

One of the `QUANTITY` options listed in *Result Display Options*.

Output Parameters

`Integers[0..5]`

`[ipLegendPosition]` – Either lpNone, lpTopLeft, lpTopRight, lpBottomLeft, lpBottomRight or lpFloating.

`[ipOpaqueLegend]` – btTrue for **Opaque**.

`[ipShowMinMax]` – btTrue for **Show Min/Max**.

`[ipHistogram]` – btTrue for **Histogram**.

`[ipLegendWidth]` – Legend width.

`[ipLegendHeight]` – Legend height.

St7SetResultSettingsLegendFont

Sets the parameters accessible through **Results Settings/Settings.../Legend** in the GUI.

```
long St7SetResultSettingsLegendFont(long uID, long Solver, long Entity,
     long Quantity, char* FontName, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Solver`

One of the solver types listed in *Solver Types*.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK or tyLINK.

`Quantity`

One of the `QUANTITY` options listed in *Result Display Options*.

`FontName`

String containing the name of the font.

`Integers[0..4]`

`[ipFontSize]` – Font size.

`[ipFontColour]` – Font colour. See also *RGB Colours*.

`[ipFontStyleBold]` – btTrue for bold.

`[ipFontStyleItalic]` – btTrue for italic.

`[ipFontStyleUnderline]` – btTrue for underline.

St7GetResultSettingsLegendFont

Returns the parameters accessible through **Results Settings/Settings.../Legend** in the GUI.

```
long St7GetResultSettingsLegendFont(long uID, long Solver, long Entity,
     long Quantity, char* FontName, long MaxStringLen, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Solver`

One of the solver types listed in *Solver Types*.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK or tyLINK.

`Quantity`

One of the `QUANTITY` options listed in *Result Display Options*.

`MaxStringLen`

Maximum number of characters allocated for FontName.

Output Parameters

`FontName`

String containing the name of the font.

`Integers[0..4]`

`[ipFontSize]` – Font size.

`[ipFontColour]` – Font colour. See also *RGB Colours*.

`[ipFontStyleBold]` – btTrue for bold.

`[ipFontStyleItalic]` – btTrue for italic.

`[ipFontStyleUnderline]` – btTrue for underline.

St7SetResultSettingsDiagram

Sets the parameters accessible through **Results Settings/Settings.../Diagram** in the GUI.

```
long St7SetResultSettingsDiagram(long uID, long Solver, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Solver`

One of the solver types listed in *Solver Types*.

`Integers[0..5]`

`[ipDiagramStyle]` – Either dsSingleLine or dsHatched.

`[ipDiagramAxialDir]` – One of adPlus1, adMinus1, adPlus2 or adMinus2.

`[ipDiagramTorqueDir]` – One of adPlus1, adMinus1, adPlus2 or adMinus2.

`[ipDiagramRelativeLength]` – Relative length in the range 0 to 100.

`[ipDiagramThickness]` – Thickness in the range kMinThickness to kMaxThickness.

`[ipDiagramMomentSide]` – Either bmTensionSide or bmCompressionSide.

St7GetResultSettingsDiagram

Returns the parameters accessible through **Results Settings/Settings.../Diagram** in the GUI.

```
long St7GetResultSettingsDiagram(long uID, long Solver, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Solver`

One of the solver types listed in *Solver Types*.

Output Parameters

`Integers[0..5]`

`[ipDiagramStyle]` – Either dsSingleLine or dsHatched.

`[ipDiagramAxialDir]` – One of adPlus1, adMinus1, adPlus2 or adMinus2.

`[ipDiagramTorqueDir]` – One of adPlus1, adMinus1, adPlus2 or adMinus2.

`[ipDiagramRelativeLength]` – Relative length in the range 0 to 100.

`[ipDiagramThickness]` – Thickness in the range kMinThickness to kMaxThickness.

`[ipDiagramMomentSide]` – Either bmTensionSide or bmCompressionSide.

St7SetResultSettingsDiagramColours

Sets the beam diagram colours accessible through **Results Settings** in the GUI.

```
long St7SetResultSettingsDiagramColours(long uID, long Solver, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Solver`

One of the solver types listed in *Solver Types*.

`Integers[0..5]`

`[ipBeamSF1]` – Shear force 1 and shear force x diagram colour.

`[ipBeamBM1]` – Bending moment 1 and bending moment x diagram colour.

`[ipBeamSF2]` – Shear force 2 and shear force y diagram colour.

`[ipBeamBM2]` – Bending moment 2 and bending moment y diagram colour.

`[ipBeamAxialF]` – Axial force diagram colour.

`[ipBeamTorque]` – Torque diagram colour.

St7GetResultSettingsDiagramColours

Returns the beam diagram colours accessible through **Results Settings** in the GUI.

```
long St7GetResultSettingsDiagramColours(long uID, long Solver, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Solver`

One of the solver types listed in *Solver Types*.

Output Parameters

`Integers[0..5]`

`[ipBeamSF1]` – Shear force 1 and shear force x diagram colour.

`[ipBeamBM1]` – Bending moment 1 and bending moment x diagram colour.

`[ipBeamSF2]` – Shear force 2 and shear force y diagram colour.

`[ipBeamBM2]` – Bending moment 2 and bending moment y diagram colour.

`[ipBeamAxialF]` – Axial force diagram colour.

`[ipBeamTorque]` – Torque diagram colour.

Node Entity Display

St7SetNodeStyle

Sets the display symbol for nodes.

```
long St7SetNodeStyle(long uID, long Style)
```

Input Parameters

`uID`

Strand7 model file ID.

`Style`

Either psCircle or psSquare.

St7SetNodeSize

Sets the size index of the displayed nodes.

```
long St7SetNodeSize(long uID, long Size)
```

Input Parameters

`uID`

Strand7 model file ID.

`Size`

Node size index in the range kMinPointSize to kMaxPointSize.

St7SetNodeShowHideSelected

Sets whether the display of nodes considers the Show/Hide Selected setting.

```
long St7SetNodeShowHideSelected(long uID, bool UseSettings)
```

Input Parameters

`uID`

Strand7 model file ID.

`UseSettings`

If True, the visibility of selected/unselected nodes in the model window depends on the Show/Hide Selected setting.

If False, the visibility of selected/unselected nodes in the model window is not affected by the Show/Hide Selected setting.

St7SetFreeNodes

Sets the display option for free nodes.

```
long St7SetFreeNodes(long uID, long Style)
```

Input Parameters

`uID`

Strand7 model file ID.

`Style`

One of nsFreeNodeAll, nsFreeNodeNone, nsFreeNodeGroup or nsFreeNodeGroupFree.

St7SetNodeLabelStyle

Sets the display style for node labels.

```
long St7SetNodeLabelStyle(long uID, long LabelStyle)
```

Input Parameters

`uID`

Strand7 model file ID.

`LabelStyle`

One of lsNone, lsEntityNumber or lsIDNumber.

Dependencies

`Font`

Assigned using *St7SetEntityFont*.

St7SetNodeColours

Sets the colours used to display nodes.

```
long St7SetNodeColours(long uID, long* Colours, long NumCol)
```

Input Parameters

`uID`

Strand7 model file ID.

`Colours[0..1]`

`[clNodeUnselected]` – The unselected node colour.

`[clNodeSelected]` – The selected node colour.

See also *RGB Colours*.

`NumCol`

The size of Colours.

St7GetNodeStyle

Returns the display symbol for nodes.

```
long St7GetNodeStyle(long uID, long* Style)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Style`

Either psCircle or psSquare.

St7GetNodeSize

Returns the size index of the displayed nodes.

```
long St7GetNodeSize(long uID, long* Size)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Size`

Node size index in the range kMinPointSize to kMaxPointSize.

St7GetNodeShowHideSelected

Returns whether the display of nodes considers the Show/Hide Selected setting.

```
long St7GetNodeShowHideSelected(long uID, bool* UseSettings)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`UseSettings`

If True, the visibility of selected/unselected nodes in the model window depends on the Show/Hide Selected setting; if False, the visibility of selected/unselected nodes in the model window is not affected by the Show/Hide Selected setting.

St7GetFreeNodes

Returns the display option for free nodes.

```
long St7GetFreeNodes(long uID, long* Style)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Style`

One of nsFreeNodeAll, nsFreeNodeNone, nsFreeNodeGroup or nsFreeNodeGroupFree.

St7GetNodeLabelStyle

Returns the display style for node labels.

```
long St7GetNodeLabelStyle(long uID, long* LabelStyle)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`LabelStyle`

One of lsNone, lsEntityNumber or lsIDNumber.

St7GetNodeColours

Returns the colours used to display nodes.

```
long St7GetNodeColours(long uID, long* Colours, long NumCol)
```

Input Parameters

`uID`

Strand7 model file ID.

`NumCol`

The size of Colours.

Output Parameters

`Colours[0..1]`

`[clNodeUnselected]` – The unselected node colour.

`[clNodeSelected]` – The selected node colour.

See also *RGB Colours*.

Beam Entity Display

St7SetBeamStyle

Sets the display style for beams.

```
long St7SetBeamStyle(long uID, long Style)
```

Input Parameters

`uID`

Strand7 model file ID.

`Style`

One of bsLine, bsSection, bsSolid or bsSlice.

St7SetBeamCableAsLine

Sets the option for cables to ignore the beam style and use bsLine instead.

```
long St7SetBeamCableAsLine(long uID, bool AsLine)
```

Input Parameters

`uID`

Strand7 model file ID.

`AsLine`

If True, the cable is rendered using the bsLine style.

If False, the cable is rendered using the set beam style.

St7SetBeamFill

Sets the fill colour scheme for beams.

```
long St7SetBeamFill(long uID, long Fill)
```

Input Parameters

`uID`

Strand7 model file ID.

`Fill`

One of bfNone, bfProperty, bfGroup, bfColour, bfOrientation or bfContour.

St7SetBeamOutline

Sets the outline colour scheme for beams.

```
long St7SetBeamOutline(long uID, long Outline)
```

Input Parameters

`uID`

Strand7 model file ID.

`Outline`

One of blNone, blProperty, blGroup, blColour, blOrientation or blContour.

St7SetBeamLineThickness

Sets the thickness of beam outlines.

```
long St7SetBeamLineThickness(long uID, long Thickness)
```

Input Parameters

`uID`

Strand7 model file ID.

`Thickness`

Beam outline thickness in the range kMinThickness to kMaxThickness.

St7SetBeamLabelStyle

Sets the display style for beam labels.

```
long St7SetBeamLabelStyle(long uID, long LabelStyle)
```

Input Parameters

`uID`

Strand7 model file ID.

`LabelStyle`

One of lsNone, lsEntityNumber, lsIDNumber, lsPropertyNumber, lsPropertyName or lsPropertyType.

Dependencies

`Font`

Assigned using *St7SetEntityFont*.

St7SetBeamColours

Sets the colours to be used for given beam display types, where such are required.

```
long St7SetBeamColours(long uID, long* Colours, long NumCol)
```

Input Parameters

`uID`

Strand7 model file ID.

`Colours[0..4]`

`[ipBeamFillColour]` – The fill colour.

`[ipBeamLineColour]` – The outline colour.

`[ipBeamOrientation1Colour]` – The end 1 orientation colour.

`[ipBeamOrientation2Colour]` – The end 2 orientation colour.

`[ipBeamNRefColour]` – The orientation edge colour.

See also *RGB Colours*.

`NumCol`

The size of Colours.

St7SetBeamLighting

Sets the state of the **Lighting** options for beam rendering.

```
long St7SetBeamLighting(long uID, bool FillLighting, bool LineLighting)
```

Input Parameters

`uID`

Strand7 model file ID.

`FillLighting`

The lighting option for beam fill; either True or False.

`LineLighting`

The lighting option for beam outlines; either True or False.

St7SetBeamNRef

Sets whether the reference node is shown in the display of beam3 elements.

```
long St7SetBeamNRef(long uID, bool Show)
```

Input Parameters

`uID`

Strand7 model file ID.

`Show`

The state of the **Show NRef** option; either True or False.

St7SetBeamOffsetNodes

Sets whether to **Show Offset Nodes** in the display of beams.

```
long St7SetBeamOffsetNodes(long uID, bool Show)
```

Input Parameters

`uID`

Strand7 model file ID.

`Show`

The state of the **Show Offset Nodes** option; either True or False.

St7SetBeamMoveToOffset

Sets whether to **Move to Offset** the display of beams.

```
long St7SetBeamMoveToOffset(long uID, bool Show)
```

Input Parameters

`uID`

Strand7 model file ID.

`Show`

The state of the **Move to Offset** option; either True or False.

St7SetBeamDrawAxes

Sets whether to **Draw Axes** with the display of beams.

```
long St7SetBeamDrawAxes(long uID, bool Show)
```

Input Parameters

`uID`

Strand7 model file ID.

`Show`

The state of the **Draw Axes** option; either True or False.

St7SetBeamSpringCoils

Sets the number of coils displayed for spring elements.

```
long St7SetBeamSpringCoils(long uID, long Coils)
```

Input Parameters

`uID`

Strand7 model file ID.

`Coils`

The number of coils to display in the range kMinSpringCoils to kMaxSpringCoils.

St7SetBeamSpringAspect

Sets the aspect ratio displayed for spring elements.

```
long St7SetBeamSpringAspect(long uID, long Aspect)
```

Input Parameters

`uID`

Strand7 model file ID.

`Aspect`

The spring element aspect ratio in the range kMinSpringAspect to kMaxSpringAspect.

St7SetBeamRoundFacets

Sets the number of circumferential facets used to render circular beam sections and spring elements.

```
long St7SetBeamRoundFacets(long uID, long Facets)
```

Input Parameters

`uID`

Strand7 model file ID.

`Facets`

The number of facets around one revolution in the range kMinFacets to kMaxFacets.

St7SetBeamSlices

Sets the number of straight line segments used to render curved beam elements.

```
long St7SetBeamSlices(long uID, long Slices)
```

Input Parameters

`uID`

Strand7 model file ID.

`Slices`

The number of segments along a beam element in the range kMinSlices to kMaxSlices.

St7SetBeamShrink

Sets a percentage by which to shrink the display of beam elements.

```
long St7SetBeamShrink(long uID, long Shrink)
```

Input Parameters

`uID`

Strand7 model file ID.

`Shrink`

The **Shrink** percentage in the range kMinShrink to kMaxShrink.

St7GetBeamStyle

Returns the display style for beams.

```
long St7GetBeamStyle(long uID, long* Style)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Style`

One of bsLine, bsSection, bsSolid or bsSlice.

St7GetBeamCableAsLine

Returns the option for cables to ignore the beam style and use bsLine instead.

```
long St7GetBeamCableAsLine(long uID, bool* AsLine)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`AsLine`

If True, the cable is rendered using the bsLine style; if False, the cable is rendered using the set beam style.

St7GetBeamFill

Returns the fill colour scheme for beams.

```
long St7GetBeamFill(long uID, long* Fill)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Fill`

One of bfNone, bfProperty, bfGroup, bfColour, bfOrientation or bfContour.

St7GetBeamOutline

Returns the outline colour scheme for beams.

```
long St7GetBeamOutline(long uID, long* Outline)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Outline`

One of blNone, blProperty, blGroup, blColour, blOrientation or blContour.

St7GetBeamLineThickness

Returns the thickness of beam outlines.

```
long St7GetBeamLineThickness(long uID, long* Thickness)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Thickness`

Beam outline thickness in the range kMinThickness to kMaxThickness.

St7GetBeamLabelStyle

Returns the display style for beam labels.

```
long St7GetBeamLabelStyle(long uID, long* LabelStyle)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`LabelStyle`

One of lsNone, lsEntityNumber, lsIDNumber, lsPropertyNumber, lsPropertyName or lsPropertyType.

St7GetBeamColours

Returns the colours to be used for given beam display types.

```
long St7GetBeamColours(long uID, long* Colours, long NumCol)
```

Input Parameters

`uID`

Strand7 model file ID.

`NumCol`

The size of Colours.

Output Parameters

`Colours[0..4]`

`[ipBeamFillColour]` – The fill colour.

`[ipBeamLineColour]` – The outline colour.

`[ipBeamOrientation1Colour]` – The end 1 orientation colour.

`[ipBeamOrientation2Colour]` – The end 2 orientation colour.

`[ipBeamNRefColour]` – The orientation edge colour.

See also *RGB Colours*.

St7GetBeamLighting

Returns the state of the **Lighting** options for beam rendering.

```
long St7GetBeamLighting(long uID, bool* FillLighting, bool* LineLighting)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`FillLighting`

The lighting option for beam fill; either True or False.

`LineLighting`

The lighting option for beam outlines; either True or False.

St7GetBeamNRef

Returns whether the reference node is shown in the display of beam3 elements.

```
long St7GetBeamNRef(long uID, bool* Show)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Show`

The state of the **Show NRef** option; either True or False.

St7GetBeamOffsetNodes

Returns whether to **Show Offset Nodes** in the display of beams.

```
long St7GetBeamOffsetNodes(long uID, bool* Show)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Show`

The state of the **Show Offset Nodes** option; either True or False.

St7GetBeamMoveToOffset

Returns whether to **Move to Offset** the display of beams.

```
long St7GetBeamMoveToOffset(long uID, bool* Show)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Show`

The state of the **Move to Offset** option; either True or False.

St7GetBeamDrawAxes

Returns whether to **Draw Axes** with the display of beams.

```
long St7GetBeamDrawAxes(long uID, bool* Show)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Show`

The state of the **Draw Axes** option; either True or False.

St7GetBeamSpringCoils

Returns the number of coils displayed for spring elements.

```
long St7GetBeamSpringCoils(long uID, long* Coils)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Coils`

The number of coils to display in the range kMinSpringCoils to kMaxSpringCoils.

St7GetBeamSpringAspect

Returns the aspect ratio displayed for spring elements.

```
long St7GetBeamSpringAspect(long uID, long* Aspect)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Aspect`

The spring element aspect ratio in the range kMinSpringAspect to kMaxSpringAspect.

St7GetBeamRoundFacets

Returns the number of circumferential facets used to render circular beam sections and spring elements.

```
long St7GetBeamRoundFacets(long uID, long* Facets)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Facets`

The number of facets around one revolution in the range kMinFacets to kMaxFacets.

St7GetBeamSlices

Returns the number of straight line segments used to render curved beam elements.

```
long St7GetBeamSlices(long uID, long* Slices)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Slices`

The number of segments along a beam element in the range kMinSlices to kMaxSlices.

St7GetBeamShrink

Returns a percentage by which to shrink the display of beam elements.

```
long St7GetBeamShrink(long uID, long* Shrink)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Shrink`

The **Shrink** percentage in the range kMinShrink to kMaxShrink.

Plate Entity Display

St7SetPlateStyle

Sets the display style for plates.

```
long St7SetPlateStyle(long uID, long Style)
```

Input Parameters

`uID`

Strand7 model file ID.

`Style`

Plate display style; either psSurface or psSolid.

St7SetPlateAverageNormals

Sets whether faceted plates are displayed with averaged normals.

```
long St7SetPlateAverageNormals(long uID, bool AverageNormals)
```

Input Parameters

`uID`

Strand7 model file ID.

`AverageNormals`

True to display plates with averaged normals.

St7SetPlateAverageNormalsAngle

Sets the angle over which plate normal are averaged.

```
long St7SetPlateAverageNormalsAngle(long uID, long Angle)
```

Input Parameters

`uID`

Strand7 model file ID.

`Angle`

Angle (0 to 60 degrees).

St7SetPlateFill

Sets the fill colour scheme for plates.

```
long St7SetPlateFill(long uID, long Fill)
```

Input Parameters

`uID`

Strand7 model file ID.

`Fill`

One of pfNone, pfProperty, pfGroup, pfColour, pfOrientation or pfContour.

St7SetPlateOutline

Sets the outline colour scheme for plates.

```
long St7SetPlateOutline(long uID, long Outline)
```

Input Parameters

`uID`

Strand7 model file ID.

`Outline`

One of plNone, plProperty,plGroup or plColour.

St7SetPlateLineThickness

Sets the thickness of plate outlines.

```
long St7SetPlateLineThickness(long uID, long Thickness)
```

Input Parameters

`uID`

Strand7 model file ID.

`Thickness`

Plate outline thickness in the range kMinThickness to kMaxThickness.

St7SetPlateLabelStyle

Sets the display style for plate labels.

```
long St7SetPlateLabelStyle(long uID, long LabelStyle)
```

Input Parameters

`uID`

Strand7 model file ID.

`LabelStyle`

One of lsNone, lsEntityNumber, lsIDNumber, lsPropertyNumber, lsPropertyName or lsPropertyType.

Dependencies

`Font`

Assigned using *St7SetEntityFont*.

St7SetPlateColours

Sets the colours to be used for given plate display types, where such are required.

```
long St7SetPlateColours(long uID, long* Colours, long NumCol)
```

Input Parameters

`uID`

Strand7 model file ID.

`Colours[0..5]`

`[ipPlateFillColour]` – The fill colour.

`[ipPlateLineColour]` – The outline colour.

`[ipPlateOrientation1Colour]` – The -z orientation colour.

`[ipPlateOrientation2Colour]` – The +z orientation colour.

`[ipPlateOrientation3Colour]` – The orientation edge colour.

`[ipPlateOffsetColour]` – The offset line colour.

See also *RGB Colours*.

`NumCol`

The size of Colours.

St7SetPlateLighting

Sets the state of the **Lighting** options for plate rendering.

```
long St7SetPlateLighting(long uID, bool FillLighting, bool LineLighting)
```

Input Parameters

`uID`

Strand7 model file ID.

`FillLighting`

The lighting option for plate fill; either True or False.

`LineLighting`

The lighting option for plate outlines; either True or False.

St7SetPlateOffsetNodes

Sets whether to **Show Offset Nodes** in the display of plates.

```
long St7SetPlateOffsetNodes(long uID, bool Show)
```

Input Parameters

`uID`

Strand7 model file ID.

`Show`

The state of the **Show Offset Nodes** option; either True or False.

St7SetPlateMoveToOffset

Sets whether to **Move to Offset** the display of plates.

```
long St7SetPlateMoveToOffset(long uID, bool Show)
```

Input Parameters

`uID`

Strand7 model file ID.

`Show`

The state of the **Move to Offset** option; either True or False.

St7SetPlateDrawAxes

Sets whether to **Draw Axes** with the display of plates.

```
long St7SetPlateDrawAxes(long uID, bool Show)
```

Input Parameters

`uID`

Strand7 model file ID.

`Show`

The state of the **Draw Axes** option; either True or False.

St7SetPlateShrink

Sets a percentage by which to shrink the display of plate elements.

```
long St7SetPlateShrink(long uID, long Shrink)
```

Input Parameters

`uID`

Strand7 model file ID.

`Shrink`

The **Shrink** percentage in the range kMinShrink to kMaxShrink.

St7SetPlateFaceNodes

Sets whether to show **Face Nodes** in the solid render of plates.

```
long St7SetPlateFaceNodes(long uID, bool Show)
```

Input Parameters

`uID`

Strand7 model file ID.

`Show`

The state of the **Face Nodes** option; either True or False.

St7SetPlateAxisLayer

For laminate properties, this sets the laminate layer whose material axes are displayed when **Draw Axes** is set.

```
long St7SetPlateAxisLayer(long uID, long Layer)
```

Input Parameters

`uID`

Strand7 model file ID.

`Layer`

The number of the laminate layer whose axes are to be drawn, or Layer = 0 to display the plate local axes instead.

St7SetPlateOutlineMode

Sets where outlines are drawn around plates.

```
long St7SetPlateOutlineMode(long uID, long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`Mode`

One of omEdge, omPropertyBoundary, omGroupBoundary, omFacetAngle, omFacetProperty or omFacetGroup.

St7GetPlateStyle

Returns the display style for plates.

```
long St7GetPlateStyle(long uID, long* Style)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Style`

Plate display style; either psSurface or psSolid.

St7GetPlateAverageNormals

Returns whether faceted plates are displayed with averaged normals.

```
long St7GetPlateAverageNormals(long uID, bool* AverageNormals)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`AverageNormals`

True to display plates with averaged normals.

St7GetPlateAverageNormalsAngle

Returns the angle over which plate normal are averaged.

```
long St7GetPlateAverageNormalsAngle(long uID, long* Angle)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Angle`

Angle (0 to 60 degrees).

St7GetPlateFill

Returns the fill colour scheme for plates.

```
long St7GetPlateFill(long uID, long* Fill)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Fill`

One of pfNone, pfProperty, pfGroup, pfColour, pfOrientation or pfContour.

St7GetPlateOutline

Returns the outline colour scheme for plates.

```
long St7GetPlateOutline(long uID, long* Outline)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Outline`

One of plNone, plProperty,plGroup or plColour.

St7GetPlateLineThickness

Returns the thickness of plate outlines.

```
long St7GetPlateLineThickness(long uID, long* Thickness)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Thickness`

Plate outline thickness in the range kMinThickness to kMaxThickness.

St7GetPlateLabelStyle

Returns the display style for plate labels.

```
long St7GetPlateLabelStyle(long uID, long* LabelStyle)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`LabelStyle`

One of lsNone, lsEntityNumber, lsIDNumber, lsPropertyNumber, lsPropertyName or lsPropertyType.

St7GetPlateColours

Returns the colours to be used for given plate display types.

```
long St7GetPlateColours(long uID, long* Colours, long NumCol)
```

Input Parameters

`uID`

Strand7 model file ID.

`NumCol`

The size of Colours.

Output Parameters

`Colours[0..5]`

`[ipPlateFillColour]` – The fill colour.

`[ipPlateLineColour]` – The outline colour.

`[ipPlateOrientation1Colour]` – The -z orientation colour.

`[ipPlateOrientation2Colour]` – The +z orientation colour.

`[ipPlateOrientation3Colour]` – The orientation edge colour.

`[ipPlateOffsetColour]` – The offset line colour.

See also *RGB Colours*.

St7GetPlateLighting

Returns the state of the **Lighting** options for plate rendering.

```
long St7GetPlateLighting(long uID, bool* FillLighting, bool* LineLighting)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`FillLighting`

The lighting option for plate fill; either True or False.

`LineLighting`

The lighting option for plate outlines; either True or False.

St7GetPlateOffsetNodes

Returns whether to **Show Offset Nodes** in the display of plates.

```
long St7GetPlateOffsetNodes(long uID, bool* Show)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Show`

The state of the **Show Offset Nodes** option; either True or False.

St7GetPlateMoveToOffset

Returns whether to **Move to Offset** the display of plates.

```
long St7GetPlateMoveToOffset(long uID, bool* Show)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Show`

The state of the **Move to Offset** option; either True or False.

St7GetPlateDrawAxes

Returns whether to **Draw Axes** with the display of plates.

```
long St7GetPlateDrawAxes(long uID, bool* Show)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Show`

The state of the **Draw Axes** option; either True or False.

St7GetPlateShrink

Returns a percentage by which to shrink the display of plate elements.

```
long St7GetPlateShrink(long uID, long* Shrink)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Shrink`

The **Shrink** percentage in the range kMinShrink to kMaxShrink.

St7GetPlateFaceNodes

Returns whether to show **Face Nodes** in the solid render of plates.

```
long St7GetPlateFaceNodes(long uID, bool* Show)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Show`

The state of the **Face Nodes** option; either True or False.

St7GetPlateAxisLayer

For laminate properties, this returns the laminate layer whose material axes are displayed when **Draw Axes** is set.

```
long St7GetPlateAxisLayer(long uID, long* Layer)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Layer`

The number of the laminate layer whose axes are to be drawn, or Layer = 0 to display the plate local axes instead.

St7GetPlateOutlineMode

Returns where outlines are drawn around plates.

```
long St7GetPlateOutlineMode(long uID, long* Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Mode`

One of omEdge, omPropertyBoundary, omGroupBoundary, omFacetAngle, omFacetProperty or omFacetGroup.

Brick Entity Display

St7SetBrickFill

Sets the fill colour scheme for bricks.

```
long St7SetBrickFill(long uID, long Fill)
```

Input Parameters

`uID`

Strand7 model file ID.

`Fill`

One of kfNone, kfProperty, kfGroup, kfColour, or kfContour.

St7SetBrickOutline

Sets the outline colour scheme for bricks.

```
long St7SetBrickOutline(long uID, long Outline)
```

Input Parameters

`uID`

Strand7 model file ID.

`Outline`

One of klNone, klProperty,klGroup or klColour.

St7SetBrickLineThickness

Sets the thickness of the brick outlines.

```
long St7SetBrickLineThickness(long uID, long Thickness)
```

Input Parameters

`uID`

Strand7 model file ID.

`Thickness`

The brick outline thickness in the range kMinThickness to kMaxThickness.

St7SetBrickLabelStyle

Sets the display style for brick labels.

```
long St7SetBrickLabelStyle(long uID, long LabelStyle)
```

Input Parameters

`uID`

Strand7 model file ID.

`LabelStyle`

One of lsNone, lsEntityNumber, lsIDNumber, lsPropertyNumber, lsPropertyName or lsPropertyType.

Dependencies

`Font`

Assigned using *St7SetEntityFont*.

St7SetBrickColours

Sets the colours to be used for given brick display types, where such are required.

```
long St7SetBrickColours(long uID, long* Colours, long NumCol)
```

Input Parameters

`uID`

Strand7 model file ID.

`Colours[0..1]`

`[ipBrickFillColour]` – The fill colour.

`[ipBrickLineColour]` – The outline colour.

See also *RGB Colours*.

`NumCol`

The size of Colours.

St7SetBrickLighting

Sets the state of the **Lighting** options for brick rendering.

```
long St7SetBrickLighting(long uID, bool FillLighting, bool LineLighting)
```

Input Parameters

`uID`

Strand7 model file ID.

`FillLighting`

The lighting option for the brick fill; either btTrue or btFalse.

`LineLighting`

The lighting option for the brick outlines; either True or False.

St7SetBrickDrawAxes

Sets whether local brick axes are displayed with bricks.

```
long St7SetBrickDrawAxes(long uID, bool Show1, bool Show2, bool Show3)
```

Input Parameters

`uID`

Strand7 model file ID.

`Show1`

The state of the **Draw x Axis** option; either True or False.

`Show2`

The state of the **Draw y Axis** option; either True or False.

`Show3`

The state of the **Draw z Axis** option; either True or False.

St7SetBrickShrink

Sets a percentage by which to shrink the display of brick elements.

```
long St7SetBrickShrink(long uID, long Shrink)
```

Input Parameters

`uID`

Strand7 model file ID.

`Shrink`

The **Shrink** percentage in the range kMinShrink to kMaxShrink.

St7SetBrickOutlineMode

Sets where outlines are drawn around bricks.

```
long St7SetBrickOutlineMode(long uID, long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`Mode`

One of omEdge, omPropertyBoundary, omGroupBoundary, omFacetAngle, omFacetProperty or omFacetGroup.

St7SetBrickWireframeAll

Sets the state of the **Draw all faces when showing wireframes** option for brick display.

```
long St7SetBrickWireframeAll(long uID, bool Show)
```

Input Parameters

`uID`

Strand7 model file ID.

`Show`

The state of the **Draw all faces when showing wireframes** option; either True or False.

St7GetBrickFill

Returns the fill colour scheme for bricks.

```
long St7GetBrickFill(long uID, long* Fill)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Fill`

One of kfNone, kfProperty, kfGroup, kfColour, or kfContour.

St7GetBrickOutline

Returns the outline colour scheme for bricks.

```
long St7GetBrickOutline(long uID, long* Outline)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Outline`

One of klNone, klProperty,klGroup or klColour.

St7GetBrickLineThickness

Returns the thickness of the brick outlines.

```
long St7GetBrickLineThickness(long uID, long* Thickness)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Thickness`

The brick outline thickness in the range kMinThickness to kMaxThickness.

St7GetBrickLabelStyle

Returns the display style for brick labels.

```
long St7GetBrickLabelStyle(long uID, long* LabelStyle)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`LabelStyle`

One of lsNone, lsEntityNumber, lsIDNumber, lsPropertyNumber, lsPropertyName or lsPropertyType.

St7GetBrickColours

Returns the colours to be used for given brick display types.

```
long St7GetBrickColours(long uID, long* Colours, long NumCol)
```

Input Parameters

`uID`

Strand7 model file ID.

`NumCol`

The size of Colours.

Output Parameters

`Colours[0..1]`

`[ipBrickFillColour]` – The fill colour.

`[ipBrickLineColour]` – The outline colour.

See also *RGB Colours*.

St7GetBrickLighting

Returns the state of the **Lighting** options for brick rendering.

```
long St7GetBrickLighting(long uID, bool* FillLighting, bool* LineLighting)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`FillLighting`

The lighting option for the brick fill; either True or False.

`LineLighting`

The lighting option for the brick outlines; either btTrue or btFalse.

St7GetBrickDrawAxes

Returns whether local brick axes are displayed with bricks.

```
long St7GetBrickDrawAxes(long uID, bool* Show1, bool* Show2, bool* Show3)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Show1`

The state of the **Draw x Axis** option; either True or False.

`Show2`

The state of the **Draw y Axis** option; either True or False.

`Show3`

The state of the **Draw z Axis** option; either True or False.

St7GetBrickShrink

Returns a percentage by which to shrink the display of brick elements.

```
long St7GetBrickShrink(long uID, long* Shrink)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Shrink`

The **Shrink** percentage in the range kMinShrink to kMaxShrink.

St7GetBrickOutlineMode

Returns where outlines are drawn around bricks.

```
long St7GetBrickOutlineMode(long uID, long* Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Mode`

One of omEdge, omPropertyBoundary, omGroupBoundary, omFacetAngle, omFacetProperty or omFacetGroup.

St7GetBrickWireframeAll

Returns the state of the **Draw all faces when showing wireframes** option for brick display.

```
long St7GetBrickWireframeAll(long uID, bool* Show)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Show`

The state of the **Draw all faces when showing wireframes** option; either True or False.

Link Entity Display

St7SetLinkOutline

Sets the colour scheme for links.

```
long St7SetLinkOutline(long uID, long Outline)
```

Input Parameters

`uID`

Strand7 model file ID.

`Outline`

One of llType, llGroup, or llGlobal.

St7SetLinkLineThickness

Sets the thickness of the link display.

```
long St7SetLinkLineThickness(long uID, long Thickness)
```

Input Parameters

`uID`

Strand7 model file ID.

`Thickness`

The link display thickness in the range kMinThickness to kMaxThickness.

St7SetLinkLabelStyle

Sets whether to label links with **Link Numbers**.

```
long St7SetLinkLabelStyle(long uID, long LabelStyle)
```

Input Parameters

`uID`

Strand7 model file ID.

`LabelStyle`

One of lsNone, lsEntityNumber, lsIDNumber, or lsLinkType.

Dependencies

`Font`

Assigned using *St7SetEntityFont*.

St7SetLinkColours

Sets the colours used for displaying links.

```
long St7SetLinkColours(long uID, long* Colours, long NumCol)
```

Input Parameters

`uID`

Strand7 model file ID.

`Colours[0..14]`

`[ipLinkColour]` – The global link colour.

`[ipMasterSlaveColour]` – The colour of master slave links.

`[ipSectorSymmetryColour]` – The colour of sector-symmetry links.

`[ipCouplingColour]` – The colour of coupling links.

`[ipPinnedColour]` – The colour of pinned links.

`[ipRigidColour]` – The colour of rigid links.

`[ipShrinkColour]` – The colour of shrink links.

`[ipTwoPointColour]` – The colour of two-point links.

`[ipAttachmentColour]` – The colour of attachment links.

`[ipInterpolatedMPLColour]` – The colour of interpolated multi-point links.

`[ipMasterSlaveMPLColour]` – The colour of master-slave multi-point links.

`[ipPinnedMPLColour]` – The colour of pinned multi-point links.

`[ipRigidMPLColour]` – The colour of rigid multi-point links.

`[ipUserMPLColour]` – The colour of user-defined multi-point links.

`[ipReactionMPLColour]` – The colour of reaction multi-point links.

See also *RGB Colours*.

`NumCol`

The size of Colours.

St7SetLinkDashes

Sets whether dashes indicating linked degrees of freedom are drawn at the ends of links.

```
long St7SetLinkDashes(long uID, bool Show)
```

Input Parameters

`uID`

Strand7 model file ID.

`Show`

The state of the **Show DoF Dashes** option; either True or False.

St7GetLinkOutline

Returns the colour scheme for links.

```
long St7GetLinkOutline(long uID, long* Outline)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Outline`

One of llType, llGroup, or llGlobal.

St7GetLinkLineThickness

Returns the thickness of the link display.

```
long St7GetLinkLineThickness(long uID, long* Thickness)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Thickness`

The link display thickness in the range kMinThickness to kMaxThickness.

St7GetLinkLabelStyle

Returns whether to label links with **Link Numbers**.

```
long St7GetLinkLabelStyle(long uID, long* LabelStyle)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`LabelStyle`

One of lsNone, lsEntityNumber, lsIDNumber, or lsLinkType.

St7GetLinkColours

Returns the colours used for displaying links.

```
long St7GetLinkColours(long uID, long* Colours, long NumCol)
```

Input Parameters

`uID`

Strand7 model file ID.

`NumCol`

The size of Colours.

Output Parameters

`Colours[0..14]`

`[ipLinkColour]` – The global link colour.

`[ipMasterSlaveColour]` – The colour of master slave links.

`[ipSectorSymmetryColour]` – The colour of sector-symmetry links.

`[ipCouplingColour]` – The colour of coupling links.

`[ipPinnedColour]` – The colour of pinned links.

`[ipRigidColour]` – The colour of rigid links.

`[ipShrinkColour]` – The colour of shrink links.

`[ipTwoPointColour]` – The colour of two-point links.

`[ipAttachmentColour]` – The colour of attachment links.

`[ipInterpolatedMPLColour]` – The colour of interpolated multi-point links.

`[ipMasterSlaveMPLColour]` – The colour of master-slave multi-point links.

`[ipPinnedMPLColour]` – The colour of pinned multi-point links.

`[ipRigidMPLColour]` – The colour of rigid multi-point links.

`[ipUserMPLColour]` – The colour of user-defined multi-point links.

`[ipReactionMPLColour]` – The colour of reaction multi-point links.

See also *RGB Colours*.

St7GetLinkDashes

Returns whether dashes indicating linked degrees of freedom are drawn at the ends of links.

```
long St7GetLinkDashes(long uID, bool* Show)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Show`

The state of the **Show DoF Dashes** option; either True or False.

Vertex Entity Display

St7SetVertexStyle

Sets the display symbol for vertices.

```
long St7SetVertexStyle(long uID, long Style)
```

Input Parameters

`uID`

Strand7 model file ID.

`Style`

Either psCircle or psSquare.

St7SetVertexSize

Sets the size index of the displayed vertices.

```
long St7SetVertexSize(long uID, long Size)
```

Input Parameters

`uID`

Strand7 model file ID.

`Size`

The vertex size index in the range kMinPointSize to kMaxPointSize.

St7SetVertexShowHideSelected

Sets whether the display of vertices considers the Show/Hide Selected setting.

```
long St7SetVertexShowHideSelected(long uID, bool UseSettings)
```

Input Parameters

`uID`

Strand7 model file ID.

`UseSettings`

If True, the visibility of selected/unselected vertices in the model window depends on the Show/Hide Selected setting.

If False, the visibility of selected/unselected vertices in the model window is not affected by the Show/Hide Selected setting.

St7SetFreeVertices

Sets the display option for free vertices.

```
long St7SetFreeVertices(long uID, long Style)
```

Input Parameters

`uID`

Strand7 model file ID.

`Style`

One of vsFreeVertexAll, vsFreeVertexNone or vsFreeVertexGroup.

St7SetVertexLabelStyle

Sets the display style for vertex labels.

```
long St7SetVertexLabelStyle(long uID, long LabelStyle)
```

Input Parameters

`uID`

Strand7 model file ID.

`LabelStyle`

One of lsNone, lsEntityNumber or lsIDNumber.

Dependencies

`Font`

Assigned using *St7SetEntityFont*.

St7SetVertexColours

Sets the colours used to display vertices.

```
long St7SetVertexColours(long uID, long* Colours, long NumCol)
```

Input Parameters

`uID`

Strand7 model file ID.

`Colours[0..2]`

`[ipVertexFreeColour]` – The unselected vertex colour.

`[ipVertexFixedColour]` – The fixed vertex colour.

`[ipVertexSelectedColour]` – The selected vertex colour.

See also *RGB Colours*.

`NumCol`

The size of Colours.

St7GetVertexStyle

Returns the display symbol for vertices.

```
long St7GetVertexStyle(long uID, long* Style)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Style`

Either psCircle or psSquare.

St7GetVertexSize

Returns the size index of the displayed vertices.

```
long St7GetVertexSize(long uID, long* Size)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Size`

The vertex size index in the range kMinPointSize to kMaxPointSize.

St7GetVertexShowHideSelected

Returns whether the display of vertices considers the Show/Hide Selected setting.

```
long St7GetVertexShowHideSelected(long uID, bool* UseSettings)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`UseSettings`

If True, the visibility of selected/unselected vertices in the model window depends on the Show/Hide Selected setting; if False, the visibility of selected/unselected vertices in the model window is not affected by the Show/Hide Selected setting.

St7GetFreeVertices

Returns the display option for free vertices.

```
long St7GetFreeVertices(long uID, long* Style)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Style`

One of vsFreeVertexAll, vsFreeVertexNone or vsFreeVertexGroup.

St7GetVertexLabelStyle

Returns the display style for vertex labels.

```
long St7GetVertexLabelStyle(long uID, long* LabelStyle)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`LabelStyle`

One of lsNone, lsEntityNumber or lsIDNumber.

St7GetVertexColours

Returns the colours used to display vertices.

```
long St7GetVertexColours(long uID, long* Colours, long NumCol)
```

Input Parameters

`uID`

Strand7 model file ID.

`NumCol`

The size of Colours.

Output Parameters

`Colours[0..2]`

`[ipVertexFreeColour]` – The unselected vertex colour.

`[ipVertexFixedColour]` – The fixed vertex colour.

`[ipVertexSelectedColour]` – The selected vertex colour.

See also *RGB Colours*.

Face Entity Display

St7SetFaceFillStyle

Sets the display style for faces.

```
long St7SetFaceFillStyle(long uID, long Style)
```

Input Parameters

`uID`

Strand7 model file ID.

`Style`

One of fdNone, fdWireframe or fdSolid.

St7SetFaceFill

Sets the fill colour scheme for faces.

```
long St7SetFaceFill(long uID, long Fill)
```

Input Parameters

`uID`

Strand7 model file ID.

`Fill`

One of ffProperty, ffGroup, ffFaceNumber, ffFaceID, ffColour or ffOrientation.

St7SetFaceOutline

Sets the outline colour scheme for faces.

```
long St7SetFaceOutline(long uID, long Outline)
```

Input Parameters

`uID`

Strand7 model file ID.

`Outline`

One of flNone, flProperty, flGroup, flFaceNumber, flFaceID or flColour.

St7SetFaceLabelStyle

Sets the display style for face labels.

```
long St7SetFaceLabelStyle(long uID, long LabelStyle)
```

Input Parameters

`uID`

Strand7 model file ID.

`LabelStyle`

One of lsNone, lsEntityNumber, lsIDNumber or lsPropertyNumber.

Dependencies

`Font`

Assigned using *St7SetEntityFont*.

St7SetFaceColours

Sets the colours to be used for given face display types, where such are required.

```
long St7SetFaceColours(long uID, long* Colours, long NumCol)
```

Input Parameters

`uID`

Strand7 model file ID.

`Colours[0..7]`

`[ipFaceFillColour]` – The fill colour.

`[ipFaceLineColour]` – The outline colour.

`[ipFaceOrientation1Colour]` – The -z orientation colour.

`[ipFaceOrientation2Colour]` – The +z orientation colour.

`[ipFaceNIEdgesColour]` – The non-interpolated edge highlight colour.

`[ipFaceCPuColour]` – The colour of control point u-lines.

`[ipFaceCPvColour]` – The colour of control point v-lines.

`[ipFaceNormalsColour]` – The colour of normal vectors.

See also *RGB Colours*.

`NumCol`

The size of Colours.

St7SetFaceLighting

Sets the state of the **Lighting** options for face rendering.

```
long St7SetFaceLighting(long uID, bool FillLighting, bool LineLighting)
```

Input Parameters

`uID`

Strand7 model file ID.

`FillLighting`

The lighting option for the face fill; either True or False.

`LineLighting`

The lighting option for the face outlines; either True or False.

St7SetFaceLineThickness

Sets the thickness of the face outlines.

```
long St7SetFaceLineThickness(long uID, long Thickness)
```

Input Parameters

`uID`

Strand7 model file ID.

`Thickness`

The face **Outline Thickness** in the range kMinThickness to kMaxThickness.

St7SetFaceWireThickness

Sets the thickness of the face wireframes.

```
long St7SetFaceWireThickness(long uID, long Thickness)
```

Input Parameters

`uID`

Strand7 model file ID.

`Thickness`

The face **Wireframe Thickness** in the range kMinThickness to kMaxThickness.

St7SetFaceWireDensity

Sets the density of the face wireframes.

```
long St7SetFaceWireDensity(long uID, long Density)
```

Input Parameters

`uID`

Strand7 model file ID.

`Density`

The face **Wireframe Density** in the range 1 to 100.

St7SetFaceNormalsSize

Sets the size index of the face normals.

```
long St7SetFaceNormalsSize(long uID, long Size)
```

Input Parameters

`uID`

Strand7 model file ID.

`Size`

The face **Normals Size** in the range 1 to 100.

St7SetFaceNIEdges

Sets whether to highlight non-interpolated face edges.

```
long St7SetFaceNIEdges(long uID, bool Show)
```

Input Parameters

`uID`

Strand7 model file ID.

`Show`

The state of the **Show Non-Interpolated Edges** option; either True or False.

St7SetFaceControlPoints

Sets whether to **Show Control Points** on the geometry faces.

```
long St7SetFaceControlPoints(long uID, bool Show)
```

Input Parameters

`uID`

Strand7 model file ID.

`Show`

The state of the **Show Control Points** option; either True or False.

St7SetFaceNormals

Sets whether to **Show Normals** on the geometry faces.

```
long St7SetFaceNormals(long uID, bool Show)
```

Input Parameters

`uID`

Strand7 model file ID.

`Show`

The state of the **Show Normals** option; either True or False.

St7GetFaceFillStyle

Returns the display style for faces.

```
long St7GetFaceFillStyle(long uID, long* Style)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Style`

One of fdNone, fdWireframe or fdSolid.

St7GetFaceFill

Returns the fill colour scheme for faces.

```
long St7GetFaceFill(long uID, long* Fill)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Fill`

One of ffProperty, ffGroup, ffFaceNumber, ffFaceID, ffColour or ffOrientation.

St7GetFaceOutline

Returns the outline colour scheme for faces.

```
long St7GetFaceOutline(long uID, long* Outline)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Outline`

One of flNone, flProperty, flGroup, flFaceNumber, flFaceID or flColour.

St7GetFaceLabelStyle

Returns the display style for face labels.

```
long St7GetFaceLabelStyle(long uID, long* LabelStyle)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`LabelStyle`

One of lsNone, lsEntityNumber, lsIDNumber or lsPropertyNumber.

St7GetFaceColours

Returns the colours to be used for given face display types.

```
long St7GetFaceColours(long uID, long* Colours, long NumCol)
```

Input Parameters

`uID`

Strand7 model file ID.

`NumCol`

The size of Colours.

Output Parameters

`Colours[0..7]`

`[ipFaceFillColour]` – The fill colour.

`[ipFaceLineColour]` – The outline colour.

`[ipFaceOrientation1Colour]` – The -z orientation colour.

`[ipFaceOrientation2Colour]` – The +z orientation colour.

`[ipFaceNIEdgesColour]` – The non-interpolated edge highlight colour.

`[ipFaceCPuColour]` – The colour of control point u-lines.

`[ipFaceCPvColour]` – The colour of control point v-lines.

`[ipFaceNormalsColour]` – The colour of normal vectors.

See also *RGB Colours*.

St7GetFaceLighting

Returns the state of the **Lighting** options for face rendering.

```
long St7GetFaceLighting(long uID, bool* FillLighting, bool* LineLighting)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`FillLighting`

The lighting option for the face fill; either True or False.

`LineLighting`

The lighting option for the face outlines; either True or False.

St7GetFaceLineThickness

Returns the thickness of the face outlines.

```
long St7GetFaceLineThickness(long uID, long* Thickness)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Thickness`

The face **Outline Thickness** in the range kMinThickness to kMaxThickness.

St7GetFaceWireThickness

Returns the thickness of the face wireframes.

```
long St7GetFaceWireThickness(long uID, long* Thickness)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Thickness`

The face **Wireframe Thickness** in the range kMinThickness to kMaxThickness.

St7GetFaceWireDensity

Returns the density of the face wireframes.

```
long St7GetFaceWireDensity(long uID, long* Density)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Density`

The face **Wireframe Density** in the range 1 to 100.

St7GetFaceNormalsSize

Returns the size index of the face normals.

```
long St7GetFaceNormalsSize(long uID, long* Size)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Size`

The face **Normals Size** in the range 1 to 100.

St7GetFaceNIEdges

Returns whether to highlight non-interpolated face edges.

```
long St7GetFaceNIEdges(long uID, bool* Show)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Show`

The state of the **Show Non-Interpolated Edges** option; either True or False.

St7GetFaceControlPoints

Returns whether to **Show Control Points** on the geometry faces.

```
long St7GetFaceControlPoints(long uID, bool* Show)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Show`

The state of the **Show Control Points** option; either True or False.

St7GetFaceNormals

Returns whether to **Show Normals** on the geometry faces.

```
long St7GetFaceNormals(long uID, bool* Show)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Show`

The state of the **Show Normals** option; either True or False.

Load Path Entity Display

St7SetPathFill

Sets the fill colour scheme for load paths.

```
long St7SetPathFill(long uID, long Fill)
```

Input Parameters

`uID`

Strand7 model file ID.

`Fill`

One of tfNone, tfTemplate, tfGroup, tfPathNumber, tfColour or tfOrientation.

St7SetPathOutline

Sets the outline colour scheme for load paths.

```
long St7SetPathOutline(long uID, long Outline)
```

Input Parameters

`uID`

Strand7 model file ID.

`Outline`

One of tlNone, tlTemplate, tlGroup, tlPathNumber or tlColour.

St7SetPathLabelStyle

Sets the display style for load path labels.

```
long St7SetPathLabelStyle(long uID, long LabelStyle)
```

Input Parameters

`uID`

Strand7 model file ID.

`LabelStyle`

One of lsNone, lsEntityNumber or lsLaneNumber.

Dependencies

`Font`

Assigned using *St7SetEntityFont*.

St7SetPathColours

Sets the colours to be used for given load path display types, where such are required.

```
long St7SetPathColours(long uID, long* Colours, long NumCol)
```

Input Parameters

`uID`

Strand7 model file ID.

`Colours[0..3]`

`[ipPathFillColour]` – The fill colour.

`[ipPathLineColour]` – The outline colour.

`[ipPathOrientation1Colour]` – The -z orientation colour.

`[ipPathOrientation2Colour]` – The +z orientation colour.

See also *RGB Colours*.

`NumCol`

The size of Colours.

St7SetPathLighting

Sets the state of the **Lighting** options for load path rendering.

```
long St7SetPathLighting(long uID, bool FillLighting, bool LineLighting)
```

Input Parameters

`uID`

Strand7 model file ID.

`FillLighting`

The lighting option for the load path fill; either True or False.

`LineLighting`

The lighting option for the load path outlines; either True or False.

St7SetPathLineThickness

Sets the thickness of the load path outlines.

```
long St7SetPathLineThickness(long uID, long Thickness)
```

Input Parameters

`uID`

Strand7 model file ID.

`Thickness`

The load path outline thickness in the range kMinThickness to kMaxThickness.

St7SetPathDivisions

Sets whether **Divisions** are shown on load paths.

```
long St7SetPathDivisions(long uID, bool Show)
```

Input Parameters

`uID`

Strand7 model file ID.

`Show`

The state of the **Divisions** option; either True or False.

St7GetPathFill

Returns the fill colour scheme for load paths.

```
long St7GetPathFill(long uID, long* Fill)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Fill`

One of tfNone, tfTemplate, tfGroup, tfPathNumber, tfColour or tfOrientation.

St7GetPathOutline

Returns the outline colour scheme for load paths.

```
long St7GetPathOutline(long uID, long* Outline)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Outline`

One of tlNone, tlTemplate, tlGroup, tlPathNumber or tlColour.

St7GetPathLabelStyle

Returns the display style for load path labels.

```
long St7GetPathLabelStyle(long uID, long* LabelStyle)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`LabelStyle`

One of lsNone, lsEntityNumber or lsLaneNumber.

St7GetPathColours

Returns the colours to be used for given load path display types.

```
long St7GetPathColours(long uID, long* Colours, long NumCol)
```

Input Parameters

`uID`

Strand7 model file ID.

`NumCol`

The size of Colours.

Output Parameters

`Colours[0..3]`

`[ipPathFillColour]` – The fill colour.

`[ipPathLineColour]` – The outline colour.

`[ipPathOrientation1Colour]` – The -z orientation colour.

`[ipPathOrientation2Colour]` – The +z orientation colour.

See also *RGB Colours*.

St7GetPathLighting

Returns the state of the **Lighting** options for load path rendering.

```
long St7GetPathLighting(long uID, bool* FillLighting, bool* LineLighting)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`FillLighting`

The lighting option for the load path fill; either True or False.

`LineLighting`

The lighting option for the load path outlines; either True or False.

St7GetPathLineThickness

Returns the thickness of the load path outlines.

```
long St7GetPathLineThickness(long uID, long* Thickness)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Thickness`

The load path outline thickness in the range kMinThickness to kMaxThickness.

St7GetPathDivisions

Returns whether **Divisions** are shown on load paths.

```
long St7GetPathDivisions(long uID, bool* Show)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Show`

The state of the **Divisions** option; either True or False.

Attribute Display

St7SetAttributeDisplay

Sets options related to the display of attributes in the model window.

```
long St7SetAttributeDisplay(long uID, long AttributeOrd, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`AttributeOrd`

Attribute identifier. See *Attribute Types* for additional information.

`Integers[0..9]`

`[ipAttribDisplayShow]` – btTrue to show the attribute, or btFalse to hide it.

`[ipAttribDisplayLabel]` – btTrue to label the attribute with its value, or btFalse to show the attribute without a label.

`[ipAttribDisplayResultant]` – btTrue to show the resultant vector, or btFalse to show the components.

`[ipAttribDisplayAnchorTail]` – btTrue to anchor the vector at the tail, or btFalse to anchor the vector at the head.

`[ipAttribDisplayScaled]` – btTrue to proportionally scale the attribute based on its value, or btFalse to draw the attribute at the unscaled default size independently of its value.

`[ipAttribDisplaySize]` – Attribute size in the range 0 to 100. Note that some attributes have a minimum size; the minimum size will be set if the requested size is less than the minimum.

`[ipAttribDisplayThickness]` – Thickness in the range kMinThickness to kMaxThickness.

`[ipAttribDisplayCol1]` – First colour. See also *RGB Colours*.

`[ipAttribDisplayCol2]` – Second colour. See also *RGB Colours*.

`[ipAttribDisplayCol3]` – Third colour. See also *RGB Colours*.

St7GetAttributeDisplay

Returns options related to the display of attributes in the model window.

```
long St7GetAttributeDisplay(long uID, long AttributeOrd, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`AttributeOrd`

Attribute identifier. See *Attribute Types* for additional information.

Output Parameters

`Integers[0..9]`

`[ipAttribDisplayShow]` – btTrue to show the attribute, or btFalse to hide it.

`[ipAttribDisplayLabel]` – btTrue to label the attribute with its value, or btFalse to show the attribute without a label.

`[ipAttribDisplayResultant]` – btTrue to show the resultant vector, or btFalse to show the components.

`[ipAttribDisplayAnchorTail]` – btTrue to anchor the vector at the tail, or btFalse to anchor the vector at the head.

`[ipAttribDisplayScaled]` – btTrue to proportionally scale the attribute based on its value, or btFalse to draw the attribute at the unscaled default size independently of its value.

`[ipAttribDisplaySize]` – Attribute size in the range 0 to 100.

`[ipAttribDisplayThickness]` – Thickness in the range kMinThickness to kMaxThickness.

`[ipAttribDisplayCol1]` – First colour. See also *RGB Colours*.

`[ipAttribDisplayCol2]` – Second colour. See also *RGB Colours*.

`[ipAttribDisplayCol3]` – Third colour. See also *RGB Colours*.

Markers

Markers are a way for API developers to highlight or annotate particular entities in the model window. Markers are drawn 'always on top' and are saved with the model file.

St7SetMarker

Assigns a marker to an element or element face.

```
long St7SetMarker(long uID, long Entity, long EntityNum, long FaceNum,
     long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyLINK, tyVERTEX, tyGEOMETRYFACE or tyLOADPATH.

`EntityNum`

Entity number.

`FaceNum`

Brick element face number (1-6). For marker type mtEntityHighlight, 0 is also valid and indicates the marker applies to the whole brick.

`Integers[0..11]`

`[ipMarkerType]` – One of mtCircleMarker, mtSquareMarker, mtTriangleMarker, mtRectangleMarker, mtEntityHighlight or mtBanner.

`[ipMarkerStyle]` – One of msFilled, msOutlined or msFilledOutlined.

`[ipMarkerFillColour]` – Fill colour. See also *RGB Colours*.

`[ipMarkerLineColour]` – Line colour. See also *RGB Colours*.

`[ipMarkerLineThickness]` – Line thickness.

`[ipMarkerSize]` – Marker size.

`[ipMarkerHeight]` – Marker height.

`[ipMarkerAnchorX]` – The model window X anchor for banner markers. X values increase from left to right in the model window.

`[ipMarkerAnchorY]` – The model window Y anchor for banner markers. Y values increase from top to bottom in the model window.

`[ipMarkerVisible]` – btTrue to show the marker; btFalse to hide the marker.

`[ipMarkerNumber]` – btTrue to show the entity number; btFalse to hide the entity number.

`[ipMarkerLabelled]` – btTrue to show the entity label with the entity number (e.g., “Node 123”); btFalse to show only the entity number (e.g., “123”).

St7GetMarker

Returns marker associated with an element or element face.

```
long St7GetMarker(long uID, long Entity, long EntityNum, long FaceNum,
     long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyLINK, tyVERTEX, tyGEOMETRYFACE or tyLOADPATH.

`EntityNum`

Entity number.

`FaceNum`

Brick element face number (1-6). For marker type mtEntityHighlight, 0 is also valid and indicates the marker applies to the whole brick.

Output Parameters

`Integers[0..11]`

`[ipMarkerType]` – One of mtCircleMarker, mtSquareMarker, mtTriangleMarker, mtRectangleMarker, mtEntityHighlight or mtBanner.

`[ipMarkerStyle]` – One of msFilled, msOutlined or msFilledOutlined.

`[ipMarkerFillColour]` – Fill colour. See also *RGB Colours*.

`[ipMarkerLineColour]` – Line colour. See also *RGB Colours*.

`[ipMarkerLineThickness]` – Line thickness.

`[ipMarkerSize]` – Marker size.

`[ipMarkerHeight]` – Marker height (applies to mtRectangleMarker) .

`[ipMarkerAnchorX]` – The model window X anchor for banner markers. X values increase from left to right in the model window.

`[ipMarkerAnchorY]` – The model window Y anchor for banner markers. Y values increase from top to bottom in the model window.

`[ipMarkerVisible]` – btTrue to show the marker; btFalse to hide the marker.

`[ipMarkerNumber]` – btTrue to show the entity number; btFalse to hide the entity number.

`[ipMarkerLabelled]` – btTrue to show the entity label with the entity number (e.g., “Node 123”); btFalse to show only the entity number (e.g., “123”).

St7DeleteMarker

Deletes a marker from an element or element face.

```
long St7DeleteMarker(long uID, long Entity, long EntityNum, long FaceNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyLINK, tyVERTEX, tyGEOMETRYFACE or tyLOADPATH.

`EntityNum`

Entity number.

`FaceNum`

Brick element face number (1-6). For marker type mtEntityHighlight, 0 is also valid and indicates the marker applies to the whole brick.

St7ShowMarker

Shows a previously created marker.

```
long St7ShowMarker(long uID, long Entity, long EntityNum, long FaceNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyLINK, tyVERTEX, tyGEOMETRYFACE or tyLOADPATH.

`EntityNum`

Entity number.

`FaceNum`

Brick element face number (1-6). For marker type mtEntityHighlight, 0 is also valid and indicates the marker applies to the whole brick.

St7HideMarker

Hides a previously created marker.

```
long St7HideMarker(long uID, long Entity, long EntityNum, long FaceNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyLINK, tyVERTEX, tyGEOMETRYFACE or tyLOADPATH.

`EntityNum`

Entity number.

`FaceNum`

Brick element face number (1-6). For marker type mtEntityHighlight, 0 is also valid and indicates the marker applies to the whole brick.

Element Contouring

St7SetEntityContourFile

Sets a user defined contour file for beam, plate or brick elements.

```
long St7SetEntityContourFile(long uID, long Entity, long FileType,
     char* FileName)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

`FileType`

Basis for the contour values; either ucNode or ucElement.

`FileName`

Full path and name of the text file containing the user defined contour values.

St7GetEntityContourFile

Returns the user defined contour file specified for beam, plate or brick elements.

```
long St7GetEntityContourFile(long uID, long Entity, long* FileType,
     char* FileName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

`MaxStringLen`

Maximum number of characters allocated for FileName.

Output Parameters

`FileType`

Basis of the contour values; either ucNode or ucElement.

`FileName`

Full path and name of the text file containing the user defined contour values.

St7SetEntityContourIndex

Sets a contour type from the combo box menus in **Entity Display** for beam, plate or brick elements.

```
long St7SetEntityContourIndex(long uID, long Entity, long Index)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

`Index`

A contour specified by the constants listed in *Entity Contours*.

St7GetEntityContourIndex

Returns the contour type specified by the combo box menus in **Entity Display** for beam, plate or brick elements.

```
long St7GetEntityContourIndex(long uID, long Entity, long* Index)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

Output Parameters

`Index`

A contour specified by the constants listed in *Entity Contours*.

St7SetEntityContourSettingsStyle

Sets the parameters accessible through **Entity Display/Contour/Settings.../Style** in the GUI.

```
long St7SetEntityContourSettingsStyle(long uID, long Entity, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

`Integers[0..11]`

`[ipContourStyle]` – Either csRainbow, csRainbowEnds, csMono, csLines or csBands.

`[ipReverse]` – btTrue if the contour is reversed.

`[ipSeparator]` – btTrue if a separator is included between ranges for the user defined contour.

`[ipBand1Colour]` – Colour for **Band 1**. See also *RGB Colours*.

`[ipBand2Colour]` – Colour for **Band 2**. See also *RGB Colours*.

`[ipSeparatorColour]` – Colour for **Separator**. See also *RGB Colours*.

`[ipLineBackColour]` – Colour for **Line Back**. See also *RGB Colours*.

`[ipMonoColour]` – Colour for **Mono**. See also *RGB Colours*.

`[ipMinColour]` – Colour for **< Min**. See also *RGB Colours*.

`[ipMaxColour]` – Colour for **> Max**. See also *RGB Colours*.

`[ipLimitMin]` – btTrue if the colour specified in ipMinColour is enabled for the out-of-bound contor.

`[ipLimitMax]` – btTrue if the colour specified in ipMaxColour is enabled for the out-of-bound contor.

St7GetEntityContourSettingsStyle

Returns the parameters accessible through **Entity Display/Contour/Settings.../Style** in the GUI.

```
long St7GetEntityContourSettingsStyle(long uID, long Entity, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

Output Parameters

`Integers[0..11]`

`[ipContourStyle]` – Either csRainbow, csRainbowEnds, csMono, csLines or csBands.

`[ipReverse]` – btTrue if the contour is reversed.

`[ipSeparator]` – btTrue if a separator is included between ranges for the user defined contour.

`[ipBand1Colour]` – Colour for **Band 1**. See also *RGB Colours*.

`[ipBand2Colour]` – Colour for **Band 2**. See also *RGB Colours*.

`[ipSeparatorColour]` – Colour for **Separator**. See also *RGB Colours*.

`[ipLineBackColour]` – Colour for **Line Back**. See also *RGB Colours*.

`[ipMonoColour]` – Colour for **Mono**. See also *RGB Colours*.

`[ipMinColour]` – Colour for **< Min**. See also *RGB Colours*.

`[ipMaxColour]` – Colour for **> Max**. See also *RGB Colours*.

`[ipLimitMin]` – btTrue if the colour specified in ipMinColour is enabled for the out-of-bound contor.

`[ipLimitMax]` – btTrue if the colour specified in ipMaxColour is enabled for the out-of-bound contor.

St7SetEntityContourSettingsLimits

Sets the parameters accessible through **Entity Display/Contour/Settings.../Limits** in the GUI.

```
long St7SetEntityContourSettingsLimits(long uID, long Entity, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

`Integers[0..4]`

`[ipContourLimit]` – Either clDefault, clUserRange, clRounded or clUserSpecified.

`[ipContourMode]` – Either cmContinuous or cmDiscrete.

`[ipNumContours]` – Number of contours with the discrete setting.

`[ipSetMinLimit]` – btTrue if the lower limit is enabled for the user range.

`[ipSetMaxLimit]` – btTrue if the upper limit is enabled for the user range.

`Doubles[0..1]`

`[ipMinLimit]` – Lower limit for the user range.

`[ipMaxLimit]` – Upper limit for the user range.

St7GetEntityContourSettingsLimits

Returns the parameters accessible through **Entity Display/Contour/Settings.../Limits** in the GUI.

```
long St7GetEntityContourSettingsLimits(long uID, long Entity, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

Output Parameters

`Integers[0..4]`

`[ipContourLimit]` – Either clDefault, clUserRange, clRounded or clUserSpecified.

`[ipContourMode]` – Either cmContinuous or cmDiscrete.

`[ipNumContours]` – Number of contours with the discrete setting.

`[ipSetMinLimit]` – btTrue if the lower limit is enabled for the user range.

`[ipSetMaxLimit]` – btTrue if the upper limit is enabled for the user range.

`Doubles[0..1]`

`[ipMinLimit]` – Lower limit for the user range.

`[ipMaxLimit]` – Upper limit for the user range.

St7SetEntityContourSettingsLimitsString

Sets the parameters accessible through **Entity Display/Contour/Settings.../Limits** in the GUI.

```
long St7SetEntityContourSettingsLimitsString(long uID, long Entity,
     char* LimitsString)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

`LimitsString`

String containing the user defined contour boundaries. Use *St7GetListSeparatorCode* to get the system's list separator character.

St7GetEntityContourSettingsLimitsString

Returns the parameters accessible through **Entity Display/Contour/Settings.../Limits** in the GUI.

```
long St7GetEntityContourSettingsLimitsString(long uID, long Entity,
     char* LimitsString, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

`MaxStringLen`

Maximum number of characters allocated for LimitsString.

Output Parameters

`LimitsString`

String containing the user defined contour boundaries. Use *St7GetListSeparatorCode* to get the system's list separator character.

St7SetEntityContourSettingsLegend

Sets the parameters accessible through **Entity Display/Contour/Settings.../Legend** in the GUI.

```
long St7SetEntityContourSettingsLegend(long uID, long Entity, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

`Integers[0..5]`

`[ipLegendPosition]` – Either lpNone, lpTopLeft, lpTopRight, lpBottomLeft, lpBottomRight or lpFloating.

`[ipOpaqueLegend]` – btTrue for **Opaque**.

`[ipShowMinMax]` – btTrue for **Show Min/Max**.

`[ipHistogram]` – btTrue for **Histogram**.

`[ipLegendWidth]` – Legend width.

`[ipLegendHeight]` – Legend height.

St7GetEntityContourSettingsLegend

Returns the parameters accessible through **Entity Display/Contour/Settings.../Legend** in the GUI.

```
long St7GetEntityContourSettingsLegend(long uID, long Entity, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

Output Parameters

`Integers[0..5]`

`[ipLegendPosition]` – Either lpNone, lpTopLeft, lpTopRight, lpBottomLeft, lpBottomRight or lpFloating.

`[ipOpaqueLegend]` – btTrue for **Opaque**.

`[ipShowMinMax]` – btTrue for **Show Min/Max**.

`[ipHistogram]` – btTrue for **Histogram**.

`[ipLegendWidth]` – Legend width.

`[ipLegendHeight]` – Legend height.

St7SetEntityContourSettingsLegendFont

Sets the parameters accessible through **Entity Display/Contour/Settings.../Legend** in the GUI.

```
long St7SetEntityContourSettingsLegendFont(long uID, long Entity, char* FontName,
     long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

`FontName`

String containing the name of the font.

`Integers[0..4]`

`[ipFontSize]` – Font size.

`[ipFontColour]` – Font colour. See also *RGB Colours*.

`[ipFontStyleBold]` – btTrue for bold.

`[ipFontStyleItalic]` – btTrue for italic.

`[ipFontStyleUnderline]` – btTrue for underline.

St7GetEntityContourSettingsLegendFont

Returns the parameters accessible through **Entity Display/Contour/Settings.../Legend** in the GUI.

```
long St7GetEntityContourSettingsLegendFont(long uID, long Entity, char* FontName,
     long MaxStringLen, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

`MaxStringLen`

Maximum number of characters allocated for FontName.

Output Parameters

`FontName`

String containing the name of the font.

`Integers[0..4]`

`[ipFontSize]` – Font size.

`[ipFontColour]` – Font colour. See also *RGB Colours*.

`[ipFontStyleBold]` – btTrue for bold.

`[ipFontStyleItalic]` – btTrue for italic.

`[ipFontStyleUnderline]` – btTrue for underline.

Post-Processing

St7SetDisplacementScale

Sets the **Displacement Scale** used to draw the deformed model, when an associated model result file is open.

```
long St7SetDisplacementScale(long uID, double DispScale, long ScaleType)
```

Input Parameters

`uID`

Strand7 model file ID.

`DispScale`

The scaling factor or percentage to be applied.

`ScaleType`

The manner of scaling to be used; either dsPercent or dsAbsolute.

St7GetDisplacementScale

Returns the **Displacement Scale** used to draw the deformed model when an associated model result file is open.

```
long St7GetDisplacementScale(long uID, double* DispScale, long* ScaleType)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`DispScale`

The scaling factor or percentage applied.

`ScaleType`

The manner of scaling used; either dsPercent or dsAbsolute.

St7DeleteAllGraphs

Deletes all result graphs.

```
long St7DeleteAllGraphs(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

Import/Export Utilities

St7ImportST7

Imports the specified Strand7 text file format model.

```
long St7ImportST7(long uID, char* FileName, long* Integers, long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the Strand7 text-file to be imported.

`Integers[0..1]`

`[ipSt7ImportRemoveCases]` – btTrue to remove the existing load and freedom cases prior to import so those from the imported model are the only cases; only applicable to empty models (that is, models with no entities).

`[ipSt7ImportMatchUCSNames]` – btTrue to map incoming UCS definitions to existing UCS definitions with the same name. Note that if the definition of the incoming UCS is not the same as that of the existing UCS, the incoming UCS definition will be lost.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

St7ImportIGES

Imports a geometry file in IGES format.

```
long St7ImportIGES(long uID, char* FileName, long* Integers, double* Doubles,
    long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the IGES file to be imported.

`Integers[0..5]`

`[ipGeomImportProperty]` – Default property ID.

`[ipGeomImportCurvesToBeams]` – Convert unreferenced curves to beam elements; either btTrue or btFalse.

`[ipGeomImportGroupsAs]` – Geometry groups import; one of ggNone, ggAuto, ggSubfigures or ggLevels.

`[ipGeomImportColourAsProperty]` – Import geometry colour definitions as property definitions; either btTrue or btFalse.

`[ipGeomImportMatchExistingProperty]` – If importing geometry colour definitions as property definitions, and a property of the same colour already exists, use that property, do not create a new one; either btTrue or btFalse.

`[ipGeomImportLengthUnit]` – Specifies a length unit for the import file; one of luGeomNone, luGeomInch, luGeomMillimetre, luGeomFoot, luGeomMile, luGeomMetre, luGeomKilometre, luGeomMil, luGeomMicron, luGeomCentimetre, luGeomMicroinch, or luGeomUnspecified.

`Doubles[0..0]`

`[ipGeomImportTol]` – Relative tolerance used when importing geometry.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

St7ImportACIS

Imports a geometry file in the ACIS format.

```
long St7ImportACIS(long uID, char* FileName, long* Integers, double* Doubles,
    long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the ACIS file to be imported.

`Integers[0..5]`

`[ipGeomImportProperty]` – Default property ID.

`[ipGeomImportCurvesToBeams]` – Convert unreferenced curves to beam elements; either btTrue or btFalse.

`[ipGeomImportGroupsAs]` – Geometry groups import; either ggNone or ggBodies.

`[ipGeomImportColourAsProperty]` – Import geometry colour definitions as property definitions; either btTrue or btFalse.

`[ipGeomImportMatchExistingProperty]` – If importing geometry colour definitions as property definitions, and a property of the same colour already exists, use that property, do not create a new one; either btTrue or btFalse.

`[ipGeomImportLengthUnit]` – Specifies a length unit for the import file; one of luGeomNone, luGeomInch, luGeomMillimetre, luGeomFoot, luGeomMile, luGeomMetre, luGeomKilometre, luGeomMil, luGeomMicron, luGeomCentimetre, luGeomMicroinch, or luGeomUnspecified.

`Doubles[0..0]`

`[ipGeomImportTol]` – Relative tolerance used when importing geometry.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

St7ImportSTEP

Imports a geometry file in the STEP format.

```
long St7ImportSTEP(long uID, char* FileName, long* Integers, double* Doubles,
    long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the STEP file to be imported.

`Integers[0..5]`

`[ipGeomImportProperty]` – Default property ID.

`[ipGeomImportGroupsAs]` – Geometry groups import; either ggNone or ggAssemblies.

`[ipGeomImportColourAsProperty]` – Import geometry colour definitions as property definitions; either btTrue or btFalse.

`[ipGeomImportMatchExistingProperty]` – If importing geometry colour definitions as property definitions, and a property of the same colour already exists, use that property, do not create a new one; either btTrue or btFalse.

`[ipGeomImportLengthUnit]` – Specifies a length unit for the import file; one of luGeomNone, luGeomInch, luGeomMillimetre, luGeomFoot, luGeomMile, luGeomMetre, luGeomKilometre, luGeomMil, luGeomMicron, luGeomCentimetre, luGeomMicroinch, or luGeomUnspecified.

`Doubles[0..0]`

`[ipGeomImportTol]` – Relative tolerance used when importing geometry.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

St7ImportRhino

Imports a geometry file in the Rhino format.

```
long St7ImportRhino(long uID, char* FileName, long* Integers, double* Doubles,
    long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the Rhino file to be imported.

`Integers[0..5]`

`[ipGeomImportProperty]` – Default property ID.

`[ipGeomImportCurvesToBeams]` – Convert unreferenced curves to beam elements; either btTrue or btFalse.

`[ipGeomImportGroupsAs]` – Geometry groups import; one of ggNone, ggAuto, ggBlocks or ggLayers.

`[ipGeomImportColourAsProperty]` – Import geometry colour definitions as property definitions; either btTrue or btFalse.

`[ipGeomImportMatchExistingProperty]` – If importing geometry colour definitions as property definitions, and a property of the same colour already exists, use that property, do not create a new one; either btTrue or btFalse.

`[ipGeomImportLengthUnit]` – Specifies a length unit for the import file; one of luGeomNone, luGeomInch, luGeomMillimetre, luGeomFoot, luGeomMile, luGeomMetre, luGeomKilometre, luGeomMil, luGeomMicron, luGeomCentimetre, luGeomMicroinch, or luGeomUnspecified.

`Doubles[0..0]`

`[ipGeomImportTol]` – Relative tolerance used when importing geometry.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

St7ImportDXF

Imports a geometry file in the DXF format.

```
long St7ImportDXF(long uID, char* FileName, long* Integers, double* Doubles,
     long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the DXF file to be imported.

`Integers[0..11]`

`[ipDXFImportFrozenLayers]` – Import frozen layers.

`[ipDXFImportLayersAsGroups]` – Import geometry layers as groups; either btTrue or btFalse.

`[ipDXFImportColoursAsProps]` – Use geometry colours as property definitions; either btTrue or btFalse.

`[ipDXFImportMatchExistingProperty]` – If importing geometry colour definitions as property definitions, and a property of the same colour already exists, use that property, do not create a new one; either btTrue or btFalse.

`[ipDXFImportPolylineAsPlates]` – Import polyline definitions as plate elements; either btTrue or btFalse.

`[ipDXFImportPolygonAsBricks]` – Import polygon definitions as brick elements; either btTrue or btFalse.

`[ipDXFImportSegmentsPerCircle]` – Number of line segments used to discretise curves.

`[ipDXFImportUseSegmentsPerCircle]` – btTrue to use **Segments per Circle**, or btFalse to use **Arc Length** for curve discretisation.

`[ipDXFImportLengthUnit]` – Specifies a length unit for the import file; one of luGeomNone, luGeomInch, luGeomMillimetre, luGeomFoot, luGeomMile, luGeomMetre, luGeomKilometre, luGeomMil, luGeomMicron, luGeomCentimetre, luGeomMicroinch, or luGeomUnspecified.

`[ipDXFImportProperty]` – Import colours as properties; either btTrue or btFalse.

`[ipDXFImportAcisBodiesAsGroups]` – Import ACIS bodies as groups; either btTrue or btFalse.

`[ipDXFImportCurvesToBeams]` – Convert unreferenced curves in ACIS data to beam elements; either btTrue or btFalse.

`Doubles[0..1]`

`[ipDXFImportArcLength]` – Length of the line segment used to discretise curves, specified in Strand7 model units.

`[ipDXFImportAcisTol]` – Relative tolerance used when importing ACIS geometry.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

St7ImportSTL

Imports a stereo-lithography file.

```
long St7ImportSTL(long uID, char* FileName, long* Integers, long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the STL file to be imported.

`Integers[0..1]`

`[ipSTLImportProperty]` – Default property for imported plates.

`[ipSTLImportLengthUnit]` – Specifies a length unit for the import file; one of luSTLNone, luSTLMillimetre, luSTLCentimetre, luSTLMetre, luSTLInch, luSTLFoot.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

St7ImportNASTRAN

Imports a NASTRAN model file.

```
long St7ImportNASTRAN(long uID, char* FileName, long* Integers, long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the NASTRAN file to be imported.

`Integers[0..0]`

`[ipNASTRANImportUnits]` – Nastran file units; one of usNASTRAN_kg_N_m, usNASTRAN_T_N_mm, usNASTRAN_sl_lbf_ft, usNASTRAN_lbm_lbf_in, usNASTRAN_sl_lbf_in or usNASTRAN_None.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

St7ImportANSYS

Imports an ANSYS model file.

```
long St7ImportANSYS(long uID, char* FileName, char* LoadCaseFilePath,
     long* Integers, long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the ANSYS file to be imported.

`LoadCaseFilePath`

Full path to the directory containing the load case data for the ANSYS file.

`Integers[0..5]`

`[ipANSYSImportFormat]` – Import format; one of ieANSYSBatchImport, ieANSYSCDBImport or ieANSYSBatchCDBImport.

`[ipANSYSArrayParameters]` – Array parameter type; one of ieANSYSArrayOverwrite, ieANSYSArrayIgnore or ieANSYSArrayPrompt.

`[ipANSYSImportLoadCaseFiles]` – Import additional load case files; either btTrue or btFalse.

`[ipANSYSImportIGESEntities]` – Import IGES geometry definitions; either btTrue or btFalse.

`[ipANSYSFixElementConnectivity]` – Fix element connectivity; either btTrue or btFalse.

`[ipANSYSRemoveDuplicateProps]` – Remove duplicate property definitions; either btTrue or btFalse.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

St7ImportSTAAD

Imports a STAAD model file.

```
long St7ImportSTAAD(long uID, char* FileName, long* Integers, long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the STAAD file to be imported.

`Integers[0..6]`

`[ipSTAADCountryType]` – Default country type; one of ieSTAADAmericanCode, ieSTAADAustralianCode or ieSTAADBritishCode.

`[ipSTAADIncludeSectionLibrary]` – Search additional beam cross section libraries; either btTrue or btFalse.

`[ipSTAADStripUnderscore]` – Remove underscore from group names; either btTrue or btFalse.

`[ipSTAADStripSectionSpaces]` – Remove spaces from section names; either btTrue or btFalse.

`[ipSTAADStripCaseQualifiers]` – Strip qualifiers from load case strings; either btTrue or btFalse.

`[ipSTAADLengthUnit]` – Length unit; one of luSTAADInch, luSTAADFoot, luSTAADCentimetre, luSTAADMetre, luSTAADMillimetre, luSTAADDecimetre or luSTAADKilometre.

`[ipSTAADForceUnit]` – Force unit; one of fuSTAADKip, fuSTAADPoundForce, fuSTAADKilogramForce, fuSTAADMegatonneForce, fuSTAADNewton, fuSTAADKilonewton, fuSTAADMeganewton or fuSTAADDecanewton.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

St7ImportSAP2000

Imports a SAP2000 model file.

```
long St7ImportSAP2000(long uID, char* FileName, long* Integers, long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the SAP2000 file to be imported.

`Integers[0..2]`

`[ipSAP2000DecimalSeparator]` – Decimal character; either ieSAP2000Period or ieSAP2000Comma.

`[ipSAP2000ThousandSeparator]` – Thousands character; one of ieSAP2000Period, ieSAP2000Comma, ieSAP2000Space or ieSAP2000None.

`[ipSAP2000MergeDuplicateFreedomSets]` – Merges duplicate freedom sets in the imported file; either btTrue or btFalse.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

St7ExportImage

Exports the Strand7 graphics as an image file.

```
long St7ExportImage(long uID, char* FileName, long ImageType, long Width,
     long Height)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the image file to be created.

`ImageType`

One of itBitmap8Bit, itBitmap16Bit, itBitmap24Bit, itJPEG or itPNG.

`Width`

Pixel width for the image.

`Height`

Pixel height for the image.

St7ExportImageToClipboard

Copies the Strand7 graphics to the clipboard.

```
long St7ExportImageToClipboard(long uID, long Width, long Height)
```

Input Parameters

`uID`

Strand7 model file ID.

`Width`

Pixel width for the image.

`Height`

Pixel height for the image.

St7ExportST7

Exports the current model in the Strand7 text file format.

```
long St7ExportST7(long uID, char* FileName, long Mode, long ExportFormat)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the Strand7 text-file to be created.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

`ExportFormat`

Controls the export format for backwards compatibility; one of ieSt7ExportCurrent, ieSt7Export106, ieSt7Export21x, ieSt7Export22x, ieSt7Export23x or ieSt7Export24x.

St7ExportIGES

Exports the current Strand7 geometry as an IGES format geometry file.

```
long St7ExportIGES(long uID, char* FileName, long* Integers, long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the IGES file to be created.

`Integers[0..6]`

`[ipGeomExportColour]` – Export colours; one of ieGeomColourNone, ieGeomFaceColour, ieGeomGroupColour or ieGeomPropertyColour.

`[ipGeomExportGroupsAsLevels]` – Export the groups as levels; either btTrue or btFalse.

`[ipGeomExportFullGroupPath]` – Export the full group definition; either btTrue or btFalse.

`[ipGeomExportFormatProtocol]` – Export format; one of ieIGESBoundedSurface, ieIGESTrimmedParametricSurface, ieIGESOpenShell or ieIGESManifoldSolidBRep.

`[ipGeomExportCurve]` – Export curves; one of ieGeomModelOnly, ieGeomParameterOnly, ieGeomModelPreferred or ieGeomParameterPreferred.

`[ipGeomExportPeriodicFace]` – Periodic face control; one of ieGeomSeamOnlyAsRequired, ieGeomSplitOnFaceBoundary or ieGeomSplitIntoHalves.

`[ipGeomExportKeepAnalytic]` – Export the analytic geometry definitions; either btTrue or btFalse.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

St7ExportSTEP

Exports the current Strand7 geometry as a STEP format geometry file.

```
long St7ExportSTEP(long uID, char* FileName, long* Integers, long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the STEP file to be created.

`Integers[0..6]`

`[ipGeomExportColour]` – Export colours; one of ieGeomColourNone, ieGeomFaceColour, ieGeomGroupColour or ieGeomPropertyColour.

`[ipGeomExportFullGroupPath]` – Export the full group definition; either btTrue or btFalse.

`[ipGeomExportFormatProtocol]` – Export format; one of ieSTEPConfigControlDesign or ieSTEPAutomotiveDesign.

`[ipGeomExportCurve]` – Export curves; one of ieGeomModelOnly, ieGeomParameterOnly, ieGeomModelPreferred or ieGeomParameterPreferred.

`[ipGeomExportPeriodicFace]` – Periodic face control; one of ieGeomSeamOnlyAsRequired, ieGeomSplitOnFaceBoundary or ieGeomSplitIntoHalves.

`[ipGeomExportKeepAnalytic]` – Export the analytic geometry definitions; either btTrue or btFalse.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

St7ExportDXF

Exports the current Strand7 geometry as a DXF format geometry file.

```
long St7ExportDXF(long uID, char* FileName, long* Integers, long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the DXF file to be created.

`Integers[0..7]`

`[ipDXFExportPlatesBricks3DFaces]` – Export plates and bricks as AutoCAD 3D faces; either btTrue or btFalse.

`[ipDXFExportGroupsAsLayers]` – Export groups as AutoCAD layers; either btTrue or btFalse.

`[ipDXFExportPropColoursAsEntityColours]` – Export property colours as AutoCAD entity colours; either btTrue or btFalse.

`[ipDXFExportBeamsAs]` – Beam element export; one of ieBeamAsLine, ieBeamAsSection or ieBeamAsSolid.

`[ipDXFExportPlatesAs]` – Plate element export; either iePlateAsSurface or iePlateAsSolid.

`[ipDXFExportBeamOffsets]` – Position beam element at offset location; either btTrue or btFalse.

`[ipDXFExportPlateOffsets]` – Position plate element at offset location; either btTrue or btFalse.

`[ipDXFExportInternalBrickFaces]` – btTrue to export **All Faces**, or btFalse to export **Free Faces**.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

St7ExportSTL

Exports the current Strand7 model as a STL file.

```
long St7ExportSTL(long uID, char* FileName, long* Integers, long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the STL file to be created.

`Integers[0..10]`

`[ipSTLExportFormat]` – Format of file; either ieSTLText or ieSTLBinary.

`[ipSTLExportGrouping]` – Grouping of elements; one of ieSTLGroupByNone, ieSTLGroupByEntityType or ieSTLGroupByGroups.

`[ipSTLExportBeams]` – Export beam elements; either btTrue or btFalse.

`[ipSTLExportPlates]` – Export plate elements; either btTrue or btFalse.

`[ipSTLExportBricks]` – Export brick faces; either btTrue or btFalse.

`[ipSTLExportGeometryFaces]` – Export geometry faces; either btTrue or btFalse

`[ipSTLExportBeamsAs]` – Export beams as either a section or solid; either ieBeamAsSection or ieBeamAsSolid.

`[ipSTLExportPlatesAs]` – Export plates as either a surface or solid; either iePlateAsSurface or iePlateAsSolid.

`[ipSTLExportBeamOffsets]` – Export Beam offsets; either btTrue or btFalse.

`[ipSTLExportPlateOffsets]` – Export Plate offsets; either btTrue or btFalse.

`[ipSTLExportInternalBrickFaces]` – btTrue to export **All Faces**, or btFalse to export **Free Faces**.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

St7ExportNASTRAN

Exports the current Strand7 model as a NASTRAN model file.

```
long St7ExportNASTRAN(long uID, char* FileName, long* Integers, double* Doubles,
    long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the NASTRAN file to be created.

`Integers[0..17]`

`[ipNASTRANFreedomCase]` – Exported freedom case.

`[ipNASTRANLoadCaseNSMass]` – Exported load case for non-structural mass attributes.

`[ipNASTRANSolver]` – Nastran solver type; one of ieNASTRANSolverLSA, ieNASTRANSolverNFA, ieNASTRANSolverLBA or ieNASTRANSolverNLA.

`[ipNASTRANExportUnits]` – Units for exported file; one of usNASTRAN_kg_N_m, usNASTRAN_T_N_mm, usNASTRAN_sl_lbf_ft, usNASTRAN_lbm_lbf_in, usNASTRAN_sl_lbf_in or usNASTRAN_None.

`[ipNASTRANBeamStressSections]` – Number of sections defined for exported beam elements.

`[ipNASTRANBeamSectionGeometry]` – Export beam section geometry; either ieNASTRANExportGeometryProps or ieNASTRANExportPropsOnly.

`[ipNASTRANExportHeatTransfer]` – Export heat transfer property data; either btTrue or btFalse.

`[ipNASTRANExportNSMass]` – Export non-structural mass attributes; either btTrue or btFalse.

`[ipNASTRANExportUnusedProps]` – Export unreferenced material properties; either btTrue or btFalse.

`[ipNASTRANTemperatureCase]` – Load case from which reference temperature is exported.

`[ipNASTRANPreLoadCase]` – Load case from which contact element pre-strain value is taken.

`[ipNASTRANNInc]` – NINC value for SOL 106 export.

`[ipNASTRANMaxIter]` – MAXITER value for SOL 106 export.

`[ipNASTRANDoEPSU]` – Include displacement in the SOL 106 convergence criteria; either btTrue or btFalse.

`[ipNASTRANDoEPSP]` – Include force in the SOL 106 convergence criteria; either btTrue or btFalse.

`[ipNASTRANDoEPSW]` – Include work in the SOL 106 convergence criteria; either btTrue or btFalse.

`[ipNASTRANExportPyramid]` – Export Pyramid elements; either ieNASTRANExportPyramidAsHexa for collapsed hexahedra or ieNASTRANExportPyramidAsPyram for CPYRAM.

`[ipNASTRANExportQuad4]` – Export Quad4 elements; either ieNASTRANExportCQUAD4 for CQUAD4 or ieNASTRANExportCQUADR for CQUADR.

`Doubles[0..3]`

`[ipNASTRANExportZeroFields]` – Zero tolerance. Parameters with magnitude less than this value are set to zero on export.

`[ipNASTRANEPSU]` – Tolerance for displacement criterion in SOL 106.

`[ipNASTRANEPSP]` – Tolerance for force criterion in SOL 106.

`[ipNASTRANEPSW]` – Tolerance for work criterion in SOL 106.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

St7ExportANSYS

Exports the current Strand7 model as an ANSYS model file.

```
long St7ExportANSYS(long uID, char* FileName, long* Integers, long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the ANSYS file to be created.

`Integers[0..9]`

`[ipANSYSExportFormat]` – Export format; one of ieANSYSBatch1Export, ieANSYSBatch3Export, ieANSYSBlockedCDBExport or ieANSYSUnblockedCDBExport.

`[ipANSYSFreedomCase]` – Exported freedom case.

`[ipANSYSLoadCase]` – Exported load case for pre-load and non-structural mass attributes.

`[ipANSYSUnits]` – Units for the exported file; one of usANSYS_None, usANSYS_kg_m_C, usANSYS_g_cm_C, usANSYS_T_mm_C, usANSYS_sl_ft_F or usANSYS_lbm_in_F.

`[ipANSYSEndRelease]` – Export partial beam end-release attributes; either ieANSYSEndReleaseFixed or ieANSYSEndReleaseFull.

`[ipANSYSExportNonlinearMat]` – Export nonlinear material data; either btTrue or btFalse.

`[ipANSYSExportHeatTransfer]` – Export heat transfer property data; either btTrue or btFalse.

`[ipANSYSExportPreLoadNSMass]` – Export pre-load and non-structural mass attributes; either btTrue or btFalse.

`[ipANSYSExportTetraOption]` – Export Tet4/Tet10 brick elements as SOLID72/SOLID92; either btTrue or btFalse.

`[ipANSYSExportQuad8Option]` – Export Quad8 plate elements as SHELL91; either btTrue or btFalse.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

Animation

St7PlayAnimationFile

Opens and plays a SAF animation file in an animation window.

```
long St7PlayAnimationFile(char* FileName, long* aHandle)
```

Input Parameters

`FileName`

Full path and name for the SAF animation file.

Output Parameters

`aHandle`

The identifier of the opened animation for use by subsequent Strand7 API calls.

St7CreateAnimation

Creates a SAF animation file and plays it in an animation window. The Strand7 model referenced by uID must have a results file open.

```
long St7CreateAnimation(long uID, long* Integers, long* aHandle)
```

Input Parameters

`uID`

Strand7 model file ID.

`Integers[0..3]`

`[ipAniCase]` – The result case index for a single case animation.

`[ipNumFrames]` – For a single animation case, the number of frames. For a multi case animation, use -1 to animate all frames or 0 to animate only those cases marked using *St7SetAnimationCase*.

`[ipAniWidth]` – The width in pixels for the animation window.

`[ipAniHeight]` – The height in pixels for the animation window.

Output Parameters

`aHandle`

The identifier of the created animation for use by subsequent Strand7 API calls.

St7CreateAnimationEmbedded

Creates a SAF animation embedded in a parent window. The Strand7 model referenced by uID must have a results file open.

```
long St7CreateAnimationEmbedded(long uID, HWND pHandle, long* Integers,
     long* aHandle)
```

Input Parameters

`uID`

Strand7 model file ID.

`pHandle`

The Windows handle of the parent window.

`Integers[0..3]`

`[ipAniCase]` – The result case index for a single case animation.

`[ipNumFrames]` – For a single animation case, the number of frames. For a multi case animation, use -1 to animate all frames or 0 to animate only those cases marked using *St7SetAnimationCase*.

`[ipAniWidth]` – The width in pixels for the animation window.

`[ipAniHeight]` – The height in pixels for the animation window.

Output Parameters

`aHandle`

The identifier of the created animation for use by subsequent Strand7 API calls.

St7CreateAnimationFile

Creates an animation file but does not play it.

```
long St7CreateAnimationFile(long uID, long* Integers, char* FileName)
```

Input Parameters

`uID`

Strand7 model file ID.

`Integers[0..4]`

`[ipAniCase]` – The result case index for a single case animation.

`[ipNumFrames]` – For a single animation case, the number of frames. For a multi case animation, use -1 to animate all frames or 0 to animate only those cases marked using *St7SetAnimationCase*.

`[ipAniWidth]` – The width in pixels for the animation window.

`[ipAniHeight]` – The height in pixels for the animation window.

`[ipAniType]` – The animation file type; one of afAniSAF, afAniEXE or afAniAVI.

`FileName`

Full path and name for the animation file.

St7CloseAnimation

Closes a SAF animation that is currently running.

```
long St7CloseAnimation(long aHandle)
```

Input Parameters

`aHandle`

The identifier of the animation to be closed.

St7SetAnimationCase

Sets the state of the specified result case for multi-case animations.

```
long St7SetAnimationCase(long uID, long CaseNum, bool Active)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`Active`

True to include the case in the animation.

St7GetAnimationCase

Returns the state assigned to the specified result case for multi-case animations.

```
long St7GetAnimationCase(long uID, long CaseNum, bool* Active)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

Output Parameters

`Active`

True if the case is included in the animation.

General Model

St7GetTotal

Returns the total number of entities of the specified entity type in a Strand7 model.

```
long St7GetTotal(long uID, long Entity, long* Total)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyLINK, tyVERTEX, tyGEOMETRYEDGE, tyGEOMETRYCOEDGE, tyGEOMETRYLOOP, tyGEOMETRYFACE or tyLOADPATH.

Output Parameters

`Total`

Number of entities in the model.

St7SetCableDroopDirection

Set the initial droop direction applied to cable elements in a Strand7 model.

```
long St7SetCableDroopDirection(long uID, long Direction)
```

Input Parameters

`uID`

Strand7 model file ID.

`Direction`

1: cable droops in positive global X direction;

-1: cable droops in negative global X direction;

2: cable droops in positive global Y direction;

-2: cable droops in negative global Y direction;

3: cable droops in positive global Z direction;

-3: cable droops in negative global Z direction;

St7GetCableDroopDirection

Returns the initial droop direction applied to cable elements in a Strand7 model.

```
long St7GetCableDroopDirection(long uID, long* Direction)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Direction`

1: cable droops in positive global X direction;

-1: cable droops in negative global X direction;

2: cable droops in positive global Y direction;

-2: cable droops in negative global Y direction;

3: cable droops in positive global Z direction;

-3: cable droops in negative global Z direction;

St7SetTitle

Sets information entered in the **NOTES** tab for the Strand7 model Title, Project, Reference or Author.

```
long St7SetTitle(long uID, long TitleType, char* TitleString)
```

Input Parameters

`uID`

Strand7 model file ID.

`TitleType`

One of tbTitle, tbProject, tbReference or tbAuthor.

`TitleString`

A string containing the text entered for the entry TitleType.

St7GetTitle

Returns the information entered in the **NOTES** tab for the Strand7 model Title, Project, Reference, Author, creation and last modification date.

```
long St7GetTitle(long uID, long TitleType, char* TitleString, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`TitleType`

One of tbTitle, tbProject, tbReference, tbAuthor, tbCreated or tbModified.

`MaxStringLen`

Maximum number of characters allocated for TitleString.

Output Parameters

`TitleString`

A string containing the text entered for the entry TitleType.

St7AddComment

Appends a line of text to the Strand7 model's notes. Note lines are identified by their index, from one up to the total number of comments.

```
long St7AddComment(long uID, char* CommentString)
```

Input Parameters

`uID`

Strand7 model file ID.

`CommentString`

String containing the text to be added. Each string is presented as a line in the Strand7 notes tab.

St7GetNumComments

Returns the number of lines of notes in a Strand7 model.

```
long St7GetNumComments(long uID, long* NumComments)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumComments`

Number of lines in the notes.

St7SetComment

Replaces the text in the specified line of a Strand7 model's notes.

```
long St7SetComment(long uID, long Comment, char* CommentString)
```

Input Parameters

`uID`

Strand7 model file ID.

`Comment`

Index number of the note line to be replaced.

`CommentString`

String containing the replacement text. Each string is presented as a line in the Strand7 notes tab.

St7GetComment

Returns the specified line of a Strand7 model's notes.

```
long St7GetComment(long uID, long Comment, char* CommentString,
     long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`Comment`

Index number for the note line to be returned.

`MaxStringLen`

Maximum number of characters allocated for CommentString.

Output Parameters

`CommentString`

String containing the returned note line.

St7DeleteComment

Deletes the specified line of a Strand7 model's notes.

```
long St7DeleteComment(long uID, long Comment)
```

Input Parameters

`uID`

Strand7 model file ID.

`Comment`

Index number of the note line to be deleted.

St7GetBeamAxisSystemInitial

Returns the beam axis system for the specified beam element based on the initial node coordinates. See *Beam Local Coordinates* for additional information.

```
long St7GetBeamAxisSystemInitial(long uID, long BeamNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

Output Parameters

`Doubles[0..8]`

`[0..2]` – A unit vector in the global XYZ system, defining the 1-direction of the beam.

`[3..5]` – A unit vector in the global XYZ system, defining the 2-direction of the beam.

`[6..8]` – A unit vector in the global XYZ system, defining the 3-direction of the beam.

St7GetBeamAxisSystemBirth

Returns the beam axis system for the specified beam element based on the node coordinates at the element's birth stage. The birth stage is determined from the requested result case. See *Beam Local Coordinates* for additional information.

```
long St7GetBeamAxisSystemBirth(long uID, long BeamNum, long ResultCase,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`ResultCase`

Result case number.

Output Parameters

`Doubles[0..8]`

`[0..2]` – A unit vector in the global XYZ system, defining the 1-direction of the beam.

`[3..5]` – A unit vector in the global XYZ system, defining the 2-direction of the beam.

`[6..8]` – A unit vector in the global XYZ system, defining the 3-direction of the beam.

St7GetBeamAxisSystemGNL

Returns the beam axis system for the specified beam element based on the absolute deformed node coordinates at the requested result case for an analysis that considers geometric nonlinearity. See *Beam Local Coordinates* for additional information.

```
long St7GetBeamAxisSystemGNL(long uID, long BeamNum, long ResultCase,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`ResultCase`

Result case number.

Output Parameters

`Doubles[0..8]`

`[0..2]` – A unit vector in the global XYZ system, defining the 1-direction of the beam.

`[3..5]` – A unit vector in the global XYZ system, defining the 2-direction of the beam.

`[6..8]` – A unit vector in the global XYZ system, defining the 3-direction of the beam.

St7GetPlateAxisSystemInitial

Returns the plate axis system for the specified plate element based on the initial node coordinates. See *Plate Local Coordinates* for additional information.

```
long St7GetPlateAxisSystemInitial(long uID, long PlateNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

Output Parameters

`Doubles[0..8]`

`[0..2]` – A unit vector in the global XYZ system, defining the local x direction of the plate.

`[3..5]` – A unit vector in the global XYZ system, defining the local y direction of the plate.

`[6..8]` – A unit vector in the global XYZ system, defining the local z direction of the plate.

St7GetPlateAxisSystemBirth

Returns the plate axis system for the specified plate element based on the node coordinates at the element’s birth stage. The birth stage is determined from the requested result case. See *Plate Local Coordinates* for additional information.

```
long St7GetPlateAxisSystemBirth(long uID, long PlateNum, long ResultCase,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`ResultCase`

Result case number.

Output Parameters

`Doubles[0..8]`

`[0..2]` – A unit vector in the global XYZ system, defining the local x direction of the plate.

`[3..5]` – A unit vector in the global XYZ system, defining the local y direction of the plate.

`[6..8]` – A unit vector in the global XYZ system, defining the local z direction of the plate.

Applicability

Applicable to stLinearStatic, stLinearBuckling, stNonlinearStatic, stQuasiStatic, stNaturalFrequency, stHarmonicResponse, stSpectralResponse, stLinearTransientDynamic and stNonlinearTransientDynamic.

St7GetPlateAxisSystemGNL

Returns the plate axis system for the specified plate element based on the absolute deformed node coordinates at the requested result case for an analysis that considers geometric nonlinearity. See *Plate Local Coordinates* for additional information.

```
long St7GetPlateAxisSystemGNL(long uID, long PlateNum, long ResultCase,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`ResultCase`

Result case number.

Output Parameters

`Doubles[0..8]`

`[0..2]` – A unit vector in the global XYZ system, defining the local x direction of the plate.

`[3..5]` – A unit vector in the global XYZ system, defining the local y direction of the plate.

`[6..8]` – A unit vector in the global XYZ system, defining the local z direction of the plate.

Applicability

Applicable to stLinearStatic, stLinearBuckling, stNonlinearStatic, stQuasiStatic, stNaturalFrequency, stHarmonicResponse, stSpectralResponse, stLinearTransientDynamic and stNonlinearTransientDynamic, as long as geometric nonlinearity was considered in the analysis.

St7GetBrickFaceAxisSystemInitial

Returns the brick face axis system for the specified brick face based on the initial node coordinates. See *Brick Local Coordinates* for additional information.

```
long St7GetBrickFaceAxisSystemInitial(long uID, long BrickNum, long FaceNum,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

Output Parameters

`Doubles[0..8]`

`[0..2]` – A unit vector in the global XYZ system, defining the local x direction of the face.

`[3..5]` – A unit vector in the global XYZ system, defining the local y direction of the face.

`[6..8]` – A unit vector in the global XYZ system, defining the local z direction of the face; note that this is directed out of the median plane of the face.

St7GetBrickFaceAxisSystemBirth

Returns the brick face axis system for the specified brick face based on the node coordinates at the element's birth stage. The birth stage is determined from the requested result case. See *Brick Local Coordinates* for additional information.

```
long St7GetBrickFaceAxisSystemBirth(long uID, long BrickNum, long FaceNum, long
    ResultCase, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`ResultCase`

Result case number.

Output Parameters

`Doubles[0..8]`

`[0..2]` – A unit vector in the global XYZ system, defining the local x direction of the face.

`[3..5]` – A unit vector in the global XYZ system, defining the local y direction of the face.

`[6..8]` – A unit vector in the global XYZ system, defining the local z direction of the face; note that this is directed out of the median plane of the face.

Applicability

Applicable to stLinearStatic, stLinearBuckling, stNonlinearStatic, stQuasiStatic, stNaturalFrequency, stHarmonicResponse, stSpectralResponse, stLinearTransientDynamic and stNonlinearTransientDynamic.

St7GetBrickFaceAxisSystemGNL

Returns the brick face axis system for the specified brick face based on the absolute deformed node coordinates at the requested result case for an analysis that considers geometric nonlinearity. See *Brick Local Coordinates* for additional information.

```
long St7GetBrickFaceAxisSystemGNL(long uID, long BrickNum, long FaceNum, long
     ResultCase, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`ResultCase`

Result case number.

Output Parameters

`Doubles[0..8]`

`[0..2]` – A unit vector in the global XYZ system, defining the local x direction of the face.

`[3..5]` – A unit vector in the global XYZ system, defining the local y direction of the face.

`[6..8]` – A unit vector in the global XYZ system, defining the local z direction of the face; note that this is directed out of the median plane of the face.

Applicability

Applicable to stLinearStatic, stLinearBuckling, stNonlinearStatic, stQuasiStatic, stNaturalFrequency, stHarmonicResponse, stSpectralResponse, stLinearTransientDynamic and stNonlinearTransientDynamic, as long as geometric nonlinearity was considered in the analysis.

St7GetPlateNumPlies

Returns the number of plies in the specified plate element in a Strand7 model.

```
long St7GetPlateNumPlies(long uID, long PlateNum, long* NumPlies)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

Output Parameters

`NumPlies`

Number of plies.

St7SetModelDefaults

Resets display related parameters in a Strand7 model to default values.

```
long St7SetModelDefaults(long uID, long Options, long Mode, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Options`

One of mdViewOptions, mdEntityOptions, mdBeamPreContourOptions, mdPlatePreContourOptions, mdBrickPreContourOptions, mdAttributeOptions, mdResultOptions, mdBeamResultContourOptions, mdPlateResultContourOptions, mdBrickResultContourOptions, mdLinkResultContourOptions or mdPrintOptions.

`Mode`

Either mdFactoryDefaults or mdUserDefaults.

`Integers[0..7]`

For mdViewOptions:

`[ipDefBackgroundTab]` – btTrue to set defaults on the **Background** tab, or btFalse to leave as-is.

`[ipDefAxisTab]` – btTrue to set defaults on the **Axes** tab, or btFalse to leave as-is.

`[ipDefRotationTab]` – btTrue to set defaults on the **Rotation** tab, or btFalse to leave as-is.

[ipDefDrawingTab] – btTrue to set defaults on the **Drawing** tab, or btFalse to leave as-is.

[ipDefPreNumbersTab] – btTrue to set defaults on the **Pre Numbers** tab, or btFalse to leave as-is.

[ipDefFreeEdgeTab] – btTrue to set defaults on the **Free Edge** tab, or btFalse to leave as-is.

[ipDefSelectionTab] – btTrue to set defaults on the **Selecting** tab, or btFalse to leave as-is.

For mdEntityOptions:

[ipDefNodeTab] – btTrue to set defaults on the **Node** tab, or btFalse to leave as-is.

[ipDefBeamTab] – btTrue to set defaults on the **Beam** tab, or btFalse to leave as-is.

[ipDefPlateTab] – btTrue to set defaults on the **Plate** tab, or btFalse to leave as-is.

[ipDefBrickTab] – btTrue to set defaults on the **Brick** tab, or btFalse to leave as-is.

[ipDefLinkTab] – btTrue to set defaults on the **Link** tab, or btFalse to leave as-is.

[ipDefPathTab] – btTrue to set defaults on the **Load Path** tab, or btFalse to leave as-is.

[ipDefVertexTab] – btTrue to set defaults on the **Vertex** tab, or btFalse to leave as-is.

[ipDefFaceTab] – btTrue to set defaults on the **Geometry Face** tab, or btFalse to leave as-is.

For mdBeamPreContourOptions, mdPlatePreContourOptions and mdBrickPreContourOptions:

[ipDefContourStyleTab] – btTrue to set defaults on the **Style** tab, or btFalse to leave as-is.

[ipDefContourLimitsTab] – btTrue to set defaults on the **Limits** tab, or btFalse to leave as-is.

[ipDefContourLegendTab] – btTrue to set defaults on the **Legend** tab, or btFalse to leave as-is.

For mdResultOptions:

[ipDefResShowHideTab] – btTrue to set defaults on the **Show/Hide** tab, or btFalse to leave as-is.

[ipDefResPostNumbersTab] – btTrue to set defaults on the **Numbers** tab, or btFalse to leave as-is.

[ipDefResCombinationsTab] – btTrue to set defaults on the **Combinations** tab, or btFalse to leave as-is.

[ipDefResEnvelopesTab] – btTrue to set defaults on the **Envelopes** tab, or btFalse to leave as-is.

[ipDefResOtherTab] – btTrue to set defaults on the **Other** tab, or btFalse to leave as-is.

For mdBeamResultContourOptions, mdPlateResultContourOptions, mdBrickResultContourOptions and mdLinkResultContourOptions:

[ipDefContourStyleTab] – btTrue to set defaults on the **Style** tab, or btFalse to leave as-is.

[ipDefContourLimitsTab] – btTrue to set defaults on the **Limits** tab, or btFalse to leave as-is.

[ipDefContourLegendTab] – btTrue to set defaults on the **Legend** tab, or btFalse to leave as-is.

[ipDefContourDiagramTab] – btTrue to set defaults on the **Diagram** tab, or btFalse to leave as-is.

For mdAttributeOptions:

[ipDefNodeAttribTab] – btTrue to set defaults on the **Node Attribute** tab, or btFalse to leave as-is.

[ipDefBeamAttribTab] – btTrue to set defaults on the **Beam Attribute** tab, or btFalse to leave as-is.

[ipDefPlateAttribTab] – btTrue to set defaults on the **Plate Attribute** tab, or btFalse to leave as-is.

[ipDefBrickAttribTab] – btTrue to set defaults on the **Brick Attribute** tab, or btFalse to leave as-is.

[ipDefPathAttribTab] – btTrue to set defaults on the **Load Path Attribute** tab, or btFalse to leave as-is.

For mdPrintOptions:

[ipHeaderFooterTab] – btTrue to set defaults on the **Header/Footer** tab, or btFalse to leave as-is.

[ipPageSetupTab] – btTrue to set defaults on the **Page Setup** tab, or btFalse to leave as-is.

[ipFontsTab] – btTrue to set defaults on the **Fonts** tab, or btFalse to leave as-is.

BXS Utilities

St7GetNumBXSLoopsAndPlates

Returns the number of loops and plates in the specified BXS.

```
long St7GetNumBXSLoopsAndPlates(long uID, long PropNum, long* NumLoops,
     long* NumPlates)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

BXS property number.

Output Parameters

`NumLoops`

Number of loops in the BXS.

`NumPlates`

Number of plates in the BXS.

St7GetNumBXSLoopPoints

Returns the number of points contained in the specified loop in a BXS.

```
long St7GetNumBXSLoopPoints(long uID, long PropNum, long LoopNum,
     long* NumPoints)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

The BXS property number.

`LoopNum`

The loop number within the BXS.

Output Parameters

`NumPoints`

The number of points within LoopNum.

St7GetBXSLoop

Returns the coordinates of the points in the specified loop of a BXS. The points are always specified in a 2D plane. Use *St7GetNumBXSLoopPoints* to determine the number of points in a loop.

```
long St7GetBXSLoop(long uID, long PropNum, long LoopNum, long MaxPoints,
     long* NumPoints, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

The BXS property number.

`LoopNum`

The loop number.

`MaxPoints`

The maximum number of points allocated in Doubles; returns all points if set greater than or equal to NumPoints.

Output Parameters

`NumPoints`

The number of points in the specified loop.

`Doubles[0..2*MaxPoints-1]`

An array containing the XY coordinates of the points in the loop. The XY coordinates of point *i* are contained in `Doubles[2*i-2..2*i-1]`.

St7GetBXSLoopType

Returns the loop type of the specified loop. Each BXS has at least one outer loop. Strand7 supports multi-part BXSs whereby each part has an outer loop.

```
long St7GetBXSLoopType(long uID, long PropNum, long LoopNum, long* LoopType)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

The BXS property number.

`LoopNum`

The loop number.

Output Parameters

`LoopType`

One of `ltUnknown`, `ltOuter` or `ltInner`.

St7GenerateBXS

Using the plate elements in the specified model, calculates and returns the BXS section properties, and saves BXS file.

```
long St7GenerateBXS(long uID, char* BXSName, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BXSName`

String containing the full path name of the BXS, for example, 'C:\Users\My Name\Custom Section.bxs'.

Output Parameters

`Doubles[0..38]`

`[ipBXSXBar]` – Centroid x coordinate.

`[ipBXSYBar]` – Centroid y coordinate.

`[ipBXSArea]` – Section area.

`[ipBXSI11]` – Second moment of area about the principal 1 axis.

`[ipBXSI22]` – Second moment of area about the principal 2 axis.

`[ipBXSAngle]` – Angle between the local x and principal 1 axes.

`[ipBXSZ11Plus]` – Section modulus about the principal 1 axis for stress on the outer fibre in the positive 2 axis direction.

`[ipBXSZ11Minus]` – Section modulus about the principal 1 axis for stress on the outer fibre in the negative 2 axis direction.

`[ipBXSZ22Plus]` – Section modulus about the principal 2 axis for stress on the outer fibre in the positive 1 axis direction.

`[ipBXSZ22Minus]` – Section modulus about the principal 2 axis for stress on the outer fibre in the negative 1 axis direction.

`[ipBXSS11]` – Plastic modulus about the principal 1 axis.

`[ipBXSS22]` – Plastic modulus about the principal 2 axis.

`[ipBXSPC1]` – Principal 1 axis direction offset for centroid of principal plastic modulus.

`[ipBXSPC2]` – Principal 2 axis direction offset for centroid of principal plastic modulus.

`[ipBXSr1]` – Radius of gyration in the principal 1 axis direction.

`[ipBXSr2]` – Radius of gyration in the principal 2 axis direction.

`[ipBXSSA1]` – Shear area in the principal 1 axis direction.

`[ipBXSSA2]` – Shear area in the principal 2 axis direction.

`[ipBXSSL1]` – Shear centre offset in the principal 1 axis direction.

`[ipBXSSL2]` – Shear centre offset in the principal 2 axis direction.

`[ipBXSIXX]` – Second moment of area about the global X axis.

`[ipBXSIYY]` – Second moment of area about the global Y axis.

`[ipBXSIXY]` – Second moment of area coupling term in the global XY axes.

`[ipBXSIxxL]` – Second moment of area about the local x axis.

`[ipBXSIyyL]` – Second moment of area about the local y axis.

`[ipBXSIxyL]` – Second moment of area coupling term in the local xy axes.

`[ipBXSZxxPlus]` – Section modulus about the local x axis for stress on the outer fibre in the positive y axis direction.

`[ipBXSZxxMinus]` – Section modulus about the local x axis for stress on the outer fibre in the negative y axis direction.

`[ipBXSZyyPlus]` – Section modulus about the local y axis for stress on the outer fibre in the positive x axis direction.

`[ipBXSZyyMinus]` – Section modulus about the local y axis for stress on the outer fibre in the negative x axis direction.

`[ipBXSSxx]` – Plastic modulus about the local x axis.

`[ipBXSSyy]` – Plastic modulus about the local y axis.

`[ipBXSPCx]` – Local x axis direction offset for centroid of local plastic modulus.

`[ipBXSPCy]` – Local y axis direction offset for centroid of local plastic modulus.

`[ipBXSrx]` – Radius of gyration in the local x axis direction.

`[ipBXSry]` – Radius of gyration in the local y axis direction.

`[ipBXSrdA]` – Radius area integral.

`[ipBXSJ]` – Torsion constant.

`[ipBXSIw]` – Warping constant.

Load and Freedom Cases

St7NewLoadCase

Creates a new load case within a Strand7 model.

```
long St7NewLoadCase(long uID, char* CaseName)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseName`

String containing the name of the new load case.

St7NewFreedomCase

Creates a new freedom case within a Strand7 model.

```
long St7NewFreedomCase(long uID, char* CaseName)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseName`

String containing the name of the new freedom case.

St7GetNumLoadCase

Returns the number of load cases in a Strand7 model.

```
long St7GetNumLoadCase(long uID, long* NumCases)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumCases`

The number of cases.

St7GetNumSeismicCase

Returns the number of primary load cases in a Strand7 model that are of Seismic type.

```
long St7GetNumSeismicCase(long uID, long* NumCases)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumCases`

The number of cases.

St7GetNumFreedomCase

Returns the number of freedom cases in a Strand7 model.

```
long St7GetNumFreedomCase(long uID, long* NumCases)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumCases`

The number of cases.

St7SetLoadCaseName

Sets the name of the specified load case in a Strand7 model.

```
long St7SetLoadCaseName(long uID, long CaseNum, char* CaseName)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

`CaseName`

String containing the new name of the load case.

St7GetLoadCaseName

Returns the name of the specified load case within a Strand7 model.

```
long St7GetLoadCaseName(long uID, long CaseNum, char* CaseName,
     long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

`MaxStringLen`

The maximum number of characters allocated for CaseName.

Output Parameters

`CaseName`

String containing the name of the specified load case.

St7SetFreedomCaseName

Sets the name of the specified freedom case within a Strand7 model.

```
long St7SetFreedomCaseName(long uID, long CaseNum, char* CaseName)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

The freedom case ID.

`CaseName`

String containing the new name for the freedom case.

St7GetFreedomCaseName

Returns the name of the specified freedom case in a Strand7 model.

```
long St7GetFreedomCaseName(long uID, long CaseNum, char* CaseName,
     long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

The freedom case ID.

`MaxStringLen`

The maximum number of characters allocated for CaseName.

Output Parameters

`CaseName`

String containing the name of the specified freedom case.

St7SetLoadCaseDefaults

Sets the defaults for the specified load case in a Strand7 model; not applicable to seismic primary load cases.

```
long St7SetLoadCaseDefaults(long uID, long CaseNum, double* Defaults)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

`Defaults[0..12]`

`[ipLoadCaseRefTemp]` – Reference temperature.

`[ipLoadCaseOrigX, ipLoadCaseOrigY, ipLoadCaseOrigZ]` – Origin for angular velocity and acceleration.

`[ipLoadCaseAccX, ipLoadCaseAccY, ipLoadCaseAccZ]` – Linear acceleration components.

`[ipLoadCaseAngVelX, ipLoadCaseAngVelY, ipLoadCaseAngVelZ]` – Angular velocity components.

`[ipLoadCaseAngAccX, ipLoadCaseAngAccY, ipLoadCaseAngAccZ]` – Angular acceleration components.

St7GetLoadCaseDefaults

Returns the default values for the specified load case within a Strand7 model; not applicable to seismic primary load cases.

```
long St7GetLoadCaseDefaults(long uID, long CaseNum, double* Defaults)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

Output Parameters

`Defaults[0..12]`

`[ipLoadCaseRefTemp]` – Reference temperature.

`[ipLoadCaseOrigX, ipLoadCaseOrigY, ipLoadCaseOrigZ]` – Origin for angular velocity and acceleration.

`[ipLoadCaseAccX, ipLoadCaseAccY, ipLoadCaseAccZ]` – Linear acceleration components.

`[ipLoadCaseAngVelX, ipLoadCaseAngVelY, ipLoadCaseAngVelZ]` – Angular velocity components.

`[ipLoadCaseAngAccX, ipLoadCaseAngAccY, ipLoadCaseAngAccZ]` – Angular acceleration components.

St7SetSeismicCaseDefaults

Sets the defaults for the specified primary load case of type lcSeismic within a Strand7 model.

```
long St7SetSeismicCaseDefaults(long uID, long CaseNum, double* Defaults)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

The primary load case ID.

`Defaults[0..9]`

`[ipSeismicCaseRefTemp]` – Reference temperature.

`[ipSeismicCaseAlpha]` – Seismic case α parameter.

`[ipSeismicCasePhi]` – Seismic case ϕ parameter.

`[ipSeismicCaseBeta]` – Seismic case β parameter.

`[ipSeismicCaseK]` – Seismic case k parameter.

`[ipSeismicCaseh0]` – Seismic base height parameter h_0.

`[ipSeismicCaseDir]` – Seismic acceleration direction; one of 1, 2 or 3 to denote the global XYZ directions respectively.

`[ipSeismicCaseLinAcc]` – Seismic acceleration value.

`[ipSeismicCaseV1]` – Component of base excitation in the first lateral direction.

`[ipSeismicCaseV2]` – Component of base excitation in the second lateral direction.

St7GetSeismicCaseDefaults

Returns the defaults for the specified primary load case of type lcSeismic within a Strand7 model.

```
long St7GetSeismicCaseDefaults(long uID, long CaseNum, double* Defaults)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

The primary load case ID.

Output Parameters

`Defaults[0..9]`

`[ipSeismicCaseRefTemp]` – Reference temperature.

`[ipSeismicCaseAlpha]` – Seismic case α parameter.

`[ipSeismicCasePhi]` – Seismic case ϕ parameter.

`[ipSeismicCaseBeta]` – Seismic case β parameter.

`[ipSeismicCaseK]` – Seismic case k parameter.

`[ipSeismicCaseh0]` – Seismic base height parameter h_0.

`[ipSeismicCaseDir]` – Seismic acceleration direction; one of 1, 2 or 3 to denote the global XYZ directions respectively.

`[ipSeismicCaseLinAcc]` – Seismic acceleration value.

`[ipSeismicCaseV1]` – Component of base excitation in the first lateral direction.

`[ipSeismicCaseV2]` – Component of base excitation in the second lateral direction.

St7SetFreedomCaseDefaults

Sets the defaults for the specified freedom case within a Strand7 model.

```
long St7SetFreedomCaseDefaults(long uID, long CaseNum, long* Defaults)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

The freedom case ID.

`Defaults[0..5]`

An array describing the global restraint conditions for each DoF in the global XYZ system. `Defaults[i] = btTrue` indicates that DoF `i` is fixed.

St7GetFreedomCaseDefaults

Returns the defaults for the specified freedom case in a Strand7 model.

```
long St7GetFreedomCaseDefaults(long uID, long CaseNum, long* Defaults)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

The freedom case ID.

Output Parameters

`Defaults[0..5]`

An array describing the global restraint conditions for each DoF in the global XYZ system. `Defaults[i] = btTrue` indicates that DoF `i` is fixed.

St7SetLoadCaseType

Sets the type for the specified load case in a Strand7 model.

```
long St7SetLoadCaseType(long uID, long CaseNum, long CaseType)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

`CaseType`

One of lcNoInertia, lcGravity, lcAccelerations or lcSeismic.

St7GetLoadCaseType

Returns the type for the specified load case in a Strand7 model.

```
long St7GetLoadCaseType(long uID, long CaseNum, long* CaseType)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

Output Parameters

`CaseType`

One of lcNoInertia, lcGravity, lcAccelerations or lcSeismic.

St7SetLoadCaseGravityDir

Sets the direction of the gravity vector for the specified load case.

```
long St7SetLoadCaseGravityDir(long uID, long CaseNum, long GravDir)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

`GravDir`

Gravity direction specified in the global XYZ system; one of 1, 2 or 3.

St7GetLoadCaseGravityDir

Returns the direction of the gravity vector assigned to the specified load case.

```
long St7GetLoadCaseGravityDir(long uID, long CaseNum, long* GravDir)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

Output Parameters

`GravDir`

Gravity direction specified in the global XYZ system; one of 1, 2 or 3.

St7SetLoadCaseGravity

Sets the value of the acceleration due to gravity for the specified load case of type lcGravity.

```
long St7SetLoadCaseGravity(long uID, long CaseNum, double Gravity)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

`Gravity`

Acceleration due to gravity.

St7GetLoadCaseGravity

Returns the value of the acceleration due to gravity for the specified load case of type lcGravity.

```
long St7GetLoadCaseGravity(long uID, long CaseNum, double* Gravity)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

Output Parameters

`Gravity`

Acceleration due to gravity.

St7SetFreedomCaseType

Sets the type for the specified freedom case in a Strand7 model.

```
long St7SetFreedomCaseType(long uID, long CaseNum, long CaseType)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

The freedom case ID.

`CaseType`

One of fcNormalFreedom, fcFreeBodyInertiaRelief, fcSingleSymmetryInertiaXY, fcSingleSymmetryInertiaYZ, fcSingleSymmetryInertiaZX, fcDoubleSymmetryInertiaX, fcDoubleSymmetryInertiaY or fcDoubleSymmetryInertiaZ.

St7GetFreedomCaseType

Returns the type of the specified freedom case with a Strand7 model.

```
long St7GetFreedomCaseType(long uID, long CaseNum, long* CaseType)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

The freedom case ID.

Output Parameters

`CaseType`

One of fcNormalFreedom, fcFreeBodyInertiaRelief, fcSingleSymmetryInertiaXY, fcSingleSymmetryInertiaYZ, fcSingleSymmetryInertiaZX, fcDoubleSymmetryInertiaX, fcDoubleSymmetryInertiaY or fcDoubleSymmetryInertiaZ.

St7SetLoadCaseMassOption

Sets the mass options for the specified load case in a Strand7 model.

```
long St7SetLoadCaseMassOption(long uID, long CaseNum, bool SMass, bool NSMass)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

`SMass`

True to apply global accelerations to structural mass.

`NSMass`

True to apply global accelerations to non-structural mass.

St7GetLoadCaseMassOption

Returns the mass options for the specified load case in a Strand7 model.

```
long St7GetLoadCaseMassOption(long uID, long CaseNum, bool* SMass, bool* NSMass)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

Output Parameters

`SMass`

If True, global accelerations are applied to structural mass.

`NSMass`

If True, global accelerations are applied to non-structural mass.

St7SetSeismicCaseDynamicNSMassState

Sets the enabled state of the dynamic factor option for the specified seismic load case in a Strand7 model.

```
long St7SetSeismicCaseDynamicNSMassState(long uID, long CaseNum, bool Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

`Enabled`

True to **Apply dynamic factors to non-structural mass attributes**.

St7GetSeismicCaseDynamicNSMassState

Returns the enabled state of the dynamic factor option for the specified seismic load case in a Strand7 model.

```
long St7GetSeismicCaseDynamicNSMassState(long uID, long CaseNum, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

Output Parameters

`Enabled`

True if **Apply dynamic factors to non-structural mass attributes**.

St7SetFreedomCaseRigidMotion

Sets the remove rigid body motion option for freedom cases of type inertia relief.

```
long St7SetFreedomCaseRigidMotion(long uID, long CaseNum, bool Remove)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Freedom case number.

`Remove`

True to remove rigid body motion.

St7GetFreedomCaseRigidMotion

Returns the remove rigid body motion option for freedom cases of type inertia relief.

```
long St7GetFreedomCaseRigidMotion(long uID, long CaseNum, bool* Remove)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Freedom case number.

Output Parameters

`Remove`

If True, rigid body motion is removed.

St7DeleteLoadCase

Deletes the specified load case from the Strand7 model. Since load cases use contiguous numbering this will cause all the following load case numbers to shift down by one.

```
long St7DeleteLoadCase(long uID, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

St7DeleteFreedomCase

Deletes the specified freedom case in the Strand7 model. Since freedom cases use contiguous numbering this will cause all the following freedom case numbers to shift down by one.

```
long St7DeleteFreedomCase(long uID, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Freedom case number.

Coordinate Systems

St7SetUCS

Sets the data for the specified User Coordinate System (UCS) in a Strand7 model, or creates a UCS if the specified UCSId does not exist.

```
long St7SetUCS(long uID, long UCSId, long UCSType, double* UCSDoubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`UCSType`

One of csCartesian, csCylindrical, csSpherical or csToroidal.

`UCSDoubles[0..kMaxUCSDoubles-1]`

An array defining the UCS axis system; see *Coordinate System Conventions*.

St7GetUCS

Returns the data for the specified UCS in a Strand7 model.

```
long St7GetUCS(long uID, long UCSId, long* UCSType, double* UCSDoubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

Output Parameters

`UCSType`

One of csCartesian, csCylindrical, csSpherical or csToroidal.

`UCSDoubles[0..kMaxUCSDoubles-1]`

An array defining the UCS axis system; see *Coordinate System Conventions*.

St7DeleteUCS

Deletes the specified UCS in a Strand7 model.

```
long St7DeleteUCS(long uID, long UCSId)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system, which cannot be deleted.

St7SetUCSName

Sets the name of the specified UCS in a Strand7 model.

```
long St7SetUCSName(long uID, long UCSId, char* UCSName)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system, which cannot be renamed.

`UCSName`

String containing the new name of the UCS.

St7GetUCSName

Returns the name of the specified UCS in a Strand7 model.

```
long St7GetUCSName(long uID, long UCSId, char* UCSName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`MaxStringLen`

The maximum number of characters allocated for UCSName.

Output Parameters

`UCSName`

String containing the name of the UCS.

St7GetUCSID

Returns the ID number corresponding to the specified UCS index in a Strand7 model.

```
long St7GetUCSID(long uID, long Index, long* UCSId)
```

Input Parameters

`uID`

Strand7 model file ID.

`Index`

The UCS index number. The list of available UCSs is always contiguous.

Output Parameters

`UCSId`

The UCS ID number corresponding to Index. UCS ID numbers are not required to be contiguous.

St7GetNumUCS

Returns the number of UCSs in a Strand7 model.

```
long St7GetNumUCS(long uID, long* NumUCS)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumUCS`

The number of UCSs.

Groups

St7SetGroupIDName

Sets the name of the specified group in a Strand7 model.

```
long St7SetGroupIDName(long uID, long ID, char* GName)
```

Input Parameters

`uID`

Strand7 model file ID.

`ID`

The ID of the specified group.

`GName`

String containing the name of the group.

St7GetGroupIDName

Returns the name of the specified group in a Strand7 model.

```
long St7GetGroupIDName(long uID, long ID, char* GName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`ID`

The ID of the specified group.

`MaxStringLen`

The maximum number of characters allocated for GName.

Output Parameters

`GName`

String containing the name of the group.

St7GetNumGroups

Returns the number of groups in a Strand7 model.

```
long St7GetNumGroups(long uID, long* NumGroups)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumGroups`

The number of groups.

St7GetGroupByIndex

Returns the group name and ID number corresponding to the specified index.

```
long St7GetGroupByIndex(long uID, long Index, char* GName, long MaxStringLen,
    long* GroupID)
```

Input Parameters

`uID`

Strand7 model file ID.

`Index`

The index of the specified group. The list of group indices is always contiguous, starting from 1.

`MaxStringLen`

The maximum number of characters allocated for GName.

Output Parameters

`GName`

String containing the name of the specified group.

`GroupID`

The ID number corresponding to the specified group. Group ID numbers are not required to be contiguous.

St7NewChildGroup

Creates a new child group within the specified group parent after its last child.

```
long St7NewChildGroup(long uID, long ParentID, char* GName, long* ChildID)
```

Input Parameters

`uID`

Strand7 model file ID.

`ParentID`

The ID number for the parent group.

`GName`

String containing the name of the new group.

Output Parameters

`ChildID`

The ID number for the newly created group.

St7GetGroupParent

Returns the parent of the specified group. ParentID is -1 if the specified group is the root group.

```
long St7GetGroupParent(long uID, long GroupID, long* ParentID)
```

Input Parameters

`uID`

Strand7 model file ID.

`GroupID`

The ID number of the specified child group.

Output Parameters

`ParentID`

The ID number of the parent group.

St7GetGroupChild

Returns the first child of the specified group. ChildID is -1 if the specified group has no children.

```
long St7GetGroupChild(long uID, long GroupID, long* ChildID)
```

Input Parameters

`uID`

Strand7 model file ID.

`GroupID`

The ID number of the specified parent group.

Output Parameters

`ChildID`

The ID number of the group child.

St7GetGroupSibling

Returns the next sibling of the specified group. SiblingID is -1 if the specified group has no subsequent siblings.

```
long St7GetGroupSibling(long uID, long GroupID, long* SiblingID)
```

Input Parameters

`uID`

Strand7 model file ID.

`GroupID`

The ID number of the specified group.

Output Parameters

`SiblingID`

The ID number of the group sibling.

St7DeleteGroup

Deletes the specified group. Elements in the group will be reassigned to the Model group.

```
long St7DeleteGroup(long uID, long GroupID)
```

Input Parameters

`uID`

Strand7 model file ID.

`GroupID`

The ID number of the group to delete.

St7SetGroupColour

Sets the colour of the specified group for entity display purposes.

```
long St7SetGroupColour(long uID, long GroupID, long GroupCol)
```

Input Parameters

`uID`

Strand7 model file ID.

`GroupID`

The ID number of the specified group.

`GroupCol`

Group colour. See also *RGB Colours*.

St7GetGroupColour

Returns the colour of the specified group for entity display purposes.

```
long St7GetGroupColour(long uID, long GroupID, long* GroupCol)
```

Input Parameters

`uID`

Strand7 model file ID.

`GroupID`

The ID number of the specified group.

Output Parameters

`GroupCol`

Group colour. See also *RGB Colours*.

St7SetDefaultGroupID

Sets the group for newly created elements where the group cannot be determined from the source.

```
long St7SetDefaultGroupID(long uID, long GroupID)
```

Input Parameters

`uID`

Strand7 model file ID.

`GroupID`

Group identifier.

St7GetDefaultGroupID

Returns the group that will be assigned to newly created elements where the group cannot be determined from the source.

```
long St7GetDefaultGroupID(long uID, long* GroupID)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`GroupID`

Group identifier.

Staged Analysis

St7AddStage

Adds a new stage to a Strand7 model.

```
long St7AddStage(long uID, char* StageName, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`StageName`

String containing the name of the new stage.

`Integers[0..4]`

`[ipStageMorph]` – Morphing option; either btTrue or btFalse.

`[ipStageMoveFixedNodes]` – Move fixed nodes option; either btTrue or btFalse.

`[ipStageRotateClusters]` – Rotate clusters option; either btTrue or btFalse.

`[ipStageSetFluidLevel]` – Set fluid level for soil in the stage; either btTrue or btFalse.

`[ipStageReset]` – Stage reset option; either btTrue or btFalse.

St7InsertStage

Inserts a new stage in a Strand7 model.

```
long St7InsertStage(long uID, long Stage, char* StageName, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Index at which to insert the new stage.

`StageName`

String containing the name of the new stage.

`Integers[0..4]`

`[ipStageMorph]` – Morphing option; either btTrue or btFalse.

`[ipStageMoveFixedNodes]` – Move fixed nodes option; either btTrue or btFalse.

`[ipStageRotateClusters]` – Rotate clusters option; either btTrue or btFalse.

`[ipStageSetFluidLevel]` – Set fluid level for soil in the stage; either btTrue or btFalse.

`[ipStageReset]` – Stage reset option; either btTrue or btFalse.

St7DeleteStage

Deletes a stage from a Strand7 model.

```
long St7DeleteStage(long uID, long Stage)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Index of the stage to be deleted.

St7GetNumStages

Returns the number of stages in a Strand7 model.

```
long St7GetNumStages(long uID, long* NumStages)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumStages`

The number of stages.

St7SetStageName

Sets the name of the specified stage.

```
long St7SetStageName(long uID, long Stage, char* StageName)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index.

`StageName`

String containing the new name of the stage.

St7GetStageName

Returns the name of the specified stage.

```
long St7GetStageName(long uID, long Stage, char* StageName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index.

`MaxStringLen`

The maximum number of characters allocated for StageName.

Output Parameters

`StageName`

String containing the name of the specified stage.

St7SetStageData

Sets the data for the specified stage.

```
long St7SetStageData(long uID, long Stage, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index.

`Integers[0..4]`

`[ipStageMorph]` – Morphing option; either btTrue or btFalse.

`[ipStageMoveFixedNodes]` – Move fixed nodes option; either btTrue or btFalse.

`[ipStageRotateClusters]` – Rotate clusters option; either btTrue or btFalse.

`[ipStageSetFluidLevel]` – Set fluid level for soil in the stage; either btTrue or btFalse.

`[ipStageReset]` – Stage reset option; either btTrue or btFalse.

St7GetStageData

Returns the data for the specified stage.

```
long St7GetStageData(long uID, long Stage, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index.

Output Parameters

`Integers[0..4]`

`[ipStageMorph]` – Morphing option; either btTrue or btFalse.

`[ipStageMoveFixedNodes]` – Move fixed nodes option; either btTrue or btFalse.

`[ipStageRotateClusters]` – Rotate clusters option; either btTrue or btFalse.

`[ipStageSetFluidLevel]` – Set fluid level for soil in the stage; either btTrue or btFalse.

`[ipStageReset]` – Stage reset option; either btTrue or btFalse.

St7SetStageFluidLevel

Sets the fluid level for soil properties in the specified stage.

```
long St7SetStageFluidLevel(long uID, long Stage, double Level)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index.

`Level`

Fluid level for soil in the stage.

St7GetStageFluidLevel

Returns the fluid level for soil properties in the specified stage.

```
long St7GetStageFluidLevel(long uID, long Stage, double* Level)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index.

Output Parameters

`Level`

Fluid level for soil in the stage.

St7EnableStageGroup

Enables the specified group for a given stage. The elements in all groups enabled for a given stage will participate in the solution once the specified stage becomes active.

```
long St7EnableStageGroup(long uID, long Stage, long GroupID)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index.

`GroupID`

The ID number for the group to be enabled for the specified stage.

St7DisableStageGroup

Disables the specified group for a given stage. The elements in all groups enabled for a given stage will participate in the solution once the specified stage becomes active.

```
long St7DisableStageGroup(long uID, long Stage, long GroupID)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index.

`GroupID`

The ID number for the group to be disabled for the specified stage.

St7GetStageGroupState

Returns whether the specified group is currently enabled for a given stage.

```
long St7GetStageGroupState(long uID, long Stage, long GroupID, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index.

`GroupID`

The ID number for the specified group.

Output Parameters

`Enabled`

True indicates that the specified group is enabled for the given stage.

Entity Sets

St7NewEntitySet

Creates a new entity set. Since entity sets are contiguously numbered, the newly created entity set number will be one greater than the previous count.

```
long St7NewEntitySet(long uID, char* SetName)
```

Input Parameters

`uID`

Strand7 model file ID.

`SetName`

The name of the entity set.

St7DeleteEntitySet

Deletes an entity set.

```
long St7DeleteEntitySet(long uID, long SetNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`SetNum`

The number of the entity set.

St7GetNumEntitySets

Returns the number of entity sets.

```
long St7GetNumEntitySets(long uID, long* NumSets)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumSets`

The number of entity sets.

St7SetEntitySetName

Sets the name of the specified entity set.

```
long St7SetEntitySetName(long uID, long SetNum, char* SetName)
```

Input Parameters

`uID`

Strand7 model file ID.

`SetNum`

The number of the entity set.

`SetName`

The name of the entity set.

St7GetEntitySetName

Returns the name of the specified entity set.

```
long St7GetEntitySetName(long uID, long SetNum, char* SetName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`SetNum`

The number of the entity set.

`MaxStringLen`

The maximum number of characters allocated for SetName.

Output Parameters

`SetName`

The name of the entity set.

St7AddSelectedToEntitySet

Adds the selected entities to an entity set.

```
long St7AddSelectedToEntitySet(long uID, long Entity, long SetNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyLINK, tyLOADPATH, tyVERTEX, tyGEOMETRYFACE or tyGEOMETRYCOEDGE.

`SetNum`

The number of the entity set.

Dependencies

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7RemoveSelectedFromEntitySet

Removes the selected entities from an entity set.

```
long St7RemoveSelectedFromEntitySet(long uID, long Entity, long SetNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyLINK, tyLOADPATH, tyVERTEX, tyGEOMETRYFACE or tyGEOMETRYCOEDGE.

`SetNum`

The number of the entity set.

Dependencies

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7GetEntitySetEntityState

Checks whether or not an entity is included in an entity set.

```
long St7GetEntitySetEntityState(long uID, long Entity, long EntityNum,
     long SetNum, bool* Included)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyLINK, tyLOADPATH, tyVERTEX, tyGEOMETRYFACE or tyGEOMETRYCOEDGE.

`EntityNum`

Entity number.

`SetNum`

The number of the entity set.

`Included`

True if the entity is included in the entity set.

Units

St7SetUnits

Sets the units used to specify and report measures of length, force, stress, mass, temperature and energy in the specified Strand7 model.

```
long St7SetUnits(long uID, long* Units)
```

Input Parameters

`uID`

Strand7 model file ID.

`Units[0..kLastUnit-1]`

`[ipLENGTHU]` – one of luMETRE, luCENTIMETRE, luMILLIMETRE, luFOOT or luINCH;
measuring length in metres, centimetres, millimetres, feet or inches respectively.

`[ipFORCEU]` – one of fuNEWTON, fuKILONEWTON, fuMEGANEWTON, fuKILOFORCE, fuPOUNDFORCE, fuTONNEFORCE or fuKIPFORCE;
measuring force in newtons, kilonewtons, meganewtons, kilograms-force, pounds-force, tonnes-force or kilopounds-force respectively.

`[ipSTRESSU]` – one of suPASCAL, suKILOPASCAL, suMEGAPASCAL, suKSCm, suPSI, suKSI or suPSF;
measuring stress in units of pascals, kilopascals, megapascals, kilograms-force per square centimetre, pounds per square inch, kilopounds per square inch, or pounds per square foot respectively.

`[ipMASSU]` – one of muKILOGRAM, muTONNE, muGRAM, muPOUND or muSLUG;
measuring mass in units of kilograms, tonnes, grams, pounds or slugs respectively.

`[ipTEMPERU]` – one of tuCELSIUS, tuFAHRENHEIT, tuKELVIN or tuRANKINE;
measuring temperature in units of Celsius, Fahrenheit, Kelvin or Rankine respectively.

`[ipENERGYU]` – one of euJOULE, euKILOJOULE, euBTU, euFTLBF or euCALORIE;
measuring energy in units of joules, kilojoules, British thermal units, foot pounds-force or calories respectively.

St7GetUnits

Returns the units used to specify and report measures of length, force, stress, mass, temperature and energy in the specified Strand7 model.

```
long St7GetUnits(long uID, long* Units)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Units[0..kLastUnit-1]`

`[ipLENGTHU]` – one of luMETRE, luCENTIMETRE, luMILLIMETRE, luFOOT or luINCH;
measuring length in metres, centimetres, millimetres, feet or inches respectively.

`[ipFORCEU]` – one of fuNEWTON, fuKILONEWTON, fuMEGANEWTON, fuKILOFORCE, fuPOUNDFORCE, fuTONNEFORCE or fuKIPFORCE;
measuring force in newtons, kilonewtons, meganewtons, kilograms-force, pounds-force, tonnes-force or kilopounds-force respectively.

`[ipSTRESSU]` – one of suPASCAL, suKILOPASCAL, suMEGAPASCAL, suKSCm, suPSI, suKSI or suPSF;
measuring stress in units of pascals, kilopascals, megapascals, kilograms-force per square centimetre, pounds per square inch, kilopounds per square inch, or pounds per square foot respectively.

`[ipMASSU]` – one of muKILOGRAM, muTONNE, muGRAM, muPOUND or muSLUG;
measuring mass in units of kilograms, tonnes, grams, pounds or slugs respectively.

`[ipTEMPERU]` – one of tuCELSIUS, tuFAHRENHEIT, tuKELVIN or tuRANKINE;
measuring temperature in units of Celsius, Fahrenheit, Kelvin or Rankine respectively.

`[ipENERGYU]` – one of euJOULE, euKILOJOULE, euBTU, euFTLBF or euCALORIE;
measuring energy in units of joules, kilojoules, British thermal units, foot pounds-force or calories respectively.

St7SetRCUnits

Sets the units used for Plate RC results (in **Results Settings**). Note that this setting is ignored unless *St7EnableModelRCUnit* is called.

```
long St7SetRCUnits(long uID, long AreaUnit, long LengthUnit)
```

Input Parameters

`uID`

Strand7 model file ID.

`AreaUnit`

One of luMETRE, luCENTIMETRE, luMILLIMETRE, luFOOT or luINCH.

`LengthUnit`

One of luMETRE, luCENTIMETRE, luMILLIMETRE, luFOOT or luINCH.

St7GetRCUnits

Returns the units used for Plate RC results (in **Results Settings**). Note that this setting is ignored unless *St7EnableModelRCUnit* is called.

```
long St7GetRCUnits(long uID, long* AreaUnit, long* LengthUnit)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`AreaUnit`

One of luMETRE, luCENTIMETRE, luMILLIMETRE, luFOOT or luINCH.

`LengthUnit`

One of luMETRE, luCENTIMETRE, luMILLIMETRE, luFOOT or luINCH.

St7ConvertUnits

Converts the current model into the specified units system.

```
long St7ConvertUnits(long uID, long* Units)
```

Input Parameters

`uID`

Strand7 model file ID.

`Units[0..kLastUnit-1]`

`[ipLENGTHU]` – luMETRE, luCENTIMETRE, luMILLIMETRE, luFOOT or luINCH.

`[ipFORCEU]` – fuNEWTON, fuKILONEWTON, fuMEGANEWTON, fuKILOFORCE, fuPOUNDFORCE, fuTONNEFORCE or fuKIPFORCE.

`[ipSTRESSU]` – suPASCAL, suKILOPASCAL, suMEGAPASCAL, suKSCm, suPSI, suKSI or suPSF.

`[ipMASSU]` – muKILOGRAM, muTONNE, muGRAM, muPOUND or muSLUG.

`[ipTEMPERU]` – tuCELSIUS, tuFAHRENHEIT, tuKELVIN or tuRANKINE.

`[ipENERGYU]` – euJOULE, euKILOJOULE, euBTU, euFTLBF or euCALORIE.

St7EnableModelStrainUnit

Allows the strain units set by *St7SetResultOptions* to override the report of absolute strains, which is the API default.

```
long St7EnableModelStrainUnit(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7DisableModelStrainUnit

Restores the API default report of absolute strains, overriding the strain units set by *St7SetResultOptions*.

```
long St7DisableModelStrainUnit(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7EnableModelRotationUnit

Allows the rotation units set by *St7SetResultOptions* to override the report of rotation in radians, which is the API default.

```
long St7EnableModelRotationUnit(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7DisableModelRotationUnit

Restores the API default report of rotations in radians, overriding the rotation units set by *St7SetResultOptions*.

```
long St7DisableModelRotationUnit(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7EnableModelRCUnit

Allows the length and area units set by *St7SetRCUnits* to override the report of plate RC results in consistent model units, which is the API default.

```
long St7EnableModelRCUnit(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7DisableModelRCUnit

Restores the API default report of plate RC results in consistent model units, overriding the length and area units set by *St7SetRCUnits*.

```
long St7DisableModelRCUnit(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

Entities – Nodes, Elements and Links

St7SetNodeXYZ

Sets the position of the specified node in the global XYZ system. A new node is created if the node number does not already exist. If the new node number is not consecutive with the existing node total a series of nodes are created at the origin such that the node list remains contiguous.

```
long St7SetNodeXYZ(long uID, long NodeNum, double* XYZ)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`XYZ[0..2]`

Node position as a 3-element array in global XYZ coordinates.

St7GetNodeXYZ

Returns the coordinates of the specified node in the global XYZ system in its undeformed position.

```
long St7GetNodeXYZ(long uID, long NodeNum, double* XYZ)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

Output Parameters

`XYZ[0..2]`

Node position as a 3-element array in global XYZ coordinates.

St7SetNodeUCS

Sets the position of the specified node in a given UCS. A new node is created if the node number does not already exist. If the new node number is not consecutive with the existing node total a series of nodes are created at the origin such that the node list remains contiguous.

```
long St7SetNodeUCS(long uID, long NodeNum, long UCSId, double* XYZ)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`XYZ[0..2]`

Node position as a 3-element array specifying the position according to the 123 axis UCS convention.

St7GetNodeUCS

Returns the coordinates of the specified node in a given UCS in its undeformed position.

```
long St7GetNodeUCS(long uID, long NodeNum, long UCSId, double* XYZ)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

Output Parameters

`XYZ[0..2]`

Node position as a 3-element array specifying the position according to the 123 axis UCS convention.

St7SetElementConnection

Sets the nodal connectivity and property ID for the specified element. A new entity is created if the element number does not already exist. If the new element number is not consecutive with the existing element total a series of null elements are created such that the element list remains contiguous. These null elements do not have any connectivity or property ID assigned.

```
long St7SetElementConnection(long uID, long Entity, long EntityNum, long PropNum,
    long* Connection)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK or tyLINK.

`EntityNum`

Entity number.

`PropNum`

The intended property ID or link type.

Where Entity is tyBEAM, tyPLATE or tyBRICK: the property ID of the new element.

Where Entity is tyLINK: one of ltAttachmentLink, ltCouplingLink, ltMasterSlaveLink, ltPinnedLink, ltRigidLink, ltSectorSymmetryLink, ltShrinkLink or ltTwoPointLink.

`Connection[0..kMaxElementNode]`

`[0]` – Number of nodes in the element.

`[1..20]` – Node numbers in the element.

See *Element Connections* for additional information.

Dependencies

`Default Group`

Target group for the element is specified by *St7SetDefaultGroupID*.

St7GetElementConnection

Returns the connectivity information for the specified element.

```
long St7GetElementConnection(long uID, long Entity, long EntityNum,
     long* Connection)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK or tyLINK.

`EntityNum`

Entity number; note that multi-point links are not supported.

Output Parameters

`Connection[0..kMaxElementNode]`

`[0]` – Number of nodes in the element.

`[1..20]` – Node numbers in the element.

See *Element Connections* for additional information.

St7GetLinkType

Returns the link type for the specified link.

```
long St7GetLinkType(long uID, long LinkNum, long* LinkType)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

Output Parameters

`LinkType`

One of the link types described in *Link Types.*

St7SetMasterSlaveLink

Assigns the parameters for the specified master-slave link.

```
long St7SetMasterSlaveLink(long uID, long LinkNum, long UCSId, long* Connection,
     long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Connection[0..2]`

`[0]` – Number of nodes in the link (2).

`[1..2]` – Node numbers.

`Integers[0..5]`

A 6-element array describing the relationship between each DoF in the linked nodes according to the UCS axis system. Entries for each DoF may be one of msFree, msFix or msFixNegate.

Dependencies

`Default Group`

Target group for the link is specified by *St7SetDefaultGroupID*.

St7GetMasterSlaveLink

Returns the parameters for the specified master-slave link.

```
long St7GetMasterSlaveLink(long uID, long LinkNum, long* UCSId, long* Connection,
     long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

Output Parameters

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Connection[0..2]`

`[0]` – Number of nodes in the link (2).

`[1..2]` – Node numbers.

`Integers[0..5]`

A 6-element array describing the relationship between each DoF in the linked nodes according to the UCS axis system. Entries for each DoF may be one of msFree, msFix or msFixNegate.

St7SetSectorSymmetryLink

Assigns the parameters for the specified sector-symmetry link.

```
long St7SetSectorSymmetryLink(long uID, long LinkNum, long Axis,
     long* Connection)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`Axis`

Axis of symmetry in the global XYZ system; one of 1, 2 or 3.

`Connection[0..2]`

`[0]` – Number of nodes in the link (2).

`[1..2]` – Node numbers.

Dependencies

`Default Group`

Target group for the link is specified by *St7SetDefaultGroupID*.

St7GetSectorSymmetryLink

Returns the parameters for the specified sector-symmetry link.

```
long St7GetSectorSymmetryLink(long uID, long LinkNum, long* Axis,
     long* Connection)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

Output Parameters

`Axis`

Axis of symmetry in the global XYZ system; one of 1, 2 or 3.

`Connection[0..2]`

`[0]` – Number of nodes in the link (2).

`[1..2]` – Node numbers.

St7SetCouplingLink

Assigns the parameters for the specified coupling link.

```
long St7SetCouplingLink(long uID, long LinkNum, long Couple, long* Connection)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`Couple`

One of cpTranslational, cpRotational or cpBoth.

`Connection[0..3]`

`[0]` – Number of nodes in the link (3).

`[1..3]` – Node numbers.

See *Element Connections* for additional information.

Dependencies

`Default Group`

Target group for the link is specified by *St7SetDefaultGroupID*.

St7GetCouplingLink

Returns the parameters for the specified coupling link.

```
long St7GetCouplingLink(long uID, long LinkNum, long* Couple, long* Connection)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

Output Parameters

`Couple`

One of cpTranslational, cpRotational or cpBoth.

`Connection[0..3]`

`[0]` – Number of nodes in the link (3).

`[1..3]` – Node numbers.

See *Element Connections* for additional information.

St7SetPinnedLink

Assigns the parameters for the specified pinned link.

```
long St7SetPinnedLink(long uID, long LinkNum, long* Connection)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`Connection[0..2]`

`[0]` – Number of nodes in the link (2).

`[1..2]` – Node numbers.

Dependencies

`Default Group`

Target group for the link is specified by *St7SetDefaultGroupID*.

St7GetPinnedLink

Returns the parameters for the specified pinned link.

```
long St7GetPinnedLink(long uID, long LinkNum, long* Connection)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

Output Parameters

`Connection[0..2]`

`[0]` – Number of nodes in the link (2).

`[1..2]` – Node numbers.

St7SetRigidLink

Assigns the parameters for the specified rigid link.

```
long St7SetRigidLink(long uID, long LinkNum, long UCSId, long Plane,
     long* Connection)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`UCSId`

ID number of the specified Cartesian UCS. UCSId = 1 refers to the global XYZ system.

`Plane`

One of rlPlaneXYZ, rlPlaneXY, rlPlaneYZ or rlPlaneZX.

`Connection[0..2]`

`[0]` – Number of nodes in the link (2).

`[1..2]` – Node numbers.

Dependencies

`Default Group`

Target group for the link is specified by *St7SetDefaultGroupID*.

St7GetRigidLink

Returns the parameters for the specified rigid link.

```
long St7GetRigidLink(long uID, long LinkNum, long* UCSId, long* Plane,
     long* Connection)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

Output Parameters

`UCSId`

ID number of the specified Cartesian UCS. UCSId = 1 refers to the global XYZ system.

`Plane`

One of rlPlaneXYZ, rlPlaneXY, rlPlaneYZ or rlPlaneZX.

`Connection[0..2]`

`[0]` – Number of nodes in the link (2).

`[1..2]` – Node numbers.

St7SetShrinkLink

Assigns the parameters for the specified shrink link.

```
long St7SetShrinkLink(long uID, long LinkNum, long* Connection, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`Connection[0..2]`

`[0]` – Number of nodes in the link (2).

`[1..2]` – Node numbers.

`Integers[0..3]`

`[0]` – btTrue to shrink in the global X direction.

`[1]` – btTrue to shrink in the global Y direction.

`[2]` – btTrue to shrink in the global Z direction.

`[3]` – Freedom case controlling the shrink factor when used in nonlinear analysis, or 0 to not set this parameter.

Dependencies

`Default Group`

Target group for the link is specified by *St7SetDefaultGroupID*.

St7GetShrinkLink

Returns the parameters for the specified shrink link.

```
long St7GetShrinkLink(long uID, long LinkNum, long* Connection, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

Output Parameters

`Connection[0..2]`

`[0]` – Number of nodes in the link (2).

`[1..2]` – Node numbers.

`Integers[0..3]`

`[0]` – btTrue to shrink in the global X direction.

`[1]` – btTrue to shrink in the global Y direction.

`[2]` – btTrue to shrink in the global Z direction.

`[3]` – Freedom case controlling the shrink factor when used in nonlinear analysis, or 0 if this parameter is not set.

St7SetTwoPointLink

Assigns the parameters for the specified two-point link.

```
long St7SetTwoPointLink(long uID, long LinkNum, long* Connection, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`Connection[0..2]`

`[0]` – Number of nodes in the link (2).

`[1..2]` – Node numbers.

`Integers[0..4]`

`[ipTwoPointDOF1]` – DoF at node 1; one of 1, 2, 3, 4, 5 or 6.

`[ipTwoPointDOF2]` – DoF at node 2; one of 1, 2, 3, 4, 5 or 6.

`[ipTwoPointUCS1]` – UCS ID number at node 1.

`[ipTwoPointUCS2]` – UCS ID number at node 2.

`[ipTwoPointFC]` – Freedom case that scales the constant coefficient in a nonlinear analysis, or 0 to not set this parameter.

`Doubles[0..2]`

`[ipTwoPointC1]` – Coefficient of node 1.

`[ipTwoPointC2]` – Coefficient of node 2.

`[ipTwoPointConst]` – Constant coefficient.

Dependencies

`Default Group`

Target group for the link is specified by *St7SetDefaultGroupID*.

St7GetTwoPointLink

Returns the parameters for the specified two-point link.

```
long St7GetTwoPointLink(long uID, long LinkNum, long* Connection, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

Output Parameters

`Connection[0..2]`

`[0]` – Number of nodes in the link (2).

`[1..2]` – Node numbers.

`Integers[0..4]`

`[ipTwoPointDOF1]` – DoF at node 1; one of 1, 2, 3, 4, 5 or 6.

`[ipTwoPointDOF2]` – DoF at node 2; one of 1, 2, 3, 4, 5 or 6.

`[ipTwoPointUCS1]` – UCS ID number at node 1.

`[ipTwoPointUCS2]` – UCS ID number at node 2.

`[ipTwoPointFC]` – Freedom case that scales the constant coefficient in a nonlinear analysis, or 0 if this parameter is not set.

`Doubles[0..2]`

`[ipTwoPointC1]` – Coefficient of node 1.

`[ipTwoPointC2]` – Coefficient of node 2.

`[ipTwoPointConst]` – Constant coefficient.

St7SetAttachmentLink

Assigns the parameters for the specified attachment link.

```
long St7SetAttachmentLink(long uID, long LinkNum, long* Connection,
     long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`Connection[0..1]`

`[0]` – Number of nodes in the link (1).

`[1]` – Node number for attached node.

`Integers[0..3]`

`[ipAttachmentElType]` – Target entity type; one of tyBEAM, tyPLATE or tyBRICK.

`[ipAttachmentElNum]` – Target element number.

`[ipAttachmentBrickFaceNum]` – Target face number for tyBRICK; either 0 to attach to the inside of the brick, or one of 1, 2, 3, 4, 5 or 6 to attach to the corresponding brick face.

`[ipAttachmentCouple]` – Connection between the degrees of freedom with target element; one of cpTranslational, cpRotational or cpBoth.

`Doubles[0..2]`

A 3-element array containing the intrinsic uvw coordinates of the target attachment location.

For ipAttachmentElType = tyBEAM, only the first element is relevant and is clipped to [0.0, +1.0].

For ipAttachmentElType = tyPLATE, only the first two elements are relevant. Both are clipped to [-1.0, +1.0].

For ipAttachmentElType = tyBRICK, if ipAttachmentBrickFaceNum=0 all three values are required. If ipAttachmentBrickFaceNum>0, only the first two are relevant. In both cases the range [-1.0, +1.0] is used.

Dependencies

`Default Group`

Target group for the link is specified by *St7SetDefaultGroupID*.

St7GetAttachmentLink

Returns the parameters for the specified attachment link.

```
long St7GetAttachmentLink(long uID, long LinkNum, long* Connection,
     long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

Output Parameters

`Connection[0..1]`

`[0]` – Number of nodes in the link (1).

`[1]` – Node number for attached node.

`Integers[0..3]`

`[ipAttachmentElType]` – Target entity type; one of tyBEAM, tyPLATE or tyBRICK.

`[ipAttachmentElNum]` – Target element number.

`[ipAttachmentBrickFaceNum]` – Target face number for tyBRICK; either 0 for the inside of the brick, or one of 1, 2, 3, 4, 5 or 6 for the corresponding brick face.

`[ipAttachmentCouple]`- Connection between the degrees of freedom with target element; one of cpTranslational, cpRotational or cpBoth.

`Doubles[0..2]`

A 3-element array containing the intrinsic uvw coordinates of the target attachment location.

For ipAttachmentElType = tyBEAM, only the first element is assigned using the range [0.0, +1.0].

For ipAttachmentElType = tyPLATE, the first two elements are assigned. Both in the range [-1.0, +1.0].

For ipAttachmentElType = tyBRICK, if ipAttachmentBrickFaceNum=0 all three values are assigned. If ipAttachmentBrickFaceNum>0, only the first two are assigned. In both cases the range [-1.0, +1.0] is used.

St7GetNumMultiPointLinkNodes

Returns the number of nodes in the specified multi-point link.

```
long St7GetNumMultiPointLinkNodes(long uID, long LinkNum, long* NumNodes)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

Output Parameters

`NumNodes`

The number of nodes in the link.

St7SetReactionMultiPointLink

Assigns the parameters for the specified reaction multi-point link.

```
long St7SetReactionMultiPointLink(long uID, long LinkNum, long NumNodes,
     long SetNum, long* Connection, double* Origin)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`NumNodes`

The number of nodes in the link.

`SetNum`

Element set number defining the entities that contribute to the reaction multi-point link cluster.

`Connection[0..NumNodes-1]`

Node numbers for linked nodes.

`Origin[0..2]`

Origin position in global XYZ coordinates.

Dependencies

`Default Group`

Target group for the link is specified by *St7SetDefaultGroupID*.

St7GetReactionMultiPointLink

Returns the parameters for the specified reaction multi-point link.

```
long St7GetReactionMultiPointLink(long uID, long LinkNum, long MaxNodes,
     long* NumNodes, long* SetNum, long* Connection, double* Origin)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`MaxNodes`

The allocated size of the Connection array. If the full connectivity of the link is required this should be at least as large as the number of nodes attached to the link, which can be obtained using *St7GetNumMultiPointLinkNodes*.

Output Parameters

`NumNodes`

The number of nodes in the link.

`SetNum`

Element set number defining the entities that contribute to the reaction multi-point link cluster.

`Connection[0..Min(NumNodes,MaxNodes)-1]`

Node numbers for linked nodes.

`Origin[0..2]`

Origin position in global XYZ coordinates.

St7SetInterpolatedMultiPointLink

Assigns the parameters for the specified interpolated multi-point link.

```
long St7SetInterpolatedMultiPointLink(long uID, long LinkNum, long NumNodes,
     long Couple, long* Connection)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`NumNodes`

The number of nodes in the link.

`Couple`

One of cpTranslational, cpRotational or cpBoth.

`Connection[0..NumNodes-1]`

Node numbers for linked nodes, with the slave node specified first.

Dependencies

`Default Group`

Target group for the link is specified by *St7SetDefaultGroupID*.

St7GetInterpolatedMultiPointLink

Returns the parameters for the specified interpolated multi-point link.

```
long St7GetInterpolatedMultiPointLink(long uID, long LinkNum, long MaxNodes,
     long* NumNodes, long* Couple, long* Connection)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`MaxNodes`

The allocated size of the Connection array. If the full connectivity of the link is required this should be at least as large as the number of nodes attached to the link, which can be obtained using *St7GetNumMultiPointLinkNodes*.

Output Parameters

`NumNodes`

The number of nodes in the link.

`Couple`

One of cpTranslational, cpRotational or cpBoth.

`Connection[0..Min(NumNodes,MaxNodes)-1]`

Node numbers for linked nodes, with the slave node specified first.

St7SetMasterSlaveMultiPointLink

Assigns the parameters for the specified master-slave multi-point link.

```
long St7SetMasterSlaveMultiPointLink(long uID, long LinkNum, long NumNodes,
     long UCSId, long DOFBits, long* Connection)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`NumNodes`

The number of nodes in the link.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`DOFBits`

Bitmask of flags for six degrees of freedom if Master-Slave links are created. For example, to couple DX, DY and RZ, set ConstraintBits = 1+2+32 = 35.

`Connection[0..NumNodes-1]`

Node numbers for linked nodes, with the slave node specified first.

Dependencies

`Default Group`

Target group for the link is specified by *St7SetDefaultGroupID*.

St7GetMasterSlaveMultiPointLink

Returns the parameters for the specified master-slave multi-point link.

```
long St7GetMasterSlaveMultiPointLink(long uID, long LinkNum, long MaxNodes,
     long* NumNodes, long* UCSId, long* DOFBits, long* Connection)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`MaxNodes`

The allocated size of the Connection array. If the full connectivity of the link is required this should be at least as large as the number of nodes attached to the link, which can be obtained using *St7GetNumMultiPointLinkNodes*.

Output Parameters

`NumNodes`

The number of nodes in the link.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`DOFBits`

Bitmask of flags for six degrees of freedom if Master-Slave links are created. For example, to couple DX, DY and RZ, set ConstraintBits = 1+2+32 = 35.

`Connection[0..Min(NumNodes,MaxNodes)-1]`

Node numbers for linked nodes, with the slave node specified first.

St7SetPinnedMultiPointLink

Assigns the parameters for the specified pinned multi-point link.

```
long St7SetPinnedMultiPointLink(long uID, long LinkNum, long NumNodes,
     long* Connection)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`NumNodes`

The number of nodes in the link.

`Connection[0..NumNodes-1]`

Node numbers for linked nodes, with the slave node specified first.

Dependencies

`Default Group`

Target group for the link is specified by *St7SetDefaultGroupID*.

St7GetPinnedMultiPointLink

Returns the parameters for the specified pinned multi-point link.

```
long St7GetPinnedMultiPointLink(long uID, long LinkNum, long MaxNodes,
     long* NumNodes, long* Connection)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`MaxNodes`

The allocated size of the Connection array. If the full connectivity of the link is required this should be at least as large as the number of nodes attached to the link, which can be obtained using *St7GetNumMultiPointLinkNodes*.

Output Parameters

`NumNodes`

The number of nodes in the link.

`Connection[0..Min(NumNodes,MaxNodes)-1]`

Node numbers for linked nodes, with the slave node specified first.

St7SetRigidMultiPointLink

Assigns the parameters for the specified rigid multi-point link.

```
long St7SetRigidMultiPointLink(long uID, long LinkNum, long NumNodes, long UCSId,
     long Axis, long* Connection)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`NumNodes`

The number of nodes in the link.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Axis`

One of rlPlaneXYZ, rlPlaneXY, rlPlaneYZ or rlPlaneZX.

`Connection[0..NumNodes-1]`

Node numbers for linked nodes, with the slave node specified first.

Dependencies

`Default Group`

Target group for the link is specified by *St7SetDefaultGroupID*.

St7GetRigidMultiPointLink

Returns the parameters for the specified rigid multi-point link.

```
long St7GetRigidMultiPointLink(long uID, long LinkNum, long MaxNodes,
     long* NumNodes, long* UCSId, long* Axis, long* Connection)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`MaxNodes`

The allocated size of the Connection array. If the full connectivity of the link is required this should be at least as large as the number of nodes attached to the link, which can be obtained using *St7GetNumMultiPointLinkNodes*.

Output Parameters

`NumNodes`

The number of nodes in the link.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Axis`

One of rlPlaneXYZ, rlPlaneXY, rlPlaneYZ or rlPlaneZX.

`Connection[0..Min(NumNodes,MaxNodes)-1]`

Node numbers for linked nodes, with the slave node specified first.

St7SetUserDefinedMultiPointLink

Assigns the parameters for the specified user defined multi-point link.

```
long St7SetUserDefinedMultiPointLink(long uID, long LinkNum, long NumNodes,
     long CaseNum, double CFactor, long* Connection, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`NumNodes`

The number of nodes in the link.

`CaseNum`

Freedom case that scales the constant factor in nonlinear and time-based analyses, or 0.

`CFactor`

Constant factor.

`Connection[0..NumNodes-1]`

Node numbers for linked nodes, with the slave node specified first.

`Integers[0..NumNodes-1]`

DoF for linked nodes, with the slave DoF specified first.

`Doubles[0..NumNodes-1]`

Factors for linked nodes, with the slave factor specified first.

Dependencies

`Default Group`

Target group for the link is specified by *St7SetDefaultGroupID*.

St7GetUserDefinedMultiPointLink

Returns the parameters for the specified user defined multi-point link.

```
long St7GetUserDefinedMultiPointLink(long uID, long LinkNum, long MaxNodes,
     long* NumNodes, long* CaseNum, double* CFactor, long* Connection,
     long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`MaxNodes`

The allocated size of the Connection array. If the full connectivity of the link is required this should be at least as large as the number of nodes attached to the link, which can be obtained using *St7GetNumMultiPointLinkNodes*.

Output Parameters

`NumNodes`

The number of nodes in the link.

`CaseNum`

Freedom case that scales the constant factor in nonlinear and time-based analyses, or 0.

`CFactor`

Constant factor.

`Connection[0..Min(NumNodes,MaxNodes)-1]`

Node numbers for linked nodes, with the slave node specified first.

`Integers[0..Min(NumNodes,MaxNodes)-1]`

DoF for linked nodes, with the slave DoF specified first.

`Doubles[0..Min(NumNodes,MaxNodes)-1]`

Factors for linked nodes, with the slave factor specified first.

St7CreateLinksFromMultiPointLink

Creates equivalent individual links from selected Rigid, Pinned and Master-Slave multi-point links.

```
long St7CreateLinksFromMultiPointLink(long uID, bool DeleteMPL)
```

Input Parameters

`uID`

Strand7 model file ID.

`DeleteMPL`

True to delete the source links.

Dependencies

`Selection`

Links can be selected using functions in *Entity Selection*.

Entities – Geometry

St7GetVertexXYZ

Returns the position of the specified vertex.

```
long St7GetVertexXYZ(long uID, long VertexNum, double* XYZ)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

Output Parameters

`XYZ[0..2]`

Vertex position as a 3-element array in global XYZ coordinates.

St7InsertVerticesOnEdge

Inserts vertices at prescribed positions along an edge.

```
long St7InsertVerticesOnEdge(long uID, long EdgeID, long NumVertex,
     long VertexType, double* Positions)
```

Input Parameters

`uID`

Strand7 model file ID.

`EdgeID`

Edge number on which to insert vertices.

`NumVertex`

Number of positions along edge to insert vertices.

`VertexType`

Either vtFree or vtFixed.

`Positions[0..NumVertex-1]`

Positions along edge to insert vertices, normalised to the range 0.0 to 1.0. Duplicated or out-of-range positions are ignored by the function.

St7GetGeometryFaceOuterLoops

Returns the outer loops in the specified geometry face, note that a geometry face may have one or two outer loops only.

```
long St7GetGeometryFaceOuterLoops(long uID, long FaceNum, long* OuterLoops)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

Output Parameters

`OuterLoops[0..1]`

`[0]` The loop number of the first outer loop.

`[1]` The loop number of the second outer loop, 0 if there is no such loop.

St7GetNumGeometryFaceCavityLoops

Returns the number of cavity loops in the specified geometry face.

```
long St7GetNumGeometryFaceCavityLoops(long uID, long FaceNum,
    long* NumCavityLoops)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

Output Parameters

`NumCavityLoops`

Number of cavity loops in the specified face.

St7GetGeometryFaceCavityLoops

Returns the cavity loop numbers for the specified geometry face. Use *St7GetNumGeometryFaceCavityLoops* to determine the number of cavity loops in the specified geometry face.

```
long St7GetGeometryFaceCavityLoops(long uID, long FaceNum, long MaxCavityLoops,
    long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`MaxCavityLoops`

Maximum amount of storage allocated for Integers. Returns all loops if set greater than or equal to NumCavityLoops.

Output Parameters

`Integers[0..MaxCavityLoops-1]`

An array containing the cavity loop numbers for the specified face such that `Integers[i-1]` contains the i^{th} cavity loop number.

St7GetNumGeometryFaceEdges

Returns the number of edges in the specified geometry face.

```
long St7GetNumGeometryFaceEdges(long uID, long FaceNum, long* NumEdges)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

Output Parameters

`NumEdges`

Number of edges in the specified face.

St7GetGeometryFaceEdges

Returns the edge numbers for the specified geometry face. Use *St7GetNumGeometryFaceEdges* to determine the number of edges in the specified geometry face.

```
long St7GetGeometryFaceEdges(long uID, long FaceNum, long MaxEdges,
     long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`MaxEdges`

Maximum amount of storage allocated for Integers. Returns all edges if set greater than or equal to NumEdges.

Output Parameters

`Integers[0..MaxEdges-1]`

An array containing the edge numbers for the specified face such that `Integers[i-1]` contains the i^{th} edge number.

St7GetNumGeometryLoopEdges

Returns the number of edges in the specified loop.

```
long St7GetNumGeometryLoopEdges(long uID, long LoopNum, long* NumEdges)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoopNum`

Loop number.

Output Parameters

`NumEdges`

Number of edges in the loop.

St7GetGeometryLoopEdges

Returns the edge numbers for the specified loop. Use *St7GetNumGeometryLoopEdges* to determine the number of edges in the specified loop.

```
long St7GetGeometryLoopEdges(long uID, long LoopNum, long MaxEdges,
    long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoopNum`

Loop number.

`MaxEdges`

Maximum amount of storage allocated for Integers. Returns all edges if set greater than or equal to NumEdges.

Output Parameters

`Integers[0..MaxEdges-1]`

An array containing the edge numbers for the specified loop such that `Integers[i-1]` contains the i^{th} edge number.

St7GetNumGeometryFaceCoedges

Returns the number of coedges in the specified geometry face.

```
long St7GetNumGeometryFaceCoedges(long uID, long FaceNum, long* NumCoedges)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

Output Parameters

`NumCoedges`

Number of coedges in the specified face.

St7GetGeometryFaceCoedges

Returns the coedge numbers for the specified geometry face. Use *St7GetNumGeometryFaceCoedges* to determine the number of coedges in the specified geometry face.

```
long St7GetGeometryFaceCoedges(long uID, long FaceNum, long MaxCoedges,
     long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`MaxCoedges`

Maximum amount of storage allocated for Integers. Returns all coedges if set greater than or equal to NumCoedges.

Output Parameters

`Integers[0..MaxCoedges-1]`

An array containing the coedge numbers for the specified face such that `Integers[i-1]` contains the i^{th} coedge number.

St7GetNumGeometryLoopCoedges

Returns the number of edges in the specified loop.

```
long St7GetNumGeometryLoopCoedges(long uID, long LoopNum, long* NumCoedges)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoopNum`

Loop number.

Output Parameters

`NumCoedges`

Number of Coedges in the loop.

St7GetGeometryLoopCoedges

Returns the edge numbers for the specified loop. Use *St7GetNumGeometryLoopCoedges* to determine the number of edges in the specified loop.

```
long St7GetGeometryLoopCoedges(long uID, long LoopNum, long MaxCoedges,
    long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoopNum`

Loop number.

`MaxCoedges`

Maximum amount of storage allocated for Integers. Returns all coedges if set greater than or equal to NumCoedges.

Output Parameters

`Integers[0..MaxCoedges-1]`

An array containing the coedge numbers for the specified loop such that `Integers[i-1]` contains the i^{th} coedge number.

St7GetGeometryCoedgeEdge

Returns the edge number of a coedge. More than one coedge can be associated with a single edge.

```
long St7GetGeometryCoedgeEdge(long uID, long CoedgeNum, long* EdgeNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

Output Parameters

`EdgeNum`

Edge number.

St7GetGeometryEdgeLength

Returns the length of the specified edge in a given geometry face.

```
long St7GetGeometryEdgeLength(long uID, long EdgeNum, double* EdgeLength)
```

Input Parameters

`uID`

Strand7 model file ID.

`EdgeNum`

Edge number.

Output Parameters

`EdgeLength`

Length of the specified edge.

St7GetNumGeometryFaceVertices

Returns the number of vertices in a given geometry face.

```
long St7GetNumGeometryFaceVertices(long uID, long FaceNum, long* NumVertices)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

Output Parameters

`NumVertices`

Number of vertices in the specified face.

St7GetGeometryFaceVertices

Returns the vertex numbers for the specified geometry face. Use *St7GetNumGeometryFaceVertices* to determine the number of vertices for the specified geometry face.

```
long St7GetGeometryFaceVertices(long uID, long FaceNum, long MaxVertices,
     long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`MaxVertices`

Maximum amount of storage allocated for Integers. Returns all vertices if set greater than or equal to NumVertices.

Output Parameters

`Integers[0..MaxVertices-1]`

An array containing the vertex numbers for the specified face such that `Integers[i-1]` contains the i^{th} vertex number.

St7GetGeometryEdgeVertices

Returns the vertex numbers in the specified geometry edge.

```
long St7GetGeometryEdgeVertices(long uID, long EdgeNum, long* EdgeVertices)
```

Input Parameters

`uID`

Strand7 model file ID.

`EdgeNum`

Edge number.

Output Parameters

`EdgeVertices[0..1]`

A 2-element array containing the start and end vertices for the specified edge.

St7GetGeometryFaceSurface

Returns the surface number for the specified geometry face.

```
long St7GetGeometryFaceSurface(long uID, long FaceNum, long* SurfaceNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

Output Parameters

`SurfaceNum`

Surface number.

St7GetGeometrySurfaceType

Returns the type of the specified surface.

```
long St7GetGeometrySurfaceType(long uID, long SurfaceNum, long* SurfaceType)
```

Input Parameters

`uID`

Strand7 model file ID.

`SurfaceNum`

Surface number.

Output Parameters

`SurfaceType`

One of suPlane, suSphere, suTorus, suCone, suBSpline, suRotSur, suPipeSur, suSumSur, suTabCyl, suRuleSur, suCubicSpline or suNull.

St7GetGeometrySize

Retrieves the relative overall size of the geometry in the specified Strand7 model. This size measure is used when calculating the relative geometry tolerance.

```
long St7GetGeometrySize(long uID, double* Size)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Size`

Relative overall geometry size.

Entities – Load Paths

St7SetLoadPath

Sets the data for a load path in the specified model. A new load path is created if a new load path ID is specified.

```
long St7SetLoadPath(long uID, long LoadPathID, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoadPathID`

Load path ID.

`Integers[0..6]`

`[ipLoadPathCase]` – Load case number.

`[ipLoadPathTemplate]` – Load path template number.

`[ipLoadPathShape]` – Load path shape; one of lpShapeStraight, lpShapeCurved, lpShapeQuadratic.

`[ipLoadPathSurface]` – Load path surface; either lpSurfaceFlat or lpSurfaceCurved. Note that this parameter is ignored when lpShapeStraight is set in `Integers[ipLoadPathShape]`.

`[ipLoadPathTarget]` – Load path target entity. lpAnyEntity for all entities, lpEntitySet to target the entity set specified at `Integers[ipLoadPathSet]`, or one of lpBeamElement, lpPlateElement or lpBrickElement for all Beams, Plates or Bricks respectively.

`[ipLoadPathDivisions]` – Number of divisions along the load path.

`[ipLoadPathSet]` – Entity set the load path targets if `Integers[ipLoadPathTarget]` is lpEntitySet.

`Doubles[0..8]`

`[0..2]` – The start XYZ point in the definition of the load path (defined in the global XYZ system).

`[3..5]` – The end XYZ point in the definition of the load path.

`[6..8]` – The lateral XYZ point in the definition of the load path; used to define the plane of the load path and its curvature (for circular load paths).

Dependencies

`Default Group`

Target group for the load path is specified by *St7SetDefaultGroupID*.

St7GetLoadPath

Returns the data assigned to the specified load path.

```
long St7GetLoadPath(long uID, long LoadPathID, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoadPathID`

Load path ID.

Output Parameters

`Integers[0..6]`

`[ipLoadPathCase]` – Load case number.

`[ipLoadPathTemplate]` – Load path template number.

`[ipLoadPathShape]` – Load path shape; one of lpShapeStraight, lpShapeCurved, lpShapeQuadratic.

`[ipLoadPathSurface]` – Load path surface; either lpSurfaceFlat or lpSurfaceCurved. Note that this parameter is ignored when lpShapeStraight is set in `Integers[ipLoadPathShape]`.

`[ipLoadPathTarget]` – Load path target entity. lpAnyEntity for all entities, lpEntitySet to target the entity set specified at `Integers[ipLoadPathSet]`, or one of lpBeamElement, lpPlateElement or lpBrickElement for all Beams, Plates or Bricks respectively.

`[ipLoadPathDivisions]` – Number of divisions along the load path.

`[ipLoadPathSet]` – Entity set the load path targets if `Integers[ipLoadPathTarget]` is lpEntitySet.

`Doubles[0..8]`

`[0..2]` – The start XYZ point in the definition of the load path (defined in the global XYZ system).

`[3..5]` – The end XYZ point in the definition of the load path.

`[6..8]` – The lateral XYZ point in the definition of the load path; used to define the plane of the load path and its curvature (for circular load paths).

St7DeleteLoadPath

Deletes the specified load path.

```
long St7DeleteLoadPath(long uID, long LoadPathID)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoadPathID`

Load path ID.

Node Attributes – Set

St7SetNodeID

Sets the ID number of the specified node.

```
long St7SetNodeID(long uID, long NodeNum, long NodeID)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`NodeID`

ID number for the specified node.

St7SetNodeRestraint6

Sets the restraint conditions at the specified node in the specified UCS.

```
long St7SetNodeRestraint6(long uID, long NodeNum, long CaseNum, long UCSId,
     long* Status, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Freedom case number.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Status[0..5]`

A 6-element array describing the restraint conditions for the six DoF at the specified node. `Status[i-1] = btTrue` indicates that the i^{th} DoF is restrained. The DoF are restrained according to the 123456 axis convention in the specified UCS.

`Doubles[0..5]`

A 6-element array describing the enforced displacement or rotation conditions for the six DoF at the specified node. `Doubles[i-1]` describes the displacement of the i^{th} DoF according to the 123456 axis convention in the specified UCS.

St7SetNodeForce3

Sets the point force acting on the specified node in the global XYZ system.

```
long St7SetNodeForce3(long uID, long NodeNum, long CaseNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

`Doubles[0..2]`

A 3-element array describing the nodal force in the global XYZ system.

St7SetNodeMoment3

Sets the point moment acting on the specified node in the global XYZ system.

```
long St7SetNodeMoment3(long uID, long NodeNum, long CaseNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

`Doubles[0..2]`

A 3-element array describing the nodal moment in the global XYZ system.

St7SetNodeTemperature1

Sets the temperature at the specified node.

```
long St7SetNodeTemperature1(long uID, long NodeNum, long CaseNum,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

`Doubles[0]`

The nodal temperature value at the specified node.

St7SetNodeTemperatureType1

Sets the type of temperature at the specified node.

```
long St7SetNodeTemperatureType1(long uID, long NodeNum, long CaseNum, long TType)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

`TType`

The type of temperature attribute applied at the specified node: ntReferenceTemperature, ntFixedTemperature, ntInitialTemperature or ntTableTemperature. If required, use *St7SetNodeTemperatureTable* to set the table.

St7SetNodeTemperatureTable

Specifies the table to be associated with the temperature at the specified node. A table can only be assigned for nodes with the appropriate table temperature type, as set using the *St7SetNodeTemperatureType1* function.

```
long St7SetNodeTemperatureTable(long uID, long NodeNum, long CaseNum,
    long TableID)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

`TableID`

Temperature vs Time table ID associated with the node temperature, or 0 for none.

St7SetNodeKTranslation3F

Sets the translational stiffness acting at the specified node.

```
long St7SetNodeKTranslation3F(long uID, long NodeNum, long CaseNum, long UCSId,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Freedom case number.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Doubles[0..2]`

A 3-element array describing the translational stiffnesses for the specified node. `Doubles[i-1]` describes the stiffness for the i^{th} translational DoF according to the 123 axis definition in the specified UCS.

St7SetNodeKRotation3F

Sets the rotational stiffness acting at the specified node.

```
long St7SetNodeKRotation3F(long uID, long NodeNum, long CaseNum, long UCSId,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Freedom case number.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Doubles[0..2]`

A 3-element array describing the rotational stiffnesses for the specified node. `Doubles[i-1]` describes the stiffness for the i^{th} rotational DoF according to the 456 axis definition in the specified UCS.

St7SetNodeTMass1

Sets the translational mass assigned to the specified node as a single value. Translational masses are active in all load and freedom cases.

```
long St7SetNodeTMass1(long uID, long NodeNum, double Mass)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`Mass`

The translational mass at the node, which acts in all three global axes directions.

St7SetNodeTMass3

Sets the translational mass assigned to the specified node as three components. Translational masses are active in all load and freedom cases.

```
long St7SetNodeTMass3(long uID, long NodeNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`Doubles[0..2]`

A 3-element array describing the translational mass for the node in the global XYZ directions.

St7SetNodeRMass3

Sets the rotational mass assigned to the specified node as three components. Rotational masses are active in all load and freedom cases.

```
long St7SetNodeRMass3(long uID, long NodeNum, long UCSId, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Doubles[0..2]`

A 3-element array defining the rotational mass for the node about the axes of the specified UCS.

St7SetNodeNSMass5ID

Sets the non-structural mass at the specified node.

```
long St7SetNodeNSMass5ID(long uID, long NodeNum, long CaseNum, long ID,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

`ID`

The non-structural mass ID.

`Doubles[0..4]`

`[0]` – The non-structural mass at the specified node.

`[1]` – The dynamic factor at the specified node. This factor is used to scale the non-structural mass when performing dynamic analysis.

`[2..4]` – A 3-element array describing the offset in the global XYZ system.

St7SetNodeKDamping3F

Sets the translational damping coefficients at the specified node.

```
long St7SetNodeKDamping3F(long uID, long NodeNum, long CaseNum, long UCSId,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Doubles[0..2]`

A 3-element array describing the damping factors for the specified node. `Doubles[i-1]` describes the damping factor for the i^{th} translational DoF according to the 123 axis definition in the specified UCS.

St7SetNodeHeatSource1

Sets the heat source at the specified node.

```
long St7SetNodeHeatSource1(long uID, long NodeNum, long CaseNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

`Doubles[0]`

Heat source.

St7SetNodeHeatSourceTables

Sets the tables associated with the specified node heat source.

```
long St7SetNodeHeatSourceTables(long uID, long NodeNum, long CaseNum,
     long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

`Tables[0..1]`

`[0]` – Factor vs Time table ID associated with the heat source, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the heat source, or 0 for none.

St7SetNodeInitialVelocity3

Sets the initial global velocity components for the specified node. These initial conditions are used when performing transient dynamic analysis.

```
long St7SetNodeInitialVelocity3(long uID, long NodeNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

`Doubles[0..2]`

A 3-element array describing the initial velocity components for the specified node in the global XYZ system.

St7SetNodeAcceleration3

Sets the global acceleration components at the specified node. These acceleration values are not used as initial conditions when performing transient analysis, rather, they are used to generate body forces when acting on masses.

```
long St7SetNodeAcceleration3(long uID, long NodeNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

`Doubles[0..2]`

A 3-element array describing the acceleration components of the specified node in the global XYZ system.

St7SetNodeResponse

Assigns a response variable to the specified node.

```
long St7SetNodeResponse(long uID, long NodeNum, long CaseNum, long ResponseType,
     long UCSId, long* Status)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

`ResponseType`

Type of response variable; either rvNodeDisplacement or rvNodeReaction.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Status[0..5]`

A 6-element array describing the active DoFs for the response variable in the UCS axis system. Each element may be set to btTrue or btFalse to enable or disable the corresponding DoF.

Node Attributes – Get

St7GetNodeID

Returns the ID number assigned to the specified node.

```
long St7GetNodeID(long uID, long NodeNum, long* NodeID)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

Output Parameters

`NodeID`

ID number for the specified node.

St7GetNodeRestraint6

Returns the restraint conditions assigned to the specified node. The UCS in which these restraints were applied is also returned. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetNodeRestraint6(long uID, long NodeNum, long CaseNum, long* UCSId,
     long* Status, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Freedom case number.

Output Parameters

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Status[0..5]`

A 6-element array describing the restraint conditions for the six DoF at the specified node. `Status[i-1] = btTrue` indicates that the i^{th} DoF is restrained. The DoF are restrained according to the 123456 axis convention in the specified UCS.

`Doubles[0..5]`

A 6-element array describing the enforced displacement conditions for the six DoF at the specified node. `Doubles[i-1]` describes the displacement of the i^{th} DoF according to the 123456 axis convention in the specified UCS.

St7GetNodeForce3

Returns the point force applied to the specified node in the global XYZ system. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetNodeForce3(long uID, long NodeNum, long CaseNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0..2]`

A 3-element array describing the nodal force in the global XYZ system.

St7GetNodeMoment3

Returns the point moment applied at the specified node in the global XYZ system. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetNodeMoment3(long uID, long NodeNum, long CaseNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0..2]`

A 3-element array describing the nodal moment in the global XYZ system.

St7GetNodeTemperature1

Returns the temperature value applied at the specified node. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetNodeTemperature1(long uID, long NodeNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0]`

The nodal temperature value at the specified node.

St7GetNodeTemperatureType1

Returns the temperature type assigned at the specified node. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetNodeTemperatureType1(long uID, long NodeNum, long CaseNum,
     long* TType)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

Output Parameters

`TType`

One of ntReferenceTemperature, ntFixedTemperature, ntInitialTemperature or ntTableTemperature. Use *St7GetNodeTemperatureTable* to return the table.

St7GetNodeTemperatureTable

Returns the table associated with the temperature at the specified node. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetNodeTemperatureTable(long uID, long NodeNum, long CaseNum,
     long* TableID)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

Output Parameters

`TableID`

Temperature vs Time table ID associated with the node temperature, or 0 for none.

St7GetNodeKTranslation3F

Returns the translational stiffness components assigned to the specified node. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetNodeKTranslation3F(long uID, long NodeNum, long CaseNum, long* UCSId,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Freedom case number.

Output Parameters

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Doubles[0..2]`

A 3-element array describing the translational stiffnesses for the specified node. `Doubles[i-1]` describes the stiffness for the i^{th} translational DoF according to the 123 axis definition in the specified UCS.

St7GetNodeKRotation3F

Returns the rotational stiffness components assigned to the specified node. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetNodeKRotation3F(long uID, long NodeNum, long CaseNum, long* UCSId,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Freedom case number.

Output Parameters

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Doubles[0..2]`

A 3-element array describing the rotational stiffnesses for the specified node. `Doubles[i-1]` describes the stiffness for the i^{th} rotational DoF according to the 456 axis definition in the specified UCS.

St7GetNodeTMass3

Returns the translational mass components assigned to the specified node. Translational masses are active in all load and freedom cases.

```
long St7GetNodeTMass3(long uID, long NodeNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

Output Parameters

`Doubles[0..2]`

A 3-element array describing the translational mass for the specified node in the global XYZ system.

St7GetNodeRMass3

Returns the rotational mass components assigned to the specified node. Rotational masses are active in all load and freedom cases.

```
long St7GetNodeRMass3(long uID, long NodeNum, long* UCSId, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

Output Parameters

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Doubles[0..2]`

A 3-element array defining the rotational mass for the node about the axes of the specified UCS.

St7GetNodeNSMass5ID

Returns the non-structural mass assigned to the specified node. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetNodeNSMass5ID(long uID, long NodeNum, long CaseNum, long ID,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

`ID`

The non-structural mass ID.

Output Parameters

`Doubles[0..4]`

`[0]` – The non-structural mass at the specified node.

`[1]` – The dynamic factor at the specified node. This factor is used to scale the non-structural mass when performing dynamic analysis.

`[2..4]` – A 3-element array describing the offset in the global XYZ system.

St7GetNodeKDamping3F

Returns the translational damping coefficients assigned at the specified node. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetNodeKDamping3F(long uID, long NodeNum, long CaseNum, long* UCSId,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Freedom case number.

Output Parameters

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Doubles[0..2]`

A 3-element array describing the damping factors for the specified node. `Doubles[i-1]` describes the damping factor for the i^{th} translational DoF according to the 123 axis definition in the specified UCS.

St7GetNodeHeatSource1

Returns the heat source assigned at the specified node. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetNodeHeatSource1(long uID, long NodeNum, long CaseNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0]`

Heat source.

St7GetNodeHeatSourceTables

Returns the tables associated with the heat source at the specified node. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetNodeHeatSourceTables(long uID, long NodeNum, long CaseNum,
     long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

Output Parameters

`Tables[0..1]`

`[0]` – Factor vs Time table ID associated with the heat source, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the heat source, or 0 for none.

St7GetNodeInitialVelocity3

Returns the initial velocity components assigned at the specified node. These initial conditions are used when performing transient dynamic analysis. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetNodeInitialVelocity3(long uID, long NodeNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0..2]`

A 3-element array describing the initial velocity for the specified node in the global XYZ system.

St7GetNodeAcceleration3

Returns the acceleration components assigned at the specified node. These acceleration values are not used as initial conditions when performing transient analysis, rather, they are used to generate body forces when acting on masses. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetNodeAcceleration3(long uID, long NodeNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0..2]`

A 3-element array describing the acceleration of the specified node in the global XYZ system.

St7GetNodeResponse

Returns the response variable assigned at the specified node. Response variables are only used by the load influence solver. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetNodeResponse(long uID, long NodeNum, long CaseNum, long ResponseType,
     long* UCSId, long* Status)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`CaseNum`

Load case number.

`ResponseType`

Type of response variable; either rvNodeDisplacement or rvNodeReaction.

Output Parameters

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Status[0..5]`

A 6-element array describing the active DoFs for the response variable in the UCS axis system. Each element may be set to btTrue or btFalse to enable or disable the corresponding DoF.

Beam Attributes – Set

St7SetBeamID

Sets the ID number of the specified beam.

```
long St7SetBeamID(long uID, long BeamNum, long BeamID)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamID`

Beam ID.

St7SetBeamReferenceAngle1

Sets the reference angle for the specified beam. This angle controls the local rotation of the beam cross section from the default orientation about the 3-axis of the beam. See *Beam Local Coordinates* for additional information.

```
long St7SetBeamReferenceAngle1(long uID, long BeamNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`Doubles[0]`

The reference angle (degrees) used to align the beam principal axis system; see *Beam Local Coordinates*.

St7SetBeamConnectionUCS

Sets the UCS used in the connection element formulation at the specified beam end. The translational and rotational stiffness components are distributed according to the 123 axis convention in the specified UCS. This attribute is only applicable to beams of connection element type.

```
long St7SetBeamConnectionUCS(long uID, long BeamNum, long BeamEnd, long UCSId)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

St7SetBeamTaper2

Sets the taper properties for the specified beam.

```
long St7SetBeamTaper2(long uID, long BeamNum, long TaperAxis, long TaperType,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`TaperAxis`

The local beam axis to be tapered: axLocalX or axLocalY. See *Beam Local Coordinates* for additional information.

`TaperType`

One of btTop, btSymm or btBottom.

`Doubles[0..1]`

A 2-element array that specifies the taper ratios at either beam end. The dimension of the beam section is scaled by this value to calculate the tapered shape.

St7SetBeamOffset2

Sets the offsets for the specified beam.

```
long St7SetBeamOffset2(long uID, long BeamNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`Doubles[0..1]`

A 2-element array describing the beam offsets in the principal 1-2 axis directions of the beam; see *Beam Local Coordinates*.

St7SetBeamSupport2

Sets the elastic support value for the specified beam.

```
long St7SetBeamSupport2(long uID, long BeamNum, long Direction, long CaseNum,
     long Status, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`Direction`

One of adMinus1, adPlus1, adMinus2 or adPlus2. See *Beam Local Coordinates*.

`CaseNum`

The freedom case number.

`Status`

Compression-only flag; either btTrue or btFalse.

`Doubles[0..1]`

`[0]` – The support stiffness.

`[1]` – The support gap. This parameter is only relevant if the compression-only flag is set to btTrue.

St7SetBeamSectionFactor7

Sets the factors for the specified beam. Each stiffness factor scales the corresponding row and column in the stiffness matrix of the beam, while the mass factor scales the entire mass matrix of the beam. Stiffness factors apply only to linear elastic beams, while the mass factor applies to all beam types.

```
long St7SetBeamSectionFactor7(long uID, long BeamNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`Doubles[0..6]`

`[0]` – Shear stiffness factor plane 1.

`[1]` – Shear stiffness factor plane 2.

`[2]` – Axial stiffness factor.

`[3]` – Bending stiffness factor plane 1.

`[4]` – Bending stiffness factor plane 2.

`[5]` – Torsional stiffness factor.

`[6]` – Mass factor.

St7SetBeamTRelease3

Sets the translational end release conditions at the specified beam.

```
long St7SetBeamTRelease3(long uID, long BeamNum, long BeamEnd, long* Status,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

`Status[0..2]`

`Status[i-1]` – defines the release conditions of the specified beam end in the principal 1-3 axis directions of the beam – one of brReleased, brFixed or brPartial for each direction. See *Beam Local Coordinates.*

`Doubles[0..2]`

A 3-element array containing the partial stiffnesses to be used in the case of partial end release conditions.

St7SetBeamRRelease3

Sets the rotational end release conditions at the specified beam.

```
long St7SetBeamRRelease3(long uID, long BeamNum, long BeamEnd, long* Status,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

`Status[0..2]`

`Status[i-1]` defines the release conditions of the specified beam end in the principal 1-3 axis directions of the beam – one of brReleased, brFixed or brPartial for each direction. See *Beam Local Coordinates.*

`Doubles[0..2]`

A 3-element array containing the partial stiffnesses to be used in the case of partial end release conditions.

St7SetBeamCableFreeLength1

Sets the free cable length for the specified beam. This is the unstressed cable length and is only active for beam of type cable.

```
long St7SetBeamCableFreeLength1(long uID, long BeamNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`Doubles[0]`

Cable free length.

St7SetBeamRadius1

Sets the bend radius of the specified beam. This attribute is only active for beams of type pipe.

```
long St7SetBeamRadius1(long uID, long BeamNum, long BeamDir, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamDir`

The axis of the bend: axPrincipal1 or axPrincipal2. The beam will be bent in the axis direction specified, not about the axis; see *Beam Local Coordinates*.

`Doubles[0]`

The radius of curvature of the bend.

St7SetPipePressure2AF

Sets the internal and external pipe pressure for the specified beam. This attribute is only active for beam of type pipe.

```
long St7SetPipePressure2AF(long uID, long BeamNum, long CaseNum, long Status,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

`Status`

Model a pipe with closed ends: btTrue or btFalse. An additional force component is assigned at the beam ends to account for the pressure acting on a close-ended pipe.

`Doubles[0..1]`

A 2-element array describing the inner and outer radial pressures acting on the element surface respectively.

St7SetPipeTemperature2OT

Sets the internal and external pipe temperatures for the specified beam. This attribute is only active for beams of type pipe.

```
long St7SetPipeTemperature2OT(long uID, long BeamNum, long CaseNum, long Status,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

`Status`

Set the external temperature equal to the nodal temperatures at each end: btTrue or btFalse. In the case of unequal end temperatures the average temperature is used.

`Doubles[0..1]`

A 2-element array describing the inner and outer surface temperatures respectively.

St7SetBeamStringGroup1

Assigns the specified beam to a string group.

```
long St7SetBeamStringGroup1(long uID, long BeamNum, long StringID)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`StringID`

The ID number of the string group.

St7SetBeamPreLoad1

Sets the pre-load conditions for the specified beam.

```
long St7SetBeamPreLoad1(long uID, long BeamNum, long CaseNum, long LoadType,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

`LoadType`

The type of pre-load, plBeamPreTension or plBeamPreStrain.

`Doubles[0]`

The pre-load value.

St7SetBeamTempGradient2

Sets the temperature gradient for the specified beam.

```
long St7SetBeamTempGradient2(long uID, long BeamNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

`Doubles[0..1]`

A 2-element array describing the temperature gradient in the 1 and 2 directions of the beam principal axis system; see *Beam Local Coordinates*.

St7SetBeamPreCurvature2

Sets the pre-curvature for the specified beam.

```
long St7SetBeamPreCurvature2(long uID, long BeamNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

`Doubles[0..1]`

A 2-element array describing the pre-curvature in the 1 and 2 directions of the beam principal axis system; see *Beam Local Coordinates*.

St7SetBeamPointForcePrincipal4ID

Assigns point force data for the specified beam element. The force is defined in the principal axis system of the beam.

```
long St7SetBeamPointForcePrincipal4ID(long uID, long BeamNum, long CaseNum,
     long ID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

`ID`

Point force ID.

`Doubles[0..3]`

`[0..2]` – The force components in the principal axis system of the beam.

`[3]` – The relative length position at which the force is applied; see *Beam Local Coordinates*.

St7SetBeamPointForceGlobal4ID

Assigns point force data for the specified beam element. The force is defined in the global XYZ system.

```
long St7SetBeamPointForceGlobal4ID(long uID, long BeamNum, long CaseNum, long ID,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

`ID`

Point force ID.

`Doubles[0..3]`

`[0..2]` – The force components in the global XYZ system.

`[3]` – The relative length position at which the force is applied; see *Beam Local Coordinates*.

St7SetBeamPointMomentPrincipal4ID

Assigns point moment data for the specified beam element. The moment is defined in the principal axis system of the beam.

```
long St7SetBeamPointMomentPrincipal4ID(long uID, long BeamNum, long CaseNum,
     long ID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

`ID`

Point moment ID.

`Doubles[0..3]`

`[0..2]` – The moment components in the principal axis system of the beam.

`[3]` – The relative length position at which the moment is applied; see *Beam Local Coordinates*.

St7SetBeamPointMomentGlobal4ID

Assigns point moment data for the specified beam element. The moment is defined in the global XYZ system.

```
long St7SetBeamPointMomentGlobal4ID(long uID, long BeamNum, long CaseNum,
     long ID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

`ID`

Point moment ID.

`Doubles[0..3]`

`[0..2]` – The moment components in the global XYZ system.

`[3]` – The relative length position at which the moment is applied; see *Beam Local Coordinates*.

St7SetBeamDistributedForcePrincipal6ID

Assigns distributed load data for the specified beam element. The force is defined in the principal axis system of the beam.

```
long St7SetBeamDistributedForcePrincipal6ID(long uID, long BeamNum, long BeamDir,
     long CaseNum, long DLType, long ID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamDir`

Principal or local axis direction. Principal directions are denoted as 1, 2 or 3; local directions are denoted as either 4 (local x) or 5 (local y); see *Beam Local Coordinates*.

`CaseNum`

Load case number.

`DLType`

One of dlConstant, dlLinear, dlTriangular, dlThreePoint0, dlThreePoint1 or dlTrapezoidal.

`ID`

Distributed load ID.

`Doubles[0..5]`

A 6-element array describing the distributed load. See *Beam Distributed Load Types* for additional information.

St7SetBeamDistributedMomentPrincipal6ID

Assigns distributed moment data for the specified beam element. The moment is defined in the principal axis system of the beam.

```
long St7SetBeamDistributedMomentPrincipal6ID(long uID, long BeamNum,
     long BeamDir, long CaseNum, long DLType, long ID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamDir`

Principal axis direction; one of 1, 2 or 3; see *Beam Local Coordinates*.

`CaseNum`

Load case number.

`DLType`

One of dlConstant, dlLinear, dlTriangular, dlThreePoint0, dlThreePoint1 or dlTrapezoidal.

`ID`

Distributed moment ID.

`Doubles[0..5]`

A 6-element array describing the distributed moment. See *Beam Distributed Load Types* for additional information.

St7SetBeamDistributedForceGlobal6ID

Assigns distributed load data for the specified beam element. The force is defined in the global XYZ system.

```
long St7SetBeamDistributedForceGlobal6ID(long uID, long BeamNum, long BeamDir,
     long ProjectFlag, long CaseNum, long DLType, long ID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamDir`

Global XYZ direction; one of 1, 2 or 3.

`ProjectFlag`

bpNone or bpProjected.

`CaseNum`

Load case number.

`DLType`

One of dlConstant, dlLinear, dlTriangular, dlThreePoint0, dlThreePoint1 or dlTrapezoidal.

`ID`

Distributed load ID.

`Doubles[0..5]`

A 6-element array describing the distributed load. See *Beam Distributed Load Types* for additional information.

St7SetBeamNSMass10ID

Assigns non-structural mass for the specified beam.

```
long St7SetBeamNSMass10ID(long uID, long BeamNum, long CaseNum, long DLType,
     long ID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

`DLType`

One of dlConstant, dlLinear, dlTriangular, dlThreePoint0, dlThreePoint1 or dlTrapezoidal.

`ID`

Mass distribution ID.

`Doubles[0..9]`

`[0..5]` – The distributed mass parameters. See *Beam Distributed Load Types* for additional information.

`[6]` – Dynamic factor.

`[7..9]` – Offset vector in the global XYZ system.

St7SetBeamConvection2

Sets the thermal convection coefficient and ambient temperature for the specified beam. The convection is assumed to occur uniformly over the cross section of the beam.

```
long St7SetBeamConvection2(long uID, long BeamNum, long BeamEnd, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

`CaseNum`

Load case number.

`Doubles[0..1]`

`[0]` – Convection coefficient.

`[1]` – Ambient temperature.

St7SetBeamConvectionTables

Sets the tables associated with the convection coefficient for the specified beam end.

```
long St7SetBeamConvectionTables(long uID, long BeamNum, long BeamEnd,
     long CaseNum, long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

`CaseNum`

Load case number.

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the convection ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the convection coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the convection coefficient, or 0 for none.

St7SetBeamRadiation2

Sets the thermal radiation coefficient and ambient temperature for the specified beam.

```
long St7SetBeamRadiation2(long uID, long BeamNum, long BeamEnd, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

`CaseNum`

Load case number.

`Doubles[0..1]`

`[0]` – Radiation coefficient.

`[1]` – Ambient temperature.

St7SetBeamRadiationTables

Set the tables associated with the radiation coefficient for the specified beam end.

```
long St7SetBeamRadiationTables(long uID, long BeamNum, long BeamEnd,
     long CaseNum, long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

`CaseNum`

Load case number.

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the radiation ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the radiation coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the radiation coefficient, or 0 for none.

St7SetBeamFlux1

Sets the heat flux for the specified beam.

```
long St7SetBeamFlux1(long uID, long BeamNum, long BeamEnd, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

`CaseNum`

Load case number.

`Doubles[0]`

Heat flux through the beam.

St7SetBeamFluxTables

Sets the tables associated with the heat flux for the specified beam end.

```
long St7SetBeamFluxTables(long uID, long BeamNum, long BeamEnd, long CaseNum,
    long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

`CaseNum`

Load case number.

`Tables[0..1]`

`[0]` – Factor vs Time table ID associated with the heat flux, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the heat flux, or 0 for none.

St7SetBeamHeatSource1

Sets the heat source for the specified beam.

```
long St7SetBeamHeatSource1(long uID, long BeamNum, long CaseNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

`Doubles[0]`

Heat source.

St7SetBeamHeatSourceTables

Sets the tables associated with the heat source for the specified beam.

```
long St7SetBeamHeatSourceTables(long uID, long BeamNum, long CaseNum,
     long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

`Tables[0..1]`

`[0]` – Factor vs Time table ID associated with the heat source, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the heat source, or 0 for none.

St7SetBeamResponse

Assigns a response variable to the specified beam. Response variables are only used by the load influence solver.

```
long St7SetBeamResponse(long uID, long BeamNum, long BeamEnd, long CaseNum,
     long* Status)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

`CaseNum`

Load case number.

`Status[0..5]`

`[ipBeamResponseSF1]` – Shear force in the principal 1 axis direction; either btTrue or btFalse.

`[ipBeamResponseSF2]` – Shear force in the principal 2 axis direction; either btTrue or btFalse.

`[ipBeamResponseAxial]` – Axial force; either btTrue or btFalse.

`[ipBeamResponseBM1]` – Bending moment in the principal 1 axis direction; either btTrue or btFalse.

`[ipBeamResponseBM2]` – Bending moment in the principal 2 axis direction; either btTrue or btFalse.

`[ipBeamResponseTorque]` – Torque; either btTrue or btFalse.

St7SetBeamCreepLoadingAge1

Sets the creep loading age for the specified beam. This attribute is only used when performing a creep analysis using the quasi-static or nonlinear transient dynamic solvers.

```
long St7SetBeamCreepLoadingAge1(long uID, long BeamNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`Doubles[0]`

Creep loading age in seconds.

St7SetBeamEndAttachment1

Sets the end attachment properties for the specified beam. Attachment attributes can be used to generate attachment links using the *St7CreateAttachments* function.

```
long St7SetBeamEndAttachment1(long uID, long BeamNum, long BeamEnd,
     long AttachType, long ConnectType, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

`AttachType`

One of alDirect, alRigid or alFlexible.

`ConnectType`

Attachment sub-type; either alMoment or alPinned.

`PropNum`

Beam property number to be used for flexible type connections.

`Doubles[0]`

The maximum distance within which the beam can be attached to another element using the attachment link.

St7SetBeamSideAttachment1

Sets the side attachment properties for the specified beam. Attachment attributes can be used to generate attachment links using the *St7CreateAttachments* function.

```
long St7SetBeamSideAttachment1(long uID, long BeamNum, long BeamEnd,
     long Direction, long AttachType, long ConnectType, long PropNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

`Direction`

One of adPlus1, adMinus1, adPlus2 or adMinus2.

`AttachType`

One of alDirect, alRigid or alFlexible.

`ConnectType`

Attachment sub-type; either alMoment or alPinned.

`PropNum`

Beam property number to be used for flexible type connections.

`Doubles[0]`

The maximum distance within which the beam can be attached to another element using the attachment link.

Beam Attributes – Get

St7GetBeamID

Returns the ID number for the specified beam.

```
long St7GetBeamID(long uID, long BeamNum, long* BeamID)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number

Output Parameters

`BeamID`

Beam ID.

St7GetBeamReferenceAngle1

Returns the reference angle for the specified beam. This angle controls the local rotation of the beam cross section about the 3-axis of the beam as per the beam local axis system definition. See *Beam Local Coordinates* for additional information.

```
long St7GetBeamReferenceAngle1(long uID, long BeamNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

Output Parameters

`Doubles[0]`

The reference angle (degrees) used to align the principal axis system of the beam; see *Beam Local Coordinates*.

St7GetBeamConnectionUCS

Returns the UCS used to define the connection element formulation for the specified beam end. The translational and rotational stiffness components are distributed according to the 123 axis convention in the specified UCS. This attribute is only applicable to beams of connection element type.

```
long St7GetBeamConnectionUCS(long uID, long BeamNum, long BeamEnd, long* UCSId)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

Output Parameters

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

St7GetBeamTaper2

Returns the taper properties for the specified beam.

```
long St7GetBeamTaper2(long uID, long BeamNum, long TaperAxis, long* TaperType,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`TaperAxis`

The local beam axis to be tapered: axLocalX or axLocalY. See *Beam Local Coordinates* for additional information.

Output Parameters

`TaperType`

One of btTop, btSymm or btBottom.

`Doubles[0..1]`

A 2-element array that specifies the taper ratios at either beam end. The dimensions of the beam section are scaled by these values to calculate the tapered shape.

St7GetBeamOffset2

Returns the offsets assigned to the specified beam.

```
long St7GetBeamOffset2(long uID, long BeamNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

Output Parameters

`Doubles[0..1]`

A 2-element array describing the beam offsets in the principal 1-2 axis directions of the beam; see *Beam Local Coordinates*.

St7GetBeamSupport2

Returns the elastic support value assigned to the specified beam. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBeamSupport2(long uID, long BeamNum, long Direction, long CaseNum,
     long* Status, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`Direction`

The support direction in principal axes of the beam. One of adMinus1, adPlus1, adMinus2 or adPlus2. See *Beam Local Coordinates*.

`CaseNum`

The freedom case number.

Output Parameters

`Status`

Compression-only flag; either btTrue or btFalse.

`Doubles[0..1]`

`[0]` – The support stiffness.

`[1]` – The support gap. This parameter is only relevant if the compression-only flag is set to btTrue.

St7GetBeamSectionFactor7

Returns the factors for the specified beam. Each stiffness factor scales the corresponding row and column in the stiffness matrix of the beam, while the mass factor scales the entire mass matrix of the beam. Stiffness factors apply only to linear elastic beams, while the mass factor applies to all beam types.

```
long St7GetBeamSectionFactor7(long uID, long BeamNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

Output Parameters

`Doubles[0..6]`

`[0]` – Shear stiffness factor plane 1.

`[1]` – Shear stiffness factor plane 2.

`[2]` – Axial stiffness factor.

`[3]` – Bending stiffness factor plane 1.

`[4]` – Bending stiffness factor plane 2.

`[5]` – Torsional stiffness factor.

`[6]` – Mass factor.

St7GetBeamTRelease3

Returns the translational end release conditions assigned to the specified beam.

```
long St7GetBeamTRelease3(long uID, long BeamNum, long BeamEnd, long* Status,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

Output Parameters

`Status[0..2]`

`Status[i-1]` describes the release conditions of the specified beam end in the principal 1-3 axis directions of the beam – one of brReleased, brFixed or brPartial for each direction. See *Beam Local Coordinates.*

`Doubles[0..2]`

A 3-element array containing the partial stiffnesses to be used in the case of partial end release conditions.

St7GetBeamRRelease3

Returns the rotational end release conditions assigned to the specified beam.

```
long St7GetBeamRRelease3(long uID, long BeamNum, long BeamEnd, long* Status,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

Output Parameters

`Status[0..2]`

`Status[i-1]` describes the release conditions of the specified beam end in the principal 1-3 axis directions of the beam – one of brReleased, brFixed or brPartial for each direction. See *Beam Local Coordinates.*

`Doubles[0..2]`

A 3-element array containing the partial stiffnesses to be used in the case of partial end release conditions.

St7GetBeamCableFreeLength1

Returns the free cable length for the specified beam. This is the unstressed cable length and is only active for beam of type cable.

```
long St7GetBeamCableFreeLength1(long uID, long BeamNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

Output Parameters

`Doubles[0]`

The free cable length.

St7GetBeamRadius1

Returns the bend radius for the specified beam. This attribute is only active for beams of type pipe.

```
long St7GetBeamRadius1(long uID, long BeamNum, long* BeamDir, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

Output Parameters

`BeamDir`

The axis of the bend: axPrincipal1 or axPrincipal2. The beam will be bent in the axis direction specified, not about the axis; see *Beam Local Coordinates*.

`Doubles[0]`

The radius of curvature of the bend.

St7GetPipePressure2AF

Returns the internal and external pressures applied to the specified beam. This attribute is only active for beams of type pipe. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPipePressure2AF(long uID, long BeamNum, long CaseNum, long* Status,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

Output Parameters

`Status`

Model a pipe with closed ends: btTrue or btFalse. An additional force component is assigned at the beam ends to account for the pressure acting on a close-ended pipe.

`Doubles[0..1]`

A 2-element array describing the inner and outer radial pressures acting on the element surface respectively.

St7GetPipeTemperature2OT

Returns the internal and external temperatures applied to the specified beam. This attribute of only active for beams of type pipe. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPipeTemperature2OT(long uID, long BeamNum, long CaseNum, long* Status,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

Output Parameters

`Status`

Set the external temperature equal to the nodal temperatures at each end: btTrue or btFalse. In the case of unequal end temperatures the average temperature is used.

`Doubles[0..1]`

A 2-element array describing the inner and outer surface temperatures respectively.

St7GetBeamStringGroup1

Returns the ID number of the string group the specified beam is assigned to. The string group attribute is only active for truss elements and will ensure that the axial force in all members is equal.

```
long St7GetBeamStringGroup1(long uID, long BeamNum, long* StringID)
```

Input Parameters

`uID`

Strand7 model file ID.

BeamNum

Beam number.

Output Parameters

StringID

The ID number of the string group.

St7GetBeamPreLoad1

Returns the pre-load assigned to the specified beam. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBeamPreLoad1(long uID, long BeamNum, long CaseNum, long* LoadType,
     double* Doubles)
```

Input Parameters

uID

Strand7 model file ID.

BeamNum

Beam number.

CaseNum

Load case number.

Output Parameters

LoadType

Either plBeamPreTension or plBeamPreStrain.

Doubles[0]

Pre-load value.

St7GetBeamTempGradient2

Returns the temperature gradients assigned to the specified beam. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBeamTempGradient2(long uID, long BeamNum, long CaseNum,
     double* Doubles)
```

Input Parameters

uID

Strand7 model file ID.

BeamNum

Beam number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0..1]`

A 2-element array describing the temperature gradient in the 1 and 2 directions of the principal axis system of the beam; see *Beam Local Coordinates*.

St7GetBeamPreCurvature2

Returns the pre-curvature assigned to the specified beam. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBeamPreCurvature2(long uID, long BeamNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0..1]`

A 2-element array describing the pre-curvature in the 1 and 2 directions of the principal axis system of the beam; see *Beam Local Coordinates*.

St7GetBeamPointForcePrincipal4ID

Returns point force data assigned to the specified beam element. The force is applied according to the principal axis system of the beam. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBeamPointForcePrincipal4ID(long uID, long BeamNum, long CaseNum,
     long ID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

`ID`

Point force ID.

Output Parameters

`Doubles[0..3]`

`[0..2]` – The force components in the principal axis system of the beam.

`[3]` – The relative length position at which the force is applied; see *Beam Local Coordinates*.

St7GetBeamPointForceGlobal4ID

Returns point force data assigned to the specified beam element. The force components are defined in the global XYZ system. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBeamPointForceGlobal4ID(long uID, long BeamNum, long CaseNum, long ID,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

`ID`

Point force ID.

Output Parameters

`Doubles[0..3]`

`[0..2]` – The force components in the global XYZ system.

`[3]` – The relative length position at which the force is applied; see *Beam Local Coordinates*.

St7GetBeamPointMomentPrincipal4ID

Returns point moment data assigned to the specified beam element. The moment is applied according to the principal axis system of the beam. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBeamPointMomentPrincipal4ID(long uID, long BeamNum, long CaseNum,
     long ID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

`ID`

Point moment ID.

Output Parameters

`Doubles[0..3]`

`[0..2]` – The moment components in the principal axis system of the beam.

`[3]` – The relative length position at which the moment is applied; see *Beam Local Coordinates*.

St7GetBeamPointMomentGlobal4ID

Returns point moment data assigned to the specified beam element. The moment components are defined in the global XYZ system. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBeamPointMomentGlobal4ID(long uID, long BeamNum, long CaseNum,
     long ID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

`ID`

Point moment ID.

Output Parameters

`Doubles[0..3]`

`[0..2]` – The moment components in the global XYZ system.

`[3]` – The relative length position at which the moment is applied; see *Beam Local Coordinates*.

St7GetBeamDistributedForcePrincipal6ID

Returns distributed load data assigned to the specified beam element. The force is applied according to the principal axis system of the beam. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBeamDistributedForcePrincipal6ID(long uID, long BeamNum, long BeamDir,
     long CaseNum, long ID, long* DLType, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamDir`

Principal or local axis direction. Principal directions are denoted as 1, 2 or 3; local directions are denoted as either 4 (local x) or 5 (local y); see *Beam Local Coordinates*.

`CaseNum`

Load case number.

`ID`

Distributed load ID.

Output Parameters

`DLType`

One of dlConstant, dlLinear, dlTriangular, dlThreePoint0, dlThreePoint1 or dlTrapezoidal.

`Doubles[0..5]`

A 6-element array describing the distributed load. See *Beam Distributed Load Types* for additional information.

St7GetBeamDistributedMomentPrincipal6ID

Returns distributed moment data assigned to the specified beam element. The moment is applied according to the principal axis system of the beam. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBeamDistributedMomentPrincipal6ID(long uID, long BeamNum,
     long BeamDir, long CaseNum, long ID, long* DLType, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamDir`

Principal axis direction; one of 1, 2 or 3; see *Beam Local Coordinates*.

`CaseNum`

Load case number.

`ID`

Distributed moment ID.

Output Parameters

`DLType`

One of dlConstant, dlLinear, dlTriangular, dlThreePoint0, dlThreePoint1 or dlTrapezoidal.

`Doubles[0..5]`

A 6-element array describing the distributed moment. See *Beam Distributed Load Types* for additional information.

St7GetBeamDistributedForceGlobal6ID

Returns distributed load data assigned to the specified beam element. The force components are defined in the global XYZ system. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBeamDistributedForceGlobal6ID(long uID, long BeamNum, long BeamDir,
    long CaseNum, long ID, long* ProjectFlag, long* DLType, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamDir`

Global XYZ direction; one of 1, 2 or 3.

`CaseNum`

Load case number.

`ID`

Distributed load ID.

Output Parameters

`ProjectFlag`

bpNone or bpProjected.

`DLType`

One of dlConstant, dlLinear, dlTriangular, dlThreePoint0, dlThreePoint1 or dlTrapezoidal.

`Doubles[0..5]`

A 6-element array describing the distributed load. See *Beam Distributed Load Types* for additional information.

St7GetBeamNSMass10ID

Returns non-structural mass assigned to the specified beam element. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBeamNSMass10ID(long uID, long BeamNum, long CaseNum, long ID,
     long* DLType, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

`ID`

Mass distribution ID.

Output Parameters

`DLType`

One of dlConstant, dlLinear, dlTriangular, dlThreePoint0, dlThreePoint1 or dlTrapezoidal.

`Doubles[0..9]`

`[0..5]` – The distributed mass parameters. See *Beam Distributed Load Types* for additional information.

`[6]` – Dynamic factor.

`[7..9]` – Offset vector in the global XYZ system.

St7GetBeamConvection2

Returns the thermal convection coefficient and ambient temperature assigned to the specified beam. The convection is assumed to occur uniformly over the beam cross section. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBeamConvection2(long uID, long BeamNum, long BeamEnd, long CaseNum,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0..1]`

`[0]` – Convection coefficient.

`[1]` – Ambient temperature.

St7GetBeamConvectionTables

Returns the tables associated with convection coefficient for the specified beam end. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBeamConvectionTables(long uID, long BeamNum, long BeamEnd,
    long CaseNum, long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

`CaseNum`

Load case number.

Output Parameters

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the convection ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the convection coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the convection coefficient, or 0 for none.

St7GetBeamRadiation2

Returns the thermal radiation coefficient and ambient temperature assigned to the specified beam. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBeamRadiation2(long uID, long BeamNum, long BeamEnd, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0..1]`

`[0]` – Radiation coefficient.

`[1]` – Ambient temperature.

St7GetBeamRadiationTables

Returns the tables associated with the radiation coefficient for the specified beam end. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBeamRadiationTables(long uID, long BeamNum, long BeamEnd,
     long CaseNum, long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

`CaseNum`

Load case number.

Output Parameters

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the radiation ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the radiation coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the radiation coefficient, or 0 for none.

St7GetBeamFlux1

Returns the heat flux assigned to the specified beam. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBeamFlux1(long uID, long BeamNum, long BeamEnd, long CaseNum,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0]`

The heat flux through the beam.

St7GetBeamFluxTables

Returns the tables associated with the heat flux for the specified beam end. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBeamFluxTables(long uID, long BeamNum, long BeamEnd, long CaseNum,
    long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

`CaseNum`

Load case number.

Output Parameters

`Tables[0..1]`

`[0]` – Factor vs Time table ID associated with the heat flux, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the heat flux, or 0 for none.

St7GetBeamHeatSource1

Returns the heat source assigned to the specified beam. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBeamHeatSource1(long uID, long BeamNum, long CaseNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0]`

Heat source.

St7GetBeamHeatSourceTables

Returns the tables associated with the heat source for the specified beam. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBeamHeatSourceTables(long uID, long BeamNum, long CaseNum,
     long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`CaseNum`

Load case number.

Output Parameters

`Tables[0..1]`

`[0]` – Factor vs Time table ID associated with the heat source, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the heat source, or 0 for none.

St7GetBeamResponse

Returns the response variable assigned for the specified beam. Response variables are only used by the load influence solver. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBeamResponse(long uID, long BeamNum, long BeamEnd, long CaseNum,
    long* Status)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

`CaseNum`

Load case number.

Output Parameters

`Status[0..5]`

`[ipBeamResponseSF1]` – Shear force in the principal 1 axis direction; either btTrue or btFalse.

`[ipBeamResponseSF2]` – Shear force in the principal 2 axis direction; either btTrue or btFalse.

`[ipBeamResponseAxial]` – Axial force; either btTrue or btFalse.

`[ipBeamResponseBM1]` – Bending moment in the principal 1 axis direction; either btTrue or btFalse.

`[ipBeamResponseBM2]` – Bending moment in the principal 2 axis direction; either btTrue or btFalse.

`[ipBeamResponseTorque]` – Torque; either btTrue or btFalse.

St7GetBeamCreepLoadingAge1

Returns the creep loading age for the specified beam. This attribute is only used when performing a creep analysis using the quasi-static or nonlinear transient dynamic solvers.

```
long St7GetBeamCreepLoadingAge1(long uID, long BeamNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

Output Parameters

`Doubles[0]`

Creep loading age in seconds.

St7GetBeamEndAttachment1

Returns the end attachment properties for the specified beam. This attribute can be used to generate attachment links using the *St7CreateAttachments* function.

```
long St7GetBeamEndAttachment1(long uID, long BeamNum, long BeamEnd,
     long* AttachType, long* ConnectType, long* PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`BeamEnd`

Beam end; either 1 or 2.

Output Parameters

`AttachType`

One of alDirect, alRigid or alFlexible.

`ConnectType`

Either alMoment or alPinned.

`PropNum`

Beam property number to be used for flexible type connections.

Doubles[0]

The maximum distance within which the beam can be attached to another element using the attachment link.

St7GetBeamSideAttachment1

Returns the side attachment properties for the specified beam. This attribute can be used to generate attachment links using the *St7CreateAttachments* function.

```
long St7GetBeamSideAttachment1(long uID, long BeamNum, long BeamEnd,
     long Direction, long* AttachType, long* ConnectType, long* PropNum,
     double* Doubles)
```

Input Parameters

uID

Strand7 model file ID.

BeamNum

Beam number.

BeamEnd

Beam end; either 1 or 2.

Direction

One of adPlus1, adMinus1, adPlus2 or adMinus2.

Output Parameters

AttachType

One of alDirect, alRigid or alFlexible.

ConnectType

Attachment sub-type; either alMoment or alPinned.

PropNum

Beam property number to be used for flexible type connections.

Doubles[0]

The maximum distance within which the beam can be attached to another element using the attachment link.

Plate Attributes – Set

St7SetPlateID

Sets the ID number for the specified plate.

```
long St7SetPlateID(long uID, long PlateNum, long PlateID)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`PlateID`

Plate ID.

St7SetPlateXAngle1

Sets the local axis angle for the specified plate. This angle controls the rotation of the plate local xy axes about the local z axis.

```
long St7SetPlateXAngle1(long uID, long PlateNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`Doubles[0]`

The angle describing the rotation of the plate local axis system about the local z axis. See *Plate Local Coordinates* for additional information.

St7SetPlateThickness2

Sets the thickness attribute of the specified plate element, overriding the plate property thickness – see *St7SetPlateThickness* to set the plate property thickness.

```
long St7SetPlateThickness2(long uID, long PlateNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`Doubles[0..1]`

`[0]` – Membrane thickness of the plate.

`[1]` – Bending thickness of the plate.

St7SetPlateOffset1

Sets the offset for the specified plate element. The offset is applied according to the plate local z axis direction and is uniform over the element surface.

```
long St7SetPlateOffset1(long uID, long PlateNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`Doubles[0]`

Plate offset in the local z axis direction.

St7SetPlateEdgeSupport4

Sets the elastic edge support value for the specified plate.

```
long St7SetPlateEdgeSupport4(long uID, long PlateNum, long EdgeNum, long CaseNum,
    long* Status, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`EdgeNum`

Edge number; one of 1, 2, 3 or 4.

`CaseNum`

Freedom case number.

`Status[0..1]`

`[0]` – Compression-only support; either btTrue or btFalse.

`[1]` – Limited bearing capacity; either btTrue or btFalse.

`Doubles[0..3]`

`[0]` – Elastic support value in the normal direction.

`[1]` – Elastic support value in the lateral direction.

`[2]` – Support gap. Only relevant if Status[0] is set to btTrue.

`[3]` – Bearing capacity. Only relevant if Status[1] is set to btTrue.

St7SetPlateFaceSupport4

Sets the elastic face support value for the specified plate.

```
long St7SetPlateFaceSupport4(long uID, long PlateNum, long Surface, long CaseNum,
    long* Status, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`Surface`

Plate surface; either psPlateMinusZ or psPlatePlusZ.

`CaseNum`

Freedom case number.

`Status[0..1]`

`[0]` – Compression-only support; either btTrue or btFalse.

`[1]` – Limited bearing capacity; either btTrue or btFalse.

`Doubles[0..3]`

`[0]` – Elastic support value in the normal direction.

`[1]` – Elastic support value in the lateral direction.

`[2]` – Support gap. Only relevant if Status[0] is set to btTrue.

`[3]` – Bearing capacity. Only relevant if Status[1] is set to btTrue.

St7SetPlateEdgeRelease1

Sets the edge release conditions for the specified plate.

```
long St7SetPlateEdgeRelease1(long uID, long PlateNum, long EdgeNum, long* Status)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`EdgeNum`

Edge number; one of 1, 2, 3 or 4.

Output Parameters

`Status[0]`

Release status; either prReleased or prFixed.

St7SetPlateSectionFactor10

Sets the factors for the specified plate. Each stiffness factor scales the corresponding row and column in the stiffness matrix of the plate, while the mass factor scales the entire mass matrix of the plate. Stiffness factors apply only to linear elastic plates, while the mass factor applies to all plate types.

```
long St7SetPlateSectionFactor10(long uID, long PlateNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`Doubles[0..9]`

`[0]` – Membrane stiffness factor Cxx.

`[1]` – Membrane stiffness factor Cyy.

`[2]` – Membrane shear stiffness factor Cgg.

`[3]` – Normal stiffness factor Czz for plane strain and axisymmetric plates.

`[4]` – Bending stiffness factor Dxx.

`[5]` – Bending stiffness factor Dyy.

`[6]` – Twisting stiffness factor Dgg.

`[7]` – Transverse shear stiffness factor Gxx.

`[8]` – Transverse shear stiffness factor Gyy.

`[9]` – Mass factor.

St7SetPlatePreLoad3

Sets the pre-load conditions for the specified plate.

```
long St7SetPlatePreLoad3(long uID, long PlateNum, long CaseNum, long LoadType,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`LoadType`

Pre-load type; either plPlatePreStrain or plPlatePreStress.

`Doubles[0..2]`

A 3-element array describing the pre-load condition. `Doubles[i-1]` describes the pre-load in the i^{th} local axis direction.

St7SetPlatePreCurvature2

Sets the pre-curvature conditions for the specified plate.

```
long St7SetPlatePreCurvature2(long uID, long PlateNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`Doubles[0..1]`

A 2-element array describing the pre-curvature in the local x and y directions, respectively.

St7SetPlateTempGradient1

Sets the temperature gradient for the specified plate. The temperature gradient acts according to the plate local z axis direction and is constant over the element surface. This attribute is only active for static and dynamic structural analysis.

```
long St7SetPlateTempGradient1(long uID, long PlateNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`Doubles[0]`

Temperature gradient in the local z axis direction.

St7SetPlatePointForce6

Assigns a point force to the specified plate.

```
long St7SetPlatePointForce6(long uID, long PlateNum, long CaseNum, long Position,
     long ID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`Position`

Position identifier; either axUCS or axLocal.

`ID`

Local ID number for the point force.

`Doubles[0..5]`

`[0..2]` – Components of the applied force in the global XYZ system.

Where Position is axUCS,
`[3..5]` – XYZ position of the point force in global XYZ coordinates.

Where Position is axLocal,
`[3..4]` – uv position of the point force in local element coordinates.

St7SetPlatePointMoment6

Assigns a point moment to the specified plate.

```
long St7SetPlatePointMoment6(long uID, long PlateNum, long CaseNum,
     long Position, long ID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`Position`

Position identifier; either axUCS or axLocal.

`ID`

Local ID number for the point moment.

`Doubles[0..5]`

`[0..2]` – Components of the applied moment in the global XYZ system.

Where Position is axUCS:
`[3..5]` – XYZ position of the point moment in global XYZ coordinates.

Where Position is axLocal:
`[3..4]` – uv position of point moment in the local element coordinate system.

St7SetPlateEdgePressure1

Assigns a normal pressure to the specified plate edge. The pressure is applied in the plane of the element, perpendicular to the plate edge.

```
long St7SetPlateEdgePressure1(long uID, long PlateNum, long CaseNum,
     long EdgeNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`EdgeNum`

Edge number; one of 1, 2, 3 or 4.

`Doubles[0]`

Edge pressure for the specified plate edge, with positive pressures directed away from the plate.

St7SetPlateEdgeShear1

Assigns a shear stress to the specified plate edge. The shear stress is applied tangential to the plate edge.

```
long St7SetPlateEdgeShear1(long uID, long PlateNum, long CaseNum, long EdgeNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`EdgeNum`

Edge number; one of 1, 2, 3 or 4.

`Doubles[0]`

Shear stress.

St7SetPlateEdgeTransverseShear1

Assigns a transverse shear stress to the specified plate edge. The shear stress acts normal to the plate surface at its edge, in the local +z direction.

```
long St7SetPlateEdgeTransverseShear1(long uID, long PlateNum, long CaseNum,
     long EdgeNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`EdgeNum`

Edge number; one of 1, 2, 3 or 4.

`Doubles[0]`

Transverse shear stress.

St7SetPlateEdgePressure3

Assigns a constant pressure to the specified plate edge. The pressure is defined in the global XYZ system.

```
long St7SetPlateEdgePressure3(long uID, long PlateNum, long CaseNum,
     long EdgeNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`EdgeNum`

Edge number; one of 1, 2, 3 or 4.

`Doubles[0..2]`

A 3-element array containing the pressure components in the global XYZ system.

St7SetPlateNormalPressure2

Assigns constant normal pressures to the outer +z and -z surfaces of the specified plate. The pressure is specified as a positive pressure on each face, directed inwards towards the mid-plane of the plate.

```
long St7SetPlateNormalPressure2(long uID, long PlateNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`Doubles[0..1]`

`[0]` – Normal face pressure over the -z surface of the plate.

`[1]` – Normal face pressure over the +z surface of the plate.

Usage

Positive pressure is directed from the relevant surface onto the plate.

St7SetPlateGlobalPressure3S

Assigns a constant face pressure to the specified plate surface. The pressure is defined in the global XYZ system.

```
long St7SetPlateGlobalPressure3S(long uID, long PlateNum, long Surface,
    long ProjectFlag, long CaseNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`Surface`

Plate surface; either psPlateMinusZ or psPlatePlusZ.

`ProjectFlag`

One of ppNone, ppProjResultant or ppProjComponents.

`CaseNum`

Load case number.

`Doubles[0..2]`

A 3-element array containing the pressure components in the global XYZ system.

St7SetPlateShear2

Assigns a face shear stress to the specified plate. The shear stress is applied in the plane of the element.

```
long St7SetPlateShear2(long uID, long PlateNum, long CaseNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`Doubles[0..1]`

A 2-element array that describes the applied shear stress according to the local plate xy axis system.

St7SetPlateNSMass5ID

Sets the non-structural mass for the specified plate.

```
long St7SetPlateNSMass5ID(long uID, long PlateNum, long CaseNum, long ID,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`ID`

Non-structural mass ID.

`Doubles[0..5]`

`[0]` – Non-structural mass for the specified plate.

`[1]` – Dynamic factor for the specified plate. This factor is used to scale the non-structural mass when performing dynamic analyses.

`[2..5]` – A 3-element array describing the offset in the global XYZ system.

St7SetPlateEdgeConvection2

Sets the thermal convection coefficient and ambient temperature for the specified plate edge. This attribute is only used when performing heat transfer analysis.

```
long St7SetPlateEdgeConvection2(long uID, long PlateNum, long CaseNum,
    long EdgeNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`EdgeNum`

Local edge number; one of 1, 2, 3 or 4.

`Doubles[0..1]`

`[0]` – Edge convection coefficient.

`[1]` – Ambient temperature.

St7SetPlateEdgeConvectionTables

Set the tables associated with convection coefficient for the specified plate edge.

```
long St7SetPlateEdgeConvectionTables(long uID, long PlateNum, long CaseNum,
    long EdgeNum, long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`EdgeNum`

Local edge number; one of 1, 2, 3 or 4.

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the convection ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the convection coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the convection coefficient, or 0 for none.

St7SetPlateEdgeRadiation2

Sets the thermal radiation coefficient and ambient temperature for the specified plate edge.

```
long St7SetPlateEdgeRadiation2(long uID, long PlateNum, long CaseNum,
     long EdgeNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`EdgeNum`

Local edge number; one of 1, 2, 3 or 4.

`Doubles[0..1]`

`[0]` – Radiation coefficient.

`[1]` – Ambient temperature.

St7SetPlateEdgeRadiationTables

Sets the tables associated with the radiation coefficient for the specified plate edge.

```
long St7SetPlateEdgeRadiationTables(long uID, long PlateNum, long CaseNum,
     long EdgeNum, long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`EdgeNum`

Local edge number; one of 1, 2, 3 or 4.

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the radiation ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the radiation coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the radiation coefficient, or 0 for none.

St7SetPlateFlux1

Sets the heat flux for the specified plate edge.

```
long St7SetPlateFlux1(long uID, long PlateNum, long CaseNum, long EdgeNum,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`EdgeNum`

Local edge number; one of 1, 2, 3 or 4.

`Doubles[0]`

Heat flux through the plate edge.

St7SetPlateFluxTables

Sets the tables associated with the heat flux for the specified plate edge.

```
long St7SetPlateFluxTables(long uID, long PlateNum, long CaseNum, long EdgeNum,
    long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`EdgeNum`

Local edge number; one of 1, 2, 3 or 4.

`Tables[0..1]`

`[0]` – Factor vs Time table ID associated with the heat flux, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the heat flux, or 0 for none.

St7SetPlateFaceConvection2

Sets the thermal convection coefficient and ambient temperature for the specified plate surface. This attribute is only used when performing heat transfer analysis.

```
long St7SetPlateFaceConvection2(long uID, long PlateNum, long CaseNum,
     long Surface, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`Surface`

Local plate surface; either psPlateMinusZ or psPlatePlusZ.

`Doubles[0..1]`

`[0]` – Convection coefficient.

`[1]` – Ambient temperature.

St7SetPlateFaceConvectionTables

Sets the tables associated with convection coefficient for the specified plate surface.

```
long St7SetPlateFaceConvectionTables(long uID, long PlateNum, long CaseNum,
     long Surface, long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`Surface`

Local plate surface number; either psPlateMinusZ or psPlatePlusZ.

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the convection ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the convection coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the convection coefficient, or 0 for none.

St7SetPlateFaceRadiation2

Sets the thermal radiation coefficient and ambient temperature for the specified plate surface.

```
long St7SetPlateFaceRadiation2(long uID, long PlateNum, long CaseNum,
    long Surface, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`Surface`

Local plate surface; either psPlateMinusZ or psPlatePlusZ.

`Doubles[0..1]`

`[0]` – Radiation coefficient.

`[1]` – Ambient temperature.

St7SetPlateFaceRadiationTables

Sets the tables associated with the radiation coefficient for the specified plate surface.

```
long St7SetPlateFaceRadiationTables(long uID, long PlateNum, long CaseNum,
    long Surface, long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`Surface`

Local plate surface; either psPlateMinusZ or psPlatePlusZ.

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the radiation ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the radiation coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the radiation coefficient, or 0 for none.

St7SetPlateHeatSource1

Sets the heat source for the specified plate.

```
long St7SetPlateHeatSource1(long uID, long PlateNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`Doubles[0]`

Heat source.

St7SetPlateHeatSourceTables

Sets the tables associated with the heat source for the specified plate.

```
long St7SetPlateHeatSourceTables(long uID, long PlateNum, long CaseNum,
     long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`Tables[0..1]`

`[0]` – Factor vs Time table ID associated with the heat source, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the heat source, or 0 for none.

St7SetPlateSoilStress2

Sets the in-situ soil stress for the specified plate. This attribute is only active for plates of property type soil.

```
long St7SetPlateSoilStress2(long uID, long PlateNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`Doubles[0..1]`

A 2-element array containing the initial vertical stress and the horizontal stress ratio.

St7SetPlateSoilRatio2

Sets the in-situ soil ratios for the specified plate. This attribute is only active for plates of property type soil.

```
long St7SetPlateSoilRatio2(long uID, long PlateNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`Doubles[0..1]`

A 2-element array containing the overconsolidation ratio and the initial void ratio.

St7SetPlateResponse

Assigns a response variable to the specified plate. Response variables are only used by the load influence solver.

```
long St7SetPlateResponse(long uID, long PlateNum, long CaseNum,
     long ResponseType, long UCSId, long* Status)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`ResponseType`

Response variable type; either rvPlateForce or rvPlateMoment.

`UCSId`

0 to use the plate local axis system, or the ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Status[0..5]`

A 6-element array describing the force/moment components that are flagged as response variables according to the 123 axis convention in the specified UCS – [11, 22, 33, 12, 23, 31].

St7SetPlateLoadPatch4

Sets the load patch type for the specified plate. This attribute is only active for plates of property type load patch.

```
long St7SetPlateLoadPatch4(long uID, long PlateNum, long PatchType,
     long EdgeBits, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`PatchType`

One of ptAuto4, ptAuto3, ptAuto2, ptAuto1, ptAngleSplit or ptManual.

`EdgeBits`

A 32-bit value in which the four least significant bits specify the selection of up to four edges. See *Load Patch Types* for additional information.

`Doubles[0..3]`

Edge weights. See *Load Patch Types* for additional information.

St7SetPlateReinforcement2

Assigns the concrete reinforcement properties for the specified plate.

```
long St7SetPlateReinforcement2(long uID, long PlateNum, long LayoutID,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`LayoutID`

Reinforcement layout ID.

`Doubles[0..1]`

A 2-element array describing the angular orientation (degrees) of the 1-3 and 2-4 reinforcement layers respectively.

St7SetPlateCreepLoadingAge1

Sets the creep loading age for the specified plate. This attribute is only active when conducting creep analysis using the quasi-static or nonlinear transient dynamic solvers.

```
long St7SetPlateCreepLoadingAge1(long uID, long PlateNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`Doubles[0]`

Creep loading age in seconds.

St7SetPlateEdgeAttachment1

Assigns an edge attachment to the specified plate edge. Attachment attributes can be used to generate attachment links using the *St7CreateAttachments* function.

```
long St7SetPlateEdgeAttachment1(long uID, long PlateNum, long EdgeNum,
     long Direction, long AttachType, long ConnectType, long PropNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`EdgeNum`

Local edge number; one of 1, 2, 3 or 4.

`Direction`

Direction of attachment; one of adPlanar, adPlusZ or adMinusZ.

`AttachType`

One of alDirect, alRigid or alFlexible.

`ConnectType`

Attachment sub-type; either alMoment or alPinned.

`PropNum`

Beam property number used for flexible attachment types.

`Doubles[0]`

The maximum distance within which the plate edge can be connected to another element using the attachment link.

St7SetPlateFaceAttachment1

Assigns a face attachment to the specified plate surface. Attachment attributes can be used to generate attachment links using the *St7CreateAttachments* function.

```
long St7SetPlateFaceAttachment1(long uID, long PlateNum, long Surface,
     long AttachType, long ConnectType, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`Surface`

Local plate surface; either psPlateMinusZ or psPlatePlusZ.

`AttachType`

One of alDirect, alRigid or alFlexible.

`ConnectType`

Attachment sub-type; either alMoment or alPinned.

`PropNum`

Beam property number used for flexible attachment types.

`Doubles[0]`

The maximum distance within which the plate surface can be connected to another element using the attachment link.

St7SetPlateCavityFluid

Assigns a cavity fluid layout to a surface of the specified plate.

```
long St7SetPlateCavityFluid(long uID, long PlateNum, long Surface, long CavityID)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`Surface`

Local plate surface; either psPlateMinusZ or psPlatePlusZ.

`CavityID`

Cavity fluid layout ID.

Plate Attributes – Get

St7GetPlateID

Returns the ID number for the specified plate.

```
long St7GetPlateID(long uID, long PlateNum, long* PlateID)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

Output Parameters

`PlateID`

Plate ID.

St7GetPlateXAngle1

Returns the local axis angle for the specified plate. This angle controls the rotations of the plate local xy axes about the local z axis.

```
long St7GetPlateXAngle1(long uID, long PlateNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

Output Parameters

`Doubles[0]`

The angle describing the rotation of the plate local axis system about the local z axis. See *Plate Local Coordinates* for additional information.

St7GetPlateThickness2

Returns the thickness attribute of the specified plate, if the thickness attribute is set – see *St7GetPlateThickness* to get the default plate property thickness.

```
long St7GetPlateThickness2(long uID, long PlateNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

Output Parameters

`Doubles[0..1]`

`[0]` – Plate membrane thickness.

`[1]` – Plate bending thickness.

St7GetPlateOffset1

Returns the offset for the specified plate. The offset is applied according to the plate local z axis direction and is uniform over the element surface.

```
long St7GetPlateOffset1(long uID, long PlateNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

Output Parameters

`Doubles[0]`

Plate offset in the local z axis direction.

St7GetPlateEdgeSupport4

Returns the elastic support applied at the specified plate edge. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateEdgeSupport4(long uID, long PlateNum, long EdgeNum, long CaseNum,
    long* Status, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`EdgeNum`

Edge number; one of 1, 2, 3 or 4.

`CaseNum`

Freedom case number.

Output Parameters

`Status[0..1]`

`[0]` – Compression-only support; either btTrue or btFalse.

`[1]` – Limited bearing capacity; either btTrue or btFalse.

`Doubles[0..3]`

`[0]` – Elastic support value in the normal direction.

`[1]` – Elastic support value in the lateral direction.

`[2]` – Support gap. Only relevant if Status[0] is set to btTrue.

`[3]` – Bearing capacity. Only relevant if Status[1] is set to btTrue.

St7GetPlateFaceSupport4

Returns the elastic support applied to the specified plate surface. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateFaceSupport4(long uID, long PlateNum, long Surface, long CaseNum,
     long* Status, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`Surface`

Plate surface; either psPlateMinusZ or psPlatePlusZ.

`CaseNum`

Freedom case number.

Output Parameters

`Status[0..1]`

`[0]` – Compression-only support; either btTrue or btFalse.

`[1]` – Limited bearing capacity; either btTrue or btFalse.

`Doubles[0..3]`

`[0]` – Elastic support value in the normal direction.

`[1]` – Elastic support value in the lateral direction.

`[2]` – Support gap. Only relevant if Status[0] is set to btTrue.

`[3]` – Bearing capacity. Only relevant if Status[1] is set to btTrue.

St7GetPlateSectionFactor10

Returns the factors for the specified plate. Each stiffness factor scales the corresponding row and column in the stiffness matrix of the plate, while the mass factor scales the entire mass matrix of the plate. Stiffness factors apply only to linear elastic plates, while the mass factor applies to all plate types.

```
long St7GetPlateSectionFactor10(long uID, long PlateNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

Output Parameters

`Doubles[0..9]`

`[0]` – Membrane stiffness factor Cxx.

`[1]` – Membrane stiffness factor Cyy.

`[2]` – Membrane shear stiffness factor Cgg.

`[3]` – Normal stiffness factor Czz for plane strain and axisymmetric plates.

`[4]` – Bending stiffness factor Dxx.

`[5]` – Bending stiffness factor Dyy.

`[6]` – Twisting stiffness factor Dgg.

`[7]` – Transverse shear stiffness factor Gxx.

`[8]` – Transverse shear stiffness factor Gyy.

`[9]` – Mass factor.

St7GetPlateEdgeRelease1

Returns the edge release condition for the specified plate edge.

```
long St7GetPlateEdgeRelease1(long uID, long PlateNum, long EdgeNum, long* Status)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`EdgeNum`

Local edge number; one of 1, 2, 3 or 4.

Output Parameters

`Status[0]`

Release status; either prReleased or prFixed.

St7GetPlatePreLoad3

Returns the pre-load conditions for the specified plate. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlatePreLoad3(long uID, long PlateNum, long CaseNum, long* LoadType,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

Output Parameters

`LoadType`

Pre-load type; either plPlatePreStrain or plPlatePreStress.

`Doubles[0..2]`

A 3-element array describing the pre-load condition. `Doubles[i-1]` describes the pre-load in the i^{th} local axis direction.

St7GetPlatePreCurvature2

Returns the pre-curvature conditions for the specified plate. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlatePreCurvature2(long uID, long PlateNum, long CaseNum,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0..1]`

A 2-element array describing the pre-curvature in the local x and y directions, respectively.

St7GetPlateTempGradient1

Returns the temperature gradient for the specified plate surface. The temperature gradient acts according to the plate local z axis direction and is constant over the element surface. This attribute is only active for static and dynamic structural analysis. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateTempGradient1(long uID, long PlateNum, long CaseNum,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0]`

Temperature gradient in the local z axis direction.

St7GetPlatePointForce6

Returns the point force assigned to the specified plate. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlatePointForce6(long uID, long PlateNum, long CaseNum, long Position,
    long ID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`Position`

Position identifier; either axUCS or axLocal.

`ID`

Local ID number for the point force.

Output Parameters

`Doubles[0..5]`

`[0..2]` – Components of the applied force in the global XYZ system.

Where Position is axUCS,
`[3..5]` – The position of the point force specified in global XYZ coordinates.

Where Position is axLocal,
`[3..4]` – The uv position of the point force in the local element coordinate system.

St7GetPlatePointMoment6

Returns the point moment assigned to the specified plate. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlatePointMoment6(long uID, long PlateNum, long CaseNum,
     long Position, long ID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`Position`

Position identifier; either axUCS or axLocal.

`ID`

Local ID number for the point moment.

Output Parameters

`Doubles[0..5]`

`[0..2]` – Components of the applied moment in the global XYZ system.

Where Position is axUCS:
`[3..5]` – The position of the point moment in global XYZ coordinates.

Where Position is axLocal:
`[3..4]` – The uv position of the point moment in the local element coordinate system.

St7GetPlateEdgePressure1

Returns the normal edge pressure assigned to the specified plate edge. The pressure is applied in the plane of the element, perpendicular to the plate edge. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateEdgePressure1(long uID, long PlateNum, long CaseNum,
     long EdgeNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number

`CaseNum`

Load case number.

`EdgeNum`

Local edge number; one of 1, 2, 3 or 4.

Output Parameters

`Doubles[0]`

Edge pressure for the plate edge, with positive pressure directed away from the plate.

St7GetPlateEdgeShear1

Returns the shear stress assigned to the specified plate edge. The shear stress is applied tangential to the plate edge. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateEdgeShear1(long uID, long PlateNum, long CaseNum, long EdgeNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`EdgeNum`

Local edge number; one of 1, 2, 3 or 4.

Output Parameters

`Doubles[0]`

Shear stress.

St7GetPlateEdgeTransverseShear1

Returns the transverse shear stress assigned to the specified plate edge. The shear stress acts normal to the plate surface at its edge, in the local +z direction. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateEdgeTransverseShear1(long uID, long PlateNum, long CaseNum,
     long EdgeNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`EdgeNum`

Local edge number; one of 1, 2, 3 or 4.

Output Parameters

`Doubles[0]`

Transverse shear stress.

St7GetPlateEdgePressure3

Returns the global edge pressure assigned to the specified plate edge. The pressure is constant and is defined in the global XYZ system. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateEdgePressure3(long uID, long PlateNum, long CaseNum,
     long EdgeNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number

`CaseNum`

Load case number.

`EdgeNum`

Local edge number; one of 1, 2, 3 or 4.

Output Parameters

`Doubles[0..2]`

A 3-element array containing the pressure components in the global XYZ system.

St7GetPlateNormalPressure2

Returns the normal pressures applied to the outer +z and -z surfaces of the specified plate. The pressure is constant and is specified as a positive pressure on each face, directed inwards towards the mid-plane of the plate. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateNormalPressure2(long uID, long PlateNum, long CaseNum,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0..1]`

`[0]` – Normal face pressure over the -z surface of the plate.

`[1]` – Normal face pressure over the +z surface of the plate.

Positive pressure is directed from the relevant surface into the plate.

St7GetPlateGlobalPressure3S

Returns the pressure applied to the specified plate surface. The pressure is constant and is defined in the global XYZ system. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateGlobalPressure3S(long uID, long PlateNum, long Surface,
    long CaseNum, long* ProjectFlag, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`Surface`

Plate surface; either psPlateMinusZ or psPlatePlusZ.

`CaseNum`

Load case number.

Output Parameters

`ProjectFlag`

One of ppNone, ppProjResultant or ppProjComponents.

`Doubles[0..2]`

A 3-element array containing the pressure components in the global XYZ system.

St7GetPlateShear2

Returns the shear stress assigned to the specified plate surface. The shear stress is applied in the plane of the element. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateShear2(long uID, long PlateNum, long CaseNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0..1]`

A 2-element array that describes the applied shear stress according to the local plate xy axis system.

St7GetPlateNSMass5ID

Returns the non-structural mass assigned to the specified plate. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateNSMass5ID(long uID, long PlateNum, long CaseNum, long ID,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`ID`

Non-structural mass ID.

Output Parameters

`Doubles[0..5]`

`[0]` – Non-structural mass for the specified plate.

`[1]` – Dynamic factor for the specified plate. This factor is used to scale the non-structural mass when performing dynamic analyses.

`[2..5]` – A 3-element array describing the offset in the global XYZ system.

St7GetPlateEdgeConvection2

Returns the edge thermal convection coefficient and ambient temperature assigned to the specified plate. This attribute is only used when performing heat transfer analysis. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateEdgeConvection2(long uID, long PlateNum, long CaseNum,
     long EdgeNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`EdgeNum`

Local edge number; one of 1, 2, 3 or 4.

Output Parameters

`Doubles[0..1]`

`[0]` – Convection coefficient.

`[1]` – Ambient temperature.

St7GetPlateEdgeConvectionTables

Returns the tables associated with convection coefficient assigned to the specified plate edge. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateEdgeConvectionTables(long uID, long PlateNum, long CaseNum,
     long EdgeNum, long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`EdgeNum`

Local edge number; one of 1, 2, 3 or 4.

Output Parameters

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the convection ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the convection coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the convection coefficient, or 0 for none.

St7GetPlateEdgeRadiation2

Returns the thermal radiation coefficient and ambient temperature assigned to the specified plate edge. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateEdgeRadiation2(long uID, long PlateNum, long CaseNum,
     long EdgeNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`EdgeNum`

Local edge number.

Output Parameters

`Doubles[0..1]`

`[0]` – Radiation coefficient.

`[1]` – Ambient temperature.

St7GetPlateEdgeRadiationTables

Returns the tables associated with the radiation coefficient assigned to the specified plate edge. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateEdgeRadiationTables(long uID, long PlateNum, long CaseNum,
     long EdgeNum, long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`EdgeNum`

Local edge number; one of 1, 2, 3 or 4.

Output Parameters

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the radiation ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the radiation coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the radiation coefficient, or 0 for none.

St7GetPlateFlux1

Returns the heat flux assigned to the specified plate edge. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateFlux1(long uID, long PlateNum, long CaseNum, long EdgeNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`EdgeNum`

Local edge number; one of 1, 2, 3 or 4.

Output Parameters

`Doubles[0]`

Heat flux through the plate edge.

St7GetPlateFluxTables

Returns the tables associated with the heat flux assigned to the specified plate edge. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateFluxTables(long uID, long PlateNum, long CaseNum, long EdgeNum,
     long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`EdgeNum`

Local edge number; one of 1, 2, 3 or 4.

Output Parameters

`Tables[0..1]`

`[0]` – Factor vs Time table ID associated with the heat flux, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the heat flux, or 0 for none.

St7GetPlateFaceConvection2

Returns the thermal convection coefficient and ambient temperature assigned to the specified plate. This attribute is only used when performing heat transfer analysis. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateFaceConvection2(long uID, long PlateNum, long CaseNum,
     long Surface, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`Surface`

Local plate surface; either psPlateMinusZ or psPlatePlusZ.

Output Parameters

`Doubles[0..1]`

`[0]` – Convection coefficient.

`[1]` – Ambient temperature.

St7GetPlateFaceConvectionTables

Returns the tables associated with convection coefficient assigned to the specified plate surface. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateFaceConvectionTables(long uID, long PlateNum, long CaseNum,
     long Surface, long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`Surface`

Local plate surface; either psPlateMinusZ or psPlatePlusZ.

Output Parameters

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the convection ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the convection coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the convection coefficient, or 0 for none.

St7GetPlateFaceRadiation2

Returns the thermal radiation coefficient and ambient temperature assigned to the specified plate. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateFaceRadiation2(long uID, long PlateNum, long CaseNum,
     long Surface, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`Surface`

Local plate surface; either psPlateMinusZ or psPlatePlusZ.

Output Parameters

`Doubles[0..1]`

`[0]` – Radiation coefficient.

`[1]` – Ambient temperature.

St7GetPlateFaceRadiationTables

Returns the tables associated with the radiation coefficient assigned to the specified plate surface. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateFaceRadiationTables(long uID, long PlateNum, long CaseNum,
     long Surface, long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`Surface`

Local plate surface; either psPlateMinusZ or psPlatePlusZ.

Output Parameters

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the radiation ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the radiation coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the radiation coefficient, or 0 for none.

St7GetPlateHeatSource1

Returns the heat source assigned to the specified plate. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateHeatSource1(long uID, long PlateNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0]`

Heat source.

St7GetPlateHeatSourceTables

Returns the tables associated with the heat source assigned to the specified plate. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateHeatSourceTables(long uID, long PlateNum, long CaseNum,
     long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

Output Parameters

`Tables[0..1]`

`[0]` – Factor vs Time table ID associated with the heat source, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the heat source, or 0 for none.

St7GetPlateSoilStress2

Returns the in-situ soil stress assigned to the specified plate. This attribute is only active for plates of property type soil.

```
long St7GetPlateSoilStress2(long uID, long PlateNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

Output Parameters

`Doubles[0..1]`

A 2-element array containing the initial vertical stress and the horizontal stress ratio.

St7GetPlateSoilRatio2

Returns the in-situ soil ratios assigned to the specified plate. This attribute is only active for plates of property type soil.

```
long St7GetPlateSoilRatio2(long uID, long PlateNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

Output Parameters

`Doubles[0..1]`

A 2-element array containing the overconsolidation ratio and the initial void ratio.

St7GetPlateResponse

Returns the response variable assigned to the specified plate. Response variables are only used by the load influence solver. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetPlateResponse(long uID, long PlateNum, long CaseNum,
     long ResponseType, long* UCSId, long* Status)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`CaseNum`

Load case number.

`ResponseType`

Response variable type; either rvPlateForce or rvPlateMoment.

Output Parameters

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Status[0..5]`

A 6-element array describing the force/moment components that are flagged as response variables according to the 123 axis convention in the specified UCS – [11, 22, 33, 12, 23, 31].

St7GetPlateLoadPatch4

Returns the load patch type assigned to the specified plate. This attribute is only active for plates of property type load patch.

```
long St7GetPlateLoadPatch4(long uID, long PlateNum, long* PatchType,
     long* EdgeBits, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

Output Parameters

`PatchType`

One of ptAuto4, ptAuto3, ptAuto2, ptAuto1, ptAngleSplit or ptManual.

`EdgeBits`

A 32-bit value in which the four least significant bits specify the selection of up to four edges. See *Load Patch Types* for additional information.

`Doubles[0..3]`

Edge weights. See *Load Patch Types* for additional information.

St7GetPlateReinforcement2

Returns the concrete reinforcement conditions for the specified plate.

```
long St7GetPlateReinforcement2(long uID, long PlateNum, long* LayoutID,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

Output Parameters

`LayoutID`

Reinforcement layout ID.

`Doubles[0..1]`

A 2-element array describing the angular orientation (degrees) of the 1-3 and 2-4 reinforcement layers respectively.

St7GetPlateCreepLoadingAge1

Returns the creep loading age assigned to the specified plate. This attribute is only active when conducting creep analysis using the quasi-static or nonlinear transient dynamic solvers.

```
long St7GetPlateCreepLoadingAge1(long uID, long PlateNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

Output Parameters

`Doubles[0]`

Creep loading age in seconds.

St7GetPlateEdgeAttachment1

Returns the attachment assigned to the specified plate edge. Attachment attributes can be used to generate attachment links using the *St7CreateAttachments* function.

```
long St7GetPlateEdgeAttachment1(long uID, long PlateNum, long EdgeNum,
     long Direction, long* AttachType, long* ConnectType, long* PropNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`EdgeNum`

Local edge number; one of 1, 2, 3 or 4.

`Direction`

One of adPlanar, adPlusZ or adMinusZ.

Output Parameters

`AttachType`

One of alDirect, alRigid or alFlexible.

`ConnectType`

Either alMoment or alPinned.

`PropNum`

Beam property number used for flexible attachment types.

`Doubles[0]`

The maximum distance within which the plate edge can be connected to another element using the attachment link.

St7GetPlateFaceAttachment1

Returns the attachment assigned to the specified plate surface. Attachment attributes can be used to generate attachment links using the *St7CreateAttachments* function.

```
long St7GetPlateFaceAttachment1(long uID, long PlateNum, long Surface,
     long* AttachType, long* ConnectType, long* PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`Surface`

Local plate surface; either psPlateMinusZ or psPlatePlusZ.

Output Parameters

`AttachType`

One of alDirect, alRigid or alFlexible.

`ConnectType`

Either alMoment or alPinned.

`PropNum`

Beam property number used for flexible attachment types.

`Doubles[0]`

The maximum distance within which the plate surface can be connected to another element using the attachment link.

St7GetPlateCavityFluid

Returns the cavity fluid layout assigned to a surface of the specified plate.

```
long St7GetPlateCavityFluid(long uID, long PlateNum, long Surface,
     long* CavityID)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`Surface`

Local plate surface; either psPlateMinusZ or psPlatePlusZ.

Output Parameters

`CavityID`

Cavity fluid layout ID.

Brick Attributes – Set

St7SetBrickID

Sets the ID number for the specified brick.

```
long St7SetBrickID(long uID, long BrickNum, long BrickID)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`BrickID`

Brick ID.

St7SetBrickLocalAxes1

Aligns the brick local axis system with the specified UCS. See *Brick Local Coordinates* for additional information.

```
long St7SetBrickLocalAxes1(long uID, long BrickNum, long UCSId)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

St7SetBrickSupport4

Sets the elastic support conditions for the specified brick face.

```
long St7SetBrickSupport4(long uID, long BrickNum, long FaceNum, long CaseNum,
     long* Status, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number; see *Brick Local Coordinates*.

`CaseNum`

Freedom case number.

`Status[0..1]`

`[0]` – Compression-only support; either btTrue or btFalse.

`[1]` – Limited bearing capacity; either btTrue or btFalse.

`Doubles[0..3]`

`[0]` – Elastic support value in the normal direction.

`[1]` – Elastic support value in the lateral direction.

`[2]` – Support gap. Only relevant if Status[0] is set to btTrue.

`[3]` – Bearing capacity. Only relevant if Status[1] is set to btTrue.

St7SetBrickPreLoad3

Sets the pre-load conditions for the specified brick. The pre-loads are defined in the local axis system of the brick.

```
long St7SetBrickPreLoad3(long uID, long BrickNum, long CaseNum, long LoadType,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`CaseNum`

Load case number.

`LoadType`

plBrickPreStress or plBrickPreStrain.

`Doubles[0..2]`

A 3-element array describing the pre-load magnitudes according to the orientation of the local brick axis system.

St7SetBrickPointForce6

Assigns a point force to the specified brick face.

```
long St7SetBrickPointForce6(long uID, long BrickNum, long FaceNum, long CaseNum,
     long Position, long ID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Load case number.

`Position`

Position identifier; either axUCS or axLocal.

`ID`

Point force ID.

`Doubles[0..5]`

`[0..2]` – Components of the applied force in the global XYZ system.

Where Position is axUCS:
`[3..5]` – The position of the applied force in global XYZ coordinates.

Where Position is axLocal:
`[3..4]` – The uv position of the applied force in the local element coordinate system of the brick face.

St7SetBrickNormalPressure1

Assigns a pressure to the specified brick face. The pressure acts into the element, normal to the plane of the face and is constant over the surface.

```
long St7SetBrickNormalPressure1(long uID, long BrickNum, long FaceNum,
     long CaseNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Load case number.

`Doubles[0]`

Normal pressure.

St7SetBrickGlobalPressure3

Assigns a constant pressure to the specified brick face. The pressure is defined in the global XYZ system.

```
long St7SetBrickGlobalPressure3(long uID, long BrickNum, long FaceNum,
     long ProjectFlag, long CaseNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`ProjectFlag`

One of ppNone, ppProjResultant or ppProjComponents.

`CaseNum`

Load case number.

`Doubles[0..2]`

A 3-element array containing the pressure components in the global XYZ system.

St7SetBrickShear2

Assigns a shear stress to the specified brick face. The shear stress acts in the plane of the face and is constant over the surface.

```
long St7SetBrickShear2(long uID, long BrickNum, long FaceNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Load case number.

`Doubles[0..1]`

A 2-element array describing the shear stress components in the local face xy axis system. See *Brick Local Coordinates* for details.

St7SetBrickNSMass5ID

Assigns a non-structural mass to the specified brick.

```
long St7SetBrickNSMass5ID(long uID, long BrickNum, long FaceNum, long CaseNum,
     long ID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Load case number.

`ID`

Non-structural mass ID.

`Doubles[0..5]`

`[0]` – Non-structural mass for the specified brick face.

`[1]` – Dynamic factor for the specified brick face. This factor is used to scale the non-structural mass when performing dynamic analyses.

`[2..5]` – A 3-element array describing the offset in the global XYZ system.

St7SetBrickConvection2

Assigns the thermal convection coefficient and ambient temperature for the specified brick face.

```
long St7SetBrickConvection2(long uID, long BrickNum, long FaceNum, long CaseNum,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Load case number.

`Doubles[0..1]`

`[0]` – Convection coefficient.

`[1]` – Ambient temperature.

St7SetBrickConvectionTables

Sets the tables associated with the convection coefficient assigned to the specified brick face.

```
long St7SetBrickConvectionTables(long uID, long BrickNum, long FaceNum,
    long CaseNum, long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Load case number.

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the convection ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the convection coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the convection coefficient, or 0 for none.

St7SetBrickRadiation2

Assigns the thermal radiation coefficient and ambient temperature for the specified brick face.

```
long St7SetBrickRadiation2(long uID, long BrickNum, long FaceNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Load case number.

`Doubles[0..1]`

`[0]` – Radiation coefficient.

`[1]` – Ambient temperature.

St7SetBrickRadiationTables

Sets the tables associated with the radiation coefficient assigned to the specified brick face.

```
long St7SetBrickRadiationTables(long uID, long BrickNum, long FaceNum,
     long CaseNum, long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Load case number.

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the radiation ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the radiation coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the radiation coefficient, or 0 for none.

St7SetBrickFlux1

Assigns a heat flux to the specified brick face.

```
long St7SetBrickFlux1(long uID, long BrickNum, long FaceNum, long CaseNum,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Load case number.

`Doubles[0]`

The heat flux through the brick face.

St7SetBrickFluxTables

Sets the tables associated with the heat flux assigned to the specified brick face.

```
long St7SetBrickFluxTables(long uID, long BrickNum, long FaceNum, long CaseNum,
    long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Load case number.

`Tables[0..1]`

`[0]` – Factor vs Time table ID associated with the heat flux, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the heat flux, or 0 for none.

St7SetBrickHeatSource1

Sets a heat source to the specified brick.

```
long St7SetBrickHeatSource1(long uID, long BrickNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`CaseNum`

Load case number.

`Doubles[0]`

Heat source.

St7SetBrickHeatSourceTables

Sets the tables associated with the heat source assigned to the specified brick.

```
long St7SetBrickHeatSourceTables(long uID, long BrickNum, long CaseNum,
     long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`CaseNum`

Load case number.

`Tables[0..1]`

`[0]` – Factor vs Time table ID associated with the heat source, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the heat source, or 0 for none.

St7SetBrickSoilStress2

Assigns the in-situ soil stress for the specified brick. This attribute is only active for bricks of property type soil.

```
long St7SetBrickSoilStress2(long uID, long BrickNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`Doubles[0..1]`

A 2-element array containing the initial vertical stress and the horizontal stress ratio.

St7SetBrickSoilRatio2

Assigns the in-situ soil ratios for the specified brick. This attribute is only active for bricks of property type soil.

```
long St7SetBrickSoilRatio2(long uID, long BrickNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`Doubles[0..1]`

A 2-element array containing the overconsolidation ratio and the initial void ratio.

St7SetBrickResponse

Assigns a response variable to the specified brick.

```
long St7SetBrickResponse(long uID, long BrickNum, long CaseNum, long UCSId,
    long* Status)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`CaseNum`

Load case number.

`UCSId`

0 to use the local axis system of the brick, or the ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Status[0..5]`

A 6-element array describing the stress components that are flagged as response variables according to the 123 axis convention in the specified UCS, [11, 22, 33, 12, 23, 31].

St7SetBrickCreepLoadingAge1

Assigns a creep loading age for the specified brick.

```
long St7SetBrickCreepLoadingAge1(long uID, long BrickNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`Doubles[0]`

Creep loading age in seconds.

St7SetBrickFaceAttachment1

Assigns an attachment to the specified brick face. Attachment attributes can be used to generate attachment links using the *St7CreateAttachments* function.

```
long St7SetBrickFaceAttachment1(long uID, long BrickNum, long FaceNum,
     long AttachType, long ConnectType, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`AttachType`

One of alDirect, alRigid or alFlexible.

`ConnectType`

Attachment sub-type; either alMoment or alPinned.

`PropNum`

Beam property number used for flexible attachment types.

`Doubles[0]`

The maximum distance within which the brick face can be connected to another element using the attachment link.

St7SetBrickCavityFluid

Assigns a cavity fluid layout to a face of the specified brick.

```
long St7SetBrickCavityFluid(long uID, long BrickNum, long FaceNum, long CavityID)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CavityID`

Cavity fluid layout ID.

Brick Attributes – Get

St7GetBrickID

Returns the ID number assigned to the specified brick.

```
long St7GetBrickID(long uID, long BrickNum, long* BrickID)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

Output Parameters

`BrickID`

Brick ID.

St7GetBrickLocalAxes1

Returns the UCS used as the local axis system for the specified brick.

```
long St7GetBrickLocalAxes1(long uID, long BrickNum, long* UCSId)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

Output Parameters

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

St7GetBrickSupport4

Returns the elastic support assigned to the specified brick face. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBrickSupport4(long uID, long BrickNum, long FaceNum, long CaseNum,
    long* Status, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Freedom case number.

Output Parameters

`Status[0..1]`

`[0]` – Compression-only support; either btTrue or btFalse.

`[1]` – Limited bearing capacity; either btTrue or btFalse.

`Doubles[0..3]`

`[0]` – Elastic support value in the normal direction.

`[1]` – Elastic support value in the lateral direction.

`[2]` – Support gap. Only relevant if Status[0] is set to btTrue.

`[3]` – Bearing capacity. Only relevant if Status[1] is set to btTrue.

St7GetBrickPreLoad3

Returns the pre-load conditions assigned to the specified brick. The pre-loads are defined in the local axis system of the brick. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBrickPreLoad3(long uID, long BrickNum, long CaseNum, long* LoadType,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`CaseNum`

Load case number.

Output Parameters

`LoadType`

plBrickPreStress or plBrickPreStrain.

`Doubles[0..2]`

A 3-element array describing the pre-load magnitudes according to the orientation of the local brick axis system.

St7GetBrickPointForce6

Returns the point force assigned to the specified brick face. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBrickPointForce6(long uID, long BrickNum, long FaceNum, long CaseNum,
     long Position, long ID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Load case number.

`Position`

Position identifier; either axUCS or axLocal.

`ID`

Point force ID.

Output Parameters

`Doubles[0..5]`

`[0..2]` – Components of the applied force in the global XYZ system.

Where Position is axUCS:
`[3..5]` – The position of the applied force in global XYZ coordinates.

Where Position is axLocal:
`[3..4]` – The uv position of the applied force in the local element coordinate system.

St7GetBrickNormalPressure1

Returns the pressure assigned to the specified brick face. The pressure acts into the element, normal to the plane of the face and is constant over the surface. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBrickNormalPressure1(long uID, long BrickNum, long FaceNum,
     long CaseNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0]`

Normal pressure value.

St7GetBrickGlobalPressure3

Returns the pressure assigned to the specified brick face. The pressure is constant and is defined in the global XYZ system. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBrickGlobalPressure3(long uID, long BrickNum, long FaceNum,
     long CaseNum, long* ProjectFlag, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Load case number.

Output Parameters

`ProjectFlag`

One of ppNone, ppProjResultant or ppProjComponents.

`Doubles[0..2]`

A 3-element array containing the pressure components in the global XYZ system.

St7GetBrickShear2

Returns the shear stress assigned to the specified brick face. The shear stress acts in the plane of the face and is constant over the surface. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBrickShear2(long uID, long BrickNum, long FaceNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0..1]`

A 2-element array describing the shear stress components in the local face xy axis system. See *Brick Local Coordinates* for details.

St7GetBrickNSMass5ID

Returns the non-structural mass assigned to the specified brick. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBrickNSMass5ID(long uID, long BrickNum, long FaceNum, long CaseNum,
     long ID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Load case number.

`ID`

Non-structural mass ID.

Output Parameters

`Doubles[0..5]`

`[0]` – Non-structural mass for the specified brick face.

`[1]` – Dynamic factor for the specified brick face. This factor is used to scale the non-structural mass when performing dynamic analyses.

`[2..5]` – A 3-element array describing the offset in the global XYZ system.

St7GetBrickConvection2

Returns the thermal convection coefficient and ambient temperature assigned to the specified brick face. This attribute is only used when performing heat transfer analysis. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBrickConvection2(long uID, long BrickNum, long FaceNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0..1]`

`[0]` – Convection coefficient.

`[1]` – Ambient temperature.

St7GetBrickConvectionTables

Returns the tables associated with the convection coefficient assigned to the specified brick face. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBrickConvectionTables(long uID, long BrickNum, long FaceNum,
     long CaseNum, long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Load case number.

Output Parameters

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the convection ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the convection coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the convection coefficient, or 0 for none.

St7GetBrickRadiation2

Returns the thermal radiation coefficient and ambient temperature assigned to the specified brick face. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBrickRadiation2(long uID, long BrickNum, long FaceNum, long CaseNum,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0..1]`

`[0]` – Radiation coefficient.

`[1]` – Ambient temperature.

St7GetBrickRadiationTables

Returns the tables associated with the radiation coefficient assigned to the specified brick face. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBrickRadiationTables(long uID, long BrickNum, long FaceNum,
     long CaseNum, long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Load case number.

Output Parameters

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the radiation ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the radiation coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the radiation coefficient, or 0 for none.

St7GetBrickFlux1

Returns the heat flux assigned to the specified brick face. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBrickFlux1(long uID, long BrickNum, long FaceNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0]`

The heat flux through the brick face.

St7GetBrickFluxTables

Returns the tables associated with the heat flux assigned to the specified brick face. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBrickFluxTables(long uID, long BrickNum, long FaceNum, long CaseNum,
     long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

`CaseNum`

Load case number.

Output Parameters

`Tables[0..1]`

`[0]` – Factor vs Time table ID associated with the heat flux, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the heat flux, or 0 for none.

St7GetBrickHeatSource1

Returns the heat source assigned to the specified brick. This attribute is only used when performing heat transfer analysis. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBrickHeatSource1(long uID, long BrickNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0]`

Heat source.

St7GetBrickHeatSourceTables

Returns the tables associated with the heat source assigned to the specified brick. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBrickHeatSourceTables(long uID, long BrickNum, long CaseNum,
     long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`CaseNum`

Load case number.

Output Parameters

`Tables[0..1]`

`[0]` – Factor vs Time table ID associated with the heat source, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the heat source, or 0 for none.

St7GetBrickSoilStress2

Returns the in-situ soil stress assigned to the specified brick. This attribute is only active for bricks of property type soil.

```
long St7GetBrickSoilStress2(long uID, long BrickNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

Output Parameters

`Doubles[0..1]`

A 2-element array containing the initial vertical stress and the horizontal stress ratio.

St7GetBrickSoilRatio2

Returns the in-situ soil ratios for the specified brick. This attribute is only active for bricks of property type soil.

```
long St7GetBrickSoilRatio2(long uID, long BrickNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

Output Parameters

`Doubles[0..1]`

A 2-element array containing the overconsolidation ratio and the initial void ratio.

St7GetBrickResponse

Returns the response variable assigned to the specified brick. Response variables are only used by the load influence solver. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetBrickResponse(long uID, long BrickNum, long CaseNum, long* UCSId,
     long* Status)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`CaseNum`

Load case number.

Output Parameters

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Status[0..5]`

A 6-element array describing the stress components that are flagged as response variables – lists the 11, 22, 33, 12, 23, 31 components in the 123 axis convention in the specified UCS.

St7GetBrickCreepLoadingAge1

Returns the creep loading age assigned to the specified brick. This attribute is only used when performing creep analysis using the quasi-static or nonlinear transient dynamic solvers.

```
long St7GetBrickCreepLoadingAge1(long uID, long BrickNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

Output Parameters

`Doubles[0]`

Creep loading age in seconds.

St7GetBrickFaceAttachment1

Returns the attachment conditions assigned to the specified brick face. Attachment attributes can be used to generate attachment links using the *St7CreateAttachments* function.

```
long St7GetBrickFaceAttachment1(long uID, long BrickNum, long FaceNum,
    long* AttachType, long* ConnectType, long* PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

Output Parameters

`AttachType`

One of alDirect, alRigid or alFlexible.

`ConnectType`

Either alMoment or alPinned.

`PropNum`

Beam property number used for flexible attachment types.

`Doubles[0]`

The maximum distance within which the brick face can be connected to another element using the attachment link.

St7GetBrickCavityFluid

Returns the cavity fluid layout assigned to a face of the specified brick.

```
long St7GetBrickCavityFluid(long uID, long BrickNum, long FaceNum,
     long* CavityID)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`FaceNum`

Local face number. See *Brick Local Coordinates* for additional information.

Output Parameters

`CavityID`

Cavity fluid layout ID.

Link Attributes – Set

St7SetLinkID

Assigns an ID number to the specified link.

```
long St7SetLinkID(long uID, long LinkNum, long LinkID)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`LinkID`

Link ID.

St7SetInterpolatedMultiPointLinkAttributes

Assigns attributes to an interpolated multi-point link.

```
long St7SetInterpolatedMultiPointLinkAttributes(long uID, long LinkNum,
     long Couple)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`Couple`

One of cpTranslational, cpRotational or cpBoth.

St7SetMasterSlaveMultiPointLinkAttributes

Assigns attributes to a master-slave multi-point link.

```
long St7SetMasterSlaveMultiPointLinkAttributes(long uID, long LinkNum,
     long UCSId, long DoFBits)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`DOFBits`

Bitmask of flags for six degrees of freedom if Master-Slave links are created. For example, to couple DX, DY and RZ, set ConstraintBits = 1+2+32 = 35.

St7SetRigidMultiPointLinkAttributes

Assigns attributes to a rigid multi-point link.

```
long St7SetRigidMultiPointLinkAttributes(long uID, long LinkNum, long UCSId,
     long Axis)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Axis`

One of rlPlaneXYZ, rlPlaneXY, rlPlaneYZ or rlPlaneZX.

St7SetReactionMultiPointLinkAttributes

Assigns attributes to a reaction multi-point link.

```
long St7SetReactionMultiPointLinkAttributes(long uID, long LinkNum, long SetNum,
     double* Origin)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`SetNum`

Element set number defining the entities that contribute to the reaction multi-point link cluster.

`Origin[0..2]`

Origin position in global XYZ coordinates.

Link Attributes – Get

St7GetLinkID

Returns the ID number assigned to the specified link.

```
long St7GetLinkID(long uID, long LinkNum, long* LinkID)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

Output Parameters

`LinkID`

Link ID.

St7GetInterpolatedMultiPointLinkAttributes

Returns attributes assigned to an interpolated multi-point link.

```
long St7GetInterpolatedMultiPointLinkAttributes(long uID, long LinkNum,
     long* Couple)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

Output Parameters

`Couple`

One of cpTranslational, cpRotational or cpBoth.

St7GetMasterSlaveMultiPointLinkAttributes

Returns attributes assigned to a master-slave multi-point link.

```
long St7GetMasterSlaveMultiPointLinkAttributes(long uID, long LinkNum,
     long* UCSId, long* DoFBits)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

Output Parameters

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`DOFBits`

Bitmask of flags for six degrees of freedom if Master-Slave links are created. For example, to couple DX, DY and RZ, set ConstraintBits = 1+2+32 = 35.

St7GetRigidMultiPointLinkAttributes

Returns attributes assigned to a rigid multi-point link.

```
long St7GetRigidMultiPointLinkAttributes(long uID, long LinkNum, long* UCSId,
    long* Axis)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

Output Parameters

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Axis`

One of rlPlaneXYZ, rlPlaneXY, rlPlaneYZ or rlPlaneZX.

St7GetReactionMultiPointLinkAttributes

Returns attributes assigned to a reaction multi-point link.

```
long St7GetReactionMultiPointLinkAttributes(long uID, long LinkNum, long* SetNum,
    double* Origin)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

Output Parameters

`SetNum`

Element set number defining the entities that contribute to the reaction multi-point link cluster.

`Origin[0..2]`

Origin position in global XYZ coordinates.

Vertex Attributes – Set

St7SetVertexType

Sets the type for the specified vertex.

```
long St7SetVertexType(long uID, long VertexNum, long VertexType)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`VertexType`

Vertex type; either vtFree or vtFixed.

St7SetVertexID

Assigns an ID number to the specified vertex.

```
long St7SetVertexID(long uID, long VertexNum, long VertexID)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`VertexID`

Vertex ID.

St7SetVertexMeshSize1

Assigns a desired mesh size at the specified vertex. This value is used to control the local mesh resolution when using the surface automeshing tools.

```
long St7SetVertexMeshSize1(long uID, long VertexNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`Doubles[0]`

Desired mesh size at the specified vertex. This value is used to determine the desired edge length of adjacent plate elements generated during surface auto-meshing.

St7SetVertexRestraint6

Assigns structural restraint conditions at the specified vertex.

```
long St7SetVertexRestraint6(long uID, long VertexNum, long CaseNum, long UCSId,
     long* Status, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Load case number.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Status[0..5]`

An array describing the restraint conditions for the six DoF at the specified vertex. `Status[i-1] = btTrue` indicates that the i^{th} DoF is restrained. The DoF are restrained according to the 123456 axis convention in the specified UCS.

`Doubles[0..5]`

An array describing the enforced displacement conditions for the six DoF at the specified vertex. `Doubles[i-1]` describes the displacement of the i^{th} DoF according to the 123456 axis convention in the specified UCS.

St7SetVertexForce3

Assigns a point force to the specified vertex.

```
long St7SetVertexForce3(long uID, long VertexNum, long CaseNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Load case number.

`Doubles[0..2]`

A 3-element array containing the force defined in the global XYZ system for the specified vertex.

St7SetVertexMoment3

Assigns a point moment to the specified vertex.

```
long St7SetVertexMoment3(long uID, long VertexNum, long CaseNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Load case number.

`Doubles[0..2]`

A 3-element array containing the moment defined in the global XYZ system for the specified vertex.

St7SetVertexTemperature1

Assigns a temperature to the specified vertex.

```
long St7SetVertexTemperature1(long uID, long VertexNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Load case number.

`Doubles[0]`

Temperature value at the specified vertex.

St7SetVertexTemperatureType1

Sets the temperature type assigned at the specified vertex. This attribute is used when performing both structural and heat transfer analysis.

```
long St7SetVertexTemperatureType1(long uID, long VertexNum, long CaseNum,
     long TType)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Load case number.

`TType`

One of ntReferenceTemperature, ntFixedTemperature, ntInitialTemperature or ntTableTemperature.

St7SetVertexTemperatureTable

Specifies the table to be associated with the temperature assigned to the specified vertex. This attribute is used when performing both structural and heat transfer analysis.

```
long St7SetVertexTemperatureTable(long uID, long VertexNum, long CaseNum,
     long TableID)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Load case number.

`TableID`

Temperature vs Time table ID associated with the vertex temperature, or 0 for none.

St7SetVertexKTranslation3F

Assigns a translational stiffness to the specified vertex.

```
long St7SetVertexKTranslation3F(long uID, long VertexNum, long CaseNum,
     long UCSId, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Freedom case number.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Doubles[0..2]`

A 3-element array describing the translational stiffnesses for the specified vertex. `Doubles[i-1]` describes the stiffness for the i^{th} translational DoF according to the 123 axis definition in the specified UCS.

St7SetVertexKRotation3F

Assigns a rotational stiffness to the specified vertex.

```
long St7SetVertexKRotation3F(long uID, long VertexNum, long CaseNum, long UCSId,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Freedom case number.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Doubles[0..2]`

A 3-element array describing the rotational stiffnesses for the specified vertex. `Doubles[i-1]` describes the stiffness for the i^{th} rotational DoF according to the 456 axis definition in the specified UCS.

St7SetVertexTMass1

Assigns a translational mass to the specified vertex as a single value.

```
long St7SetVertexTMass1(long uID, long VertexNum, double Mass)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`Mass`

The translational mass.

St7SetVertexTMass3

Assigns a translational mass to the specified vertex as three components.

```
long St7SetVertexTMass3(long uID, long VertexNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`Doubles[0..2]`

A 3-element array describing the translational mass for the specified vertex in the global XYZ system.

St7SetVertexRMass3

Assigns a rotational mass to the specified vertex.

```
long St7SetVertexRMass3(long uID, long VertexNum, long UCSId, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Doubles[0..2]`

A 3-element array defining the rotational mass for the vertex about the axes of the specified UCS.

St7SetVertexNSMass5ID

Assigns a non-structural mass to the specified vertex.

```
long St7SetVertexNSMass5ID(long uID, long VertexNum, long CaseNum, long ID,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Load case number.

`ID`

Non-structural mass ID.

`Doubles[0..4]`

`[0]` – Non-structural mass at the specified vertex.

`[1]` – Dynamic factor for the specified vertex. This factor is used to scale the non-structural mass when performing dynamic analysis.

`[2..4]` – A 3-element array describing the offset in the global XYZ system.

St7SetVertexKDamping3F

Assigns the translational damping coefficients for the specified vertex.

```
long St7SetVertexKDamping3F(long uID, long VertexNum, long CaseNum, long UCSId,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Load case number.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Doubles[0..2]`

A 3-element array describing the damping factors for the specified vertex. `Doubles[i-1]` describes the damping factor for the i^{th} translational DoF according to the 123 axis definition in the specified UCS.

St7SetVertexHeatSource1

Assigns a heat source to the specified vertex.

```
long St7SetVertexHeatSource1(long uID, long VertexNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Load case number.

`Doubles[0]`

Heat source.

St7SetVertexHeatSourceTables

Sets the tables associated with the heat source assigned to the specified vertex.

```
long St7SetVertexHeatSourceTables(long uID, long VertexNum, long CaseNum,
     long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Load case number.

`Tables[0..1]`

`[0]` – Factor vs Time table ID associated with the heat source, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the heat source, or 0 for none.

Vertex Attributes – Get

St7GetVertexType

Returns the type assigned to the specified vertex.

```
long St7GetVertexType(long uID, long VertexNum, long* VertexType)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

Output Parameters

`VertexType`

Vertex type; either vtFree or vtFixed.

St7GetVertexID

Returns the ID number assigned to the specified vertex.

```
long St7GetVertexID(long uID, long VertexNum, long* VertexID)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

Output Parameters

`VertexID`

Vertex ID.

St7GetVertexMeshSize1

Returns the desired mesh size assigned to the specified vertex. This value is used to control the local mesh resolution when using the surface automeshing tools.

```
long St7GetVertexMeshSize1(long uID, long VertexNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

Output Parameters

`Doubles[0]`

Desired mesh size at the specified vertex. This value is used to determine the desired edge length of adjacent plate elements generated during surface auto-meshing.

St7GetVertexRestraint6

Returns the restraint conditions assigned at the specified vertex. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetVertexRestraint6(long uID, long VertexNum, long CaseNum, long* UCSId,
     long* Status, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Load case number.

Output Parameters

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Status[0..5]`

An array describing the restraint conditions for the six DoF at the specified vertex. `Status[i-1] = btTrue` indicates that the i^{th} DoF is restrained. The DoF are restrained according to the 123456 axis convention in the specified UCS.

`Doubles[0..5]`

An array describing the enforced displacement conditions for the six DoF at the specified vertex. `Doubles[i-1]` describes the displacement of the i^{th} DoF according to the 123456 axis convention in the specified UCS.

St7GetVertexForce3

Returns the point force assigned to the specified vertex. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetVertexForce3(long uID, long VertexNum, long CaseNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0..2]`

A 3-element array containing the force in the global XYZ system for the specified vertex.

St7GetVertexMoment3

Returns the point moment assigned at the specified vertex. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetVertexMoment3(long uID, long VertexNum, long CaseNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0..2]`

A 3-element array containing the force in the global XYZ system for the specified vertex.

St7GetVertexTemperature1

Returns the temperature assigned to the specified vertex. This attribute is used when conducting both structural and heat transfer analysis. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetVertexTemperature1(long uID, long VertexNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0]`

Applied temperature value.

St7GetVertexTemperatureType1

Returns the temperature type assigned to the specified vertex. This attribute is used when performing both structural and heat transfer analysis. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetVertexTemperatureType1(long uID, long VertexNum, long CaseNum,
     long* TType)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Load case number.

Output Parameters

`TType`

One of ntReferenceTemperature, ntFixedTemperature, ntInitialTemperature or ntTableTemperature.

St7GetVertexTemperatureTable

Returns the table associated with the specified vertex. This attribute is used when performing both structural and heat transfer analysis. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetVertexTemperatureTable(long uID, long VertexNum, long CaseNum,
     long* TableID)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Load case number.

Output Parameters

`TableID`

Temperature vs Time table ID associated with the vertex temperature, or 0 for none.

St7GetVertexKTranslation3F

Returns the translational stiffness assigned to the specified vertex. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetVertexKTranslation3F(long uID, long VertexNum, long CaseNum,
     long* UCSId, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Freedom case number.

Output Parameters

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Doubles[0..2]`

A 3-element array describing the translational stiffnesses for the vertex in the directions of the specified UCS.

St7GetVertexKRotation3F

Returns the rotational stiffness assigned to the specified vertex. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetVertexKRotation3F(long uID, long VertexNum, long CaseNum, long* UCSId,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Freedom case number.

Output Parameters

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Doubles[0..2]`

A 3-element array describing the rotational stiffnesses for the vertex about the axes of the specified UCS.

St7GetVertexTMass3

Returns the translational mass assigned to the specified vertex.

```
long St7GetVertexTMass3(long uID, long VertexNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

Output Parameters

`Doubles[0..2]`

A 3-element array describing the translational mass for the specified vertex in the global XYZ system.

St7GetVertexRMass3

Returns the rotational mass assigned to the specified vertex.

```
long St7GetVertexRMass3(long uID, long VertexNum, long* UCSId, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

Output Parameters

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Doubles[0..2]`

A 3-element array defining the rotational mass for the vertex about the axes of the specified UCS.

St7GetVertexNSMass5ID

Returns the non-structural mass assigned to the specified vertex. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetVertexNSMass5ID(long uID, long VertexNum, long CaseNum, long ID,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Load case number.

`ID`

Non-structural mass ID.

Output Parameters

`Doubles[0..4]`

`[0]` – Non-structural mass at the specified vertex.

`[1]` – Dynamic factor for the specified vertex. This factor is used to scale the non-structural mass when performing dynamic analysis.

`[2..4]` – A 3-element array describing the offset in the global XYZ system.

St7GetVertexKDamping3F

Returns the translational damping coefficients assigned to the specified vertex. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetVertexKDamping3F(long uID, long VertexNum, long CaseNum, long* UCSId,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Freedom case number.

Output Parameters

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Doubles[0..2]`

A 3-element array describing the damping factors for the specified vertex. `Doubles[i-1]` describes the damping factor for the i^{th} translational DoF according to the 123 axis definition in the specified UCS.

St7GetVertexHeatSource1

Returns the heat source assigned to the specified vertex. This attribute is only used when performing heat transfer analysis. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetVertexHeatSource1(long uID, long VertexNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0]`

Heat source.

St7GetVertexHeatSourceTables

Returns the tables associated with the heat source assigned to the specified vertex. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetVertexHeatSourceTables(long uID, long VertexNum, long CaseNum,
     long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Vertex number.

`CaseNum`

Load case number.

Output Parameters

`Tables[0..1]`

`[0]` – Factor vs Time table ID associated with the heat source, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the heat source, or 0 for none.

Edge Attributes – Set

St7SetGeometryEdgeType

Sets the type for the specified geometry edge. This determines how its attributes are inherited by elements created by the automesher.

```
long St7SetGeometryEdgeType(long uID, long EdgeNum, long EdgeType)
```

Input Parameters

`uID`

Strand7 model file ID.

`EdgeNum`

Edge number.

`EdgeType`

Edge type; either etInterpolated or etNonInterpolated.

St7SetGeometryEdgeMinDivisions

Sets the minimum number of elements to be created by the automesher along the specified geometry edge.

```
long St7SetGeometryEdgeMinDivisions(long uID, long EdgeNum, long Divisions)
```

Input Parameters

`uID`

Strand7 model file ID.

`EdgeNum`

Edge number.

`Divisions`

Minimum number of elements along the geometry edge.

St7SetGeometryEdgeBeamProperty

Sets the beam property number to be created by the automesher on the specified geometry edge.

```
long St7SetGeometryEdgeBeamProperty(long uID, long EdgeNum, long PropNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`EdgeNum`

Edge number.

`PropNum`

Edge beam property on the geometry edge.

St7SetGeometryEdgeCluster

Sets the cluster type and ID on the edge. After automeshing, all nodes along edges with the same cluster type and ID are connected according to the parameters.

```
long St7SetGeometryEdgeCluster(long uID, long EdgeNum, long ClusterID,
    long Entity, long EntityType, long OriginCode, double* Origin)
```

Input Parameters

`uID`

Strand7 model file ID.

`EdgeNum`

Edge number.

`ClusterID`

Cluster ID.

`Entity`

Either tyBEAM to create a beam cluster or tyLINK to create a multi-point link.

`EntityType`

For beams, the property number of beams to be created.

For links, one of ltInterpolatedMultiPointLink, ltMasterSlaveMultiPointLink, ltPinnedMultiPointLink or ltRigidMultiPointLink.

`OriginCode`

Either coAutoClusterOrigin or coManualClusterOrigin.

`Origin[0..2]`

Origin position in global XYZ coordinates when OriginCode is coManualClusterOrigin.

Edge Attributes – Get

St7GetGeometryEdgeType

Returns the type assigned to the specified geometry edge. This determines how its attributes are inherited by elements created by the automesher.

```
long St7GetGeometryEdgeType(long uID, long EdgeNum, long* EdgeType)
```

Input Parameters

`uID`

Strand7 model file ID.

`EdgeNum`

Edge number.

Output Parameters

`EdgeType`

Edge type; either etInterpolated or etNonInterpolated.

St7GetGeometryEdgeMinDivisions

Returns the minimum number of elements to be created by the automesher along the specified geometry edge.

```
long St7GetGeometryEdgeMinDivisions(long uID, long EdgeNum, long* Divisions)
```

Input Parameters

`uID`

Strand7 model file ID.

`EdgeNum`

Edge number.

Output Parameters

`Divisions`

Minimum number of elements along the geometry edge.

St7GetGeometryEdgeBeamProperty

Returns the beam property number to be created by the automesher on the specified geometry edge.

```
long St7GetGeometryEdgeBeamProperty(long uID, long EdgeNum, long* PropNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`EdgeNum`

Edge number.

Output Parameters

`PropNum`

Edge beam property on the geometry edge.

St7GetGeometryEdgeCluster

Returns the cluster type and ID on the edge. After automeshing, all nodes along edges with the same cluster type and ID are connected according to the parameters.

```
long St7GetGeometryEdgeCluster(long uID, long EdgeNum, long* ClusterID,
     long* Entity, long* EntityType, long* OriginCode, double* Origin)
```

Input Parameters

`uID`

Strand7 model file ID.

`EdgeNum`

Edge number.

Output Parameters

`ClusterID`

Cluster ID.

`Entity`

Either tyBEAM to create a beam cluster or tyLINK to create a multi-point link.

`EntityType`

For beams, the property number of beams to be created.

For links, one of ltInterpolatedMultiPointLink, ltMasterSlaveMultiPointLink, ltPinnedMultiPointLink or ltRigidMultiPointLink.

`OriginCode`

Either coAutoClusterOrigin or coManualClusterOrigin.

`Origin[0..2]`

Origin position in global XYZ coordinates when OriginCode is coManualClusterOrigin.

Coedge Attributes – Set

St7SetGeometryCoedgeRelease1

Sets the release condition on the specified geometry coedge.

```
long St7SetGeometryCoedgeRelease1(long uID, long CoedgeNum, long* Status)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`Status`

Either prReleased or prFixed.

St7SetGeometryCoedgeSupport4

Assigns an elastic support to the specified geometry coedge.

```
long St7SetGeometryCoedgeSupport4(long uID, long CoedgeNum, long CaseNum,
     long* Status, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Freedom case number.

`Status[0..1]`

`[0]` – Compression-only support; either btTrue or btFalse.

`[1]` – Limited bearing capacity; either btTrue or btFalse.

`Doubles[0..3]`

`[0]` – Elastic support value in the normal direction.

`[1]` – Elastic support value in the lateral direction.

`[2]` – Support gap. Only relevant if Status[0] is set to btTrue.

`[3]` – Bearing capacity. Only relevant if Status[1] is set to btTrue.

St7SetGeometryCoedgePressure1

Assigns a normal pressure to the specified geometry coedge.

```
long St7SetGeometryCoedgePressure1(long uID, long CoedgeNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Load case number.

`Doubles[0]`

Edge pressure value.

St7SetGeometryCoedgePressure3

Assigns a global pressure to the specified geometry coedge.

```
long St7SetGeometryCoedgePressure3(long uID, long CoedgeNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Load case number.

`Doubles[0..2]`

A 3-element array containing the pressure components in the global XYZ system.

St7SetGeometryCoedgeShear1

Assigns a shear stress along the specified geometry coedge.

```
long St7SetGeometryCoedgeShear1(long uID, long CoedgeNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Load case number.

`Doubles[0]`

Shear stress.

St7SetGeometryCoedgeTransverseShear1

Assigns a transverse shear stress to the specified geometry coedge.

```
long St7SetGeometryCoedgeTransverseShear1(long uID, long CoedgeNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Load case number.

`Doubles[0]`

Transverse shear stress.

St7SetGeometryCoedgeConvection2

Assigns a thermal convection coefficient and ambient temperature to the specified geometry coedge.

```
long St7SetGeometryCoedgeConvection2(long uID, long CoedgeNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Load case number.

`Doubles[0..1]`

`[0]` – Convection coefficient.

`[1]` – Ambient temperature.

St7SetGeometryCoedgeConvectionTables

Sets the tables associated with the convection coefficient assigned to the specified geometry coedge.

```
long St7SetGeometryCoedgeConvectionTables(long uID, long CoedgeNum, long CaseNum,
    long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Load case number.

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the convection ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the convection coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the convection coefficient, or 0 for none.

St7SetGeometryCoedgeRadiation2

Assigns a thermal radiation coefficient and ambient temperature to the specified geometry coedge.

```
long St7SetGeometryCoedgeRadiation2(long uID, long CoedgeNum, long CaseNum,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Load case number.

`Doubles[0..1]`

`[0]` – Radiation coefficient.

`[1]` – Ambient temperature.

St7SetGeometryCoedgeRadiationTables

Sets the tables associated with the radiation coefficient assigned to the specified geometry coedge.

```
long St7SetGeometryCoedgeRadiationTables(long uID, long CoedgeNum, long CaseNum,
     long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Load case number.

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the radiation ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the radiation coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the radiation coefficient, or 0 for none.

St7SetGeometryCoedgeFlux1

Assigns a heat flux to the specified geometry coedge.

```
long St7SetGeometryCoedgeFlux1(long uID, long CoedgeNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Load case number.

`Doubles[0]`

The heat flux through the edge.

St7SetGeometryCoedgeFluxTables

Sets the tables associated with the heat flux attribute assigned to the specified geometry coedge.

```
long St7SetGeometryCoedgeFluxTables(long uID, long CoedgeNum, long CaseNum,
     long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Load case number.

`Tables[0..1]`

`[0]` – Factor vs Time table ID associated with the heat flux, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the heat flux, or 0 for none.

St7SetGeometryCoedgeAttachment1

Assigns an attachment attribute to the specified geometry coedge. Attachment attributes can be used to generate attachment links using the *St7CreateAttachments* function.

```
long St7SetGeometryCoedgeAttachment1(long uID, long CoedgeNum, long Direction,
     long AttachType, long ConnectType, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`Direction`

Direction of attachment; one of adPlanar, adPlusZ or adMinusZ.

`AttachType`

One of alDirect, alRigid or alFlexible.

`ConnectType`

Attachment sub-type; either alMoment or alPinned.

`PropNum`

Beam property number used for flexible attachment types.

`Doubles[0]`

The maximum distance within which the edge can be connected to another element using the attachment link.

Coedge Attributes – Get

St7GetGeometryCoedgeRelease1

Returns the edge release conditions assigned to the specified geometry coedge.

```
long St7GetGeometryCoedgeRelease1(long uID, long CoedgeNum, long* Status)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

Output Parameters

`Status`

Either prReleased or prFixed.

St7GetGeometryCoedgeSupport4

Returns the elastic edge support assigned to the specified geometry coedge. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryCoedgeSupport4(long uID, long CoedgeNum, long CaseNum,
    long* Status, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Freedom case number.

Output Parameters

`Status[0..1]`

`[0]` – Compression-only support; either btTrue or btFalse.

`[1]` – Limited bearing capacity; either btTrue or btFalse.

`Doubles[0..3]`

`[0]` – Elastic support value in the normal direction.

`[1]` – Elastic support value in the lateral direction.

[2] – Support gap. Only relevant if Status[0] is set to btTrue.

[3] – Bearing capacity. Only relevant if Status[1] is set to btTrue.

St7GetGeometryCoedgePressure1

Returns the normal edge pressure applied to the specified geometry coedge. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryCoedgePressure1(long uID, long CoedgeNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0]`

Edge pressure value.

St7GetGeometryCoedgeShear1

Returns the shear stress applied along the specified geometry coedge. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryCoedgeShear1(long uID, long CoedgeNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0]`

Shear stress.

St7GetGeometryCoedgeTransverseShear1

Returns the transverse shear stress assigned to the specified geometry coedge. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryCoedgeTransverseShear1(long uID, long CoedgeNum, long CaseNum,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0]`

Transverse shear stress.

St7GetGeometryCoedgePressure3

Returns the global edge pressure applied to the specified geometry coedge. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryCoedgePressure3(long uID, long CoedgeNum, long CaseNum,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0..2]`

A 3-element array containing the pressure components in the global XYZ system.

St7GetGeometryCoedgeConvection2

Returns the thermal convection coefficient and ambient temperature assigned to the specified geometry coedge. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryCoedgeConvection2(long uID, long CoedgeNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0..1]`

`[0]` – Convection coefficient.

`[1]` – Ambient temperature.

St7GetGeometryCoedgeConvectionTables

Returns the tables associated with the convection coefficient assigned to the specified geometry coedge. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryCoedgeConvectionTables(long uID, long CoedgeNum, long CaseNum,
     long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Load case number.

Output Parameters

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the convection ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the convection coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the convection coefficient, or 0 for none.

St7GetGeometryCoedgeRadiation2

Returns the thermal radiation coefficient and ambient temperature assigned to the specified geometry coedge. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryCoedgeRadiation2(long uID, long CoedgeNum, long CaseNum,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0..1]`

`[0]` – Radiation coefficient.

`[1]` – Ambient temperature.

St7GetGeometryCoedgeRadiationTables

Returns the tables associated with the radiation coefficient assigned to the specified geometry coedge. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryCoedgeRadiationTables(long uID, long CoedgeNum, long CaseNum,
    long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Load case number.

Output Parameters

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the radiation ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the radiation coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the radiation coefficient, or 0 for none.

St7GetGeometryCoedgeFlux1

Returns the heat flux assigned to the specified geometry coedge. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryCoedgeFlux1(long uID, long CoedgeNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0]`

The heat flux through the edge.

St7GetGeometryCoedgeFluxTables

Returns the tables associated with the heat flux assigned to the specified geometry coedge. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryCoedgeFluxTables(long uID, long CoedgeNum, long CaseNum,
     long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`CaseNum`

Load case number.

Output Parameters

`Tables[0..1]`

`[0]` – Factor vs Time table ID associated with the heat flux, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the heat flux, or 0 for none.

St7GetGeometryCoedgeAttachment1

Returns the edge attachment attribute assigned to the specified geometry coedge. Attachment attributes can be used to generate attachment links using the *St7CreateAttachments* function.

```
long St7GetGeometryCoedgeAttachment1(long uID, long CoedgeNum, long Direction,
     long* AttachType, long* ConnectType, long* PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CoedgeNum`

Coedge number.

`Direction`

One of adPlanar, adPlusZ or adMinusZ.

Output Parameters

`AttachType`

One of alDirect, alRigid or alFlexible.

`ConnectType`

Attachment sub-type; either alMoment or alPinned.

`PropNum`

Beam property number used for flexible attachment types.

`Doubles[0]`

The maximum distance within which the brick face can be connected to another element using the attachment link.

Face Attributes – Set

St7SetGeometryFaceProperty

Assigns a property number to the specified geometry face.

```
long St7SetGeometryFaceProperty(long uID, long FaceNum, long PropNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`PropNum`

Property number.

St7SetGeometryFaceID

Assigns an ID number to the specified geometry face.

```
long St7SetGeometryFaceID(long uID, long FaceNum, long FaceID)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`FaceID`

Face ID.

St7SetGeometryFaceThickness2

Assigns a thickness to the specified geometry face. This value is constant over the surface.

```
long St7SetGeometryFaceThickness2(long uID, long FaceNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`Doubles[0..1]`

`[0]` – Membrane thickness.

`[1]` – Bending thickness.

St7SetGeometryFaceOffset1

Assigns an offset to the specified geometry face. This value is constant over the surface.

```
long St7SetGeometryFaceOffset1(long uID, long FaceNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`Doubles[0]`

Offset value.

St7SetGeometryFaceSupport4

Assigns an elastic support condition to the specified geometry face.

```
long St7SetGeometryFaceSupport4(long uID, long FaceNum, long Surface,
     long CaseNum, long* Status, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`Surface`

Face surface; either psPlateMinusZ or psPlatePlusZ.

`CaseNum`

Freedom case number.

`Status[0..1]`

`[0]` – Compression-only support; either btTrue or btFalse.

`[1]` – Limited bearing capacity; either btTrue or btFalse.

`Doubles[0..3]`

`[0]` – Elastic support value in the normal direction.

`[1]` – Elastic support value in the lateral direction.

`[2]` – Support gap. Only relevant if Status[0] is set to btTrue.

`[3]` – Bearing capacity. Only relevant if Status[1] is set to btTrue.

St7SetGeometryFaceTempGradient1

Assigns a temperature gradient to the specified geometry face.

```
long St7SetGeometryFaceTempGradient1(long uID, long FaceNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`CaseNum`

Load case number.

`Doubles[0]`

Temperature gradient.

St7SetGeometryFaceNormalPressure2

Assigns normal pressure attributes to the specified geometry face. Two pressures are specified, corresponding to the normal pressure attributes eventually inherited by automeshed plates on their outer +z and -z surfaces.

```
long St7SetGeometryFaceNormalPressure2(long uID, long FaceNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`CaseNum`

Load case number.

`Doubles[0..1]`

`[0]` – Normal face pressure over the -Z surface of plates automeshed from the face.

`[1]` – Normal face pressure over the +Z surface of plates automeshed from the face.

Positive pressure is directed from the relevant plate surface into the plate.

St7SetGeometryFaceGlobalPressure3S

Assigns components of face pressure to the specified geometry face surface in the global XYZ system.

```
long St7SetGeometryFaceGlobalPressure3S(long uID, long FaceNum, long Surface,
     long ProjectFlag, long CaseNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`Surface`

Plate surface that inherits the attribute; either psPlateMinusZ or psPlatePlusZ.

`ProjectFlag`

One of ppNone, ppProjResultant or ppProjComponents.

`CaseNum`

Load case number.

`Doubles[0..2]`

A 3-element array containing the pressure components in the global XYZ system.

St7SetGeometryFaceNSMass5ID

Assigns a non-structural mass to the specified geometry face.

```
long St7SetGeometryFaceNSMass5ID(long uID, long FaceNum, long CaseNum, long ID,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`CaseNum`

Load case number.

`ID`

Non-structural mass ID.

`Doubles[0..4]`

`[0]` – Non-structural mass for the specified face.

`[1]` – Dynamic factor for the specified face. This factor is used to scale the non-structural mass when performing dynamic analysis.

`[2..4]` – A 3-element array describing the offset in the global XYZ system.

St7SetGeometryFaceConvection2

Assigns the thermal convection coefficient and ambient temperature for the specified geometry face.

```
long St7SetGeometryFaceConvection2(long uID, long FaceNum, long CaseNum,
     long Surface, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`CaseNum`

Load case number.

`Surface`

Local plate surface; either psPlateMinusZ or psPlatePlusZ.

`Doubles[0..1]`

`[0]` – Convection coefficient.

`[1]` – Ambient temperature.

St7SetGeometryFaceConvectionTables

Sets the tables associated with the convection coefficient assigned to the specified geometry face surface.

```
long St7SetGeometryFaceConvectionTables(long uID, long FaceNum, long CaseNum,
     long Surface, long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`CaseNum`

Load case number.

`Surface`

Local plate surface; either psPlateMinusZ or psPlatePlusZ.

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the convection ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the convection coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the convection coefficient, or 0 for none.

St7SetGeometryFaceRadiation2

Assigns the thermal radiation coefficient and ambient temperature for the specified geometry face.

```
long St7SetGeometryFaceRadiation2(long uID, long FaceNum, long CaseNum,
    long Surface, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`CaseNum`

Load case number.

`Surface`

Local plate surface; either psPlateMinusZ or psPlatePlusZ.

`Doubles[0..1]`

`[0]` – Radiation coefficient.

`[1]` – Ambient temperature.

St7SetGeometryFaceRadiationTables

Specifies the tables associated with the radiation coefficient assigned to the specified geometry face surface.

```
long St7SetGeometryFaceRadiationTables(long uID, long FaceNum, long CaseNum,
    long Surface, long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`CaseNum`

Load case number.

`Surface`

Local plate surface; either psPlateMinusZ or psPlatePlusZ.

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the radiation ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the radiation coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the radiation coefficient, or 0 for none.

St7SetGeometryFaceHeatSource1

Assigns a heat source to the specified geometry face.

```
long St7SetGeometryFaceHeatSource1(long uID, long FaceNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`CaseNum`

Load case number.

`Doubles[0]`

Heat source.

St7SetGeometryFaceHeatSourceTables

Specifies the tables associated with the heat source assigned to the specified geometry face.

```
long St7SetGeometryFaceHeatSourceTables(long uID, long FaceNum, long CaseNum,
     long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`CaseNum`

Load case number.

`Tables[0..1]`

`[0]` – Factor vs Time table ID associated with the heat source, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the heat source, or 0 for none.

St7SetGeometryFaceAttachment1

Assigns an attachment condition to the specified geometry face. Attachment attributes can be sued to generate attachment links using the *St7CreateAttachments* function.

```
long St7SetGeometryFaceAttachment1(long uID, long FaceNum, long Surface,
     long AttachType, long ConnectType, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`Surface`

Local plate surface; either psPlateMinusZ or psPlatePlusZ.

`AttachType`

One of alDirect, alRigid or alFlexible.

`ConnectType`

Attachment sub-type; either alMoment or alPinned.

`PropNum`

Beam property number used for flexible attachment types.

`Doubles[0]`

The maximum distance within which the face can be connected to another element using the attachment link.

Face Attributes – Get

St7GetGeometryFaceProperty

Returns the property assigned to the specified geometry face.

```
long St7GetGeometryFaceProperty(long uID, long FaceNum, long* PropNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

Output Parameters

`PropNum`

Face property number.

St7GetGeometryFaceID

Returns the ID number assigned to the specified geometry face.

```
long St7GetGeometryFaceID(long uID, long FaceNum, long* FaceID)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

Output Parameters

`FaceID`

Face ID.

St7GetGeometryFaceThickness2

Returns the thickness assigned to the specified geometry face.

```
long St7GetGeometryFaceThickness2(long uID, long FaceNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

Output Parameters

`Doubles[0..1]`

`[0]` – Membrane thickness.

`[1]` – Bending thickness.

St7GetGeometryFaceOffset1

Returns the offset assigned to the specified geometry face.

```
long St7GetGeometryFaceOffset1(long uID, long FaceNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

Output Parameters

`Doubles[0]`

Offset value.

St7GetGeometryFaceSupport4

Returns the elastic support condition assigned to the specified geometry face. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryFaceSupport4(long uID, long FaceNum, long Surface,
     long CaseNum, long* Status, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`Surface`

Face surface; either psPlateMinusZ or psPlatePlusZ.

`CaseNum`

Freedom case number.

Output Parameters

`Status[0..1]`

`[0]` – Compression-only support; either btTrue or btFalse.

`[1]` – Limited bearing capacity; either btTrue or btFalse.

`Doubles[0..3]`

`[0]` – Elastic support value in the normal direction.

`[1]` – Elastic support value in the lateral direction.

`[2]` – Support gap. Only relevant if Status[0] is set to btTrue.

`[3]` – Bearing capacity. Only relevant if Status[1] is set to btTrue.

St7GetGeometryFaceTempGradient1

Returns the temperature gradient assigned to the specified geometry face. This attribute is only used when performing structural analysis. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryFaceTempGradient1(long uID, long FaceNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0]`

Temperature gradient.

St7GetGeometryFaceNormalPressure2

Returns the normal pressure attributes assigned to the specified geometry face. Two pressures are returned, corresponding to the normal pressure attributes eventually inherited by automeshed plates on their outer +z and -z surfaces. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryFaceNormalPressure2(long uID, long FaceNum, long CaseNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0..1]`

`[0]` – Normal face pressure over the -Z surface of plates automeshed from the face.

`[1]` – Normal face pressure over the +Z surface of plates automeshed from the face.

Positive pressure is directed from the relevant plate surface into the plate.

St7GetGeometryFaceGlobalPressure3S

Returns the components of face pressure applied to the specified geometry face surface in the global XYZ system. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryFaceGlobalPressure3S(long uID, long FaceNum, long Surface,
    long CaseNum, long* ProjectFlag, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`Surface`

Plate surface that inherits the attribute; either psPlateMinusZ or psPlatePlusZ.

`CaseNum`

Load case number.

Output Parameters

`ProjectFlag`

One of ppNone, ppProjResultant or ppProjComponents.

`Doubles[0..2]`

A 3-element array containing the pressure components in the global XYZ system.

St7GetGeometryFaceNSMass5ID

Returns the non-structural mass assigned to the specified geometry face. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryFaceNSMass5ID(long uID, long FaceNum, long CaseNum, long ID,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`CaseNum`

Load case number.

`ID`

Non-structural mass ID.

Output Parameters

`Doubles[0..4]`

`[0]` – Non-structural mass for the specified face.

`[1]` – Dynamic factor for the specified face. This factor is used to scale the non-structural mass when performing dynamic analysis.

`[2..4]` – A 3-element array describing the offset in the global XYZ system.

St7GetGeometryFaceConvection2

Returns the thermal convection coefficient and ambient temperature assigned to the specified geometry face. This attribute is only used when performing heat transfer analysis. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryFaceConvection2(long uID, long FaceNum, long CaseNum,
     long Surface, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`CaseNum`

Load case number.

`Surface`

Local plate surface; either psPlateMinusZ or psPlatePlusZ.

Output Parameters

`Doubles[0..1]`

`[0]` – Convection coefficient.

`[1]` – Ambient temperature.

St7GetGeometryFaceConvectionTables

Returns the tables associated with the convection coefficient assigned to the specified geometry face surface. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryFaceConvectionTables(long uID, long FaceNum, long CaseNum,
     long Surface, long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`CaseNum`

Load case number.

`Surface`

Local plate surface; either psPlateMinusZ or psPlatePlusZ.

Output Parameters

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the convection ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the convection coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the convection coefficient, or 0 for none.

St7GetGeometryFaceRadiation2

Returns the thermal radiation coefficient and ambient temperature assigned to the specified geometry face. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryFaceRadiation2(long uID, long FaceNum, long CaseNum,
     long Surface, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`CaseNum`

Load case number.

`Surface`

Local plate surface; either psPlateMinusZ or psPlatePlusZ.

Output Parameters

`Doubles[0..1]`

`[0]` – Radiation coefficient.

`[1]` – Ambient temperature.

St7GetGeometryFaceRadiationTables

Returns the tables associated with the radiation coefficient assigned to the specified geometry face surface. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryFaceRadiationTables(long uID, long FaceNum, long CaseNum,
     long Surface, long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`CaseNum`

Load case number.

`Surface`

Local plate surface; either psPlateMinusZ or psPlatePlusZ.

Output Parameters

`Tables[0..2]`

`[0]` – Temperature vs Time table ID associated with the radiation ambient temperature, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the radiation coefficient, or 0 for none.

`[2]` – Factor vs Time table ID associated with the radiation coefficient, or 0 for none.

St7GetGeometryFaceHeatSource1

Returns the heat source assigned to the specified geometry face. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryFaceHeatSource1(long uID, long FaceNum, long CaseNum,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`CaseNum`

Load case number.

Output Parameters

`Doubles[0]`

Heat source.

St7GetGeometryFaceHeatSourceTables

Returns the tables associated with the heat source assigned to the specified geometry face. See also *St7GetEntityAttributeSequenceCount* and *St7GetEntityAttributeSequence*.

```
long St7GetGeometryFaceHeatSourceTables(long uID, long FaceNum, long CaseNum,
    long* Tables)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`CaseNum`

Load case number.

Output Parameters

`Tables[0..1]`

`[0]` – Factor vs Time table ID associated with the heat source, or 0 for none.

`[1]` – Factor vs Temperature table ID associated with the heat source, or 0 for none.

St7GetGeometryFaceAttachment1

Returns the attachment conditions assigned to the specified geometry face. Attachment attributes can be used to generate attachment links using the *St7CreateAttachments* function.

```
long St7GetGeometryFaceAttachment1(long uID, long FaceNum, long Surface,
     long* AttachType, long* ConnectType, long* PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`Surface`

Local plate surface; either psPlateMinusZ or psPlatePlusZ.

Output Parameters

`AttachType`

One of alDirect, alRigid or alFlexible.

`ConnectType`

Attachment sub-type; either alMoment or alPinned.

`PropNum`

Beam property number used for flexible attachment types.

`Doubles[0]`

The maximum distance within which the face can be connected to another element using the attachment link.

Attributes General

St7SetElementProperty

Sets the property for the specified element. The property does not need to be created in advance.

```
long St7SetElementProperty(long uID, long Entity, long EntityNum, long PropNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP or ptBRICKPROP.

`EntityNum`

Entity number.

`PropNum`

Property number.

St7GetElementProperty

Returns the property assigned to the specified element.

```
long St7GetElementProperty(long uID, long Entity, long EntityNum, long* PropNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP or ptBRICKPROP.

`EntityNum`

Entity number.

Output Parameters

`PropNum`

Property number.

St7SetElementPropertySwitch

Specifies a property switch for a staged analysis.

```
long St7SetElementPropertySwitch(long uID, long Entity, long EntityNum,
     long PropID, long Stage)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP or ptBRICKPROP.

`EntityNum`

Entity number.

`PropID`

Property number, as either a positive or negative integer.

`Stage`

Stage index.

Usage

A positive PropID signifies that the element is re-born at the switched stage (i.e., its birth stage becomes the stage index associated with the property switch). A negative PropID signifies that the element changes its property type but its birth stage is inherited from the current element (i.e., the Inherited Birth Stage option is set).

St7GetElementPropertySequence

Returns the stage-dependent property sequence assigned to the specified element for staged analysis. *St7GetNumStages* can be used to determine the number of stages in the model.

```
long St7GetElementPropertySequence(long uID, long Entity, long EntityNum,
     long MaxProps, long* Props)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP or ptBRICKPROP.

`EntityNum`

Entity number.

`MaxProps`

Maximum amount of storage allocated for the Props array.

Output Parameters

`Props[0..MaxProps-1]`

An array containing the property number assigned at each stage of the analysis. Property numbers can be either positive or negative, depending on the Inherited Birth Stage setting.

Usage

A positive property number signifies that the element is re-born at the switched stage (i.e., its birth stage becomes the stage index associated with the property switch). A negative PropID signifies that the element changes its property type but its birth stage is inherited from the current element (i.e., the Inherited Birth Stage option is set).

St7DeleteAttribute

Deletes the specified attribute. See *Attribute Types* for additional information.

```
long St7DeleteAttribute(long uID, long Entity, long EntityNum, long AttributeOrd,
    long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyVERTEX, tyGEOMETRYEDGE, tyGEOMETRYCOEDGE, tyGEOMETRYFACE or tyLOADPATH.

`EntityNum`

Entity number.

`AttributeOrd`

Attribute identifier; see *Attribute Types*.

`Integers[0..3]`

`[ipAttrLocal]` – Local attribute number; see *Attribute Types*.

`[ipAttrAxis]` – Axis or direction of the attribute; see *Attribute Types*.

`[ipAttrCase]` – Attribute load/freedom case number; see *Attribute Types*.

`[ipAttrID]` – Attribute ID number; see *Attribute Types*.

St7SetEntityGroup

Assigns the specified entity to a given group.

```
long St7SetEntityGroup(long uID, long Entity, long EntityNum, long GroupID)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK, tyLINK, tyGEOMETRYFACE or tyLOADPATH.

`EntityNum`

Entity number.

`GroupID`

Group ID.

St7GetEntityGroup

Returns the group number assigned to the specified entity.

```
long St7GetEntityGroup(long uID, long Entity, long EntityNum, long* GroupID)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK, tyLINK, tyGEOMETRYFACE or tyLOADPATH.

`EntityNum`

Entity number.

Output Parameters

`GroupID`

Group ID.

St7GetEntityAttributeSequenceCount

Returns the number of instances of the attribute on the requested entity.

```
long St7GetEntityAttributeSequenceCount(long uID, long Entity, long EntityNum,
     long AttributeOrd, long* NumSets)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyVERTEX, tyGEOMETRYEDGE or tyGEOMETRYFACE.

`EntityNum`

Entity number.

`AttributeOrd`

Attribute identifier; see *Attribute Types*.

Output Parameters

`NumSets`

Number of instances of the attribute on the element. Use *St7GetEntityAttributeSequence* to get details of each instance.

St7GetEntityAttributeSequence

Returns details of the attribute instances on the requested entity.

```
long St7GetEntityAttributeSequence(long uID, long Entity, long EntityNum,
     long AttributeOrd, long MaxSets, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyNODE, tyBEAM, tyPLATE, tyBRICK, tyVERTEX, tyGEOMETRYEDGE or tyGEOMETRYFACE.

`EntityNum`

Entity number.

`AttributeOrd`

Attribute identifier; see *Attribute Types*.

`MaxSets`

Number of instances of the attribute for which space has been allocated in Integers. Each attribute instance requires four integers.

Output Parameters

`Integers[0..4*NumSets]`

For each `i` attribute instance,

`[4*i+ ipAttrLocal]` – Local number if applicable.
`[4*i+ ipAttrAxis]` – Axis or direction if applicable.
`[4*i+ ipAttrCase]` – Load or freedom case number if applicable.
`[4*i+ ipAttrID]` – ID number (1-192) if applicable.

For the interpretation of the four values with respect to the specific attribute types; see *Attribute Types*.

Use *St7GetEntityAttributeSequenceCount* to determine the number of attribute instances that will be returned as NumSets, then ensure the length of the Integers array is at least 4*NumSets.

Properties – Beams, Plates and Bricks

St7GetTotalProperties

Returns the total number and highest property index for each of the Strand7 property types in the specified model.

```
long St7GetTotalProperties(long uID, long* NumProperties, long* LastProperty)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumProperties[0..kMaxEntityTotals-1]`

`[ipBeamPropTotal]` – the total number of beam property types.

`[ipPlatePropTotal]` – the total number of plate property types.

`[ipBrickPropTotal]` – the total number of brick property types.

`[ipPlyPropTotal]` – the total number of ply property types.

`LastProperty[0..kMaxEntityTotals-1]`

`[ipBeamPropTotal]` – the highest beam property number.

`[ipPlatePropTotal]` – the highest plate property number.

`[ipBrickPropTotal]` – the highest brick property number.

`[ipPlyPropTotal]` – the highest ply property number.

St7GetPropertyNumByIndex

Returns the property number associated with the specified property index. The property indices are stored internally and are based on a contiguous numbering system.

```
long St7GetPropertyNumByIndex(long uID, long Entity, long PropIndex,
     long* PropNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP, ptBRICKPROP or ptPLYPROP.

`PropIndex`

Property index position.

Output Parameters

`PropNum`

Property number.

St7SetPropertyName

Sets the name of the specified property.

```
long St7SetPropertyName(long uID, long Entity, long PropNum, char* PropName)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP, ptBRICKPROP or ptPLYPROP.

`PropNum`

Property number.

`PropName`

Name of the property.

St7GetPropertyName

Returns the name of the specified property.

```
long St7GetPropertyName(long uID, long Entity, long PropNum, char* PropName,
     long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP, ptBRICKPROP or ptPLYPROP.

`PropNum`

Property number.

`MaxStringLen`

Maximum number of characters allocated for PropName.

Output Parameters

`PropName`

Name of the property.

St7SetPropertyColour

Sets the colour of the specified property.

```
long St7SetPropertyColour(long uID, long Entity, long PropNum, long PropCol)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP, ptBRICKPROP or ptPLYPROP.

`PropNum`

Property number.

`PropCol`

Property colour. See also *RGB Colours*.

St7GetPropertyColour

Returns the colour assigned to the specified property.

```
long St7GetPropertyColour(long uID, long Entity, long PropNum, long* PropCol)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP, ptBRICKPROP or ptPLYPROP.

`PropNum`

Property number.

Output Parameters

`PropCol`

Property colour. See also *RGB Colours*.

St7SetPropertyTable

Assigns a table to the specified property.

```
long St7SetPropertyTable(long uID, long PropTableType, long PropNum,
     long TableID)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropTableType`

Property table type; see *Table Types*.

`PropNum`

Property number.

`TableID`

Table ID number, or 0 for none.

St7GetPropertyTable

Returns a table assigned to the specified property.

```
long St7GetPropertyTable(long uID, long PropTableType, long PropNum,
     long* TableID)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropTableType`

Property table type; see *Table Types*.

`PropNum`

Property number.

Output Parameters

`TableID`

Table ID number, or 0 for none.

St7SetPropertyCreepID

Assigns the creep definition to the specified property.

```
long St7SetPropertyCreepID(long uID, long Entity, long PropNum, long CreepID)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP or ptBRICKPROP.

`PropNum`

Property number.

`CreepID`

Creep layout ID or 0 for none.

St7GetPropertyCreepID

Returns the creep definition of the specified property.

```
long St7GetPropertyCreepID(long uID, long Entity, long PropNum, long* CreepID)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP or ptBRICKPROP.

`PropNum`

Property number.

Output Parameters

`CreepID`

Creep layout ID or 0 for none.

St7SetMaterialName

Sets the name of the material referenced by the specified property.

```
long St7SetMaterialName(long uID, long Entity, long PropNum, char* MaterialName)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP, ptBRICKPROP or ptPLYPROP.

`PropNum`

Property number.

`MaterialName`

Name of the material.

St7GetMaterialName

Returns the name of the material referenced by the specified property.

```
long St7GetMaterialName(long uID, long Entity, long PropNum, char* MaterialName,
     long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP, ptBRICKPROP or ptPLYPROP.

`PropNum`

Property number.

`MaxStringLen`

Maximum number of characters allocated for MaterialName.

Output Parameters

`MaterialName`

Name of the material.

St7SetHardeningType

Sets the hardening model used for the specified property.

```
long St7SetHardeningType(long uID, long Entity, long PropNum, long HardType)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

Property type; only ptBEAMPROP is valid.

`PropNum`

Property number.

`HardType`

One of htIsotropic, htKinematic or htTakeda.

St7GetHardeningType

Returns the hardening model used for the specified property.

```
long St7GetHardeningType(long uID, long Entity, long PropNum, long* HardType)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

Property type; only ptBEAMPROP is valid.

`PropNum`

Property number.

Output Parameters

`HardType`

One of htIsotropic, htKinematic or htTakeda.

St7SetTimeDependentModType

Sets the type of temperature/time dependence for the specified property. This setting controls the scaling used to update the material modulus values. This option is only used when an associated Factor vs Temperature/Time table is assigned to the specified property.

```
long St7SetTimeDependentModType(long uID, long Entity, long PropNum,
    long ModType)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP or ptBRICKPROP.

`PropNum`

Property number.

`ModType`

Type of temperature/time dependence; either mtElastic or mtPlastic.

St7GetTimeDependentModType

Returns the type of temperature/time dependence assigned to the specified property. This setting controls the scaling used to update the material modulus values. This option is only used when an associated Factor vs Temperature/Time table is assigned to the specified property.

```
long St7GetTimeDependentModType(long uID, long Entity, long PropNum,
     long* ModType)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP or ptBRICKPROP.

`PropNum`

Property number.

Output Parameters

`ModType`

Type of temperature/time dependence; either mtElastic or mtPlastic.

St7SetPropertyRayleighFactors

Sets the Rayleigh damping parameters for the specified property.

```
long St7SetPropertyRayleighFactors(long uID, long Entity, long PropNum,
     long RayleighMode, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP or ptBRICKPROP.

`PropNum`

Property number.

`RayleighMode`

Type of Rayleigh factors specified; either rmSetFrequencies or rmSetAlphaBeta.

`Doubles[0..5]`

rmSetAlphaBeta:

`[ipRayleighAlpha]` – Alpha.

`[ipRayleighBeta]` – Beta.

rmSetFrequencies:

`[ipRayleighF1]` – Rayleigh damping frequency 1.

`[ipRayleighF2]` – Rayleigh damping frequency 2.

`[ipRayleighR1]` – Rayleigh damping ratio 1.

[ipRayleighR2] – Rayleigh damping ratio 2.

Both:

[ipRayleighDisplayF1] – Rayleigh damping graph display frequency 1.

[ipRayleighDisplayF2] – Rayleigh damping graph display frequency 2.

St7GetPropertyRayleighFactors

Returns the Rayleigh damping parameters for the specified property.

```
long St7GetPropertyRayleighFactors(long uID, long Entity, long PropNum,
    long* RayleighMode, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP or ptBRICKPROP.

`PropNum`

Property number.

Output Parameters

`RayleighMode`

Type of Rayleigh factors specified; either rmSetFrequencies or rmSetAlphaBeta.

`Doubles[0..5]`

rmSetAlphaBeta:

[ipRayleighAlpha] – Alpha

[ipRayleighBeta] - Beta.

rmSetFrequencies:

[ipRayleighF1] – Rayleigh damping frequency 1.

[ipRayleighF2] – Rayleigh damping frequency 2.

[ipRayleighR1] – Rayleigh damping ratio 1.

[ipRayleighR2] – Rayleigh damping ratio 2.

Both:

[ipRayleighDisplayF1] – Rayleigh damping graph display frequency 1.

[ipRayleighDisplayF2] – Rayleigh damping graph display frequency 2.

St7SetAlphaTempType

Sets the thermal expansion table type for the specified property.

```
long St7SetAlphaTempType(long uID, long Entity, long PropNum, long AlphaTempType)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP or ptBRICKPROP.

`PropNum`

Property number.

`AlphaTempType`

Table type; either atIntegrated or atInstantaneous.

St7GetAlphaTempType

Returns the type of thermal expansion table assigned to the specified property.

```
long St7GetAlphaTempType(long uID, long Entity, long PropNum,
    long* AlphaTempType)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP or ptBRICKPROP.

`PropNum`

Property number.

Output Parameters

`AlphaTempType`

Table type; either atIntegrated or atInstantaneous.

St7NewBeamProperty

Creates a new beam property.

```
long St7NewBeamProperty(long uID, long PropNum, long BeamType, char* PropName)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Property number.

`BeamType`

One of btSpring, btCable, btTruss, btCutoff, btContact, btBeam, btUser, btPipe or btConnection.

`PropName`

Name of the property.

St7GetBeamPropertyData

Returns the specified beam property.

```
long St7GetBeamPropertyData(long uID, long PropNum, long* Integers,
    double* SectionData, double* MaterialData)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Property number.

Output Parameters

`Integers[0..4]`

`[ipBeamPropBeamType]` – Type of beam element; one of btNull, btSpring, btCable, btTruss, btCutoff, btContact, btBeam, btUser, btPipe or btConnection.

`[ipBeamPropUsePoisson]` – btTrue to use E and rho for material properties, btFalse to use E and G.

`[ipBeamPropSectionType]` – One of the section shapes defined in *Beam Cross Section Shape*.

`[ipBeamPropMirrorType]` – One of the section mirror types defined in *Beam Cross Section Mirror Options*.

`[ipBeamPropCompatibleTwist]` – Compatible twist option for mirrored sections; either btTrue or btFalse.

`SectionData[0..kNumBeamSectionData-1]`

`[ipAREA]` – Section area.

`[ipI11]` – Second moment of area about the principal 1 axis.

`[ipI22]` – Second moment of area about the principal 2 axis.

[ipJ] – Torsion constant.

[ipSL1] – Shear centre offset in the principal 1 axis direction.

[ipSL2] – Shear centre offset in the principal 2 axis direction.

[ipSA1] – Shear area in the principal 1 axis direction.

[ipSA2] – Shear area in the principal 2 axis direction.

[ipXBAR] – Centroid x coordinate.

[ipYBAR] – Centroid y coordinate.

[ipANGLE] – Principal axis 1 angle in radians w.r.t. the local section coordinates.

[ipD1] – Section geometry D1 parameter; see *Beam Cross Section Shape*.

[ipD2] – Section geometry D2 parameter.

[ipD3] – Section geometry D3 parameter.

[ipT1] – Section geometry T1 parameter.

[ipT2] – Section geometry T2 parameter.

[ipT3] – Section geometry T3 parameter.

[ipGapA] – Mirrored section gap parameter A.

[ipGapB] – Mirrored section gap parameter B.

MaterialData[0..kNumMaterialData-1]

[ipModulus] – Material modulus.

[ipPoisson] – Material Poisson's ratio.

[ipDensity] – Material density.

[ipShearModulus] – Material shear modulus.

Note that MaterialData values are as stored in the model, not as used by the solver. The solver will take into account Integers[ipBeamPropUsePoisson] to determine the material values to use.

St7SetBeamSectionName

Sets the section name referenced by the specified beam property.

```
long St7SetBeamSectionName(long uID, long PropNum, char* SectionName)
```

Input Parameters

uID

Strand7 model file ID.

PropNum

Property number.

`SectionName`

Name of the section.

St7GetBeamSectionName

Returns the name of the section referenced by the specified beam property.

```
long St7GetBeamSectionName(long uID, long PropNum, char* SectionName,
     long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Property number.

`MaxStringLen`

Maximum number of characters allocated for SectionName.

Output Parameters

`SectionName`

Name of the section.

St7SetBeamPropertyType

Sets the beam type for the specified beam property.

```
long St7SetBeamPropertyType(long uID, long PropNum, long BeamType)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Property number.

`BeamType`

One of btSpring, btCable, btTruss, btCutoff, btContact, btBeam, btUser, btPipe or btConnection.

St7GetBeamPropertyType

Returns the beam type for the specified beam property.

```
long St7GetBeamPropertyType(long uID, long PropNum, long* BeamType)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Property number.

Output Parameters

`BeamType`

One of btNull, btSpring, btCable, btTruss, btCutoff, btContact, btBeam, btUser, btPipe or btConnection.

St7SetBeamMirrorOption

Sets the section mirror type for the specified beam property.

```
long St7SetBeamMirrorOption(long uID, long PropNum, long MirrorType,
     long CompatibleTwist, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`MirrorType`

One of the section mirror types defined in *Beam Cross Section Mirror Options*.

`CompatibleTwist`

Compatible twist option for mirrored sections; either btTrue or btFalse.

`Doubles[0..1]`

A 2-element array containing the mirror gap parameters A and B respectively.

St7GetBeamMirrorOption

Returns the section mirror type for the specified beam property.

```
long St7GetBeamMirrorOption(long uID, long PropNum, long* MirrorType,
     long* CompatibleTwist, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

Output Parameters

`MirrorType`

One of the section mirror types defined in *Beam Cross Section Mirror Options*.

`CompatibleTwist`

Compatible twist option for mirrored sections; either btTrue or btFalse.

`Doubles[0..1]`

A 2-element array containing the mirror gap parameters A and B respectively.

St7SetBeamNonlinearType

Sets the nonlinear material type for the specified beam property.

```
long St7SetBeamNonlinearType(long uID, long PropNum, long NonlinType,
     long YieldType)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`NonlinType`

Nonlinear material type; either ntNonlinElastic or ntElastoPlastic.

`YieldType`

One of ycBeamFibre, ycBeamTresca or ycBeamVonMises.

St7GetBeamNonlinearType

Returns the nonlinear material type assigned to the specified beam property.

```
long St7GetBeamNonlinearType(long uID, long PropNum, long* NonlinType,
     long* YieldType)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

Output Parameters

`NonlinType`

Nonlinear material type; either ntNonlinElastic or ntElastoPlastic.

`YieldType`

One of ycBeamFibre, ycBeamTresca or ycBeamVonMises.

St7SetBeamSectionPropertyData

Sets the beam section property data for the specified beam property.

```
long St7SetBeamSectionPropertyData(long uID, long PropNum, long* Integers,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`Integers[0]`

Number of length-wise integration slices.

`Doubles[0..10]`

`[ipAREA]` – Section area.

`[ipI11]` – Second moment of area about the principal 1 axis.

`[ipI22]` – Second moment of area about the principal 2 axis.

`[ipJ]` – Torsion constant.

`[ipSL1]` – Shear centre offset in the principal 1 axis direction.

`[ipSL2]` – Shear centre offset in the principal 2 axis direction.

`[ipSA1]` – Shear area in the principal 1 axis direction.

`[ipSA2]` – Shear area in the principal 2 axis direction.

`[ipXBAR]` – Centroid x coordinate.

`[ipYBAR]` – Centroid y coordinate.

`[ipANGLE]` – Principal axis 1 angle in radians w.r.t. the local section coordinates.

St7GetBeamSectionPropertyData

Returns the beam section property data assigned to the specified beam property.

```
long St7GetBeamSectionPropertyData(long uID, long PropNum, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

Output Parameters

`Integers[0]`

Number of length-wise integration slices.

`Doubles[0..10]`

`[ipAREA]` – Section area.

`[ipI11]` – Second moment of area about the principal 1 axis.

`[ipI22]` – Second moment of area about the principal 2 axis.

`[ipJ]` – Torsion constant.

`[ipSL1]` – Shear centre offset in the principal 1 axis direction.

`[ipSL2]` – Shear centre offset in the principal 2 axis direction.

`[ipSA1]` – Shear area in the principal 1 axis direction.

`[ipSA2]` – Shear area in the principal 2 axis direction.

`[ipXBAR]` – Centroid x coordinate.

`[ipYBAR]` – Centroid y coordinate.

`[ipANGLE]` – Principal axis 1 angle in radians w.r.t. the local section coordinates.

St7SetBeamSectionGeometry

Sets the beam cross section geometry data for the specified beam property.

```
long St7SetBeamSectionGeometry(long uID, long PropNum, long SectionType,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`SectionType`

One of the section shapes defined in *Beam Cross Section Shape*.

`Doubles[0..5]`

A 6-element array containing the beam cross section D1, D2, D3, T1, T2 and T3 parameters respectively; see *Beam Cross Section Shape*.

St7GetBeamSectionGeometry

Returns the beam cross section data assigned to the specified beam property.

```
long St7GetBeamSectionGeometry(long uID, long PropNum, long* SectionType,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

Output Parameters

`SectionType`

One of the section shapes defined in *Beam Cross Section Shape*.

`Doubles[0..5]`

A 6-element array containing the beam cross section D1, D2, D3, T1, T2 and T3 parameters respectively; see *Beam Cross Section Shape*.

St7SetBeamSectionGeometryBGL

Sets the beam cross section geometry by specifying the dimensions of a cross section from the beam geometry library (BGL).

```
long St7SetBeamSectionGeometryBGL(long uID, long PropNum, long Shape,
     double* Dimensions)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`Shape`

One of bgNullSection, bgRectangularHollow, bgISection, bgChannel, bgTSection, bgAngle or bgBulbFlat.

`Dimensions[0..kMaxBGLDimensions-1]`

Dimensions that define the beam section geometry; see *Beam Geometry Library (BGL) Cross Section Shapes*.

St7GetBeamSectionGeometryBGL

Returns the values that define the geometry of a cross section from the beam geometry library (BGL) contained in the specified beam property.

```
long St7GetBeamSectionGeometryBGL(long uID, long PropNum, long* Shape,
     double* Dimensions)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

Output Parameters

`Shape`

One of bgNullSection, bgRectangularHollow, bgISection, bgChannel, bgTSection, bgAngle or bgBulbFlat.

`Dimensions[0..kMaxBGLDimensions-1]`

Dimensions that define the beam section geometry; see *Beam Geometry Library (BGL) Cross Section Shapes*.

St7SetBeamSectionNominalDiscretisation

Sets the discretisation values used when discretising the cross section for nonlinear beam types.

```
long St7SetBeamSectionNominalDiscretisation(long uID, long PropNum,
     long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`Integers[0..2]`

`[0]` – nominal divisions used to discretise the beam (along the longest ordinate), or divisions in the x ordinate when divisions are specified.

`[1]` – divisions in the y ordinate used to discretise the beam when divisions are specified.

`[2]` – btTrue to use nominal divisions, btFalse to specify the divisions in each ordinate explicitly.

St7GetBeamSectionNominalDiscretisation

Returns the discretisation values used when discretising the cross section for nonlinear beam types.

```
long St7GetBeamSectionNominalDiscretisation(long uID, long PropNum,
     long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

Output Parameters

`Integers[0..2]`

`[0]` – nominal divisions used to discretise the beam (along the longest ordinate), or divisions in the x ordinate when divisions are specified.

`[1]` – divisions in the y ordinate used to discretise the beam when divisions are specified.

`[2]` – btTrue to use nominal divisions, btFalse to specify the divisions in each ordinate explicitly.

St7SetBeamSectionCircularDiscretisation

Sets the number of circumferential divisions used when discretising circular cross sections for nonlinear beam types.

```
long St7SetBeamSectionCircularDiscretisation(long uID, long PropNum,
     long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`Integers[0]`

Number of circumferential divisions used to discretise the circular beam cross section; one of
0 – for **Auto** divisions
1 – for 8 divisions
2 – for 16 divisions
3 – for 24 divisions
4 – for 32 divisions
5 – for 40 divisions
6 – for 48 divisions.

St7GetBeamSectionCircularDiscretisation

Returns the number of circumferential divisions used when discretising circular cross sections for nonlinear beam types.

```
long St7GetBeamSectionCircularDiscretisation(long uID, long PropNum,
     long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

Output Parameters

`Integers[0]`

Number of circumferential divisions used to discretise the circular beam cross section; one of
0 – for **Auto** divisions
1 – for 8 divisions
2 – for 16 divisions
3 – for 24 divisions
4 – for 32 divisions
5 – for 40 divisions
6 – for 48 divisions.

St7CheckBeamSectionQuality

Checks the quality of the section discretisation in response to unit shear forces and torque. Values closer to 1.0 are more accurate. Section quality discretisation is applicable to MNL solves.

```
long St7CheckBeamSectionQuality(long uID, long PropNum, double* Shear1,
     double* Shear2, double* Torque)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

Output Parameters

`Shear1`

Unit shear 1 response.

`Shear2`

Unit shear 2 response.

`Torque`

Unit torque response.

St7SaveBeamSectionMesh

Creates a Strand7 model file with a plate mesh of the discretised beam section. The function is applicable to all of the cross section definition options (standard sections, beam section library (BSL) and beam geometry library (BGL)).

```
long St7SaveBeamSectionMesh(long uID, long PropNum, char* FileName)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`FileName`

Full path and filename for the Strand7 model containing the beam section discretisation.

St7CalculateBeamSectionProperties

Calculates the section properties for the standard beam cross sections defined in *Beam Cross Section Shape*, with the exception of bsNullSection. The function will return an error if the property does not use one of the standard section types.

```
long St7CalculateBeamSectionProperties(long uID, long PropNum, bool DoShear)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`DoShear`

Include the shear area values; either True or False. If the shear areas are included the “thick” beam formulation is used.

St7AssignBXS

Assigns a BXS to the specified beam property.

```
long St7AssignBXS(long uID, long PropNum, char* BXSName)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`BXSName`

Full path and filename of the pre-existing .bxs file to assign to the property.

St7ExportBXS

Exports a BXS contained in a beam property to the specified file.

```
long St7ExportBXS(long uID, long PropNum, char* BXSName)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`BXSName`

Full path and filename of the .bxs file.

St7SetSpringDamperData

Sets the spring-damper element parameters for the specified beam property.

```
long St7SetSpringDamperData(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`Doubles[0..6]`

`[ipSpringAxialStiff]` – Axial stiffness.

`[ipSpringLateralStiff]` – Lateral stiffness.

`[ipSpringTorsionStiff]` – Torsional stiffness.

`[ipSpringAxialDamp]` – Axial damping.

[ipSpringLateralDamp] – Lateral damping.

[ipSpringTorsionDamp] – Torsional damping.

[ipSpringMass] – Element mass.

St7GetSpringDamperData

Returns the spring-damper element parameters assigned to the specified beam property.

```
long St7GetSpringDamperData(long uID, long PropNum, double* Doubles)
```

Input Parameters

uID

Strand7 model file ID.

PropNum

Beam property number.

Output Parameters

Doubles[0..6]

[ipSpringAxialStiff] – Axial stiffness.

[ipSpringLateralStiff] – Lateral stiffness.

[ipSpringTorsionStiff] – Torsional stiffness.

[ipSpringAxialDamp] – Axial damping.

[ipSpringLateralDamp] – Lateral damping.

[ipSpringTorsionDamp] – Torsional damping.

[ipSpringMass] – Element mass.

St7SetCableData

Sets the cable element parameters for the specified beam property.

```
long St7SetCableData(long uID, long PropNum, long* Integers, double* Doubles)
```

Input Parameters

uID

Strand7 model file ID.

PropNum

Beam property number.

Integers[0..0]

[ipCablePreStrainScalesMass] – btTrue for pre-strain to scale the cable mass.

`Doubles[0..0]`

`[ipCableDiameter]` – Cable diameter.

St7GetCableData

Returns the cable element parameters for the specified beam property.

```
long St7GetCableData(long uID, long PropNum, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

Output Parameters

`Integers[0..0]`

`[ipCablePreStrainScalesMass]` – btTrue for pre-strain to scale the cable mass.

`Doubles[0..0]`

`[ipCableDiameter]` – Cable diameter.

St7SetTrussData

Sets the truss element parameters for the specified beam property.

```
long St7SetTrussData(long uID, long PropNum, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`Integers[0..0]`

`[ipTrussIncludeTorsion]` – Include torsion; either btTrue or btFalse.

St7GetTrussData

Returns the truss element parameters for the specified beam property.

```
long St7GetTrussData(long uID, long PropNum, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

Output Parameters

`Integers[0..0]`

`[ipTrussIncludeTorsion]` – Include torsion; either btTrue or btFalse.

St7SetCutoffBarData

Sets the cut-off bar parameters for the specified beam property.

```
long St7SetCutoffBarData(long uID, long PropNum, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`Integers[0..1]`

`[ipCutoffType]` – Type of cut-off bar; either cbBrittle or cbDuctile.

`[ipKeepMass]` – Use element mass; either btTrue or btFalse.

`Doubles[0..1]`

`[ipCutoffTension]` – Tensile force limit.

`[ipCutoffCompression]` – Compressive force limit.

St7GetCutoffBarData

Returns the cut-off bar parameters assigned to the specified beam property.

```
long St7GetCutoffBarData(long uID, long PropNum, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

Output Parameters

`Integers[0..1]`

`[ipCutoffType]` – Type of cut-off bar; either cbBrittle or cbDuctile.

`[ipKeepMass]` – Use element mass; either btTrue or btFalse.

`Doubles[0..1]`

`[ipCutoffTension]` – Tensile force limit.

`[ipCutoffCompression]` – Compressive force limit.

St7SetPointContactData

Sets the point contact element parameters for the specified beam property.

```
long St7SetPointContactData(long uID, long PropNum, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`Integers[0..6]`

`[ipContactType]` – Type of contact element; one of ctZeroGap, ctNormal, ctTension or ctTakeup.

`[ipContactSubType]` – Type of Takeup contact; either tuTension or tuCompression.

`[ipDynamicStiffness]` – Update the stiffness of the contact element; either btTrue or btFalse.

`[ipUpdateDirection]` – Update the direction of the contact throughout solution; either btTrue or btFalse.

`[ipFrictionModel]` – Type of friction model used; either cfElastic or cfPlastic.

`[ipFrictionYieldType]` – Type of yield; either cyRectangular or cyElliptical.

`[ipTensionLateralStiffness]` – Use lateral stiffness with ctTension elements; either btTrue or btFalse.

`Doubles[0..5]`

`[ipContactAxialStiffness]` – Penalty stiffness value in axial direction. This value is updated dynamically based on the **Update Direction** settings.

`[ipContactLateralStiffness]` – Penalty stiffness value in lateral direction.

`[ipFrictionC1]` – Lateral friction coefficient in the 1 axis direction.

`[ipFrictionC2]` – Lateral friction coefficient in the 2 axis direction.

`[ipContactMaxTension]` – Maximum tensile force value.

`[ipContactStrainTol]` – Target strain range for active point contacts with dynamic stiffness.

St7GetPointContactData

Returns the point contact element parameters assigned to the specified beam property.

```
long St7GetPointContactData(long uID, long PropNum, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

Output Parameters

`Integers[0..6]`

`[ipContactType]` – Type of contact element; one of ctZeroGap, ctNormal, ctTension or ctTakeup.

`[ipContactSubType]` – Type of Takeup contact; either tuTension or tuCompression.

`[ipDynamicStiffness]` – Update the stiffness of the contact element; either btTrue or btFalse.

`[ipUpdateDirection]` – Update the direction of the contact throughout solution; either btTrue or btFalse.

`[ipFrictionModel]` – Type of friction model used; either cfElastic or cfPlastic.

`[ipFrictionYieldType]` – Type of yield; either cyRectangular or cyElliptical.

`[ipTensionLateralStiffness]` – Use lateral stiffness with tuTension elements; either btTrue or btFalse.

`Doubles[0..5]`

`[ipContactAxialStiffness]` – Penalty stiffness value in axial direction. This value is updated dynamically based on the **Update Direction** settings.

`[ipContactLateralStiffness]` – Penalty stiffness value in lateral direction.

`[ipFrictionC1]` – Lateral friction coefficient in the 1 axis direction.

`[ipFrictionC2]` – Lateral friction coefficient in the 2 axis direction.

`[ipContactMaxTension]` – Maximum tensile force value.

`[ipContactStrainTol]` – Target strain range for active point contacts with Dynamic Stiffness.

St7SetPipeData

Sets the pipe element parameters for the specified beam property.

```
long St7SetPipeData(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`Doubles[0..3]`

`[ipPipeFlexibility]` – Flexibility factor.

`[ipPipeFluidDensity]` – Density of contained fluid.

`[ipPipeOuterDiameter]` – Outer diameter.

`[ipPipeThickness]` – Wall thickness.

St7GetPipeData

Returns the pipe element parameters for the specified beam property.

```
long St7GetPipeData(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

Output Parameters

`Doubles[0..3]`

`[ipPipeFlexibility]` – Flexibility factor.

`[ipPipeFluidDensity]` – Density of contained fluid.

`[ipPipeOuterDiameter]` – Outer diameter.

`[ipPipeThickness]` – Wall thickness.

St7SetConnectionData

Sets the connection element parameters for the specified beam property.

```
long St7SetConnectionData(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`Doubles[0..5]`

`[ipConnectionShear1]` – Translational stiffness in the 1 axis direction; this is a shear stiffness for Connection elements that do not reference a UCS.

`[ipConnectionShear2]` – Translational stiffness in the 2 axis direction; this is a shear stiffness for Connection elements that do not reference a UCS.

`[ipConnectionAxial]` – Translational stiffness in the 3 axis direction; this is an axial stiffness for Connection elements that do not reference a UCS.

`[ipConnectionBend1]` – Rotational stiffness in the plane of the 1 axis; this is a bending stiffness in plane 1 for Connection elements that do not reference a UCS.

`[ipConnectionBend2]` – Rotational stiffness in the plane of the 2 axis; this is a bending stiffness in plane 2 for Connection elements that do not reference a UCS.

`[ipConnectionTorque]` – Rotational stiffness about the 3 axis; this is a torque stiffness for Connection elements that do not reference a UCS.

Usage

A UCS may be assigned to the ends of Connection elements to override the principal axis system using *St7SetBeamConnectionUCS*.

St7GetConnectionData

Returns the connection element parameters for the specified beam property.

```
long St7GetConnectionData(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

Output Parameters

`Doubles[0..5]`

`[ipConnectionShear1]` – Translational stiffness in the 1 axis direction; this is a shear stiffness for Connection elements that do not reference a UCS.

`[ipConnectionShear2]` – Translational stiffness in the 2 axis direction; this is a shear stiffness for Connection elements that do not reference a UCS.

`[ipConnectionAxial]` – Translational stiffness in the 3 axis direction; this is an axial stiffness for Connection elements that do not reference a UCS.

`[ipConnectionBend1]` – Rotational stiffness in the plane of the 1 axis; this is a bending stiffness in plane 1 for Connection elements that do not reference a UCS.

`[ipConnectionBend2]` – Rotational stiffness in the plane of the 2 axis; this is a bending stiffness in plane 2 for Connection elements that do not reference a UCS.

`[ipConnectionTorque]` – Rotational stiffness about the 3 axis; this is a torque stiffness for Connection elements that do not reference a UCS.

Usage

A UCS may be assigned to the ends of Connection elements to override the principal axis system using *St7SetBeamConnectionUCS*.

St7SetUserBeamData

Sets the user defined element parameters for the specified beam property.

```
long St7SetUserBeamData(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`Doubles[0..21]`

`[0..20]` – User defined material matrix K defined by the upper triangular matrix of coefficients K_{ij}. See *User Defined Material Matrix* for indexing.

`[21]` – Element mass.

St7GetUserBeamData

Returns the user defined element property for the specified beam property.

```
long St7GetUserBeamData(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

Output Parameters

`Doubles[0..21]`

`[0..20]` – User defined material matrix K defined by the upper triangular matrix of coefficients $K_{ij.}$ See *User Defined Material Matrix* for indexing.

`[21]` – Element mass.

St7SetSpringDamperThermalData

Sets the thermal data required by the heat solvers for the specified Spring Damper property.

```
long St7SetSpringDamperThermalData(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`Doubles[0..1]`

`[ipThermalArea]` – Effective cross sectional area used to calculate heat capacity and flux.

`[ipThermalMass]` – The mass used to calculate heat capacity.

St7GetSpringDamperThermalData

Returns the thermal data set for the specified Spring Damper property that is required by the heat solvers.

```
long St7GetSpringDamperThermalData(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

Output Parameters

`Doubles[0..1]`

`[ipThermalArea]` – Effective cross sectional area used to calculate heat capacity and flux.

`[ipThermalMass]` – The mass used to calculate heat capacity.

St7SetPointContactThermalData

Sets the thermal data required by the heat solvers for the specified Point Contact property.

```
long St7SetPointContactThermalData(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`Doubles[0..1]`

`[ipThermalArea]` – Effective cross sectional area used to calculate heat capacity and flux.

`[ipThermalMass]` – The mass used to calculate heat capacity.

St7GetPointContactThermalData

Returns the thermal data set for the specified Point Contact property that is required by the heat solvers.

```
long St7GetPointContactThermalData(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

Output Parameters

`Doubles[0..1]`

`[ipThermalArea]` – Effective cross sectional area used to calculate heat capacity and flux.

`[ipThermalMass]` – The mass used to calculate heat capacity.

St7SetUserBeamThermalData

Sets the thermal data required by the heat solvers for the specified user defined beam property.

```
long St7SetUserBeamThermalData(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`Doubles[0..1]`

`[ipThermalArea]` – Effective cross sectional area used to calculate heat capacity and flux.

`[ipThermalMass]` – The mass used to calculate heat capacity.

St7GetUserBeamThermalData

Returns the thermal data set for the specified user defined beam property that is required by the heat solvers.

```
long St7GetUserBeamThermalData(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

Output Parameters

`Doubles[0..1]`

`[ipThermalArea]` – Effective cross sectional area used to calculate heat capacity and flux.

`[ipThermalMass]` – The mass used to calculate heat capacity.

St7SetConnectionThermalData

Sets the thermal data required by the heat solvers for the specified Connection property.

```
long St7SetConnectionThermalData(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`Doubles[0..1]`

`[ipThermalArea]` – Effective cross sectional area used to calculate heat capacity and flux.

`[ipThermalMass]` – The mass used to calculate heat capacity.

St7GetConnectionThermalData

Returns the thermal data set for the specified Connection property that is required by the heat solvers.

```
long St7GetConnectionThermalData(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

Output Parameters

`Doubles[0..1]`

`[ipThermalArea]` – Effective cross sectional area used to calculate heat capacity and flux.

`[ipThermalMass]` – The mass used to calculate heat capacity.

St7SetBeamMaterialData

Sets the material properties for the specified beam property.

```
long St7SetBeamMaterialData(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`Doubles[0..8]`

`[ipBeamModulus]` – Modulus.

`[ipBeamShear]` – Shear modulus.

`[ipBeamPoisson]` – Poisson's ratio.

`[ipBeamDensity]` – Density.

`[ipBeamAlpha]` – Thermal expansion coefficient.

`[ipBeamViscosity]` – Viscous damping coefficient.

`[ipBeamDampingRatio]` – Damping ratio.

`[ipBeamConductivity]` – Thermal conductivity coefficient.

`[ipBeamSpecificHeat]` – Specific heat coefficient.

Usage

The function can be called for all beam property types (i.e. btSpring, btCable, btTruss, btCutoff, btContact, btBeam, btUser, btPipe and btConnection), although not all types support the full set of material parameters.

St7GetBeamMaterialData

Returns the material properties assigned to the specified beam property.

```
long St7GetBeamMaterialData(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

PropNum

Beam property number.

Output Parameters

Doubles[0..8]

[ipBeamModulus] – Modulus.

[ipBeamShear] – Shear modulus.

[ipBeamPoisson] – Poisson's ratio.

[ipBeamDensity] – Density.

[ipBeamAlpha] – Thermal expansion coefficient.

[ipBeamViscosity] – Viscous damping coefficient.

[ipBeamDampingRatio] – Damping ratio.

[ipBeamConductivity] – Thermal conductivity coefficient.

[ipBeamSpecificHeat] – Specific heat coefficient.

Usage

Note that the values of [ipBeamShear] and [ipBeamPoisson] are those used by the solver, and take into account the setting of *St7SetBeamShearModulusMode*.

The function can be called for all beam property types (i.e. btSpring, btCable, btTruss, btCutoff, btContact, btBeam, btUser, btPipe and btConnection), although not all types support the full set of material parameters.

St7SetBeamShearModulusMode

Sets the specified beam property to use either Poisson's ratio or shear modulus for defining the material matrix.

```
long St7SetBeamShearModulusMode(long uID, long PropNum, long Mode)
```

Input Parameters

uID

Strand7 model file ID.

PropNum

Beam property number.

Mode

Either smUsePoissonsRatio or smUseShearModulus.

Usage

If Mode=smUsePoissonsRatio, the shear modulus is calculated from Young's modulus and Poisson's ratio assuming an isotropic material. If Mode=smUseShearModulus, the shear modulus is used directly.

St7GetBeamShearModulusMode

Returns whether the specified beam property uses Poisson's ratio or shear modulus for defining the material matrix.

```
long St7GetBeamShearModulusMode(long uID, long PropNum, long* Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

Output Parameters

`Mode`

Either smUsePoissonsRatio or smUseShearModulus.

Usage

If Mode=smUsePoissonsRatio, the shear modulus is calculated from Young's modulus and Poisson's ratio assuming an isotropic material. If Mode=smUseShearModulus, the shear modulus is used directly.

St7SetBeamNonlinearMode

Sets the specified beam property to use either the Fibre Stress or the Moment-Curvature option for material nonlinearity.

```
long St7SetBeamNonlinearMode(long uID, long PropNum, long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`Mode`

Either nmUseFibreStress or nmUseMomentCurvature.

St7GetBeamNonlinearMode

Returns whether the specified beam property uses the Fibre Stress or the Moment-Curvature option for material nonlinearity.

```
long St7GetBeamNonlinearMode(long uID, long PropNum, long* Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

Output Parameters

`Mode`

Either nmUseFibreStress or nmUseMomentCurvature.

St7NewPlateProperty

Creates a new plate property.

```
long St7NewPlateProperty(long uID, long PropNum, long PlateType,
     long MaterialType, char* PropName)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

`PlateType`

One of ptPlaneStress, ptPlaneStrain, ptAxisymmetric, ptPlateShell, ptShearPanel, ptMembrane or ptLoadPatch.

`MaterialType`

One of mtIsotropic, mtOrthotropic, mtAnisotropic, mtRubber, mtSoil, mtLaminate, mtUserDefined or mtFluid.

`PropName`

Name of the plate property.

St7SetPlatePropertyType

Sets the property type for the specified plate property.

```
long St7SetPlatePropertyType(long uID, long PropNum, long PlateType,
     long MaterialType)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

`PlateType`

One of ptPlaneStress, ptPlaneStrain, ptAxisymmetric, ptPlateShell, ptShearPanel, ptMembrane or ptLoadPatch.

`MaterialType`

One of mtIsotropic, mtOrthotropic, mtAnisotropic, mtRubber, mtSoil, mtLaminate, mtUserDefined or mtFluid.

St7GetPlatePropertyType

Returns the property type for the specified plate property.

```
long St7GetPlatePropertyType(long uID, long PropNum, long* PlateType,
     long* MaterialType)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

Output Parameters

`PlateType`

One of ptNull, ptPlaneStress, ptPlaneStrain, ptAxisymmetric, ptPlateShell, ptShearPanel, ptMembrane or ptLoadPatch.

`MaterialType`

One of mtNull, mtIsotropic, mtOrthotropic, mtAnisotropic, mtRubber, mtSoil, mtLaminate, mtUserDefined or mtFluid.

St7SetPlateNonlinearType

Sets the nonlinear material type for the specified plate property.

```
long St7SetPlateNonlinearType(long uID, long PropNum, long NonlinType,
     long YieldType)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

`NonlinType`

Nonlinear material type; either ntNonlinElastic or ntElastoPlastic.

`YieldType`

One of ycTresca, ycVonMises, ycMaxStress, ycMohrCoulomb or ycDruckerPrager.

St7GetPlateNonlinearType

Returns the nonlinear material type assigned to the specified plate property.

```
long St7GetPlateNonlinearType(long uID, long PropNum, long* NonlinType,
     long* YieldType)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

Output Parameters

`NonlinType`

Nonlinear material type; either ntNonlinElastic or ntElastoPlastic.

`YieldType`

One of ycTresca, ycVonMises, ycMaxStress, ycMohrCoulomb or ycDruckerPrager.

St7SetPlateThickness

Sets the thickness for the specified plate property.

```
long St7SetPlateThickness(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

`Doubles[0..1]`

`[0]` – Plate membrane thickness.

`[1]` – Plate bending thickness.

St7GetPlateThickness

Returns the thickness assigned to the specified plate property.

```
long St7GetPlateThickness(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

Output Parameters

`Doubles[0..1]`

`[0]` – Plate membrane thickness.

`[1]` – Plate bending thickness.

St7SetPlateIsotropicMaterial

Sets the isotropic material parameters for the specified plate property.

```
long St7SetPlateIsotropicMaterial(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

`Doubles[0..7]`

`[ipPlateIsoModulus]` – Modulus.

`[ipPlateIsoPoisson]` – Poisson’s ratio.

`[ipPlateIsoDensity]` – Density.

`[ipPlateIsoAlpha]` – Thermal expansion coefficient.

`[ipPlateIsoViscosity]` – Viscous damping coefficient.

`[ipPlateIsoDampingRatio]` – Damping ratio.

`[ipPlateIsoConductivity]` – Conductivity coefficient.

`[ipPlateIsoSpecificHeat]` – Specific heat coefficient.

St7GetPlateIsotropicMaterial

Returns the isotropic material properties for the specified plate property.

```
long St7GetPlateIsotropicMaterial(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

Output Parameters

`Doubles[0..7]`

`[ipPlateIsoModulus]` – Modulus.

`[ipPlateIsoPoisson]` – Poisson’s ratio.

`[ipPlateIsoDensity]` – Density.

`[ipPlateIsoAlpha]` – Thermal expansion coefficient.

`[ipPlateIsoViscosity]` – Viscous damping coefficient.

`[ipPlateIsoDampingRatio]` – Damping ratio.

`[ipPlateIsoConductivity]` – Conductivity coefficient.

`[ipPlateIsoSpecificHeat]` – Specific heat coefficient.

St7SetPlateOrthotropicMaterial

Sets the orthotropic material properties for the specified plate property.

```
long St7SetPlateOrthotropicMaterial(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

`Doubles[0..17]`

`[ipPlateOrthoModulus1]` – Modulus in the 1 axis direction.

`[ipPlateOrthoModulus2]` – Modulus in the 2 axis direction.

`[ipPlateOrthoModulus3]` – Modulus in the 3 axis direction.

`[ipPlateOrthoShear12]` – Shear modulus in the 12 axis direction.

[ipPlateOrthoShear23] – Shear modulus in the 23 axis direction.

[ipPlateOrthoShear31] – Shear modulus in the 31 axis direction.

[ipPlateOrthoPoisson12] – Poisson's ratio in the 12 axis direction.

[ipPlateOrthoPoisson23] – Poisson's ratio in the 23 axis direction.

[ipPlateOrthoPoisson31] – Poisson's ratio in the 31 axis direction.

[ipPlateOrthoDensity] – Density.

[ipPlateOrthoAlpha1] – Thermal expansion coefficient in the 12 axis direction.

[ipPlateOrthoAlpha2] – Thermal expansion coefficient in the 23 axis direction.

[ipPlateOrthoAlpha3] – Thermal expansion coefficient in the 31 axis direction.

[ipPlateOrthoViscosity] – Viscous damping coefficient.

[ipPlateOrthoDampingRatio] – Damping ratio.

[ipPlateOrthoConductivity1] – Thermal conductivity in the 1 axis direction.

[ipPlateOrthoConductivity2] – Thermal conductivity in the 2 axis direction.

[ipPlateOrthoSpecificHeat] – Specific heat coefficient.

St7GetPlateOrthotropicMaterial

Returns the orthotropic material properties assigned to the specified plate property.

```
long St7GetPlateOrthotropicMaterial(long uID, long PropNum, double* Doubles)
```

Input Parameters

uID

Strand7 model file ID.

PropNum

Plate property number.

Output Parameters

Doubles[0..17]

[ipPlateOrthoModulus1] – Modulus in the 1 axis direction.

[ipPlateOrthoModulus2] – Modulus in the 2 axis direction.

[ipPlateOrthoModulus3] – Modulus in the 3 axis direction.

[ipPlateOrthoShear12] – Shear modulus in the 12 axis direction.

[ipPlateOrthoShear23] – Shear modulus in the 23 axis direction.

[ipPlateOrthoShear31] – Shear modulus in the 31 axis direction.

[ipPlateOrthoPoisson12] – Poisson's ratio in the 12 axis direction.

[ipPlateOrthoPoisson23] – Poisson's ratio in the 23 axis direction.

[ipPlateOrthoPoisson31] – Poisson's ratio in the 31 axis direction.

[ipPlateOrthoDensity] – Density.

[ipPlateOrthoAlpha1] – Thermal expansion coefficient in the 12 axis direction.

[ipPlateOrthoAlpha2] – Thermal expansion coefficient in the 23 axis direction.

[ipPlateOrthoAlpha3] – Thermal expansion coefficient in the 31 axis direction.

[ipPlateOrthoViscosity] – Viscous damping coefficient.

[ipPlateOrthoDampingRatio] – Damping ratio.

[ipPlateOrthoConductivity1] – Thermal conductivity in the 1 axis direction.

[ipPlateOrthoConductivity2] – Thermal conductivity in the 2 axis direction.

[ipPlateOrthoSpecificHeat] – Specific heat coefficient.

St7SetPlateRubberMaterial

Sets the rubber material properties for the specified plate property.

```
long St7SetPlateRubberMaterial(long uID, long PropNum, long RubberType,
     double* Doubles)
```

Input Parameters

uID

Strand7 model file ID.

PropNum

Plate property number.

RubberType

One of rtNeoHookean, rtMooneyRivlin, rtGeneralisedMooneyRivlin or rtOgden.

Doubles[0..15]

An array describing the rubber material coefficients. The format depends on the material sub-type, with different sub-types requiring a varying number of rubber coefficients following the common data:

[ipRubberBulk] – Bulk modulus.

[ipRubberDensity] – Density.

[ipRubberAlpha] – Thermal expansion coefficient.

[ipRubberViscosity] – Viscous damping coefficient.

[ipRubberDampingRatio] – Damping ratio.

[ipRubberConductivity] – Conductivity.

[ipRubberSpecificHeat] – Specific heat.

[ipRubberConstC1..ipRubberConstC1+Num] – Rubber coefficients,

where:
Num = 0 (Neo-Hookean)
Num = 1 (Mooney-Rivlin)
Num = 8 (Generalised Mooney-Rivlin)
Num = 5 (Ogden)

St7GetPlateRubberMaterial

Returns the rubber material properties assigned to the specified plate property.

```
long St7GetPlateRubberMaterial(long uID, long PropNum, long* RubberType,
    double* Doubles)
```

Input Parameters

uID

Strand7 model file ID.

PropNum

Plate property number.

Output Parameters

RubberType

One of rtNeoHookean, rtMooneyRivlin, rtGeneralisedMooneyRivlin or rtOgden.

Doubles[0..15]

An array describing the rubber material coefficients. The format depends on the material sub-type, with different sub-types requiring a varying number of rubber coefficients following the common data:

[ipRubberBulk] – Bulk modulus.

[ipRubberDensity] – Density.

[ipRubberAlpha] – Thermal expansion coefficient.

[ipRubberViscosity] – Viscous damping coefficient.

[ipRubberDampingRatio] – Damping ratio.

[ipRubberConductivity] – Conductivity.

[ipRubberSpecificHeat] – Specific heat.

[ipRubberConstC1..ipRubberConstC1+Num] – Rubber coefficients,

where:
Num = 0 (Neo-Hookean)
Num = 1 (Mooney-Rivlin)
Num = 8 (Generalised Mooney-Rivlin)
Num = 5 (Ogden)

St7SetPlateAnisotropicMaterial

Sets the anisotropic material properties for the specified plate property.

```
long St7SetPlateAnisotropicMaterial(long uID, long PropNum, long MatType,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

`MatType`

Matrix type; either mtStiffness or mtCompliance.

`Doubles[0..22]`

`[0..5]` – In-plane components of the anisotropic material stress-strain matrix D defined by the coefficients D_{11}, D_{12}, D_{13}, D_{22}, D_{23} and D_{33}, respectively.

`[6..9]` – Out-of-plane components of the anisotropic material stress-strain matrix D defined by the coefficients D_{14}, D_{24}, D_{34}, and D_{44}, respectively (plane stress and plane strain elements only).

`[ipPlateAnisoTransShear1]` – Transverse shear modulus in the 13 plane (plate/shell elements only).

`[ipPlateAnisoTransShear2]` – Transverse shear modulus in the 23 plane (plate/shell elements only).

`[ipPlateAnisoTransShear3]` – Transverse shear coupling modulus (plate/shell elements only).

`[ipPlateAnisoDensity]` – Density.

`[ipPlateAnisoAlpha1]` – Thermal expansion coefficient in the 1 axis direction.

`[ipPlateAnisoAlpha2]` – Thermal expansion coefficient in the 2 axis direction.

`[ipPlateAnisoAlpha3]` – Thermal expansion coefficient in the 3 axis direction.

`[ipPlateAnisoAlpha12]` – Thermal expansion coefficient in the 12 axis direction.

`[ipPlateAnisoViscosity]` – Viscous damping coefficient.

`[ipPlateAnisoDampingRatio]` – Damping ratio.

`[ipPlateAnisoConductivity1]` – Conductivity coefficient in the 1 axis direction.

`[ipPlateAnisoConductivity2]` – Conductivity coefficient in the 2 axis direction.

`[ipPlateAnisoSpecificHeat]` – Specific heat coefficient.

St7GetPlateAnisotropicMaterial

Returns the anisotropic material properties assigned to the specified plate property.

```
long St7GetPlateAnisotropicMaterial(long uID, long PropNum, long* MatType,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

Output Parameters

`MatType`

Matrix type; either mtStiffness or mtCompliance.

`Doubles[0..22]`

`[0..5]` – In-plane components of the anisotropic material stress-strain matrix D defined by the coefficients D_{11}, D_{12}, D_{13}, D_{22}, D_{23} and D_{33}, respectively.

`[6..9]` – Out-of-plane components of the anisotropic material stress-strain matrix D defined by the coefficients D_{14}, D_{24}, D_{34}, and D_{44}, respectively (plane stress and plane strain elements only).

`[ipPlateAnisoTransShear1]` – Transverse shear modulus in the 13 plane.

`[ipPlateAnisoTransShear2]` – Transverse shear modulus in the 23 plane.

`[ipPlateAnisoTransShear3]` – Transverse shear coupling modulus.

`[ipPlateAnisoDensity]` – Density.

`[ipPlateAnisoAlpha1]` – Thermal expansion coefficient in the 1 axis direction.

`[ipPlateAnisoAlpha2]` – Thermal expansion coefficient in the 2 axis direction.

`[ipPlateAnisoAlpha3]` – Thermal expansion coefficient in the 3 axis direction.

`[ipPlateAnisoAlpha12]` – Thermal expansion coefficient in the 12 axis direction.

`[ipPlateAnisoViscosity]` – Viscous damping coefficient.

`[ipPlateAnisoDampingRatio]` – Damping ratio.

`[ipPlateAnisoConductivity1]` – Conductivity coefficient in the 1 axis direction.

`[ipPlateAnisoConductivity2]` – Conductivity coefficient in the 2 axis direction.

`[ipPlateAnisoSpecificHeat]` – Specific heat coefficient.

St7SetPlateLaminateMaterial

Sets the laminate material properties for the specified plate property. Not all parameters returned by *St7GetPlateLaminateMaterial* can be set, as some parameters are based on the ply properties in the layup.

```
long St7SetPlateLaminateMaterial(long uID, long PropNum, long LaminateID,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

`LaminateID`

Laminate layout ID.

`Doubles[0..4]`

`[ipLaminateViscosity]` – Viscous damping coefficient.

`[ipLaminateDampingRatio]` – Damping ratio.

`[ipLaminateConductivity1]` – Conductivity coefficient in the x axis direction.

`[ipLaminateConductivity2]` – Conductivity coefficient in the y axis direction.

`[ipLaminateSpecificHeat]` – Specific heat coefficient.

St7GetPlateLaminateMaterial

Returns the laminate material properties for the specified plate property.

```
long St7GetPlateLaminateMaterial(long uID, long PropNum, long* LaminateID,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

Output Parameters

`LaminateID`

Laminate layout ID.

`Doubles[0..17]`

`[ipLaminateViscosity]` – Viscous damping coefficient.

[ipLaminateDampingRatio] – Damping ratio.

[ipLaminateConductivity1] – Conductivity coefficient in the x axis direction.

[ipLaminateConductivity2] – Conductivity coefficient in the y axis direction.

[ipLaminateSpecificHeat] – Specific heat coefficient.

[ipLaminateDensity] – Mass density per unit volume.

[ipLaminateAlphax] – Effective membrane thermal expansion coefficient in the x axis direction.

[ipLaminateAlphay] – Effective membrane thermal expansion coefficient in the y axis direction.

[ipLaminateAlphaxy] – Effective membrane shear thermal expansion coefficient in the xy plane.

[ipLaminateBetax] – Effective bending thermal expansion coefficient in the x direction.

[ipLaminateBetay] – Effective bending thermal expansion coefficient in the y direction.

[ipLaminateBetaxy] – Effective twisting thermal expansion coefficient out of the xy plane.

[ipLaminateModulusx] – Effective modulus in the x axis direction.

[ipLaminateModulusy] – Effective modulus in the y axis direction.

[ipLaminateShearxy] – Effective shear modulus in the xy plane.

[ipLaminatePoissonxy] – Effective Poisson's ratio v_{xy}.

[ipLaminatePoissonyx] – Effective Poisson's ratio v_{yx}.

[ipLaminateThickness] – Thickness.

St7SetPlateUserDefinedMaterial

Sets the user defined material properties for the specified plate property.

```
long St7SetPlateUserDefinedMaterial(long uID, long PropNum, long MatType,
    double* Doubles)
```

Input Parameters

uID

Strand7 model file ID.

PropNum

Plate property number.

MatType

Matrix type; either mtStiffness or mtCompliance.

Doubles[0..35]

[0..20] – Material membrane, bending and coupling matrices with indexing as defined in *User Defined Material Matrix*.

[ipPlateUserTransShearxz] – Transverse shear modulus G_{13}.

[ipPlateUserTransShearyz] – Transverse shear modulus G_{23}.

[ipPlateUserTransShearcz] – Transverse shear coupling modulus G_{c3}.

[ipPlateUserDensity] – Mass density per unit volume.

[ipPlateUserAlphax] – Thermal expansion coefficient in the 1 axis direction.

[ipPlateUserAlphay] – Thermal expansion coefficient in the 2 axis direction.

[ipPlateUserAlphaxy] – Thermal expansion coefficient in the 12 axis direction.

[ipPlateUserBetax] – Thermal curvature expansion coefficient along the 1 axis direction.

[ipPlateUserBetay] – Thermal curvature expansion coefficient along the 2 axis direction.

[ipPlateUserBetaxy] – Thermal twist expansion coefficient.

[ipPlateUserViscosity] – Viscous damping coefficient.

[ipPlateUserDampingRatio] – Damping ratio.

[ipPlateUserConductivity1] – Conductivity coefficient in the 1 axis direction.

[ipPlateUserConductivity2] – Conductivity coefficient in the 2 axis direction.

[ipPlateUserSpecificHeat] – Specific heat coefficient.

St7GetPlateUserDefinedMaterial

Returns the user defined material properties assigned to the specified plate property.

```
long St7GetPlateUserDefinedMaterial(long uID, long PropNum, long* MatType,
     double* Doubles)
```

Input Parameters

uID

Strand7 model file ID.

PropNum

Plate property number.

Output Parameters

MatType

Matrix type; either mtStiffness or mtCompliance.

Doubles[0..35]

[0..20] – Material membrane, bending and coupling matrices with indexing as defined in *User Defined Material Matrix.*

[ipPlateUserTransShearxz] – Transverse shear modulus G_{13}.

[ipPlateUserTransShearyz] – Transverse shear modulus G_{23}.

[ipPlateUserTransShearcz] – Transverse shear modulus G_{c3}.

[ipPlateUserDensity] – Mass density per unit volume.

[ipPlateUserAlphax] – Thermal expansion coefficient in the 1 axis direction.

[ipPlateUserAlphay] – Thermal expansion coefficient in the 2 axis direction.

[ipPlateUserAlphaxy] – Thermal expansion coefficient in the 12 axis direction.

[ipPlateUserBetax] – Thermal curvature expansion coefficient along the 1 axis direction.

[ipPlateUserBetay] – Thermal curvature expansion coefficient along the 2 axis direction.

[ipPlateUserBetaxy] – Thermal twist expansion coefficient.

[ipPlateUserViscosity] – Viscous damping coefficient.

[ipPlateUserDampingRatio] – Damping ratio.

[ipPlateUserConductivity1] – Conductivity coefficient in the 1 axis direction.

[ipPlateUserConductivity2] – Conductivity coefficient in the 2 axis direction.

[ipPlateUserSpecificHeat] – Specific heat coefficient.

St7SetPlateMCDPMaterial

Sets the material properties for the specified Mohr-Coulomb or Drucker-Prager plate property. Note that these properties correspond to the isotropic yield criterion; to set soil material properties with these yield criteria use *St7SetPlateSoilMCMaterial* or *St7SetPlateSoilDPMaterial*.

```
long St7SetPlateMCDPMaterial(long uID, long PropNum, double* Doubles)
```

Input Parameters

uID

Strand7 model file ID.

PropNum

Plate property number.

Doubles[0..1]

[ipFrictionAngle] – Friction angle.

[ipCohesion] – Cohesion value.

St7GetPlateMCDPMaterial

Returns the material properties assigned to the specified Mohr-Coulomb or Drucker-Prager plate property. Note that these properties correspond to the isotropic yield criterion; to get soil material properties with these yield criteria use *St7GetPlateSoilMCMaterial* or *St7GetPlateSoilDPMaterial*.

```
long St7GetPlateMCDPMaterial(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

Output Parameters

`Doubles[0..1]`

`[ipFrictionAngle]` – Friction angle.

`[ipCohesion]` – Cohesion value.

St7GetPlateSoilType

Returns the soil type for a plate property assigned as a soil material.

```
long St7GetPlateSoilType(long uID, long PropNum, long* SoilType)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

Output Parameters

`SoilType`

One of stDuncanChang, stModifiedCamClay, stMohrCoulomb, stDruckerPrager or stLinearElastic.

St7SetPlateSoilDCMaterial

Sets the soil material properties for the specified Duncan-Chang plate property.

```
long St7SetPlateSoilDCMaterial(long uID, long PropNum, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

`Integers[0..2]`

`[ipSoilDCUseBulkModulus]` – Use bulk modulus; either btTrue or btFalse.

`[ipSoilDCSetPropLevel]` – btTrue to set fluid level to Property value; btFalse to set fluid level to Stage/Global level.

`[ipSoilDCDrainedState]` – Drained state; either btTrue or btFalse.

`Doubles[0..19]`

`[ipSoilDCModulusK]` – Modulus.

`[ipSoilDCModulusKUR]` – Unloading/reloading modulus.

`[ipSoilDCModulusN]` – Modulus exponent.

`[ipSoilDCPoisson]` – Poisson's ratio.

`[ipSoilDCBulkK]` – Bulk modulus.

`[ipSoilDCBulkM]` – Bulk modulus exponent.

`[ipSoilDCFrictionAngle]` – Friction angle.

`[ipSoilDCDeltaAngle]` – Friction angle change.

`[ipSoilDCCohesion]` – Cohesion value.

`[ipSoilDCFailureRatio]` – Failure ratio.

`[ipSoilDCFailureMod]` – Failure modulus.

`[ipSoilDCReferenceP]` – Reference pressure.

`[ipSoilDCDensity]` – Mass density per unit volume.

`[ipSoilDCHorizontalRatio]` – Horizontal stress ratio.

`[ipSoilDCER]` – Reference void ratio.

`[ipSoilDCConductivity]` – Conductivity.

`[ipSoilDCSpecificHeat]` – Specific heat.

`[ipSoilDCFluidLevel]` – Fluid level.

`[ipSoilDCViscosity]` – Viscous damping coefficient.

`[ipSoilDCDampingRatio]` – Damping ratio.

St7GetPlateSoilDCMaterial

Returns the soil material properties assigned to the specified Duncan-Chang plate property.

```
long St7GetPlateSoilDCMaterial(long uID, long PropNum, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

Output Parameters

`Integers[0..2]`

`[ipSoilDCUseBulkModulus]` – Use bulk modulus; either btTrue or btFalse.

`[ipSoilDCSetPropLevel]` – btTrue to set fluid level to Property value; btFalse to set fluid level to Stage/Global level.

`[ipSoilDCDrainedState]` – Drained state; either btTrue or btFalse.

`Doubles[0..19]`

`[ipSoilDCModulusK]` – Modulus.

`[ipSoilDCModulusKUR]` – Unloading/reloading modulus.

`[ipSoilDCModulusN]` – Modulus exponent.

`[ipSoilDCPoisson]` – Poisson's ratio.

`[ipSoilDCBulkK]` – Bulk modulus.

`[ipSoilDCBulkM]` – Bulk modulus exponent.

`[ipSoilDCFrictionAngle]` – Friction angle.

`[ipSoilDCDeltaAngle]` – Friction angle change.

`[ipSoilDCCohesion]` – Cohesion value.

`[ipSoilDCFailureRatio]` – Failure ratio.

`[ipSoilDCFailureMod]` – Failure modulus.

`[ipSoilDCReferenceP]` – Reference pressure.

`[ipSoilDCDensity]` – Mass density per unit volume.

`[ipSoilDCHorizontalRatio]` – Horizontal stress ratio.

`[ipSoilDCER]` – Reference void ratio.

`[ipSoilDCConductivity]` – Conductivity.

`[ipSoilDCSpecificHeat]` – Specific heat.

`[ipSoilDCFluidLevel]` – Fluid level.

`[ipSoilDCViscosity]` – Viscous damping coefficient.

`[ipSoilDCDampingRatio]` – Damping ratio.

St7SetPlateSoilCCMaterial

Sets the soil material properties for the specified Cam-Clay plate property.

```
long St7SetPlateSoilCCMaterial(long uID, long PropNum, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

`Integers[0..3]`

`[ipSoilCCUsePoisson]` – Use Poisson's ratio; either btTrue or btFalse.

`[ipSoilCCDrainedState]` – Drained state; either btTrue or btFalse.

`[ipSoilCCUseOCR]` – Overconsolidation; either btTrue or btFalse.

`[ipSoilCCSetPropLevel]` – btTrue to set fluid level to Property value; btFalse to set fluid level to Stage/Global level.

`Doubles[0..16]`

`[ipSoilCCCriticalStateLine]` – Critical state line slope.

`[ipSoilCCConsolidationLine]` – Normal consolidation line slope.

`[ipSoilCCSwellingLine]` – Swelling line slope.

`[ipSoilCCDensity]` – Mass density per unit volume.

`[ipSoilCCPoisson]` – Poisson's ratio.

`[ipSoilCCModulusGa]` – Shear modulus at point A.

`[ipSoilCCModulusGb]` – Shear modulus at point B.

`[ipSoilCCHorizontalRatio]` – Horizontal stress ratio.

`[ipSoilCCER]` – Reference void ratio.

`[ipSoilCCPR]` – Unit pressure ratio.

`[ipSoilCCPC0]` – Initial consolidation pressure.

`[ipSoilCCOCR]` – Overconsolidation ratio.

`[ipSoilCCConductivity]` – Conductivity.

`[ipSoilCCSpecificHeat]` – Specific heat.

`[ipSoilCCFluidLevel]` – Fluid level.

`[ipSoilCCViscosity]` – Viscous damping coefficient.

`[ipSoilCCDampingRatio]` – Damping ratio.

St7GetPlateSoilCCMaterial

Returns the soil material properties assigned to the specified Cam-Clay plate property.

```
long St7GetPlateSoilCCMaterial(long uID, long PropNum, long* Integers,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

Output Parameters

`Integers[0..3]`

`[ipSoilCCUsePoisson]` – Use Poisson’s ratio; either btTrue or btFalse.

`[ipSoilCCDrainedState]` – Drained state; either btTrue or btFalse.

`[ipSoilCCUseOCR]` – Over-consolidation; either btTrue or btFalse.

`[ipSoilCCSetPropLevel]` – btTrue to set fluid level to Property value; btFalse to set fluid level to Stage/Global level.

`Doubles[0..16]`

`[ipSoilCCCriticalStateLine]` – Critical state line slope.

`[ipSoilCCConsolidationLine]` – Normal consolidation line slope.

`[ipSoilCCSwellingLine]` – Swelling line slope.

`[ipSoilCCDensity]` – Mass density per unit volume.

`[ipSoilCCPoisson]` – Poisson’s ratio.

`[ipSoilCCModulusGa]` – Shear modulus at point A.

`[ipSoilCCModulusGb]` – Shear modulus at point B.

`[ipSoilCCHorizontalRatio]` – Horizontal stress ratio.

`[ipSoilCCER]` – Reference void ratio.

`[ipSoilCCPR]` – Unit pressure ratio.

`[ipSoilCCPC0]` – Initial consolidation pressure.

`[ipSoilCCOCR]` – Overconsolidation ratio.

`[ipSoilCCConductivity]` – Conductivity.

`[ipSoilCCSpecificHeat]` – Specific heat.

`[ipSoilCCFluidLevel]` – Fluid level.

[ipSoilCCViscosity] – Viscous damping coefficient.

[ipSoilCCDampingRatio] – Damping ratio.

St7SetPlateSoilMCMaterial

Assigns the Mohr-Coulomb soil parameters for the specified plate property.

```
long St7SetPlateSoilMCMaterial(long uID, long PropNum, long* Integers,
    double* Doubles)
```

Input Parameters

uID

Strand7 model file ID.

PropNum

Property number.

Integers[0..1]

[ipSoilMCSetPropLevel] – btTrue to set fluid level to Property value; btFalse to set fluid level to Stage/Global level.

[ipSoilMCDrainedState] – Drained state; either btTrue or btFalse.

Doubles[0..11]

[ipSoilMCModulus] – Modulus.

[ipSoilMCPoisson] – Poisson's ratio.

[ipSoilMCDensity] – Mass density per unit volume.

[ipSoilMCCohesion] – Cohesion value.

[ipSoilMCFrictionAngle] – Friction angle.

[ipSoilMCHorizontalRatio] – Horizontal stress ratio.

[ipSoilMCER] – Void ratio.

[ipSoilMCConductivity] – Conductivity.

[ipSoilMCSpecificHeat] – Specific heat.

[ipSoilMCFluidLevel] – Fluid level.

[ipSoilMCViscosity] – Viscous damping coefficient.

[ipSoilMCDampingRatio] – Damping ratio.

St7GetPlateSoilMCMaterial

Returns the Mohr-Coulomb soil parameters assigned to the specified plate property.

```
long St7GetPlateSoilMCMaterial(long uID, long PropNum, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Property number.

Output Parameters

`Integers[0..1]`

`[ipSoilMCSetPropLevel]` – btTrue to set fluid level to Property value; btFalse to set fluid level to Stage/Global level.

`[ipSoilMCDrainedState]` – Drained state; either btTrue or btFalse.

`Doubles[0..11]`

`[ipSoilMCModulus]` – Modulus.

`[ipSoilMCPoisson]` – Poisson's ratio.

`[ipSoilMCDensity]` – Mass density per unit volume.

`[ipSoilMCCohesion]` – Cohesion value.

`[ipSoilMCFrictionAngle]` – Friction angle.

`[ipSoilMCHorizontalRatio]` – Horizontal stress ratio.

`[ipSoilMCER]` – Void ratio.

`[ipSoilMCConductivity]` – Conductivity.

`[ipSoilMCSpecificHeat]` – Specific heat.

`[ipSoilMCFluidLevel]` – Fluid level.

`[ipSoilMCViscosity]` – Viscous damping coefficient.

`[ipSoilMCDampingRatio]` – Damping ratio.

St7SetPlateSoilDPMaterial

Assigns the Drucker-Prager soil parameters to the specified plate property.

```
long St7SetPlateSoilDPMaterial(long uID, long PropNum, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Property number.

`Integers[0..1]`

`[ipSoilDPSetPropLevel]` – btTrue to set fluid level to Property value; btFalse to set fluid level to Stage/Global level.

`[ipSoilDPDrainedState]` – Drained state; either btTrue or btFalse.

`Doubles[0..11]`

`[ipSoilDPModulus]` – Modulus.

`[ipSoilDPPoisson]` – Poisson's ratio.

`[ipSoilDPDensity]` – Mass density per unit volume.

`[ipSoilDPCohesion]` – Cohesion value.

`[ipSoilDPFrictionAngle]` – Friction angle.

`[ipSoilDPHorizontalRatio]` – Horizontal stress ratio.

`[ipSoilDPER]` – Void ratio.

`[ipSoilDPConductivity]` – Conductivity.

`[ipSoilDPSpecificHeat]` – Specific heat.

`[ipSoilDPFluidLevel]` – Fluid level.

`[ipSoilDPViscosity]` – Viscous damping coefficient.

`[ipSoilDPDampingRatio]` – Damping ratio.

St7GetPlateSoilDPMaterial

Returns the Drucker-Prager soil parameters assigned to the specified plate property.

```
long St7GetPlateSoilDPMaterial(long uID, long PropNum, long* Integers,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Property number.

Output Parameters

`Integers[0..1]`

`[ipSoilDPSetPropLevel]` – btTrue to set fluid level to Property value; btFalse to set fluid level to Stage/Global level.

`[ipSoilDPDrainedState]` – Drained state; either btTrue or btFalse.

`Doubles[0..11]`

`[ipSoilDPModulus]` – Modulus.

`[ipSoilDPPoisson]` – Poisson's ratio.

`[ipSoilDPDensity]` – Mass density per unit volume.

`[ipSoilDPCohesion]` – Cohesion value.

`[ipSoilDPFrictionAngle]` – Friction angle.

`[ipSoilDPHorizontalRatio]` – Horizontal stress ratio.

`[ipSoilDPER]` – Void ratio.

`[ipSoilDPConductivity]` – Conductivity.

`[ipSoilDPSpecificHeat]` – Specific heat.

`[ipSoilDPFluidLevel]` – Fluid level.

`[ipSoilDPViscosity]` – Viscous damping coefficient.

`[ipSoilDPDampingRatio]` – Damping ratio.

St7SetPlateSoilLSMaterial

Sets the linear elastic soil parameters for the specified plate property.

```
long St7SetPlateSoilLSMaterial(long uID, long PropNum, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Property number.

`Integers[0..1]`

`[ipSoilLSSetPropLevel]` – btTrue to set fluid level to Property value; btFalse to set fluid level to Stage/Global level.

`[ipSoilLSDrainedState]` – Drained state; either btTrue or btFalse.

`Doubles[0..9]`

`[ipSoilLSModulus]` – Modulus.

`[ipSoilLSPoisson]` – Poisson's ratio.

`[ipSoilLSDensity]` – Mass density per unit volume.

`[ipSoilLSHorizontalRatio]` – Horizontal stress ratio.

[ipSoilLSER] – Void ratio.

[ipSoilLSConductivity] – Conductivity.

[ipSoilLSSpecificHeat] – Specific heat.

[ipSoilLSFluidLevel] – Fluid level.

[ipSoilLSViscosity] – Viscous damping coefficient.

[ipSoilLSDampingRatio] – Damping ratio.

St7GetPlateSoilLSMaterial

Returns the linear elastic soil parameters for the specified plate property.

```
long St7GetPlateSoilLSMaterial(long uID, long PropNum, long* Integers,
    double* Doubles)
```

Input Parameters

uID

Strand7 model file ID.

PropNum

Property number.

Output Parameters

Integers[0..1]

[ipSoilLSSetPropLevel] – btTrue to set fluid level to Property value; btFalse to set fluid level to Stage/Global level.

[ipSoilLSDrainedState] – Drained state; either btTrue or btFalse.

Doubles[0..9]

[ipSoilLSModulus] – Modulus.

[ipSoilLSPoisson] – Poisson's ratio.

[ipSoilLSDensity] – Mass density per unit volume.

[ipSoilLSHorizontalRatio] – Horizontal stress ratio.

[ipSoilLSER] – Void ratio.

[ipSoilLSConductivity] – Conductivity.

[ipSoilLSSpecificHeat] – Specific heat.

[ipSoilLSFluidLevel] – Fluid level.

[ipSoilLSViscosity] – Viscous damping coefficient.

[ipSoilLSDampingRatio] – Damping ratio.

St7SetPlateFluidMaterial

Sets the material properties for the specified fluid plate property.

```
long St7SetPlateFluidMaterial(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

`Doubles[0..7]`

`[ipFluidModulus]` – Modulus.

`[ipFluidPenaltyParam]` – Penalty parameter.

`[ipFluidDensity]` – Mass density per unit volume.

`[ipFluidAlpha]` – Thermal expansion coefficient.

`[ipFluidViscosity]` – Viscous damping coefficient.

`[ipFluidDampingRatio]` – Damping ratio.

`[ipFluidConductivity]` – Conductivity.

`[ipFluidSpecificHeat]` – Specific heat.

St7GetPlateFluidMaterial

Returns the material properties assigned to the specified fluid plate property.

```
long St7GetPlateFluidMaterial(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

Output Parameters

`Doubles[0..7]`

`[ipFluidModulus]` – Modulus.

`[ipFluidPenaltyParam]` – Penalty parameter.

`[ipFluidDensity]` – Mass density per unit volume.

`[ipFluidAlpha]` – Thermal expansion coefficient.

`[ipFluidViscosity]` – Viscous damping coefficient.

`[ipFluidDampingRatio]` – Damping ratio.

`[ipFluidConductivity]` – Conductivity.

`[ipFluidSpecificHeat]` – Specific heat.

St7SetPlateUseReducedInt

Sets the state of the **Reduced Integration** flag for the specified plate property. This option is only used for the 2D and Axisymmetric plate property types, applicable to quadratic elements.

```
long St7SetPlateUseReducedInt(long uID, long PropNum, bool UseReducedInt)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

`UseReducedInt`

True to use a reduced order integration scheme.

St7GetPlateUseReducedInt

Returns the state of the **Reduced Integration** flag for the specified plate property. This option is only used for the 2D and Axisymmetric plate property types, applicable to quadratic elements.

```
long St7GetPlateUseReducedInt(long uID, long PropNum, bool* UseReducedInt)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

Output Parameters

`UseReducedInt`

True if a reduced order integration scheme is used.

St7SetPlateAddBubbleFunction

Sets the state of the **Add Bubble Function** option for the specified plate property. This option is only used for Quad4 plane stress, plane strain and axisymmetric element types.

```
long St7SetPlateAddBubbleFunction(long uID, long PropNum, bool AddBubbleFunction)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

`AddBubbleFunction`

True to add the “bubble” contribution to the element shape functions.

St7GetPlateAddBubbleFunction

Returns the state of the **Add Bubble Function** option for the specified plate property. This option is only used for Quad4 plane stress, plane strain and axisymmetric element types.

```
long St7GetPlateAddBubbleFunction(long uID, long PropNum,
     bool* AddBubbleFunction)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

Output Parameters

`AddBubbleFunction`

True if the “bubble” contribution is added to the element shape functions.

St7SetPlateLayers

Sets the number of layers used for MNL integrations through the plate thickness. The default is ten, and a maximum of 100 layers may be set.

```
long St7SetPlateLayers(long uID, long PropNum, long NumLayers)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

`NumLayers`

Number of integration layers.

St7GetPlateLayers

Returns the number of layers used for MNL integrations through the plate thickness.

```
long St7GetPlateLayers(long uID, long PropNum, long* NumLayers)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

Output Parameters

`NumLayers`

Number of integration layers.

St7SetPlatePatchTol

Sets the relative tolerance for load patch plate properties.

```
long St7SetPlatePatchTol(long uID, long PropNum, double PatchTol)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

`PatchTol`

Relative tolerance.

St7GetPlatePatchTol

Returns the relative tolerance for load patch plate properties.

```
long St7GetPlatePatchTol(long uID, long PropNum, double* PatchTol)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Plate property number.

Output Parameters

`PatchTol`

Relative tolerance.

St7NewBrickProperty

Creates a new brick property.

```
long St7NewBrickProperty(long uID, long PropNum, long MaterialType,
     char* PropName)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

`MaterialType`

One of mtIsotropic, mtOrthotropic, mtAnisotropic, mtRubber, mtSoil or mtFluid.

`PropName`

Name of the property.

St7SetBrickPropertyType

Sets the material type for the specified brick property.

```
long St7SetBrickPropertyType(long uID, long PropNum, long MaterialType)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

`MaterialType`

One of mtIsotropic, mtOrthotropic, mtAnisotropic, mtRubber, mtSoil or mtFluid.

St7GetBrickPropertyType

Returns the material type for the specified brick property.

```
long St7GetBrickPropertyType(long uID, long PropNum, long* MaterialType)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

Output Parameters

`MaterialType`

One of mtNull, mtIsotropic, mtOrthotropic, mtAnisotropic, mtRubber, mtSoil or mtFluid.

St7SetBrickNonlinearType

Sets the nonlinear material type for the specified brick property.

```
long St7SetBrickNonlinearType(long uID, long PropNum, long NonlinType,
     long YieldType)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

`NonlinType`

Nonlinear material type; either ntNonlinElastic or ntElastoPlastic.

`YieldType`

One of ycTresca, ycVonMises, ycMaxStress, ycMohrCoulomb or ycDruckerPrager.

St7GetBrickNonlinearType

Returns the nonlinear material type assigned to the specified brick property.

```
long St7GetBrickNonlinearType(long uID, long PropNum, long* NonlinType,
     long* YieldType)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

Output Parameters

`NonlinType`

Nonlinear material type; either ntNonlinElastic or ntElastoPlastic.

`YieldType`

One of ycTresca, ycVonMises, ycMaxStress, ycMohrCoulomb or ycDruckerPrager.

St7SetBrickIsotropicMaterial

Set the isotropic material properties for the specified brick property.

```
long St7SetBrickIsotropicMaterial(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

`Doubles[0..7]`

`[ipBrickIsoModulus]` – Modulus.

`[ipBrickIsoPoisson]` – Poisson's ratio.

`[ipBrickIsoDensity]` – Mass density per unit volume.

`[ipBrickIsoAlpha]` – Thermal expansion coefficient.

`[ipBrickIsoViscosity]` – Viscous damping coefficient.

`[ipBrickIsoDampingRatio]` – Damping ratio.

`[ipBrickIsoConductivity]` – Conductivity coefficient.

`[ipBrickIsoSpecificHeat]` – Specific heat coefficient.

St7GetBrickIsotropicMaterial

Returns the isotropic material properties assigned to the specified brick property.

```
long St7GetBrickIsotropicMaterial(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

Output Parameters

`Doubles[0..7]`

`[ipBrickIsoModulus]` – Modulus.

`[ipBrickIsoPoisson]` – Poisson's ratio.

`[ipBrickIsoDensity]` – Mass density per unit volume.

`[ipBrickIsoAlpha]` – Thermal expansion coefficient.

`[ipBrickIsoViscosity]` – Viscous damping coefficient.

`[ipBrickIsoDampingRatio]` – Damping ratio.

`[ipBrickIsoConductivity]` – Conductivity coefficient.

`[ipBrickIsoSpecificHeat]` – Specific heat coefficient.

St7SetBrickOrthotropicMaterial

Sets the orthotropic material properties for the specified brick property.

```
long St7SetBrickOrthotropicMaterial(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

`Doubles[0..18]`

`[ipBrickOrthoModulus1]` – Modulus in the 1 axis direction.

`[ipBrickOrthoModulus2]` – Modulus in the 2 axis direction.

`[ipBrickOrthoModulus3]` – Modulus in the 3 axis direction.

`[ipBrickOrthoShear12]` – Shear modulus in the 12 axis direction.

`[ipBrickOrthoShear23]` – Shear modulus in the 23 axis direction.

`[ipBrickOrthoShear31]` – Shear modulus in the 31 axis direction.

`[ipBrickOrthoPoisson12]` – Poisson's ratio in the 12 axis direction.

`[ipBrickOrthoPoisson23]` – Poisson's ratio in the 23 axis direction.

`[ipBrickOrthoPoisson31]` – Poisson's ratio in the 31 axis direction.

`[ipBrickOrthoDensity]` – Mass density per unit volume.

`[ipBrickOrthoAlpha1]` – Thermal expansion coefficient in the 1 axis direction.

`[ipBrickOrthoAlpha2]` – Thermal expansion coefficient in the 2 axis direction.

[ipBrickOrthoAlpha3] – Thermal expansion coefficient in the 3 axis direction.

[ipBrickOrthoViscosity] – Viscous damping coefficient.

[ipBrickOrthoDampingRatio] – Damping ratio.

[ipBrickOrthoConductivity1] – Conductivity coefficient in the 1 axis direction.

[ipBrickOrthoConductivity2] – Conductivity coefficient in the 2 axis direction.

[ipBrickOrthoConductivity3] – Conductivity coefficient in the 3 axis direction.

[ipBrickOrthoSpecificHeat] – Specific heat coefficient.

St7GetBrickOrthotropicMaterial

Returns the orthotropic material properties assigned to the specified brick property.

```
long St7GetBrickOrthotropicMaterial(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

Output Parameters

`Doubles[0..18]`

[ipBrickOrthoModulus1] – Modulus in the 1 axis direction.

[ipBrickOrthoModulus2] – Modulus in the 2 axis direction.

[ipBrickOrthoModulus3] – Modulus in the 3 axis direction.

[ipBrickOrthoShear12] – Shear modulus in the 12 axis direction.

[ipBrickOrthoShear23] – Shear modulus in the 23 axis direction.

[ipBrickOrthoShear31] – Shear modulus in the 31 axis direction.

[ipBrickOrthoPoisson12] – Poisson’s ratio in the 12 axis direction.

[ipBrickOrthoPoisson23] – Poisson’s ratio in the 23 axis direction.

[ipBrickOrthoPoisson31] – Poisson’s ratio in the 31 axis direction.

[ipBrickOrthoDensity] – Mass density per unit volume.

[ipBrickOrthoAlpha1] – Thermal expansion coefficient in the 1 axis direction.

[ipBrickOrthoAlpha2] – Thermal expansion coefficient in the 2 axis direction.

[ipBrickOrthoAlpha3] – Thermal expansion coefficient in the 3 axis direction.

[ipBrickOrthoViscosity] – Viscous damping coefficient.

`[ipBrickOrthoDampingRatio]` – Damping ratio.

`[ipBrickOrthoConductivity1]` – Conductivity coefficient in the 1 axis direction.

`[ipBrickOrthoConductivity2]` – Conductivity coefficient in the 2 axis direction.

`[ipBrickOrthoConductivity3]` – Conductivity coefficient in the 3 axis direction.

`[ipBrickOrthoSpecificHeat]` – Specific heat coefficient.

St7SetBrickAnisotropicMaterial

Sets the anisotropic material properties for the specified brick property.

```
long St7SetBrickAnisotropicMaterial(long uID, long PropNum, long MatType,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

`MatType`

Matrix type; either mtStiffness or mtCompliance.

`Doubles[0..33]`

`[0..20]` – Complete anisotropic material stress-strain matrix D defined by the upper triangular matrix of coefficients D_{ij} where $i < j$ and i varies quickest: D_{11}, D_{12},… D_{22}, D_{23},… D_{66}, respectively.

`[ipBrickAnisoDensity]` – Mass density per unit volume.

`[ipBrickAnisoAlpha1]` – Thermal expansion coefficient in the 1 axis direction.

`[ipBrickAnisoAlpha2]` – Thermal expansion coefficient in the 2 axis direction.

`[ipBrickAnisoAlpha3]` – Thermal expansion coefficient in the 3 axis direction.

`[ipBrickAnisoAlpha12]` – Thermal expansion coefficient in the 12 axis direction.

`[ipBrickAnisoAlpha23]` – Thermal expansion coefficient in the 23 axis direction.

`[ipBrickAnisoAlpha31]` – Thermal expansion coefficient in the 31 axis direction.

`[ipBrickAnisoViscosity]` – Viscous damping coefficient.

`[ipBrickAnisoDampingRatio]` – Damping ratio.

`[ipBrickAnisoConductivity1]` – Conductivity coefficient in the 1 axis direction.

`[ipBrickAnisoConductivity2]` – Conductivity coefficient in the 2 axis direction.

`[ipBrickAnisoConductivity3]` – Conductivity coefficient in the 3 axis direction.

`[ipBrickAnisoSpecificHeat]` – Specific heat coefficient.

St7GetBrickAnisotropicMaterial

Returns the anisotropic material properties assigned to the specified brick property.

```
long St7GetBrickAnisotropicMaterial(long uID, long PropNum, long* MatType,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

Output Parameters

`MatType`

Matrix type; either mtStiffness or mtCompliance.

`Doubles[0..33]`

`[0..20]` – Complete anisotropic material stress-strain matrix D defined by the upper triangular matrix of coefficients D_{ij} where $i < j$ and i varies quickest: D_{11}, D_{12},… D_{22}, D_{23},… D_{66}, respectively.

`[ipBrickAnisoDensity]` – Mass density per unit volume.

`[ipBrickAnisoAlpha1]` – Thermal expansion coefficient in the 1 axis direction.

`[ipBrickAnisoAlpha2]` – Thermal expansion coefficient in the 2 axis direction.

`[ipBrickAnisoAlpha3]` – Thermal expansion coefficient in the 3 axis direction.

`[ipBrickAnisoAlpha12]` – Thermal expansion coefficient in the 12 axis direction.

`[ipBrickAnisoAlpha23]` – Thermal expansion coefficient in the 23 axis direction.

`[ipBrickAnisoAlpha31]` – Thermal expansion coefficient in the 31 axis direction.

`[ipBrickAnisoViscosity]` – Viscous damping coefficient.

`[ipBrickAnisoDampingRatio]` – Damping ratio.

`[ipBrickAnisoConductivity1]` – Conductivity coefficient in the 1 axis direction.

`[ipBrickAnisoConductivity2]` – Conductivity coefficient in the 2 axis direction.

`[ipBrickAnisoConductivity3]` – Conductivity coefficient in the 3 axis direction.

`[ipBrickAnisoSpecificHeat]` – Specific heat coefficient.

St7SetBrickRubberMaterial

Sets the rubber material properties for the specified brick property.

```
long St7SetBrickRubberMaterial(long uID, long PropNum, long RubberType,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

`RubberType`

One of rtNeoHookean, rtMooneyRivlin or rtGeneralisedMooneyRivlin.

`Doubles[0..15]`

An array describing the rubber material coefficients. The format depends on the material sub-type, with different sub-types requiring a varying number of rubber coefficients following the common data:

`[ipRubberBulk]` – Bulk modulus.

`[ipRubberDensity]` – Mass density per unit volume.

`[ipRubberAlpha]` – Thermal expansion coefficient.

`[ipRubberViscosity]` – Viscous damping coefficient.

`[ipRubberDampingRatio]` – Damping ratio.

`[ipRubberConductivity]` – Conductivity.

`[ipRubberSpecificHeat]` – Specific heat.

`[ipRubberConstC1..ipRubberConstC1+Num]` – Rubber coefficients,

where:
`Num = 0` (Neo-Hookean)
`Num = 1` (Mooney-Rivlin)
`Num = 8` (Generalised Mooney-Rivlin)

St7GetBrickRubberMaterial

Returns the rubber material properties assigned to the specified brick property.

```
long St7GetBrickRubberMaterial(long uID, long PropNum, long* RubberType,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

Output Parameters

`RubberType`

One of rtNeoHookean, rtMooneyRivlin or rtGeneralisedMooneyRivlin.

`Doubles[0..15]`

An array describing the rubber material coefficients. The format depends on the material sub-type, with different sub-types requiring a varying number of rubber coefficients following the common data:

`[ipRubberBulk]` – Bulk modulus.

`[ipRubberDensity]` – Mass density per unit volume.

`[ipRubberAlpha]` – Thermal expansion coefficient.

`[ipRubberViscosity]` – Viscous damping coefficient.

`[ipRubberDampingRatio]` – Damping ratio.

`[ipRubberConductivity]` – Conductivity.

`[ipRubberSpecificHeat]` – Specific heat.

`[ipRubberConstC1..ipRubberConstC1+Num]` – Rubber coefficients,

where:
`Num = 0` (Neo-Hookean)
`Num = 1` (Mooney-Rivlin)
`Num = 8` (Generalised Mooney-Rivlin)

St7SetBrickMCDPMaterial

Sets the Mohr-Coulomb and Drucker-Prager material properties for the specified brick property. Note that these properties correspond to the isotropic yield criterion; to set soil material properties with these yield criteria use *St7SetBrickSoilMCMaterial* or *St7SetBrickSoilDPMaterial*.

```
long St7SetBrickMCDPMaterial(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

`Doubles[0..1]`

`[ipFrictionAngle]` – Friction angle.

`[ipCohesion]` – Cohesion value.

St7GetBrickMCDPMaterial

Returns the Mohr-Coulomb and Drucker-Prager material properties assigned to the specified brick property. Note that these properties correspond to the isotropic yield criterion; to get soil material properties with these yield criteria use *St7GetBrickSoilMCMaterial* or *St7GetBrickSoilDPMaterial*.

```
long St7GetBrickMCDPMaterial(long uID, long PropNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

Output Parameters

`Doubles[0..1]`

`[ipFrictionAngle]` – Friction angle.

`[ipCohesion]` – Cohesion value.

St7GetBrickSoilType

Returns the soil type for a brick property assigned as a soil material.

```
long St7GetBrickSoilType(long uID, long PropNum, long* SoilType)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

Output Parameters

`SoilType`

One of stDuncanChang, stModifiedCamClay, stMohrCoulomb, stDruckerPrager or stLinearElastic.

St7SetBrickSoilDCMaterial

Sets the Duncan-Chang soil material properties for the specified brick property.

```
long St7SetBrickSoilDCMaterial(long uID, long PropNum, long* Integers,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

`Integers[0..2]`

`[ipSoilDCUseBulkModulus]` – Use bulk modulus; either btTrue or btFalse.

`[ipSoilDCSetPropLevel]` – btTrue to set fluid level to Property value; btFalse to set fluid level to Stage/Global level.

`[ipSoilDCDrainedState]` – Drained state; either btTrue or btFalse.

`Doubles[0..19]`

`[ipSoilDCModulusK]` – Modulus.

`[ipSoilDCModulusKUR]` – Unloading/reloading modulus.

`[ipSoilDCModulusN]` – Modulus exponent.

`[ipSoilDCPoisson]` – Poisson's ratio.

`[ipSoilDCBulkK]` – Bulk modulus.

`[ipSoilDCBulkM]` – Bulk modulus exponent.

`[ipSoilDCFrictionAngle]` – Friction angle.

`[ipSoilDCDeltaAngle]` – Friction angle change.

`[ipSoilDCCohesion]` – Cohesion value.

`[ipSoilDCFailureRatio]` – Failure ratio.

`[ipSoilDCFailureMod]` – Failure modulus.

`[ipSoilDCReferenceP]` – Reference pressure.

`[ipSoilDCDensity]` – Mass density per unit volume.

`[ipSoilDCHorizontalRatio]` – Horizontal stress ratio.

`[ipSoilDCER]` – Reference void ratio.

`[ipSoilDCConductivity]` – Conductivity.

`[ipSoilDCSpecificHeat]` – Specific heat.

`[ipSoilDCFluidLevel]` – Fluid level.

`[ipSoilDCViscosity]` – Viscous damping coefficient.

`[ipSoilDCDampingRatio]` – Damping ratio.

St7GetBrickSoilDCMaterial

Returns the Duncan-Chang soil material properties assigned to the specified brick property.

```
long St7GetBrickSoilDCMaterial(long uID, long PropNum, long* Integers,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

Output Parameters

`Integers[0..2]`

`[ipSoilDCUseBulkModulus]` – Use bulk modulus; either btTrue or btFalse.

`[ipSoilDCSetPropLevel]` – btTrue to set fluid level to Property value; btFalse to set fluid level to Stage/Global level.

`[ipSoilDCDrainedState]` – Drained state; either btTrue or btFalse.

`Doubles[0..19]`

`[ipSoilDCModulusK]` – Modulus.

`[ipSoilDCModulusKUR]` – Unloading/reloading modulus.

`[ipSoilDCModulusN]` – Modulus exponent.

`[ipSoilDCPoisson]` – Poisson's ratio.

`[ipSoilDCBulkK]` – Bulk modulus.

`[ipSoilDCBulkM]` – Bulk modulus exponent.

`[ipSoilDCFrictionAngle]` – Friction angle.

`[ipSoilDCDeltaAngle]` – Friction angle change.

`[ipSoilDCCohesion]` – Cohesion value.

`[ipSoilDCFailureRatio]` – Failure ratio.

`[ipSoilDCFailureMod]` – Failure modulus.

`[ipSoilDCReferenceP]` – Reference pressure.

`[ipSoilDCDensity]` – Mass density per unit volume.

`[ipSoilDCHorizontalRatio]` – Horizontal stress ratio.

`[ipSoilDCER]` – Reference void ratio.

`[ipSoilDCConductivity]` – Conductivity.

`[ipSoilDCSpecificHeat]` – Specific heat.

`[ipSoilDCFluidLevel]` – Fluid level.

`[ipSoilDCViscosity]` – Viscous damping coefficient.

`[ipSoilDCDampingRatio]` – Damping ratio.

St7SetBrickSoilCCMaterial

Sets the Cam-Clay soil material properties for the specified brick property.

```
long St7SetBrickSoilCCMaterial(long uID, long PropNum, long* Integers,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

`Integers[0..3]`

`[ipSoilCCUsePoisson]` – Use Poisson's ratio; either btTrue or btFalse.

`[ipSoilCCDrainedState]` – Drained state; either btTrue or btFalse.

`[ipSoilCCUseOCR]` – Overconsolidation; either btTrue or btFalse.

`[ipSoilCCSetPropLevel]` – btTrue to set fluid level to Property value; btFalse to set fluid level to Stage/Global level.

`Doubles[0..16]`

`[ipSoilCCCriticalStateLine]` – Critical state line slope.

`[ipSoilCCConsolidationLine]` – Normal consolidation line slope.

`[ipSoilCCSwellingLine]` – Swelling line slope.

`[ipSoilCCDensity]` – Mass density per unit volume.

`[ipSoilCCPoisson]` – Poisson's ratio.

`[ipSoilCCModulusGa]` – Shear modulus at point A.

`[ipSoilCCModulusGb]` – Shear modulus at point B.

`[ipSoilCCHorizontalRatio]` – Horizontal stress ratio.

`[ipSoilCCER]` – Reference void ratio.

`[ipSoilCCPR]` – Unit pressure ratio.

`[ipSoilCCPC0]` – Initial consolidation pressure.

`[ipSoilCCOCR]` – Overconsolidation ratio.

`[ipSoilCCConductivity]` – Conductivity.

`[ipSoilCCSpecificHeat]` – Specific heat.

`[ipSoilCCFluidLevel]` – Fluid level.

`[ipSoilCCViscosity]` – Viscous damping coefficient.

`[ipSoilCCDampingRatio]` – Damping ratio.

St7GetBrickSoilCCMaterial

Returns the Cam-Clay soil material properties assigned to the specified brick property.

```
long St7GetBrickSoilCCMaterial(long uID, long PropNum, long* Integers,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

Output Parameters

`Integers[0..3]`

`[ipSoilCCUsePoisson]` – Use Poisson's ratio; either btTrue or btFalse.

`[ipSoilCCDrainedState]` – Drained state; either btTrue or btFalse.

`[ipSoilCCUseOCR]` – Over-consolidation; either btTrue or btFalse.

`[ipSoilCCSetPropLevel]` – btTrue to set fluid level to Property value; btFalse to set fluid level to Stage/Global level.

`Doubles[0..16]`

`[ipSoilCCCriticalStateLine]` – Critical state line slope.

`[ipSoilCCConsolidationLine]` – Normal consolidation line slope.

`[ipSoilCCSwellingLine]` – Swelling line slope.

`[ipSoilCCDensity]` – Mass density per unit volume.

`[ipSoilCCPoisson]` – Poisson's ratio.

`[ipSoilCCModulusGa]` – Shear modulus at point A.

`[ipSoilCCModulusGb]` – Shear modulus at point B.

`[ipSoilCCHorizontalRatio]` – Horizontal stress ratio.

`[ipSoilCCER]` – Reference void ratio.

`[ipSoilCCPR]` – Unit pressure ratio.

`[ipSoilCCPC0]` – Initial consolidation pressure.

`[ipSoilCCOCR]` – Overconsolidation ratio.

`[ipSoilCCConductivity]` – Conductivity.

[ipSoilCCSpecificHeat] – Specific heat.

[ipSoilCCFluidLevel] – Fluid level.

[ipSoilCCViscosity] – Viscous damping coefficient.

[ipSoilCCDampingRatio] – Damping ratio.

St7SetBrickSoilMCMaterial

Assigns the Mohr-Coulomb soil parameters for the specified brick property.

```
long St7SetBrickSoilMCMaterial(long uID, long PropNum, long* Integers,
     double* Doubles)
```

Input Parameters

uID

Strand7 model file ID.

PropNum

Brick property number.

Integers[0..1]

[ipSoilMCSetPropLevel] – btTrue to set fluid level to Property value; btFalse to set fluid level to Stage/Global level.

[ipSoilMCDrainedState] – Drained state; either btTrue or btFalse.

Doubles[0..11]

[ipSoilMCModulus] – Modulus.

[ipSoilMCPoisson] – Poisson’s ratio.

[ipSoilMCDensity] – Mass density per unit volume.

[ipSoilMCCohesion] – Cohesion value.

[ipSoilMCFrictionAngle] – Friction angle.

[ipSoilMCHorizontalRatio] – Horizontal stress ratio.

[ipSoilMCER] – Void ratio.

[ipSoilMCConductivity] – Conductivity.

[ipSoilMCSpecificHeat] – Specific heat.

[ipSoilMCFluidLevel] – Fluid level.

[ipSoilMCViscosity] – Viscous damping coefficient.

[ipSoilMCDampingRatio] – Damping ratio.

St7GetBrickSoilMCMaterial

Returns the Mohr-Coulomb soil parameters assigned to the specified brick property.

```
long St7GetBrickSoilMCMaterial(long uID, long PropNum, long* Integers,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

Output Parameters

`Integers[0..1]`

`[ipSoilMCSetPropLevel]` – btTrue to set fluid level to Property value; btFalse to set fluid level to Stage/Global level.

`[ipSoilMCDrainedState]` – Drained state; either btTrue or btFalse.

`Doubles[0..11]`

`[ipSoilMCModulus]` – Modulus.

`[ipSoilMCPoisson]` – Poisson’s ratio.

`[ipSoilMCDensity]` – Mass density per unit volume.

`[ipSoilMCCohesion]` – Cohesion value.

`[ipSoilMCFrictionAngle]` – Friction angle.

`[ipSoilMCHorizontalRatio]` – Horizontal stress ratio.

`[ipSoilMCER]` – Void ratio.

`[ipSoilMCConductivity]` – Conductivity.

`[ipSoilMCSpecificHeat]` – Specific heat.

`[ipSoilMCFluidLevel]` – Fluid level.

`[ipSoilMCViscosity]` – Viscous damping coefficient.

`[ipSoilMCDampingRatio]` – Damping ratio.

St7SetBrickSoilDPMaterial

Assigns the Drucker-Prager soil parameters for the specified brick property.

```
long St7SetBrickSoilDPMaterial(long uID, long PropNum, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

`Integers[0..1]`

`[ipSoilDPSetPropLevel]` – btTrue to set fluid level to Property value; btFalse to set fluid level to Stage/Global level.

`[ipSoilDPDrainedState]` – Drained state; either btTrue or btFalse.

`Doubles[0..11]`

`[ipSoilDPModulus]` – Modulus.

`[ipSoilDPPoisson]` – Poisson's ratio.

`[ipSoilDPDensity]` – Mass density per unit volume.

`[ipSoilDPCohesion]` – Cohesion value.

`[ipSoilDPFrictionAngle]` – Friction angle.

`[ipSoilDPHorizontalRatio]` – Horizontal stress ratio.

`[ipSoilDPER]` – Void ratio.

`[ipSoilDPConductivity]` – Conductivity.

`[ipSoilDPSpecificHeat]` – Specific heat.

`[ipSoilDPFluidLevel]` – Fluid level.

`[ipSoilDPViscosity]` – Viscous damping coefficient.

`[ipSoilDPDampingRatio]` – Damping ratio.

St7GetBrickSoilDPMaterial

Returns the Drucker-Prager soil parameters assigned to the specified brick property.

```
long St7GetBrickSoilDPMaterial(long uID, long PropNum, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

Output Parameters

`Integers[0..1]`

`[ipSoilDPSetPropLevel]` – btTrue to set fluid level to Property value; btFalse to set fluid level to Stage/Global level.

`[ipSoilDPDrainedState]` – Drained state; either btTrue or btFalse.

`Doubles[0..11]`

`[ipSoilDPModulus]` – Modulus.

`[ipSoilDPPoisson]` – Poisson's ratio.

`[ipSoilDPDensity]` – Mass density per unit volume.

`[ipSoilDPCohesion]` – Cohesion value.

`[ipSoilDPFrictionAngle]` – Friction angle.

`[ipSoilDPHorizontalRatio]` – Horizontal stress ratio.

`[ipSoilDPER]` – Void ratio.

`[ipSoilDPConductivity]` – Conductivity.

`[ipSoilDPSpecificHeat]` – Specific heat.

`[ipSoilDPFluidLevel]` – Fluid level.

`[ipSoilDPViscosity]` – Viscous damping coefficient.

`[ipSoilDPDampingRatio]` – Damping ratio.

St7SetBrickSoilLSMaterial

Assigns the linear elastic soil parameters for the specified brick property.

```
long St7SetBrickSoilLSMaterial(long uID, long PropNum, long* Integers,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

`Integers[0..1]`

`[ipSoilLSSetPropLevel]` – btTrue to set fluid level to Property value; btFalse to set fluid level to Stage/Global level.

`[ipSoilLSDrainedState]` – Drained state; either btTrue or btFalse.

`Doubles[0..9]`

`[ipSoilLSModulus]` – Modulus.

`[ipSoilLSPoisson]` – Poisson's ratio.

`[ipSoilLSDensity]` – Mass density per unit volume.

`[ipSoilLSHorizontalRatio]` – Horizontal stress ratio.

`[ipSoilLSER]` – Void ratio.

`[ipSoilLSConductivity]` – Conductivity.

`[ipSoilLSSpecificHeat]` – Specific heat.

`[ipSoilLSFluidLevel]` – Fluid level.

`[ipSoilLSViscosity]` – Viscous damping coefficient.

`[ipSoilLSDampingRatio]` – Damping ratio.

St7GetBrickSoilLSMaterial

Returns the linear elastic soil parameters assigned to the specified brick property.

```
long St7GetBrickSoilLSMaterial(long uID, long PropNum, long* Integers,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

Output Parameters

`Integers[0..1]`

`[ipSoilLSSetPropLevel]` – btTrue to set fluid level to Property value; btFalse to set fluid level to Stage/Global level.

`[ipSoilLSDrainedState]` – Drained state; either btTrue or btFalse.

`Doubles[0..9]`

`[ipSoilLSModulus]` – Modulus.

`[ipSoilLSPoisson]` – Poisson's ratio.

`[ipSoilLSDensity]` – Mass density per unit volume.

`[ipSoilLSHorizontalRatio]` – Horizontal stress ratio.

`[ipSoilLSER]` – Void ratio.

[ipSoilLSConductivity] – Conductivity.

[ipSoilLSSpecificHeat] – Specific heat.

[ipSoilLSFluidLevel] – Fluid level.

[ipSoilLSViscosity] – Viscous damping coefficient.

[ipSoilLSDampingRatio] – Damping ratio.

St7SetBrickFluidMaterial

Sets the fluid material properties for the specified brick property.

```
long St7SetBrickFluidMaterial(long uID, long PropNum, double* Doubles)
```

Input Parameters

uID

Strand7 model file ID.

PropNum

Brick property number.

Doubles[0..7]

[ipFluidModulus] – Modulus.

[ipFluidPenaltyParam] – Penalty parameter.

[ipFluidDensity] – Mass density per unit volume.

[ipFluidAlpha] – Thermal expansion coefficient.

[ipFluidViscosity] – Viscous damping coefficient.

[ipFluidDampingRatio] – Damping ratio.

[ipFluidConductivity] – Conductivity.

[ipFluidSpecificHeat] – Specific heat.

St7GetBrickFluidMaterial

Returns the fluid material properties assigned to the specified brick property.

```
long St7GetBrickFluidMaterial(long uID, long PropNum, double* Doubles)
```

Input Parameters

uID

Strand7 model file ID.

PropNum

Brick property number.

Output Parameters

`Doubles[0..7]`

`[ipFluidModulus]` – Modulus.

`[ipFluidPenaltyParam]` – Penalty parameter.

`[ipFluidDensity]` – Mass density per unit volume.

`[ipFluidAlpha]` – Thermal expansion coefficient.

`[ipFluidViscosity]` – Viscous damping coefficient.

`[ipFluidDampingRatio]` – Damping ratio.

`[ipFluidConductivity]` – Conductivity.

`[ipFluidSpecificHeat]` – Specific heat.

St7SetBrickAddBubbleFunction

Sets the state of the **Add Bubble Function** option for the specified brick property. This option is only used for Hex8 element types.

```
long St7SetBrickAddBubbleFunction(long uID, long PropNum, bool AddBubbleFunction)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

`AddBubbleFunction`

True to add the “bubble” contribution to the element shape functions.

St7GetBrickAddBubbleFunction

Returns the state of the **Add Bubble Function** option for the specified brick property. This option is only used for Hex8 element types.

```
long St7GetBrickAddBubbleFunction(long uID, long PropNum,
     bool* AddBubbleFunction)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

Output Parameters

`AddBubbleFunction`

True if the “bubble” contribution is added to the element shape functions.

St7SetBrickIntegrationPoints

Sets the number of integration (Gauss) points used by the solver along each intrinsic coordinate direction.

```
long St7SetBrickIntegrationPoints(long uID, long PropNum, long Xi, long Eta,
    long Zeta)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

`Xi`

Number of integration points in the first intrinsic ordinate; one of 1, 2 or 3.

`Eta`

Number of integration points in the second intrinsic ordinate; one of 1, 2 or 3.

`Zeta`

Number of integration points in the third intrinsic ordinate; one of 1, 2 or 3.

St7GetBrickIntegrationPoints

Returns the number of integration (Gauss) points used by the solver along each intrinsic coordinate direction.

```
long St7GetBrickIntegrationPoints(long uID, long PropNum, long* Xi, long* Eta,
    long* Zeta)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Brick property number.

Output Parameters

`Xi`

Number of integration points in the first intrinsic ordinate.

`Eta`

Number of integration points in the second intrinsic ordinate.

`Zeta`

Number of integration points in the third intrinsic ordinate.

St7DeleteProperty

Deletes the specified material property.

```
long St7DeleteProperty(long uID, long Entity, long PropNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP, ptBRICKPROP or ptPLYPROP.

`PropNum`

Entity property number.

St7DeleteUnusedProperties

Deletes all unused properties in the specified model.

```
long St7DeleteUnusedProperties(long uID, long Entity, long* NumDeleted)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP, ptBRICKPROP or ptPLYPROP.

Output Parameters

`NumDeleted`

Number of properties deleted.

Properties – Plies

St7NewPlyProperty

Creates a new ply property.

```
long St7NewPlyProperty(long uID, long PropNum, char* PropName)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Ply property number.

`PropName`

Name of the property.

St7SetPlyMaterial

Sets the material properties for the specified ply property.

```
long St7SetPlyMaterial(long uID, long PropNum, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Ply property number.

`Integers[0..0]`

`[ipPlyWeaveType]` – Weave type; one of wtPlyUniDirectional, wtPlyBiDirectional, wtPlyTriDirectional or wtPlyQuasiIsotropic.

`Doubles[0..20]`

`[ipPlyModulus1]` – Modulus in the 1 axis direction.

`[ipPlyModulus2]` – Modulus in the 2 axis direction.

`[ipPlyPoisson]` – Poisson's ratio.

`[ipPlyShear12]` – Shear modulus in the 12 axis direction.

`[ipPlyShear13]` – Shear modulus in the 13 axis direction.

`[ipPlyShear23]` – Shear modulus in the 23 axis direction.

`[ipPlyAlpha1]` – Thermal expansion coefficient in the 1 axis direction.

`[ipPlyAlpha2]` – Thermal expansion coefficient in the 2 axis direction.

`[ipPlyDensity]` – Mass density per unit volume.

`[ipPlyThickness]` – Thickness.

`[ipPlyS1Tension]` – Tensile stress limit in the 1 axis direction.

`[ipPlyS2Tension]` – Tensile stress limit in the 2 axis direction.

`[ipPlyS1Compression]` – Compressive stress limit in the 1 axis direction.

`[ipPlyS2Compression]` – Compressive stress limit in the 2 axis direction.

`[ipPlySShear]` – Shear stress limit.

`[ipPlyE1Tension]` – Tensile strain limit in the 1 axis direction.

`[ipPlyE2Tension]` – Tensile strain limit in the 2 axis direction.

`[ipPlyE1Compression]` – Compressive strain limit in the 1 axis direction.

`[ipPlyE2Compression]` – Compressive strain limit in the 2 axis direction.

`[ipPlyEShear]` – Shear strain limit.

`[ipPlyInterLaminaShear]` – Interlamina shear stress limit.

St7GetPlyMaterial

Returns the material properties assigned to the specified ply property.

```
long St7GetPlyMaterial(long uID, long PropNum, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Ply property number.

Output Parameters

`Integers[0..0]`

`[ipPlyWeaveType]` – Weave type; one of wtPlyUniDirectional, wtPlyBiDirectional, wtPlyTriDirectional or wtPlyQuasiIsotropic.

`Doubles[0..20]`

`[ipPlyModulus1]` – Modulus in the 1 axis direction.

`[ipPlyModulus2]` – Modulus in the 2 axis direction.

`[ipPlyPoisson]` – Poisson’s ratio.

`[ipPlyShear12]` – Shear modulus in the 12 axis direction.

`[ipPlyShear13]` – Shear modulus in the 13 axis direction.

[ipPlyShear23] – Shear modulus in the 23 axis direction.

[ipPlyAlpha1] – Thermal expansion coefficient in the 1 axis direction.

[ipPlyAlpha2] – Thermal expansion coefficient in the 2 axis direction.

[ipPlyDensity] – Mass density per unit volume.

[ipPlyThickness] – Thickness.

[ipPlyS1Tension] – Tensile stress limit in the 1 axis direction.

[ipPlyS2Tension] – Tensile stress limit in the 2 axis direction.

[ipPlyS1Compression] – Compressive stress limit in the 1 axis direction.

[ipPlyS2Compression] – Compressive stress limit in the 2 axis direction.

[ipPlySShear] – Shear stress limit.

[ipPlyE1Tension] – Tensile strain limit in the 1 axis direction.

[ipPlyE2Tension] – Tensile strain limit in the 2 axis direction.

[ipPlyE1Compression] – Compressive strain limit in the 1 axis direction.

[ipPlyE2Compression] – Compressive strain limit in the 2 axis direction.

[ipPlyEShear] – Shear strain limit.

[ipPlyInterLaminaShear] – Interlamina shear stress limit.

Properties – Laminates

St7GetTotalLaminateStacks

Returns the total number and highest ID number of the laminate stacks in the specified model.

```
long St7GetTotalLaminateStacks(long uID, long* NumStacks, long* LastStack)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumStacks`

The number of laminate stacks.

`LastStack`

The highest laminate number identifying a laminate stack.

St7GetLaminateStackNumByIndex

Returns the laminate number associated with the specified laminate index. The laminate indices are stored internally and are based on a contiguous numbering system.

```
long St7GetLaminateStackNumByIndex(long uID, long Index, long* LaminateID)
```

Input Parameters

`uID`

Strand7 model file ID.

`Index`

Laminate index.

Output Parameters

`LaminateID`

Laminate layout ID.

St7NewLaminate

Creates a new laminate.

```
long St7NewLaminate(long uID, long LaminateID, char* LamName)
```

Input Parameters

`uID`

Strand7 model file ID.

`LaminateID`

Laminate layout ID.

`LamName`

Name of the laminate.

St7SetLaminateName

Sets the name of the specified laminate.

```
long St7SetLaminateName(long uID, long LaminateID, char* LamName)
```

Input Parameters

`uID`

Strand7 model file ID.

`LaminateID`

Laminate layout ID.

`LamName`

Name of the laminate.

St7GetLaminateName

Returns the name of the specified laminate.

```
long St7GetLaminateName(long uID, long LaminateID, char* LamName,
     long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`LaminateID`

Laminate layout ID.

`MaxStringLen`

Maximum number of characters allocated for LamName.

Output Parameters

`LamName`

Name of the laminate.

St7GetLaminateNumPlies

Returns the number of plies in the specified laminate.

```
long St7GetLaminateNumPlies(long uID, long LaminateID, long* NumPlies)
```

Input Parameters

`uID`

Strand7 model file ID.

`LaminateID`

Laminate layout ID.

Output Parameters

`NumPlies`

Number of plies.

St7SetLaminatePly

Sets the ply property and ply orientation for the specified layer in a laminate.

```
long St7SetLaminatePly(long uID, long LaminateID, long Pos, long PlyPropNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`LaminateID`

Laminate layout ID.

`Pos`

Ply position within the laminate.

`PlyPropNum`

Ply property number.

`Doubles[0..1]`

`[ipLaminatePlyAngle]` – The angle (degrees) between the ply material 1 axis direction and the local x axis of the plate element.

`[ipLaminatePlyThickness]` – Ply thickness. If zero, the thickness defined in the ply property is used.

St7GetLaminatePly

Returns the ply property and ply orientation assigned to the specified layer in a laminate.

```
long St7GetLaminatePly(long uID, long LaminateID, long Pos, long* PlyPropNum,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`LaminateID`

Laminate layout ID.

`Pos`

Ply position within the laminate.

Output Parameters

`PlyPropNum`

Ply property number.

`Doubles[0..1]`

`[ipLaminatePlyAngle]` – The angle (degrees) between the ply material 1 axis direction and the local x axis of the plate element.

`[ipLaminatePlyThickness]` – Ply thickness. If zero, the thickness defined in the ply property is used.

St7AddLaminatePly

Adds a new ply to the specified laminate. The ply is appended to the end of the current laminate stack.

```
long St7AddLaminatePly(long uID, long LaminateID, long PlyPropNum,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`LaminateID`

Laminate layout ID.

`PlyPropNum`

Ply property number.

`Doubles[0..1]`

`[ipLaminatePlyAngle]` – The angle (degrees) between the ply material 1 axis direction and the local x axis of the plate element.

`[ipLaminatePlyThickness]` – Ply thickness. If zero, the thickness defined in the ply property is used.

St7InsertLaminatePly

Inserts a new ply at the specified position within a laminate. Note that if a laminate is at the maximum stack depth of 300 plies, inserting a new ply will cause the existing ply number 300 to be popped off the end of the stack and removed from the laminate.

```
long St7InsertLaminatePly(long uID, long LaminateID, long Pos, long PlyPropNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`LaminateID`

Laminate layout ID.

`Pos`

Ply position within laminate.

`PlyPropNum`

Ply property number.

`Doubles[0..1]`

`[ipLaminatePlyAngle]` – The angle (degrees) between the ply material 1 axis direction and the local x axis of the plate element.

`[ipLaminatePlyThickness]` – Ply thickness. If zero, the thickness defined in the ply property is used.

St7DeleteLaminatePly

Deletes the specified ply from a laminate.

```
long St7DeleteLaminatePly(long uID, long LaminateID, long Pos)
```

Input Parameters

`uID`

Strand7 model file ID.

`LaminateID`

Laminate layout ID.

`Pos`

Ply position within laminate.

St7SetLaminateData

Sets the user-definable data for the specified laminate.

```
long St7SetLaminateData(long uID, long LaminateID, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`LaminateID`

Laminate layout ID.

`Integers[0..1]`

`[ipLaminateIgnoreCoupling]` – Coupled membrane/bending option; either btTrue or btFalse.

`[ipLaminateAutoTransverseShear]` – Automatic transverse shear calculation; either btTrue or btFalse. If btTrue, the Doubles values are ignored and the transverse shear terms are automatically calculated based on the laminate layup.

`Doubles[0..2]`

`[0..2]` – Transverse shear terms of the material stress-strain matrix G_{xz}, G_{yz} and G_{cz} respectively.

St7GetLaminateMatrices

Returns the material matrices for the specified laminate.

```
long St7GetLaminateMatrices(long uID, long LaminateID, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`LaminateID`

Laminate layout ID.

Output Parameters

`Integers[0..2]`

`[ipLaminateIgnoreCoupling]` – Coupled membrane/bending option; either btTrue or btFalse.

`[ipLaminateAutoTransverseShear]` – Automatic transverse shear calculation; either btTrue or btFalse.

`[ipLaminateSingularMatrix]` – Either btTrue or btFalse, depending on whether the laminate matrix is singular or not.

`Doubles[0..23]`

`[0..2]` – Transverse shear terms of the material stress-strain matrix G_{xz}, G_{yz} and G_{cz} respectively.

`[3..8]` – Membrane terms of the material stress-strain matrix defined by the coefficients C_{11}, C_{12}, C_{13}, C_{22}, C_{23} and C_{33} respectively.

`[9..14]` – Bending terms of the material stress-strain matrix defined by the coefficients D_{11}, D_{12}, D_{13}, D_{22}, D_{23} and D_{33} respectively.

`[15..23]` – Coupling terms of the material stress-strain matrix defined by the coefficients B_{11}, B_{12}, B_{13}, B_{21}, B_{22}, B_{23}, B_{31}, B_{32} and B_{33} respectively.

St7DeleteLaminate

Deletes the specified laminate.

```
long St7DeleteLaminate(long uID, long LaminateID)
```

Input Parameters

`uID`

Strand7 model file ID.

`LaminateID`

Laminate layout ID.

St7DeleteUnusedLaminates

Deletes all unused laminates in the specified model.

```
long St7DeleteUnusedLaminates(long uID, long* NumDeleted)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumDeleted`

Number of laminates deleted.

Concrete Reinforcement

St7GetTotalReinforcementLayouts

Returns the total number and highest ID number of the concrete reinforcement layouts in the specified model.

```
long St7GetTotalReinforcementLayouts(long uID, long* NumLayouts,
     long* LastLayout)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumLayouts`

The number of layouts.

`LastLayout`

The highest layout ID.

St7GetReinforcementLayoutNumByIndex

Returns the reinforcement layout number associated with the specified layout index. The reinforcement layout indices are stored internally and are based on a contiguous numbering system.

```
long St7GetReinforcementLayoutNumByIndex(long uID, long Index, long* LayoutID)
```

Input Parameters

`uID`

Strand7 model file ID.

`Index`

Reinforcement layout index.

Output Parameters

`LayoutID`

Reinforcement layout ID.

St7NewReinforcementLayout

Creates a new concrete reinforcement layout.

```
long St7NewReinforcementLayout(long uID, long LayoutID, char* LayoutName)
```

Input Parameters

`uID`

Strand7 model file ID.

`LayoutID`

Reinforcement layout ID.

`LayoutName`

Name of the layout.

St7SetReinforcementName

Sets the name of the specified concrete reinforcement layout.

```
long St7SetReinforcementName(long uID, long LayoutID, char* LayoutName)
```

Input Parameters

`uID`

Strand7 model file ID.

`LayoutID`

Reinforcement layout ID.

`LayoutName`

Name of the layout.

St7GetReinforcementName

Returns the names assigned to the specified concrete reinforcement layout.

```
long St7GetReinforcementName(long uID, long LayoutID, char* LayoutName,
     long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`LayoutID`

Reinforcement layout ID.

`MaxStringLen`

Maximum number of characters allocated to LayoutName.

Output Parameters

`LayoutName`

Name of the layout.

St7SetReinforcementData

Sets the concrete reinforcement data for the specified concrete reinforcement layout.

```
long St7SetReinforcementData(long uID, long LayoutID, long* Integers,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`LayoutID`

Reinforcement layout ID.

`Integers[0..9]`

`[ipRCLayoutType]` – Layup type; either crRCSymmetric or crRCAntiSymmetric.

`[ipRCColour13]` – Layer 13 colour. See also *RGB Colours*.

`[ipRCColour24]` – Layer 24 colour. See also *RGB Colours*.

`[ipRCCalcMethod]` – Either crRCSimplified or crRCElastoPlasticIter.

`[ipRCConsiderMembrane]` – Consider membrane effects; either btTrue or btFalse.

`[ipRCAllowCompressionReo]` – Allow the steel reinforcement to support compression; either btTrue or btFalse.

`[ipRCCode]` – One of crAS3600, crEC2 or crACI318.

`[ipRCLimitConcreteStrain]` – Add reinforcement to limit concrete strain; either btTrue or btFalse.

`[ipRCUseMembraneThickness]` – If btTrue, use plate element membrane thickness, otherwise use plate element bending thickness.

`[ipRCWoodArmerForce]` – one of crRCWoodArmerShearForceMag, crRCWoodArmerShearForceSign or crRCWoodArmerShearForceNone.

`Doubles[0..20]`

`[ipRCDiam1]` – Layer 1 bar diameter.

`[ipRCDiam2]` – Layer 2 bar diameter.

`[ipRCDiam3]` – Layer 3 bar diameter.

`[ipRCDiam4]` – Layer 4 bar diameter.

`[ipRCCover1]` – Cover 1 depth.

`[ipRCCover2]` – Cover 2 depth.

`[ipRCSpacing1]` – Layer 1 bar spacing.

`[ipRCSpacing2]` – Layer 2 bar spacing.

`[ipRCSpacing3]` – Layer 3 bar spacing.

`[ipRCSpacing4]` – Layer 4 bar spacing.

`[ipRCConcreteModulus]` – Concrete modulus.

[ipRCConcreteStrain] – Concrete strain limit.

[ipRCConcreteStress] – Concrete stress limit.

[ipRCConcreteAlpha] – Concrete *alpha* parameter.

[ipRCConcreteGamma] – Concrete *gamma* parameter.

[ipRCSteelModulus] – Steel modulus.

[ipRCSteelStress] – Steel stress limit.

[ipRCSteelGamma] – Steel *gamma* parameter.

[ipRCSteelMinArea] – Steel minimum area value.

[ipRCReduction] – Reduction factor.

[ipRCConcreteEta] – Concrete *eta* parameter.

St7GetReinforcementData

Returns the concrete reinforcement data assigned to the specified concrete reinforcement layout.

```
long St7GetReinforcementData(long uID, long LayoutID, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`LayoutID`

Reinforcement layout ID.

Output Parameters

`Integers[0..9]`

[ipRCLayoutType] – Layup type; either crRCSymmetric or crRCAntiSymmetric.

[ipRCColour13] – Layer 13 colour. See also *RGB Colours*.

[ipRCColour24] – Layer 24 colour. See also *RGB Colours*.

[ipRCCalcMethod] – Either crRCSimplified or crRCElastoPlasticIter.

[ipRCConsiderMembrane] – Consider membrane effects; either btTrue or btFalse.

[ipRCAllowCompressionReo] – Allow the steel reinforcement to support compression; either btTrue or btFalse.

[ipRCCode] – One of crAS3600, crEC2 or crACI318.

[ipRCLimitConcreteStrain] – Add reinforcement to limit concrete strain; either btTrue or btFalse.

[ipRCUseMembraneThickness] – It btTrue, use plate element membrane thickness, otherwise use plate element bending thickness.

`[ipRCWoodArmerForce]` – one of crRCWoodArmerShearForceMag, crRCWoodArmerShearForceSign or crRCWoodArmerShearForceNone.

`Doubles[0..20]`

`[ipRCDiam1]` – Layer 1 bar diameter.

`[ipRCDiam2]` – Layer 2 bar diameter.

`[ipRCDiam3]` – Layer 3 bar diameter.

`[ipRCDiam4]` – Layer 4 bar diameter.

`[ipRCCover1]` – Cover 1 depth.

`[ipRCCover2]` – Cover 2 depth.

`[ipRCSpacing1]` – Layer 1 bar spacing.

`[ipRCSpacing2]` – Layer 2 bar spacing.

`[ipRCSpacing3]` – Layer 3 bar spacing.

`[ipRCSpacing4]` – Layer 4 bar spacing.

`[ipRCConcreteModulus]` – Concrete modulus.

`[ipRCConcreteStrain]` – Concrete strain limit.

`[ipRCConcreteStress]` – Concrete stress limit.

`[ipRCConcreteAlpha]` – Concrete *alpha* parameter.

`[ipRCConcreteGamma]` – Concrete *gamma* parameter.

`[ipRCSteelModulus]` – Steel modulus.

`[ipRCSteelStress]` – Steel stress limit.

`[ipRCSteelGamma]` – Steel *gamma* parameter.

`[ipRCSteelMinArea]` – Steel minimum area value.

`[ipRCReduction]` – Reduction factor.

`[ipRCConcreteEta]` – Concrete *eta* parameter.

St7DeleteReinforcementLayout

Deletes the specified concrete reinforcement layout.

```
long St7DeleteReinforcementLayout(long uID, long LayoutID)
```

Input Parameters

`uID`

Strand7 model file ID.

`LayoutID`

Reinforcement layout ID.

Creep Law Definitions

St7GetTotalCreepDefinitions

Returns the total number and highest ID number of the creep laws in the specified model.

```
long St7GetTotalCreepDefinitions(long uID, long* NumSets, long* LastSet)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumSets`

The number of creep laws.

`LastSet`

The highest creep definition ID.

St7GetCreepDefinitionNumByIndex

Returns the creep law number associated with the specified creep law index. The creep law indices are stored internally and are based on a contiguous numbering system.

```
long St7GetCreepDefinitionNumByIndex(long uID, long Index, long* CreepID)
```

Input Parameters

`uID`

Strand7 model file ID.

`Index`

Creep law index.

Output Parameters

`CreepID`

Creep layout ID.

St7NewCreepDefinition

Creates a new creep law definition.

```
long St7NewCreepDefinition(long uID, long CreepID, char* CreepDefinitionName)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`CreepDefinitionName`

Name of the creep definition.

St7SetCreepDefinitionName

Sets the name of the specified creep law definition.

```
long St7SetCreepDefinitionName(long uID, long CreepID, char* CreepDefinitionName)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`CreepDefinitionName`

Name of the creep definition.

St7GetCreepDefinitionName

Returns the name of the specified creep law definition.

```
long St7GetCreepDefinitionName(long uID, long CreepID, char* CreepDefinitionName,
    long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`MaxStringLen`

Maximum number of characters allocated for CreepDefinitionName.

Output Parameters

`CreepDefinitionName`

Name of the creep definition.

St7SetCreepLaw

Sets the type of creep law assigned to the specified creep definition.

```
long St7SetCreepLaw(long uID, long CreepID, long CreepLaw)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`CreepLaw`

One of clConcreteHyperbolic, clConcreteViscoChain, clConcreteUserDefined, clPrimaryPower, clSecondaryPower, clPrimarySecondaryPower, clSecondaryHyperbolic, clSecondaryExponential, clThetaProjection, clGenGraham, clGenBlackburn or clUserDefined.

St7GetCreepLaw

Returns the type of creep law assigned to the specified creep definition.

```
long St7GetCreepLaw(long uID, long CreepID, long* CreepLaw)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

Output Parameters

`CreepLaw`

One of clConcreteHyperbolic, clConcreteViscoChain, clConcreteUserDefined, clPrimaryPower, clSecondaryPower, clPrimarySecondaryPower, clSecondaryHyperbolic, clSecondaryExponential, clThetaProjection, clGenGraham, clGenBlackburn or clUserDefined.

St7SetCreepBasicData

Sets the basic creep coefficients for the specified creep law definition.

```
long St7SetCreepBasicData(long uID, long CreepID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`Doubles[0..15]`

An array containing the basic creep coefficients. See Creep Definitions for additional information.

St7GetCreepBasicData

Returns the basic creep coefficients assigned to the specified creep definition.

```
long St7GetCreepBasicData(long uID, long CreepID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

Output Parameters

`Doubles[0..15]`

An array containing the basic creep coefficients. See *Creep Definitions* for additional information.

St7EnableCreepUserTable

Enables a user defined Strain vs Time table for the specified creep law definition.

```
long St7EnableCreepUserTable(long uID, long CreepID, long TableID)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`TableID`

User defined Strain vs Time table ID.

St7DisableCreepUserTable

Disables a user defined Strain vs Time table for the specified creep law definition.

```
long St7DisableCreepUserTable(long uID, long CreepID, long TableID)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`TableID`

User defined Strain vs Time table ID.

St7GetCreepUserTableState

Returns the enabled state of a user defined Strain vs Time table for the specified creep law definition.

```
long St7GetCreepUserTableState(long uID, long CreepID, long TableID,
     bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`TableID`

User defined Strain vs Time table ID.

Output Parameters

`Enabled`

True if the user defined table is enabled.

St7SetCreepUserTableData

Sets the data associated with the user defined Strain vs Time data for the specified creep law definition.

```
long St7SetCreepUserTableData(long uID, long CreepID, long TableID,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`TableID`

User defined table ID.

`Doubles[0..1]`

`[0]` – Stress level associated with Strain vs Time data.

`[1]` – Temperature associated with Strain vs Time data.

St7GetCreepUserTableData

Returns the data associated with the user defined Strain vs Time table assigned to the specified creep law definition.

```
long St7GetCreepUserTableData(long uID, long CreepID, long TableID,
     double* Doubles)
```

Input Parameters

uID

Strand7 model file ID.

CreepID

Creep layout ID.

TableID

User defined table ID.

Output Parameters

Doubles[0..1]

[0] – Stress level associated with Strain vs Time data.

[1] – Temperature associated with Strain vs Time data.

St7SetCreepHardeningType

Sets the hardening type for the specified creep law definition.

```
long St7SetCreepHardeningType(long uID, long CreepID, long* Integers)
```

Input Parameters

uID

Strand7 model file ID.

CreepID

Creep layout ID.

Integers[0..1]

[ipCreepHardeningType] – Type of creep hardening; either crHardeningTime or crHardeningStrain.

[ipCreepHardeningCyclic] – Cyclic hardening option; either btTrue or btFalse.

St7GetCreepHardeningType

Returns the hardening type assigned to the specified creep law definition.

```
long St7GetCreepHardeningType(long uID, long CreepID, long* Integers)
```

Input Parameters

uID

Strand7 model file ID.

`CreepID`

Creep layout ID.

Output Parameters

`Integers[0..1]`

`[ipCreepHardeningType]` – Type of creep hardening; either crHardeningTime or crHardeningStrain.

`[ipCreepHardeningCyclic]` – Cyclic hardening option; either btTrue or btFalse.

St7SetCreepTimeUnit

Sets the time units for the specified metallic creep law definition.

```
long St7SetCreepTimeUnit(long uID, long CreepID, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`Integers[0..0]`

`[0]` – Time units; one of tuMilliSec, tuSec, tuMin, tuHour or tuDay.

St7GetCreepTimeUnit

Returns the time units for the specified metallic creep law definition.

```
long St7GetCreepTimeUnit(long uID, long CreepID, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

Output Parameters

`Integers[0..0]`

`[0]` – Time units; one of tuMilliSec, tuSec, tuMin, tuHour or tuDay.

St7SetCreepTemperatureInclude

Sets temperature dependency for the specified creep law definition, where applicable.

```
long St7SetCreepTemperatureInclude(long uID, long CreepID, bool Include)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`Include`

True to include temperature dependent terms.

St7GetCreepTemperatureInclude

Returns the temperature dependency for the specified creep law definition, where applicable.

```
long St7GetCreepTemperatureInclude(long uID, long CreepID, bool* Include)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

Output Parameters

`Include`

Returns True when temperature dependent terms are included.

St7SetCreepConcreteHyperbolicData

Sets the hyperbolic data for the specified creep law definition.

```
long St7SetCreepConcreteHyperbolicData(long uID, long CreepID, long* Integers,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`Integers[0..1]`

`[ipCreepHyperbolicTimeTable]` – Factor vs Time table ID, or 0 for none.

`[ipCreepHyperbolicConstModulus]` – **Constant Modulus** flag; either btTrue or btFalse.

`Doubles[0..3]`

`[ipCreepHyberbolicAlpha]` – Hyperbolic law *alpha* parameter.

`[ipCreepHyperbolicBeta]` – Hyperbolic law *beta* parameter.

`[ipCreepHyperbolicDelta]` – Hyperbolic law *delta* parameter.

`[ipCreepHyperbolicPhi]` – Hyperbolic law *phi* parameter.

St7GetCreepConcreteHyperbolicData

Returns the hyperbolic data assigned to the specified creep law definition.

```
long St7GetCreepConcreteHyperbolicData(long uID, long CreepID, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

Output Parameters

`Integers[0..1]`

`[ipCreepHyperbolicTimeTable]` – Factor vs Time table ID, or 0 for none.

`[ipCreepHyperbolicConstModulus]` – **Constant Modulus** flag; either btTrue or btFalse.

`Doubles[0..3]`

`[ipCreepHyberbolicAlpha]` – Hyperbolic law *alpha* parameter.

`[ipCreepHyperbolicBeta]` – Hyperbolic law *beta* parameter.

`[ipCreepHyperbolicDelta]` – Hyperbolic law *delta* parameter.

`[ipCreepHyperbolicPhi]` – Hyperbolic law *phi* parameter.

St7SetCreepConcreteViscoChainData

Sets the visco-elastic chain data for the specified creep law definition.

```
long St7SetCreepConcreteViscoChainData(long uID, long CreepID, long Pos,
     long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`Pos`

Chain number.

`Integers[0..1]`

`[ipCreepViscoTimeTable]` – Strain vs Time table ID, or 0 for none.

`[ipCreepViscoTempTable]` – Factor vs Temperature table ID, or 0 for none.

`Doubles[0..1]`

`[ipCreepViscoDamper]` – Damping value.

`[ipCreepViscoStiffness]` – Stiffness value.

St7GetCreepConcreteViscoChainData

Returns the visco-elastic chain data assigned to the specified creep law definition.

```
long St7GetCreepConcreteViscoChainData(long uID, long CreepID, long Pos,
     long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`Pos`

Chain number.

Output Parameters

`Integers[0..1]`

`[ipCreepViscoTimeTable]` – Strain vs Time table ID, or 0 for none.

`[ipCreepViscoTempTable]` – Factor vs Temperature table ID, or 0 for none.

`Doubles[0..1]`

`[ipCreepViscoDamper]` – Damping value.

`[ipCreepViscoStiffness]` – Stiffness value.

St7EnableCreepConcreteUserTable

Enables the user defined concrete Strain vs Time table for the specified creep law definition.

```
long St7EnableCreepConcreteUserTable(long uID, long CreepID, long TableID)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`TableID`

User defined Strain vs Time table ID.

St7DisableCreepConcreteUserTable

Disables the user defined concrete Strain vs Time table for the specified creep law definition.

```
long St7DisableCreepConcreteUserTable(long uID, long CreepID, long TableID)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`TableID`

User defined Strain vs Time table ID.

St7GetCreepConcreteUserTableState

Returns the enabled state of the user defined concrete Strain vs Time table for the specified creep law definition.

```
long St7GetCreepConcreteUserTableState(long uID, long CreepID, long TableID,
    bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`TableID`

User defined Strain vs Time table ID.

Output Parameters

`Enabled`

True if the user defined concrete table is enabled.

St7SetCreepConcreteUserTableData

Sets the data for the user defined concrete Strain vs Time table for the specified creep law definition.

```
long St7SetCreepConcreteUserTableData(long uID, long CreepID, long TableID,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`TableID`

User defined Strain vs Time table ID.

`Doubles[0..1]`

`[0]` – Age at first loading value.

`[1]` – Stress value associated with Strain vs Time data.

St7GetCreepConcreteUserTableData

Returns the data assigned to the user defined Strain vs Time concrete table for the specified creep law definition.

```
long St7GetCreepConcreteUserTableData(long uID, long CreepID, long TableID,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`TableID`

User defined Strain vs Time table ID.

Output Parameters

`Doubles[0..1]`

`[0]` – Age at first loading value.

`[1]` – Stress value associated with Strain vs Time data.

St7SetCreepConcreteFunctionType

Sets the concrete type assigned to the specified creep law definition.

```
long St7SetCreepConcreteFunctionType(long uID, long CreepID, long FunctionType)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`FunctionType`

Either crCreepFunction or crRelaxationFunction.

St7GetCreepConcreteFunctionType

Returns the concrete type assigned to the specified creep law definition.

```
long St7GetCreepConcreteFunctionType(long uID, long CreepID, long* FunctionType)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

Output Parameters

`FunctionType`

Either crCreepFunction or crRelaxationFunction.

St7SetCreepConcreteLoadingAge

Sets the default loading age for the specified creep law definition.

```
long St7SetCreepConcreteLoadingAge(long uID, long CreepID, double LoadingAge)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`LoadingAge`

Default creep loading age.

St7GetCreepConcreteLoadingAge

Returns the default loading age assigned to the specified creep law definition.

```
long St7GetCreepConcreteLoadingAge(long uID, long CreepID, double* LoadingAge)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

Output Parameters

`LoadingAge`

Default creep loading age.

St7SetCreepConcreteLoadingTimeUnit

Sets the time units for the specified concrete creep law definition.

```
long St7SetCreepConcreteLoadingTimeUnit(long uID, long CreepID, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`Integers[0..0]`

`[0]` – Time units; one of tuMilliSec, tuSec, tuMin, tuHour or tuDay.

St7GetCreepConcreteLoadingTimeUnit

Returns the time units for the specified concrete creep law definition.

```
long St7GetCreepConcreteLoadingTimeUnit(long uID, long CreepID, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

Output Parameters

`Integers[0..0]`

`[0]` – Time units; one of tuMilliSec, tuSec, tuMin, tuHour or tuDay.

St7SetCreepConcreteShrinkageType

Sets the shrinkage type assigned to the specified creep law definition.

```
long St7SetCreepConcreteShrinkageType(long uID, long CreepID, long ShrinkageType)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`ShrinkageType`

Either crCreepShrinkageTable or crCreepShrinkageFormula.

St7GetCreepConcreteShrinkageType

Returns the shrinkage type assigned to the specified creep law definition.

```
long St7GetCreepConcreteShrinkageType(long uID, long CreepID,
     long* ShrinkageType)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

Output Parameters

`ShrinkageType`

Either crCreepShrinkageTable or crCreepShrinkageFormula.

St7SetCreepConcreteShrinkageFormulaData

Assigns the shrinkage formula data for the specified creep law definition.

```
long St7SetCreepConcreteShrinkageFormulaData(long uID, long CreepID,
     long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`Integers[0]`

Currently unused.

`Doubles[0..3]`

`[ipCreepShrinkageAlpha]` – Concrete shrinkage *alpha* parameter.

`[ipCreepShrinkageBeta]` – Concrete shrinkage *beta* parameter.

`[ipCreepShrinkageDelta]` – Concrete shrinkage *delta* parameter.

`[ipCreepShrinkageStrain]` – Concrete shrinkage initial strain parameter.

St7GetCreepConcreteShrinkageFormulaData

Returns the shrinkage formula data assigned to the specified creep law definition.

```
long St7GetCreepConcreteShrinkageFormulaData(long uID, long CreepID,
     long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

Output Parameters

`Integers[0]`

Currently unused.

`Doubles[0..3]`

`[ipCreepShrinkageAlpha]` – Concrete shrinkage *alpha* parameter.

`[ipCreepShrinkageBeta]` – Concrete shrinkage *beta* parameter.

`[ipCreepShrinkageDelta]` – Concrete shrinkage *delta* parameter.

`[ipCreepShrinkageStrain]` – Concrete shrinkage initial strain parameter.

St7SetCreepConcreteShrinkageTableData

Associates a table with the concrete shrinkage data for the specified creep law definition.

```
long St7SetCreepConcreteShrinkageTableData(long uID, long CreepID, long TableID)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`TableID`

Strain vs Time table ID, or 0 for none.

St7GetCreepConcreteShrinkageTableData

Returns the table associated with the concrete shrinkage data for the specified creep law definition.

```
long St7GetCreepConcreteShrinkageTableData(long uID, long CreepID, long* TableID)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

Output Parameters

`TableID`

Strain vs Time table ID, or 0 for none.

St7SetCreepConcreteTemperatureData

Sets the concrete temperature data for the specified creep law definition.

```
long St7SetCreepConcreteTemperatureData(long uID, long CreepID, long* Integers,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`Integers[0..2]`

`[ipIncludeCreepTemperature]` – Include temperature effects for the creep ageing; either btTrue or btFalse.

`[ipIncludeRateTemperature]` – Include temperature effects for the creep rate; either btTrue or btFalse.

`[ipIncludeShrinkageTemperature]` – Included temperature effects for the shrinkage ageing; either btTrue or btFalse.

`Doubles[0..5]`

`[ipCreepCAAge]` – Creep age *CA* parameter.

`[ipCreepTRefAge]` – Creep age TA_{REF} parameter.

`[ipCreepCCCreep]` – Creep rate *CC* parameter.

`[ipCreepTRefCreep]` – Creep rate TC_{REF} parameter.

`[ipCreepCAShrink]` – Shrinkage age *CA* parameter.

`[ipCreepTRefShrink]` – Shrinkage age TA_{REF} parameter.

St7GetCreepConcreteTemperatureData

Returns the concrete temperature data associated with the specified creep law definition.

```
long St7GetCreepConcreteTemperatureData(long uID, long CreepID, long* Integers,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

Output Parameters

`Integers[0..2]`

`[ipIncludeCreepTemperature]` – Include temperature effects for the creep ageing; either btTrue or btFalse.

`[ipIncludeRateTemperature]` – Include temperature effects for the creep rate; either btTrue or btFalse.

`[ipIncludeShrinkageTemperature]` – Include temperature effects for the shrinkage ageing; either btTrue or btFalse.

`Doubles[0..5]`

`[ipCreepCAAge]` – Creep age *CA* parameter.

`[ipCreepTRefAge]` – Creep age TA_{REF} parameter.

`[ipCreepCCCreep]` – Creep rate *CC* parameter.

`[ipCreepTRefCreep]` – Creep rate TC_{REF} parameter.

`[ipCreepCAShrink]` – Shrinkage age *CA* parameter.

`[ipCreepTRefShrink]` – Shrinkage age TA_{REF} parameter.

St7SetCreepConcreteCementCuringData

Sets the cement curing data for the specified creep law definition.

```
long St7SetCreepConcreteCementCuringData(long uID, long CreepID, long* Integers,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`Integers[0..2]`

`[ipCreepIncludeCuring]` – Include curing effects; either btTrue or btFalse.

`[ipCreepCuringTimeTable]` – Factor vs Time table ID, or 0 for none.

`[ipCreepCuringType]` – Curing rate; one of crCementCuringRapid, crCementCuringNormal or crCementCuringSlow.

`Doubles[0..2]`

`[ipCreepCuringCT]` – Curing *CT* parameter.

`[ipCreepCuringTRef]` – Curing T_{REF} parameter.

`[ipCreepCuringT0]` – Curing T_0 parameter.

St7GetCreepConcreteCementCuringData

Returns the cement curing data associated with the specified creep law definition.

```
long St7GetCreepConcreteCementCuringData(long uID, long CreepID, long* Integers,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

Output Parameters

`Integers[0..2]`

`[ipCreepIncludeCuring]` – Include curing effects; either btTrue or btFalse.

`[ipCreepCuringTimeTable]` – Factor vs Time table ID, or 0 for none.

`[ipCreepCuringType]` – Curing rate; one of crCementCuringRapid, crCementCuringNormal or crCementCuringSlow.

`Doubles[0..2]`

`[ipCreepCuringCT]` – Curing *CT* parameter.

`[ipCreepCuringTRef]` – Curing T_{REF} parameter.

`[ipCreepCuringT0]` – Curing T_0 parameter.

St7DeleteCreepDefinition

Deletes the specified creep definition.

```
long St7DeleteCreepDefinition(long uID, long CreepID)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

Load Path Templates

St7GetTotalLoadPathTemplates

Returns the total number and highest ID number of the load path templates in the specified model.

```
long St7GetTotalLoadPathTemplates(long uID, long* NumTemplates,
     long* LastTemplate)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumTemplates`

The number of load path templates.

`LastTemplate`

The highest load path template ID.

St7GetLoadPathTemplateNumByIndex

Returns the load path template number associated with the specified template index. The load path template indices are stored internally and are based on a contiguous numbering system.

```
long St7GetLoadPathTemplateNumByIndex(long uID, long Index, long* TemplateID)
```

Input Parameters

`uID`

Strand7 model file ID.

`Index`

Load path template index.

Output Parameters

`TemplateID`

Load path template ID.

St7NewLoadPathTemplate

Creates a new load path template.

```
long St7NewLoadPathTemplate(long uID, long TemplateID, char* TemplateName)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`TemplateName`

Name of the new template.

St7SetLoadPathTemplateName

Sets the name of the specified load path template.

```
long St7SetLoadPathTemplateName(long uID, long TemplateID, char* TemplateName)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`TemplateName`

Name of the template.

St7GetLoadPathTemplateName

Returns the name assigned to the specified load path template.

```
long St7GetLoadPathTemplateName(long uID, long TemplateID, char* TemplateName,
    long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`MaxStringLen`

Maximum number of characters allocated for TemplateName.

Output Parameters

`TemplateName`

Name of the template.

St7SetLoadPathTemplateParameters

Sets the data for the specified load path template.

```
long St7SetLoadPathTemplateParameters(long uID, long TemplateID, long* Integers,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Integers[0..4]`

`[ipLPTColour]` – Load path colour. See also *RGB Colours*.

`[ipLPTNumLanes]` – Number of lanes.

`[ipLPTMultiLaneType]` – Multi lane type; either lpAllSameFactors or lpAllDifferentFactors.

`[ipLPTTransitionLoad]` – Transition trailing load path in time history analysis; either btTrue or btFalse.

`[ipLPTBeamLateralTolerance]` – Apply relative tolerance in the lateral direction of the path for loads on beam elements; either btTrue or btFalse.

`Doubles[0..1]`

`[ipLPTTolerance]` – Relative tolerance.

`[ipLPTMinLaneWidth]` – Minimum lane width.

St7GetLoadPathTemplateParameters

Returns the data assigned to the specified load path template.

```
long St7GetLoadPathTemplateParameters(long uID, long TemplateID, long* Integers,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

Output Parameters

`Integers[0..4]`

`[ipLPTColour]` – Load path colour. See also *RGB Colours*.

`[ipLPTNumLanes]` – Number of lanes.

`[ipLPTMultiLaneType]` – Multi lane type; either lpAllSameFactors or lpAllDifferentFactors.

`[ipLPTTransitionLoad]` – Transition trailing load path in time history analysis; either btTrue or btFalse.

`[ipLPTBeamLateralTolerance]` – Apply relative tolerance in the lateral direction of the path for loads on beam elements; either btTrue or btFalse.

`Doubles[0..1]`

`[ipLPTTolerance]` – Relative tolerance.

`[ipLPTMinLaneWidth]` – Minimum lane width.

St7SetLoadPathTemplateLaneFactor

Assigns the lane factor for the specified load path template.

```
long St7SetLoadPathTemplateLaneFactor(long uID, long TemplateID, long Lane,
    double Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Lane`

Lane number.

`Factor`

Lane factor.

St7GetLoadPathTemplateLaneFactor

Returns the lane factor assigned to the specified load path template.

```
long St7GetLoadPathTemplateLaneFactor(long uID, long TemplateID, long Lane,
    double* Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Lane`

Lane number.

Output Parameters

`Factor`

Lane factor.

St7AddLoadPathTemplateVehicle

Adds a vehicle to the specified load path template.

```
long St7AddLoadPathTemplateVehicle(long uID, long TemplateID)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

St7SetLoadPathTemplateVehicleName

Sets the name of a vehicle in the specified load path template.

```
long St7SetLoadPathTemplateVehicleName(long uID, long TemplateID, long Vehicle,
     char* VehicleName)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

`VehicleName`

Vehicle name.

St7GetLoadPathTemplateVehicleName

Returns the name assigned to a vehicle in the specified load path template.

```
long St7GetLoadPathTemplateVehicleName(long uID, long TemplateID, long Vehicle,
     char* VehicleName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

`MaxStringLen`

Maximum number of characters allocated for VehicleName.

Output Parameters

`VehicleName`

Vehicle name.

St7InsertLoadPathTemplateVehicle

Inserts a new vehicle at the specified position in the specified load path template.

```
long St7InsertLoadPathTemplateVehicle(long uID, long TemplateID, long Vehicle)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

New vehicle number.

St7CloneLoadPathTemplateVehicle

Creates a copy of a vehicle in the specified load path template and appends it to the vehicle list.

```
long St7CloneLoadPathTemplateVehicle(long uID, long TemplateID, long Vehicle)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number to be cloned.

St7DeleteLoadPathTemplateVehicle

Deletes a vehicle within the specified load path template.

```
long St7DeleteLoadPathTemplateVehicle(long uID, long TemplateID, long Vehicle)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

St7GetNumLoadPathTemplateVehicles

Returns the number of vehicles assigned to the specified load path template.

```
long St7GetNumLoadPathTemplateVehicles(long uID, long TemplateID,
     long* NumVehicles)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

Output Parameters

`NumVehicles`

Number of vehicles.

St7SetLoadPathTemplateVehicleData

Sets the data for a vehicle in the specified load path template.

```
long St7SetLoadPathTemplateVehicleData(long uID, long TemplateID, long Vehicle,
     long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

`Integers[0..1]`

`[ipLPTVehicleInstance]` – Vehicle instance type; either lpVehicleSingleLane or lpVehicleDoubleLane.

`[ipLPTVehicleDirection]` – Vehicle direction flag; either lpVehicleForward or lpVehicleBackward.

`Doubles[0..1]`

`[ipLPTVehicleVelocity]` – Vehicle velocity.

`[ipLPTVehicleStartTime]` – Vehicle start time.

St7GetLoadPathTemplateVehicleData

Returns the data assigned to a vehicle in the specified load path template.

```
long St7GetLoadPathTemplateVehicleData(long uID, long TemplateID, long Vehicle,
     long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

Output Parameters

`Integers[0..1]`

`[ipLPTVehicleInstance]` – Vehicle instance type; either lpVehicleSingleLane or lpVehicleDoubleLane.

`[ipLPTVehicleDirection]` – Vehicle direction flag; either lpVehicleForward or lpVehicleBackward.

`Doubles[0..1]`

`[ipLPTVehicleVelocity]` – Vehicle velocity.

`[ipLPTVehicleStartTime]` – Vehicle start time.

St7EnableLoadPathTemplateVehicleLane

Enables a vehicle/lane combination within the specified load path template.

```
long St7EnableLoadPathTemplateVehicleLane(long uID, long TemplateID,
     long Vehicle, long Lane)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

`Lane`

Lane number.

St7DisableLoadPathTemplateVehicleLane

Disables a vehicle/lane combination within the specified load path template.

```
long St7DisableLoadPathTemplateVehicleLane(long uID, long TemplateID,
     long Vehicle, long Lane)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

`Lane`

Lane number.

St7GetLoadPathTemplateVehicleLaneState

Returns the enabled state of a vehicle/lane combination within the specified load path template.

```
long St7GetLoadPathTemplateVehicleLaneState(long uID, long TemplateID,
     long Vehicle, long Lane, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

`Lane`

Lane number.

Output Parameters

`Enabled`

True if the specified vehicle/lane combination is enabled.

St7AddLoadPathTemplatePointForce

Adds a point force to the specified load path template.

```
long St7AddLoadPathTemplatePointForce(long uID, long TemplateID, long Vehicle)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

St7InsertLoadPathTemplatePointForce

Inserts a point force within the specified load path template.

```
long St7InsertLoadPathTemplatePointForce(long uID, long TemplateID, long Vehicle,
    long Pos)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

`Pos`

Point force number.

St7DeleteLoadPathTemplatePointForce

Deletes a point force from the specified load path template.

```
long St7DeleteLoadPathTemplatePointForce(long uID, long TemplateID, long Vehicle,
     long Pos)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

`Pos`

Point force number.

St7GetNumLoadPathTemplatePointForces

Returns the number of point forces assigned to the specified load path template.

```
long St7GetNumLoadPathTemplatePointForces(long uID, long TemplateID,
     long Vehicle, long* NumPointForces)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

Output Parameters

`NumPointForces`

Number of point forces.

St7SetLoadPathTemplatePointForceData

Sets the point force data for the specified load path template.

```
long St7SetLoadPathTemplatePointForceData(long uID, long TemplateID,
     long Vehicle, long Pos, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

`Pos`

Point force number.

`Integers[0..3]`

`[ipLPTMobility]` – Mobility; either lpPointForceMobilityGrouped or lpPointForceMobilityFloating.

`[ipLPTAxisSystem]` – Axis system; either lpAxisGlobal or lpAxisLocal.

`[ipLPTAdjacency]` – Consider adjacency; either btTrue or btFalse.

`[ipLPTCentrifugal]` – Consider centrifugal effects; either btTrue or btFalse.

`Doubles[0..4]`

`[0..1]` – XY position of the point force.

`[2..4]` – Components of the point force according to the 123 axis convention in the specified axis system.

St7GetLoadPathTemplatePointForceData

Returns the point force data assigned to the specified load path template.

```
long St7GetLoadPathTemplatePointForceData(long uID, long TemplateID,
     long Vehicle, long Pos, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

`Pos`

Point force number.

Output Parameters

`Integers[0..3]`

`[ipLPTMobility]` – Mobility; either lpPointForceMobilityGrouped or lpPointForceMobilityFloating.

`[ipLPTAxisSystem]` – Axis system; either lpAxisGlobal or lpAxisLocal.

`[ipLPTAdjacency]` – Consider adjacency; either btTrue or btFalse.

`[ipLPTCentrifugal]` – Consider centrifugal effects; either btTrue or btFalse.

`Doubles[0..4]`

`[0..1]` – XY position of the point force.

`[2..4]` – Components of the point force according to the 123 axis convention in the specified axis system.

St7AddLoadPathTemplateDistributedForce

Adds a distributed force to the specified load path template.

```
long St7AddLoadPathTemplateDistributedForce(long uID, long TemplateID,
     long Vehicle)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

St7InsertLoadPathTemplateDistributedForce

Inserts a new distributed force to the specified load path template.

```
long St7InsertLoadPathTemplateDistributedForce(long uID, long TemplateID,
     long Vehicle, long Pos)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

`Pos`

Distributed force number.

St7DeleteLoadPathTemplateDistributedForce

Deletes a distributed force from the specified load path template.

```
long St7DeleteLoadPathTemplateDistributedForce(long uID, long TemplateID,
     long Vehicle, long Pos)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

`Pos`

Distributed force number.

St7GetNumLoadPathTemplateDistributedForces

Returns the number of distributed forces assigned to the specified load path template.

```
long St7GetNumLoadPathTemplateDistributedForces(long uID, long TemplateID,
     long Vehicle, long* NumDistributedForces)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

Output Parameters

`NumDistributedForces`

Number of distributed forces.

St7SetLoadPathTemplateDistributedForceData

Sets the distributed force data for the specified load path template.

```
long St7SetLoadPathTemplateDistributedForceData(long uID, long TemplateID,
     long Vehicle, long Pos, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

`Pos`

Distributed force number.

`Integers[0..3]`

`[ipLPTMobility]` – Mobility; one of lpDistrForceMobilityGrouped, lpDistrForceMobilityLeading, lpDistrForceMobilityTrailing, lpDistrForceMobilityFullLength or lpDistrForceMobilityFloating.

`[ipLPTAxisSystem]` – Axis system; either lpAxisGlobal or lpAxisLocal.

`[ipLPTAdjacency]` – Consider adjacency; either btTrue or btFalse.

`[ipLPTCentrifugal]` – Consider centrifugal effects; either btTrue or btFalse.

`Doubles[0..6]`

`[0..3]` – Position of endpoints according to the [x1, x2, y1, y2] format.

`[4..6]` – Components of distributed force according to the 123 axis convention in the specified coordinate system.

St7GetLoadPathTemplateDistributedForceData

Returns the distributed force data assigned to the specified load path template.

```
long St7GetLoadPathTemplateDistributedForceData(long uID, long TemplateID,
     long Vehicle, long Pos, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

`Pos`

Distributed force number.

Output Parameters

`Integers[0..3]`

`[ipLPTMobility]` – Mobility; one of lpDistrForceMobilityGrouped, lpDistrForceMobilityLeading, lpDistrForceMobilityTrailing, lpDistrForceMobilityFullLength or lpDistrForceMobilityFloating.

`[ipLPTAxisSystem]` – Axis system; either lpAxisGlobal or lpAxisLocal.

`[ipLPTAdjacency]` – Consider adjacency; either btTrue or btFalse.

`[ipLPTCentrifugal]` – Consider centrifugal effects; either btTrue or btFalse.

`Doubles[0..6]`

`[0..3]` – Position of endpoints according to the [x1, x2, y1, y2] format.

`[4..6]` – Components of distributed force according to the 123 axis convention in the specified coordinate system.

St7AddLoadPathTemplateHeatSource

Adds a new heat source to the specified load path template.

```
long St7AddLoadPathTemplateHeatSource(long uID, long TemplateID, long Vehicle)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

St7InsertLoadPathTemplateHeatSource

Inserts a new heat source in the specified load path template.

```
long St7InsertLoadPathTemplateHeatSource(long uID, long TemplateID, long Vehicle,
     long Pos)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

Vehicle

Vehicle number.

Pos

Heat source number.

St7DeleteLoadPathTemplateHeatSource

Deletes a heat source from the specified load path template.

```
long St7DeleteLoadPathTemplateHeatSource(long uID, long TemplateID, long Vehicle,
     long Pos)
```

Input Parameters

uID

Strand7 model file ID.

TemplateID

Load path template ID.

Vehicle

Vehicle number.

Pos

Heat source number.

St7GetNumLoadPathTemplateHeatSources

Returns the number of heat sources assigned to the specified load path template.

```
long St7GetNumLoadPathTemplateHeatSources(long uID, long TemplateID,
     long Vehicle, long* NumHeatSources)
```

Input Parameters

uID

Strand7 model file ID.

TemplateID

Load path template ID.

Vehicle

Vehicle number.

Output Parameters

NumHeatSources

Number of heat sources.

St7SetLoadPathTemplateHeatSourceData

Sets the heat source data for the specified load path template.

```
long St7SetLoadPathTemplateHeatSourceData(long uID, long TemplateID,
     long Vehicle, long Pos, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

`Pos`

Heat source number.

`Integers[0]`

Currently unused.

`Doubles[0..4]`

`[0..1]` – XY position of the heat source.

`[2..3]` – XY dimensions of the heat source.

`[4]` – Heat source.

St7GetLoadPathTemplateHeatSourceData

Returns the heat source data assigned to the specified load path template.

```
long St7GetLoadPathTemplateHeatSourceData(long uID, long TemplateID,
     long Vehicle, long Pos, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

`Pos`

Heat source number.

Output Parameters

`Integers[0]`

Currently unused.

`Doubles[0..4]`

`[0..1]` – XY position of the heat source.

`[2..3]` – XY dimensions of the heat source.

`[4]` – Heat source.

St7SetLoadPathTemplateVehicleSet

Assigns a vehicle set to the specified vehicle in a given load path template.

```
long St7SetLoadPathTemplateVehicleSet(long uID, long TemplateID, long Vehicle,
    char* VehicleSet)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

`VehicleSet`

String identifying the vehicle set. A null string indicates that the specified vehicle does not belong to a set.

St7GetLoadPathTemplateVehicleSet

Returns the vehicle set assigned to the specified vehicle in a given load path template.

```
long St7GetLoadPathTemplateVehicleSet(long uID, long TemplateID, long Vehicle,
    char* VehicleSet, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`Vehicle`

Vehicle number.

`MaxStringLen`

Maximum number of characters allocated for VehicleSet.

Output Parameters

`VehicleSet`

String identifying the vehicle set. A null string indicates that the specified vehicle does not belong to a set.

St7DeleteLoadPathTemplate

Deletes the specified load path template.

```
long St7DeleteLoadPathTemplate(long uID, long TemplateID)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

St7SetLoadPathTemplateCentrifugalData

Sets the centrifugal data for the specified load path template.

```
long St7SetLoadPathTemplateCentrifugalData(long uID, long TemplateID, char* K0,
    char* K1, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`K0`

Expression for the K0 term in the centrifugal force equation: Fc = K0 + K1*Fz. This formula can be a function of R, L, V and g.

`K1`

Expression for the K1 term in the centrifugal force equation: Fc = K0 + K1*Fz. This formula can be a function of R, L, V and g.

`Integers[0..2]`

`[ipLPTLimitK1]` – Impose K1 limits; either btTrue or btFalse.

`[ipLPTLengthUnit]` – Length unit; one of luMETRE, luCENTIMETRE, luMILLIMETRE, luFOOT or luINCH.

`[ipLPTForceUnit]` – Force unit; one of fuNEWTON, fuKILONEWTON, fuMEGANEWTON, fuKILOFORCE, fuPOUNDFORCE, fuTONNEFORCE or fuKIPFORCE.

`Doubles[0..1]`

`[ipLPTMinK1]` – Minimum K1 value.

`[ipLPTMaxK1]` – Maximum K1 value.

St7GetLoadPathTemplateCentrifugalData

Returns the centrifugal data assigned to the specified load path template.

```
long St7GetLoadPathTemplateCentrifugalData(long uID, long TemplateID, char* K0,
    char* K1, long MaxStringLen, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`MaxStringLen`

Maximum number of characters allocated for K0, K1.

Output Parameters

`K0`

Expression for the K0 term in the centrifugal force equation: Fc = K0 + K1*Fz. This formula can be a function of R, L, V and g.

`K1`

Expression for the K1 term in the centrifugal force equation: Fc = K0 + K1*Fz. This formula can be a function of R, L, V and g.

`Integers[0..2]`

`[ipLPTLimitK1]` – Impose K1 limits; either btTrue or btFalse.

`[ipLPTLengthUnit]` – Length unit; one of luMETRE, luCENTIMETRE, luMILLIMETRE, luFOOT or luINCH.

`[ipLPTForceUnit]` – Force unit; one of fuNEWTON, fuKILONEWTON, fuMEGANEWTON, fuKILOFORCE, fuPOUNDFORCE, fuTONNEFORCE or fuKIPFORCE.

`Doubles[0..1]`

`[ipLPTMinK1]` – Minimum K1 value.

`[ipLPTMaxK1]` – Maximum K1 value.

Cavity Fluid

St7GetTotalCavityFluidLayouts

Returns the total number and highest ID number of the cavity fluid layouts in the specified model.

```
long St7GetTotalCavityFluidLayouts(long uID, long* NumLayouts, long* LastLayout)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumLayouts`

The number of layouts.

`LastLayout`

The highest layout ID.

St7GetCavityFluidLayoutNumByIndex

Returns the cavity fluid layout number associated with the specified layout index. The cavity fluid layout indices are stored internally and are based on a contiguous numbering system.

```
long St7GetCavityFluidLayoutNumByIndex(long uID, long Index, long* CavityID)
```

Input Parameters

`uID`

Strand7 model file ID.

`Index`

Cavity fluid layout index.

Output Parameters

`CavityID`

Cavity fluid layout ID.

St7NewCavityFluidLayout

Creates a new cavity fluid layout.

```
long St7NewCavityFluidLayout(long uID, long CavityID, char* LayoutName)
```

Input Parameters

`uID`

Strand7 model file ID.

`CavityID`

Cavity fluid layout ID.

`LayoutName`

Name of the layout.

St7SetCavityFluidName

Sets the name of the specified cavity fluid layout.

```
long St7SetCavityFluidName(long uID, long CavityID, char* LayoutName)
```

Input Parameters

`uID`

Strand7 model file ID.

`CavityID`

Cavity fluid layout ID.

`LayoutName`

Name of the layout.

St7GetCavityFluidName

Returns the name assigned to the specified cavity fluid layout.

```
long St7GetCavityFluidName(long uID, long CavityID, char* LayoutName,
    long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`CavityID`

Cavity fluid layout ID.

`MaxStringLen`

Maximum number of characters allocated to LayoutName.

Output Parameters

`LayoutName`

Name of the layout.

St7GetCavityFluidType

Returns the type of fluid contained in the specified cavity fluid layout.

```
long St7GetCavityFluidType(long uID, long CavityID, long* FluidType)
```

Input Parameters

`uID`

Strand7 model file ID.

`CavityID`

Cavity fluid layout ID.

Output Parameters

`FluidType`

Type of fluid contained in the cavity – either ftIdealGas or ftConstantBulkModulus.

St7SetCavityFluidIdealGas

Sets the cavity fluid type to Ideal Gas and assigns the data for the specified cavity fluid layout.

```
long St7SetCavityFluidIdealGas(long uID, long CavityID, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CavityID`

Cavity fluid layout ID.

`Integers[0..4]`

`[ipCFColour]` – Colour. See also *RGB Colours*.

`[ipCFMultipleAsOne]` – Either btTrue or btFalse; if btTrue, multiple volumes that reference the cavity layout are assumed to be connected, otherwise they are assumed to be independent.

`[ipCFConsiderTemperature]` – Either btTrue or btFalse.

`[ipCFPressureControlCase]` – The load case that controls the initial cavity pressure in nonlinear analysis, or 0 for none.

`[ipCFAssembleStiffness]` – One of cfNoK, cfApproximateK or cfCompleteK.

`Doubles[0..1]`

`[ipCFInitialPressure]` – Initial pressure in cavity.

`[ipCFInitialTemperature]` – Initial temperature in cavity.

St7GetCavityFluidIdealGas

Returns the data assigned to the specified cavity fluid layout, if it is of type Ideal Gas.

```
long St7GetCavityFluidIdealGas(long uID, long CavityID, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CavityID`

Cavity fluid layout ID.

Output Parameters

`Integers[0..4]`

`[ipCFColour]` – Colour. See also *RGB Colours*.

`[ipCFMultipleAsOne]` – Either btTrue or btFalse; if btTrue, multiple volumes that reference the cavity layout are assumed to be connected, otherwise they are assumed to be independent.

`[ipCFConsiderTemperature]` – Either btTrue or btFalse.

`[ipCFPressureControlCase]` – The load case number that controls the initial cavity pressure in nonlinear analysis, or 0 for none.

`[ipCFAssembleStiffness]` – One of cfNoK, cfApproximateK or cfCompleteK.

`Doubles[0..1]`

`[ipCFInitialPressure]` – Initial pressure in cavity.

`[ipCFInitialTemperature]` – Initial temperature in cavity.

St7SetCavityFluidConstBulk

Sets the cavity fluid type to Constant Bulk Modulus and assigns the data for the specified cavity fluid layout.

```
long St7SetCavityFluidConstBulk(long uID, long CavityID, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CavityID`

Cavity fluid layout ID.

`Integers[0..4]`

`[ipCFColour]` – Colour. See also *RGB Colours*.

`[ipCFMultipleAsOne]` – Either btTrue or btFalse; if btTrue, multiple volumes that reference the cavity layout are assumed to be connected, otherwise they are assumed to be independent.

`[ipCFAssembleStiffness]` – One of cfNoK, cfApproximateK or cfCompleteK.

`Doubles[0..1]`

`[ipCFBulkModulus]` – Constant bulk modulus of cavity fluid.

`[ipCFAlpha]` – Thermal expansion coefficient of cavity fluid.

St7GetCavityFluidConstBulk

Returns the data assigned to the specified cavity fluid layout, if it is of type Constant Bulk Modulus.

```
long St7GetCavityFluidConstBulk(long uID, long CavityID, long* Integers,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CavityID`

Cavity fluid layout ID.

`Integers[0..4]`

`[ipCFColour]` – Colour. See also *RGB Colours*.

`[ipCFMultipleAsOne]` – Either btTrue or btFalse; if btTrue, multiple volumes that reference the cavity layout are assumed to be connected, otherwise they are assumed to be independent.

`[ipCFAssembleStiffness]` – One of cfNoK, cfApproximateK or cfCompleteK.

`Doubles[0..1]`

`[ipCFBulkModulus]` – Constant bulk modulus of cavity fluid.

`[ipCFAlpha]` – Thermal expansion coefficient of cavity fluid.

St7SetCavityFluidPreLoad

Sets the pre load data for the specified cavity fluid layout, if it is of type Constant Bulk Modulus.

```
long St7SetCavityFluidPreLoad(long uID, long CavityID, long CaseNum, long
     PreType, double Value)
```

Input Parameters

`uID`

Strand7 model file ID.

`CavityID`

Cavity fluid layout ID.

`CaseNum`

Load case number.

`PreType`

Either plCavityPreStress or plCavityPreStrain.

`Value`

The pre load value.

St7GetCavityFluidPreLoad

Returns the pre load data for the specified cavity fluid layout, if it is of type Constant Bulk Modulus.

```
long St7GetCavityFluidPreLoad(long uID, long CavityID, long CaseNum, long*
     PreType, double* Value)
```

Input Parameters

`uID`

Strand7 model file ID.

`CavityID`

Cavity fluid layout ID.

`CaseNum`

Load case number.

Output Parameters

`PreType`

Either plCavityPreStress or plCavityPreStrain.

`Value`

The pre load value.

St7DeleteCavityFluidLayout

Deletes the specified cavity fluid layout.

```
long St7DeleteCavityFluidLayout(long uID, long CavityID)
```

Input Parameters

`uID`

Strand7 model file ID.

`CavityID`

Cavity fluid layout ID.

Material Property Libraries

St7SetLibraryPath

Sets the full path name to the directory containing the Strand7 library files. Any subsequent calls to the libraries will use the files contained in this directory.

```
long St7SetLibraryPath(char* LibraryPath)
```

Input Parameters

`LibraryPath`

Full path name to the directory containing the Strand7 library files.

St7GetLibraryPath

Returns the full path name to the directory containing the Strand7 library files.

```
long St7GetLibraryPath(char* LibraryPath, long MaxStringLen)
```

Input Parameters

`MaxStringLen`

The maximum number of characters allocated for LibraryPath.

Output Parameters

`LibraryPath`

Full path name to the directory containing the Strand7 library files.

St7GetNumLibraries

Returns the number of material libraries currently available.

```
long St7GetNumLibraries(long LibraryType, long* NumLibraries)
```

Input Parameters

`LibraryType`

One of lbMaterial, lbBeamSection, lbSectionGeometry, lbComposite, lbReinforcementLayout, lbCreepDefinition or lbLoadPathTemplate.

Output Parameters

`NumLibraries`

Number of libraries.

St7GetLibraryName

Returns the name assigned to the specified library.

```
long St7GetLibraryName(long LibraryType, long LibraryID, char* LibraryName,
     long MaxStringLen)
```

Input Parameters

`LibraryType`

One of lbMaterial, lbBeamSection, lbSectionGeometry, lbComposite, lbReinforcementLayout, lbCreepDefinition or lbLoadPathTemplate.

`LibraryID`

Library ID.

`MaxStringLen`

Maximum number of characters allocated for LibraryName.

Output Parameters

`LibraryName`

Library name.

St7GetLibraryID

Returns the ID number assigned to the specified library.

```
long St7GetLibraryID(long LibraryType, char* LibraryName, long* LibraryID)
```

Input Parameters

`LibraryType`

One of lbMaterial, lbBeamSection, lbSectionGeometry, lbComposite, lbReinforcementLayout, lbCreepDefinition or lbLoadPathTemplate.

`LibraryName`

Library name without the extension. For example, “RHS (350 Grade)” rather than “RHS (350 Grade).bsl”.

Output Parameters

`LibraryID`

Library ID.

St7GetNumLibraryItems

Returns the number of items assigned to the specified library.

```
long St7GetNumLibraryItems(long LibraryType, long LibraryID, long* NumItems)
```

Input Parameters

`LibraryType`

One of lbMaterial, lbBeamSection, lbSectionGeometry, lbComposite, lbReinforcementLayout, lbCreepDefinition or lbLoadPathTemplate.

`LibraryID`

Library ID.

Output Parameters

`NumItems`

Number of library items.

St7GetLibraryItemName

Returns the name assigned to the specified library item.

```
long St7GetLibraryItemName(long LibraryType, long LibraryID, long ItemID,
     char* ItemName, long MaxStringLen)
```

Input Parameters

`LibraryType`

One of lbMaterial, lbBeamSection, lbSectionGeometry, lbComposite, lbReinforcementLayout, lbCreepDefinition or lbLoadPathTemplate.

`LibraryID`

Library ID.

`ItemID`

Item ID.

`MaxStringLen`

Maximum number of characters allocated for ItemName.

Output Parameters

`ItemName`

Name of the item.

St7GetLibraryItemID

Returns the ID number assigned to the specified library item.

```
long St7GetLibraryItemID(long LibraryType, long LibraryID, char* ItemName,
     long* ItemID)
```

Input Parameters

`LibraryType`

One of lbMaterial, lbBeamSection, lbSectionGeometry, lbComposite, lbReinforcementLayout, lbCreepDefinition or lbLoadPathTemplate.

`LibraryID`

Library ID.

`ItemName`

Name of the item.

Output Parameters

`ItemID`

Item ID.

St7GetLibraryBeamSectionPropertyDataBSL

Returns the values that define the geometry and properties of a cross section from the beam section library (BSL).

```
long St7GetLibraryBeamSectionPropertyDataBSL(long LibraryID, long ItemID,
     long LengthUnit, char* ItemName, long MaxStringLen, long* ItemShape,
     double* SectionData)
```

Input Parameters

`LibraryID`

Library ID.

`ItemID`

Item ID.

`LengthUnit`

One of luMETRE, luCENTIMETRE, luMILLIMETRE, luFOOT or luINCH.

`MaxStringLen`

The maximum number of characters allocated for ItemName.

Output Parameters

`ItemName`

Name of the item.

`ItemShape`

One of the section shapes defined in *Beam Cross Section Shape*.

`SectionData[0..kNumBeamSectionData-1]`

`[ipAREA]` – Section area.

`[ipI11]` – Second moment of area about the principal 1 axis.

`[ipI22]` – Second moment of area about the principal 2 axis.

`[ipJ]` – Torsion constant.

`[ipSL1]` – Shear centre offset in the principal 1 axis direction.

`[ipSL2]` – Shear centre offset in the principal 2 axis direction.

`[ipSA1]` – Shear area in the principal 1 axis direction.

`[ipSA2]` – Shear area in the principal 2 axis direction.

`[ipXBAR]` – Centroid x coordinate.

`[ipYBAR]` – Centroid y coordinate.

`[ipANGLE]` – Principal axis 1 angle in radians w.r.t. the local section coordinates.

`[ipD1]` – Section geometry D1 parameter; see *Beam Cross Section Shape*.

`[ipD2]` – Section geometry D2 parameter.

`[ipD3]` – Section geometry D3 parameter.

`[ipT1]` – Section geometry T1 parameter.

`[ipT2]` – Section geometry T2 parameter.

`[ipT3]` – Section geometry T3 parameter.

`[ipGapA]` – Mirrored section gap parameter A.

`[ipGapB]` – Mirrored section gap parameter B.

St7GetLibraryBeamSectionPropertyDataBGL

Returns the properties of a cross section from the beam geometry library (BGL).

```
long St7GetLibraryBeamSectionPropertyDataBGL(long LibraryID, long ItemID,
     long LengthUnit, char* ItemName, long MaxStringLen, double* Doubles)
```

Input Parameters

`LibraryID`

Library ID.

`ItemID`

Item ID.

`LengthUnit`

One of luMETRE, luCENTIMETRE, luMILLIMETRE, luFOOT or luINCH.

`MaxStringLen`

The maximum number of characters allocated for ItemName.

Output Parameters

`ItemName`

Name of the item.

`Doubles[0..38]`

`[ipBXSXBar]` – Centroid x coordinate.

`[ipBXSYBar]` – Centroid y coordinate.

`[ipBXSArea]` – Section area.

`[ipBXSI11]` – Second moment of area about the principal 1 axis.

[ipBXSI22] – Second moment of area about the principal 2 axis.

[ipBXSAngle] – Angle between the local x and principal 1 axes.

[ipBXSZ11Plus] – Section modulus about the principal 1 axis for stress on the outer fibre in the positive 2 axis direction.

[ipBXSZ11Minus] – Section modulus about the principal 1 axis for stress on the outer fibre in the negative 2 axis direction.

[ipBXSZ22Plus] – Section modulus about the principal 2 axis for stress on the outer fibre in the positive 1 axis direction.

[ipBXSZ22Minus] – Section modulus about the principal 2 axis for stress on the outer fibre in the negative 1 axis direction.

[ipBXSS11] – Plastic modulus about the principal 1 axis.

[ipBXSS22] – Plastic modulus about the principal 2 axis.

[ipBXSPC1] – Principal 1 axis direction offset for centroid of principal plastic modulus.

[ipBXSPC2] – Principal 2 axis direction offset for centroid of principal plastic modulus.

[ipBXSr1] – Radius of gyration in the principal 1 axis direction.

[ipBXSr2] – Radius of gyration in the principal 2 axis direction.

[ipBXSSA1] – Shear area in the principal 1 axis direction.

[ipBXSSA2] – Shear area in the principal 2 axis direction.

[ipBXSSL1] – Shear centre offset in the principal 1 axis direction.

[ipBXSSL2] – Shear centre offset in the principal 2 axis direction.

[ipBXSIXX] – Second moment of area about the global X axis.

[ipBXSIYY] – Second moment of area about the global Y axis.

[ipBXSIXY] – Second moment of area coupling term in the global XY axes.

[ipBXSIxxL] – Second moment of area about the local x axis.

[ipBXSIyyL] – Second moment of area about the local y axis.

[ipBXSIxyL] – Second moment of area coupling term in the local xy axes.

[ipBXSZxxPlus] – Section modulus about the local x axis for stress on the outer fibre in the positive y axis direction.

[ipBXSZxxMinus] – Section modulus about the local x axis for stress on the outer fibre in the negative y axis direction.

[ipBXSZyyPlus] – Section modulus about the local y axis for stress on the outer fibre in the positive x axis direction.

[ipBXSZyyMinus] – Section modulus about the local y axis for stress on the outer fibre in the negative x axis direction.

`[ipBXSSxx]` – Plastic modulus about the local x axis.

`[ipBXSSyy]` – Plastic modulus about the local y axis.

`[ipBXSPCx]` – Local x axis direction offset for centroid of local plastic modulus.

`[ipBXSPCy]` – Local y axis direction offset for centroid of local plastic modulus.

`[ipBXSrx]` – Radius of gyration in the local x axis direction.

`[ipBXSry]` – Radius of gyration in the local y axis direction.

`[ipBXSrdA]` – Radius area integral.

`[ipBXSJ]` – Torsion constant.

`[ipBXSIw]` – Warping constant.

St7GetLibraryBeamSectionGeometryBGL

Returns the values that define the geometry of a cross section from the beam geometry library (BGL).

```
long St7GetLibraryBeamSectionGeometryBGL(long LibraryID, long ItemID,
     long LengthUnit, char* ItemName, long MaxStringLen, long* ItemShape,
     double* ItemDimensions)
```

Input Parameters

`LibraryID`

Library ID.

`ItemID`

Item ID.

`LengthUnit`

One of luMETRE, luCENTIMETRE, luMILLIMETRE, luFOOT or luINCH.

`MaxStringLen`

The maximum number of characters allocated for ItemName.

Output Parameters

`ItemName`

Name of the item.

`ItemShape`

One of bgNullSection, bgRectangularHollow, bgISection, bgChannel, bgTSection, bgAngle or bgBulbFlat.

`ItemDimensions[0..kMaxBGLDimensions-1]`

Dimensions that define the beam section geometry; see *Beam Geometry Library (BGL) Cross Section Shapes*.

St7AssignLibraryMaterial

Assigns the specified item from the material library (.MAT) to an entity property.

```
long St7AssignLibraryMaterial(long uID, long Entity, long PropNum,
     long LibraryID, long ItemID)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP or ptBRICKPROP.

`PropNum`

Element property number.

`LibraryID`

Library ID.

`ItemID`

Item ID.

St7AssignLibraryComposite

Assigns the specified item from the composite material library (.CML) to a ply property.

```
long St7AssignLibraryComposite(long uID, long PropNum, long LibraryID,
     long ItemID)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Ply property number.

`LibraryID`

Library ID.

`ItemID`

Item ID.

St7AssignLibraryBeamSection

Assigns the specified item from the beam section library (.BSL) to a beam property.

```
long St7AssignLibraryBeamSection(long uID, long PropNum, long LibraryID,
     long ItemID, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`LibraryID`

Library ID.

`ItemID`

Item ID.

`Integers[0..3]`

`[0]` – btTrue to import beam material data.

`[1]` – btTrue to calculate null values.

`[2]` – btTrue to import material damping values if importing material data.

`[3]` – btTrue to replace the property name with the library name.

St7AssignLibraryBeamSectionBGL

Assigns the specified item from the beam geometry library (.BGL) to a beam property.

```
long St7AssignLibraryBeamSectionBGL(long uID, long PropNum, long LibraryID,
     long ItemID, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Beam property number.

`LibraryID`

Library ID.

`ItemID`

Item ID.

`Integers[0..3]`

`[0]` – btTrue to import beam material data.

`[1]` – not used.

`[2]` – btTrue to import material damping values if importing material data.

[3] – btTrue to replace the property name with the library name.

St7AssignLibraryCreepDefinition

Assigns the specified item from the creep definition library (.CRL) to a creep layout.

```
long St7AssignLibraryCreepDefinition(long uID, long CreepID, long LibraryID,
     long ItemID)
```

Input Parameters

`uID`

Strand7 model file ID.

`CreepID`

Creep layout ID.

`LibraryID`

Library ID.

`ItemID`

Item ID.

St7AssignLibraryLoadPathTemplate

Assigns the specified item from the load path definition library (.MLT) to a load path template.

```
long St7AssignLibraryLoadPathTemplate(long uID, long TemplateID, long LibraryID,
     long ItemID)
```

Input Parameters

`uID`

Strand7 model file ID.

`TemplateID`

Load path template ID.

`LibraryID`

Library ID.

`ItemID`

Item ID.

Usage

This call is typically followed by a call to *St7SetLoadPathTemplateParameters* to set the number of lanes in the template, amongst other settings.

St7AssignLibraryReinforcementLayout

Assigns the specified item from the reinforcement definition library (.REO) to a reinforcement layout.

```
long St7AssignLibraryReinforcementLayout(long uID, long LayoutID, long LibraryID,
     long ItemID)
```

Input Parameters

`uID`

Strand7 model file ID.

`LayoutID`

Reinforcement layout ID.

`LibraryID`

Library ID.

`ItemID`

Item ID.

Tables

St7GetNumTables

Returns the number of tables of the specified type.

```
long St7GetNumTables(long uID, long TableType, long* NumTables,
    long* MaxTableNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`TableType`

One of the table types listed in *Table Types*.

Output Parameters

`NumTables`

Number of tables.

`MaxTableNum`

Maximum table ID.

St7GetTableInfoByIndex

Returns the name and ID number of the specified table. The maximum table index is returned by the *St7GetNumTables* function.

```
long St7GetTableInfoByIndex(long uID, long TableType, long Index, long* TableID,
    char* TableName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`TableType`

One of the table types listed in *Table Types*.

`Index`

Table index.

`MaxStringLen`

The maximum number of characters allocated for TableName.

Output Parameters

`TableID`

Table ID.

`TableName`

Name of the table.

St7NewTableType

Creates a new table in the specified model.

```
long St7NewTableType(long uID, long TableType, long TableID, long NumEntries,
     char* TableName, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`TableType`

One of the table types listed in *Table Types*.

`TableID`

Table ID.

`NumEntries`

Number of rows (or XY data pairs) in the table.

`TableName`

Name of the table.

`Doubles[0..2*NumEntries-1]`

An array containing the XY data for the table. Each XY pair is stored in a block of length 2, with the start of the i^{th} pair at `Doubles[(i-1)*2]`.

St7DeleteTableType

Deletes the specified table.

```
long St7DeleteTableType(long uID, long TableType, long TableID)
```

Input Parameters

`uID`

Strand7 model file ID.

`TableType`

One of the table types listed in *Table Types*.

`TableID`

Table ID.

St7SetTableTypeName

Sets the name of the specified table.

```
long St7SetTableTypeName(long uID, long TableType, long TableID, char* TableName)
```

Input Parameters

`uID`

Strand7 model file ID.

`TableType`

One of the table types listed in *Table Types*.

`TableID`

Table ID.

`TableName`

Name of the table.

St7GetTableTypeName

Returns the name of the specified table.

```
long St7GetTableTypeName(long uID, long TableType, long TableID, char* TableName,
     long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`TableType`

One of the table types listed in *Table Types*.

`TableID`

Table ID.

`MaxStringLen`

Maximum number of characters allocated for TableName.

Output Parameters

`TableName`

Name of the table.

St7GetTableID

Returns the ID number for a table specified by name. Where multiple names exist, the table ID with the lowest table index is returned.

```
long St7GetTableID(long uID, char* TableName, long TableType, long* TableID)
```

Input Parameters

`uID`

Strand7 model file ID.

`TableName`

Name of the table.

`TableType`

One of the table types listed in *Table Types*.

Output Parameters

`TableID`

Table ID.

St7GetNumTableTypeRows

Returns the number of rows in the specified table.

```
long St7GetNumTableTypeRows(long uID, long TableType, long TableID,
     long* NumRows)
```

Input Parameters

`uID`

Strand7 model file ID.

`TableType`

One of the table types listed in *Table Types*.

`TableID`

Table ID.

Output Parameters

`NumRows`

Number of rows.

St7SetTableTypeData

Sets the XY data for the specified table.

```
long St7SetTableTypeData(long uID, long TableType, long TableID, long NumEntries,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`TableType`

One of the table types listed in *Table Types*.

`TableID`

Table ID.

`NumEntries`

Number of entries in table.

`Doubles[0..2*NumEntries-1]`

An array containing the XY data for the table. Each XY pair is stored in a block of length 2, with the start of the i^{th} pair at `Doubles[(i-1)*2]`.

St7GetTableTypeData

Returns the XY data for the specified table.

```
long St7GetTableTypeData(long uID, long TableType, long TableID, long MaxRows,
     long* NumRows, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`TableType`

One of the table types listed in *Table Types*.

`TableID`

Table ID.

`MaxRows`

Maximum number of rows allocated for Doubles.

Output Parameters

`NumRows`

Number of rows used.

`Doubles[0..2*MaxRows-1]`

An array containing the XY data for the table. Each XY pair is stored in a block of length 2, with the start of the i^{th} pair at `Doubles[(i-1)*2]`.

St7SetFrequencyPeriodTableType

Sets the type of the specified Factor vs Frequency/Period table.

```
long St7SetFrequencyPeriodTableType(long uID, long TableID, long FreqType)
```

Input Parameters

`uID`

Strand7 model file ID.

`TableID`

Table ID.

`FreqType`

Type of frequency table; either ftPeriod or ftFrequency.

St7GetFrequencyPeriodTableType

Returns the type of the specified Factor vs Frequency/Period table.

```
long St7GetFrequencyPeriodTableType(long uID, long TableID, long* FreqType)
```

Input Parameters

`uID`

Strand7 model file ID.

`TableID`

Table ID.

Output Parameters

`FreqType`

Type of frequency table; either ftPeriod or ftFrequency.

St7SetFrequencyPeriodTableUnits

Sets the units for the specified Factor vs Frequency/Period table.

```
long St7SetFrequencyPeriodTableUnits(long uID, long TableID, long UnitType)
```

Input Parameters

`uID`

Strand7 model file ID.

`TableID`

Table ID.

`UnitType`

One of fuNone, fuDispResponse, fuVelResponse, fuAccelResponse, fuDispPSD, fuVelPSD, fuAccelPSD, fuAccelResponseG or fuAccelPSDG.

St7GetFrequencyPeriodTableUnits

Returns the units for the specified Factor vs Frequency/Period table.

```
long St7GetFrequencyPeriodTableUnits(long uID, long TableID, long* UnitType)
```

Input Parameters

`uID`

Strand7 model file ID.

`TableID`

Table ID.

Output Parameters

`UnitType`

One of fuNone, fuDispResponse, fuVelResponse, fuAccelResponse, fuDispPSD, fuVelPSD, fuAccelPSD, fuAccelResponseG or fuAccelPSDG.

St7SetTimeTableUnits

Sets the time units for the specified time based table.

```
long St7SetTimeTableUnits(long uID, long TableType, long TableID, long UnitType)
```

Input Parameters

`uID`

Strand7 model file ID.

`TableType`

One of ttVsTime, ttStrainTime or ttTemperatureVsTime.

`TableID`

Table ID.

`UnitType`

One of tuMilliSec, tuSec, tuMin, tuHour or tuDay.

St7GetTimeTableUnits

Returns the time units assigned to the specified time based table.

```
long St7GetTimeTableUnits(long uID, long TableType, long TableID, long* UnitType)
```

Input Parameters

`uID`

Strand7 model file ID.

`TableType`

One of ttVsTime, ttStrainTime or ttTemperatureVsTime.

`TableID`

Table ID.

Output Parameters

`UnitType`

One of tuMilliSec, tuSec, tuMin, tuHour or tuDay.

St7ConvertTimeTableUnits

Converts the time units for the specified time based table.

```
long St7ConvertTimeTableUnits(long uID, long TableType, long TableID,
     long UnitType)
```

Input Parameters

`uID`

Strand7 model file ID.

`TableType`

Either ttVsTime or ttStrainTime.

`TableID`

Table ID.

`UnitType`

One of tuMilliSec, tuSec, tuMin, tuHour or tuDay.

St7SetMomentRotationTableUnits

Sets the rotation units for the specified Moment vs Rotation table.

```
long St7SetMomentRotationTableUnits(long uID, long TableID, long UnitType)
```

Input Parameters

`uID`

Strand7 model file ID.

`TableID`

Table ID.

`UnitType`

Either auRadian or auDegree.

St7GetMomentRotationTableUnits

Returns the rotation units assigned to the specified Moment vs Rotation table.

```
long St7GetMomentRotationTableUnits(long uID, long TableID, long* UnitType)
```

Input Parameters

`uID`

Strand7 model file ID.

`TableID`

Table ID.

Output Parameters

`UnitType`

Either auRadian or auDegree.

St7SetAccVsTimeTableUnits

Sets the units assigned to the specified acceleration vs time table.

```
long St7SetAccVsTimeTableUnits(long uID, long TableID, long UnitType)
```

Input Parameters

`uID`

Strand7 model file ID.

`TableID`

Table ID.

`UnitType`

Acceleration vs time units type; either atModelUnits or atGravityUnits.

St7GetAccVsTimeTableUnits

Returns the units assigned to the specified acceleration vs time table.

```
long St7GetAccVsTimeTableUnits(long uID, long TableID, long* UnitType)
```

Input Parameters

`uID`

Strand7 model file ID.

`TableID`

Table ID.

Output Parameters

`UnitType`

Acceleration vs time units type; either atModelUnits or atGravityUnits.

Solver – Linear Static

St7EnableLSALoadCase

Enables the specified load case and freedom case combination such that it is included in linear static analysis.

```
long St7EnableLSALoadCase(long uID, long LoadCaseNum, long FreedomCaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoadCaseNum`

Load case number.

`FreedomCaseNum`

Freedom case number.

St7DisableLSALoadCase

Disables the specified load case and freedom case combination such that it is not included in linear static analysis.

```
long St7DisableLSALoadCase(long uID, long LoadCaseNum, long FreedomCaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoadCaseNum`

Load case number.

`FreedomCaseNum`

Freedom case number.

St7GetLSALoadCaseState

Returns the enabled state of the specified load case and freedom case combination for linear static analysis.

```
long St7GetLSALoadCaseState(long uID, long LoadCaseNum, long FreedomCaseNum,
     bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoadCaseNum`

Load case number.

`FreedomCaseNum`

Freedom case number.

Output Parameters

`Enabled`

True if the specified load/seismic case and freedom case combination is enabled for linear static analysis.

Solver – Linear Buckling

St7SetLBAInitial

Assigns the initial conditions file to be used for linear buckling analysis.

```
long St7SetLBAInitial(long uID, char* FileName, long VariableCaseNum,
     long FixedCaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the initial conditions file for the linear buckling solver.

`VariableCaseNum`

Result case number in FileName that provides the **Variable Case**.

`FixedCaseNum`

Result case number in FileName that provides the **Constant Case**.

St7GetLBAInitial

Returns the initial conditions file assigned for linear buckling analysis.

```
long St7GetLBAInitial(long uID, char* FileName, long* VariableCaseNum,
     long* FixedCaseNum, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxStringLen`

Maximum number of characters allocated for FileName.

Output Parameters

`FileName`

Full path and name for the initial conditions file used by the linear buckling solver.

`VariableCaseNum`

Result case number in FileName that provides the **Variable Case**.

`FixedCaseNum`

Result case number in FileName that provides the **Constant Case**.

St7SetLBANumModes

Sets the number of modes to be calculated in linear buckling analysis.

```
long St7SetLBANumModes(long uID, long NumModes)
```

Input Parameters

`uID`

Strand7 model file ID.

`NumModes`

Number of modes to be found.

St7GetLBANumModes

Returns the number of modes to be calculated in linear buckling analysis.

```
long St7GetLBANumModes(long uID, long* NumModes)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumModes`

Number of modes to be found.

St7SetLBAShift

Sets the load factor shift to be used in linear buckling analysis. The closest modes found, both above and below the shift value, will be calculated.

```
long St7SetLBAShift(long uID, double Shift)
```

Input Parameters

`uID`

Strand7 model file ID.

`Shift`

Load factor shift.

St7GetLBAShift

Returns the load factor shift to be used in linear buckling analysis. The closest modes found, both above and below the shift value, will be calculated.

```
long St7GetLBAShift(long uID, double* Shift)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Shift`

Load factor shift.

Solver – Load Influence

St7EnableLIALoadCase

Enables the specified load and freedom case combination such that it is included in load influence analysis.

```
long St7EnableLIALoadCase(long uID, long LoadCaseNum, long FreedomCaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoadCaseNum`

Load case number.

`FreedomCaseNum`

Freedom case number.

St7DisableLIALoadCase

Disables the specified load and freedom case combination such that it is not included in load influence analysis.

```
long St7DisableLIALoadCase(long uID, long LoadCaseNum, long FreedomCaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoadCaseNum`

Load case number.

`FreedomCaseNum`

Freedom case number.

St7GetLIALoadCaseState

Returns the enabled state assigned to the specified load and freedom case combination for load influence analysis.

```
long St7GetLIALoadCaseState(long uID, long LoadCaseNum, long FreedomCaseNum,
    bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoadCaseNum`

Load case number.

`FreedomCaseNum`

Freedom case number.

Output Parameters

`Enabled`

True if the specified load and freedom case combination is enabled for load influence analysis.

Solver – Nonlinear Static

St7SetNLAStagedAnalysis

Sets the state of the staged analysis option for nonlinear static analysis.

```
long St7SetNLAStagedAnalysis(long uID, bool StagedAnalysis)
```

Input Parameters

`uID`

Strand7 model file ID.

`StagedAnalysis`

True to perform staged analysis.

St7GetNLAStagedAnalysis

Returns the state of the staged analysis option for nonlinear static analysis.

```
long St7GetNLAStagedAnalysis(long uID, bool* StagedAnalysis)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`StagedAnalysis`

True if staged analysis is active.

St7EnableNLAStage

Activates the specified stage such that it is included in nonlinear static analysis.

```
long St7EnableNLAStage(long uID, long Stage)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index to be enabled.

St7DisableNLAStage

Deactivates the specified stage such that it is not included in nonlinear static analysis.

```
long St7DisableNLAStage(long uID, long Stage)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index to be disabled.

St7GetNLAStageState

Returns the enabled state assigned to the specified stage for nonlinear static analysis.

```
long St7GetNLAStageState(long uID, long Stage, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index.

Output Parameters

`Enabled`

True if the specified stage is enabled.

St7AddNLAIncrement

Adds a new blank increment to the nonlinear static analysis load table.

```
long St7AddNLAIncrement(long uID, long Stage, char* IncName)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index – use 0 for unstaged analysis.

`IncName`

String containing the increment name.

St7InsertNLAIncrement

Inserts a new blank increment at the specified position in the nonlinear static analysis load table.

```
long St7InsertNLAIncrement(long uID, long Stage, long Increment, char* IncName)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index – use 0 for unstaged analysis.

`Increment`

Increment number.

`IncName`

String containing the increment name.

St7DeleteNLAIncrement

Deletes the specified increment from the nonlinear static analysis load table.

```
long St7DeleteNLAIncrement(long uID, long Stage, long Increment)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index – use 0 for unstaged analysis.

`Increment`

Increment number.

St7GetNLAIncrementName

Returns the name of the specified increment in the nonlinear static analysis load table.

```
long St7GetNLAIncrementName(long uID, long Stage, long Increment, char* IncName,
     long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index – use 0 for unstaged analysis.

`Increment`

Increment number.

`MaxStringLen`

Maximum number of characters allocated for IncName.

Output Parameters

`IncName`

String containing the increment name.

St7GetNumNLAIncrements

Returns the total number of increments assigned in the nonlinear static analysis load table.

```
long St7GetNumNLAIncrements(long uID, long Stage, long* NumIncrements)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index – use 0 for unstaged analysis.

Output Parameters

`NumIncrements`

Total number of increments.

St7SetNLALoadIncrementFactor

Assigns the load case factors for the specified increment in the nonlinear static analysis load table.

```
long St7SetNLALoadIncrementFactor(long uID, long Stage, long Increment,
    long CaseNum, double Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index – use 0 for unstaged analysis.

`Increment`

Increment number.

`CaseNum`

Load case number.

`Factor`

Load case factor.

St7GetNLALoadIncrementFactor

Returns the load case factors assigned for the specified increment in the nonlinear static analysis load table.

```
long St7GetNLALoadIncrementFactor(long uID, long Stage, long Increment,
     long CaseNum, double* Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index – use 0 for unstaged analysis.

`Increment`

Increment number.

`CaseNum`

Load case number.

Output Parameters

`Factor`

Load case factor.

St7SetNLAFreedomIncrementFactor

Assigns the freedom case factors for the specified increment in the nonlinear static analysis load table.

```
long St7SetNLAFreedomIncrementFactor(long uID, long Stage, long Increment,
     long CaseNum, double Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index – use 0 for unstaged analysis.

`Increment`

Increment number.

`CaseNum`

Freedom case number.

`Factor`

Freedom case factor.

St7GetNLAFreedomIncrementFactor

Returns the freedom case factors assigned for the specified increment in the nonlinear static analysis load table.

```
long St7GetNLAFreedomIncrementFactor(long uID, long Stage, long Increment,
     long CaseNum, double* Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index – use 0 for unstaged analysis.

`Increment`

Increment number.

`CaseNum`

Freedom case number.

Output Parameters

`Factor`

Freedom case factor.

St7SetNLAResetAtIncrement

Sets the **RESET MODEL** flag for the specified increment in the nonlinear static analysis load table.

```
long St7SetNLAResetAtIncrement(long uID, long Increment, bool Reset)
```

Input Parameters

`uID`

Strand7 model file ID.

`Increment`

Increment number.

`Reset`

True to reset the model.

St7GetNLAResetAtIncrement

Returns the **RESET MODEL** flag for the specified increment in the nonlinear static analysis load table.

```
long St7GetNLAResetAtIncrement(long uID, long Increment, bool* Reset)
```

Input Parameters

`uID`

Strand7 model file ID.

`Increment`

Increment number.

Output Parameters

`Reset`

True to reset the model.

St7EnableNLALoadCase

Enables the specified load case such that it is included in nonlinear static analysis.

```
long St7EnableNLALoadCase(long uID, long Stage, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index – use 0 for unstaged analysis.

`CaseNum`

Load case number.

St7DisableNLALoadCase

Disables the specified load case such that it is not included in nonlinear static analysis.

```
long St7DisableNLALoadCase(long uID, long Stage, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index – use 0 for unstaged analysis.

`CaseNum`

Load case number.

St7GetNLALoadCaseState

Returns the enabled state assigned to the specified load case for nonlinear static analysis.

```
long St7GetNLALoadCaseState(long uID, long Stage, long CaseNum, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index – use 0 for unstaged analysis.

`CaseNum`

Load case number.

Output Parameters

`Enabled`

True if the specified load case is enabled for nonlinear static analysis.

St7EnableNLAFreedomCase

Enables the specified freedom case such that it is included in nonlinear static analysis.

```
long St7EnableNLAFreedomCase(long uID, long Stage, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index – use 0 for unstaged analysis.

`CaseNum`

Freedom case number.

St7DisableNLAFreedomCase

Disables the specified freedom case such that it is not included in nonlinear static analysis.

```
long St7DisableNLAFreedomCase(long uID, long Stage, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index – use 0 for unstaged analysis.

`CaseNum`

Freedom case number.

St7GetNLAFreedomCaseState

Returns the enabled state of the specified freedom case for nonlinear static analysis.

```
long St7GetNLAFreedomCaseState(long uID, long Stage, long CaseNum, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index – use 0 for unstaged analysis.

`CaseNum`

Freedom case number.

Output Parameters

`Enabled`

True if the specified freedom case is enabled for nonlinear static analysis.

St7EnableNLAPseudoTime

Enables pseudo time for nonlinear static analysis.

```
long St7EnableNLAPseudoTime(long uID, long Stage)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index – use 0 for unstaged analysis.

St7DisableNLAPseudoTime

Disables pseudo time for nonlinear static analysis.

```
long St7DisableNLAPseudoTime(long uID, long Stage)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index – use 0 for unstaged analysis.

St7GetNLAPseudoTimeState

Returns the enabled state of pseudo time for nonlinear static analysis.

```
long St7GetNLAPseudoTimeState(long uID, long Stage, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index – use 0 for unstaged analysis.

Output Parameters

`Enabled`

True if pseudo time is enabled.

St7SetNLAPseudoTime

Assigns the pseudo time to the specified increment for nonlinear static analysis.

```
long St7SetNLAPseudoTime(long uID, long Stage, long Increment, double Time)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index – use 0 for unstaged analysis.

`Increment`

Increment number.

`Time`

Pseudo time.

St7GetNLAPseudoTime

Returns the pseudo time in the specified increment for nonlinear static analysis.

```
long St7GetNLAPseudoTime(long uID, long Stage, long Increment, double* Time)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index – use 0 for unstaged analysis.

`Increment`

Increment number.

Output Parameters

`Time`

Pseudo time.

St7SetNLAInitial

Assigns the initial conditions file to be used for nonlinear static analysis.

```
long St7SetNLAInitial(long uID, char* FileName, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the initial conditions file.

`CaseNum`

Result case number within FileName to be used as the initial conditions.

St7GetNLAInitial

Returns the initial conditions file assigned for nonlinear static analysis.

```
long St7GetNLAInitial(long uID, char* FileName, long* CaseNum, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxStringLen`

Maximum number of characters allocated for FileName.

Output Parameters

`FileName`

Full path and name for the initial conditions file.

`CaseNum`

Result case number within FileName to be used as the initial conditions.

Solver – Quasi-static

St7SetQSAInitial

Assigns the initial conditions file to be used for quasi-static analysis.

```
long St7SetQSAInitial(long uID, char* FileName, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the initial conditions file.

`CaseNum`

Result case number within FileName to be used as the initial conditions.

St7GetQSAInitial

Returns the initial conditions file assigned for quasi-static analysis.

```
long St7GetQSAInitial(long uID, char* FileName, long* CaseNum, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxStringLen`

Maximum number of characters allocated for FileName.

Output Parameters

`FileName`

Full path and name for the initial conditions file.

`CaseNum`

Result case number within FileName to be used as the initial conditions.

Solver – Natural Frequency

St7SetNFAInitial

Assigns the initial conditions file to be used for natural frequency analysis.

```
long St7SetNFAInitial(long uID, char* FileName, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the initial conditions file.

`CaseNum`

Result case number within FileName to be used as the initial conditions.

St7GetNFAInitial

Returns the initial conditions file assigned for natural frequency analysis. If an initial conditions file is specified, stress stiffening/softening effects will be included in the analysis.

```
long St7GetNFAInitial(long uID, char* FileName, long* CaseNum, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxStringLen`

Maximum number of characters allocated for FileName

Output Parameters

`FileName`

Full path and name for the initial conditions file.

`CaseNum`

Result case number within FileName to be used as the initial conditions.

St7SetNFANumModes

Sets the number of modes to be calculated in natural frequency analysis.

```
long St7SetNFANumModes(long uID, long NumModes)
```

Input Parameters

`uID`

Strand7 model file ID.

`NumModes`

Number of modes.

St7GetNFANumModes

Returns the number of modes to be calculated in natural frequency analysis.

```
long St7GetNFANumModes(long uID, long* NumModes)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumModes`

Number of modes.

St7SetNFAShift

Sets the frequency shift to be used in natural frequency analysis. The closest modes found, both above and below the shift value, will be calculated.

```
long St7SetNFAShift(long uID, double Shift)
```

Input Parameters

`uID`

Strand7 model file ID.

`Shift`

Frequency shift (Hz).

St7GetNFAShift

Returns the frequency shift assigned to the natural frequency analysis. The closest modes found, both above and below the shift value, will be calculated.

```
long St7GetNFAShift(long uID, double* Shift)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Shift`

Frequency shift (Hz).

St7SetNFAModeParticipationCalculate

Sets the state of the mass participation option for natural frequency analysis.

```
long St7SetNFAModeParticipationCalculate(long uID, bool Calculate)
```

Input Parameters

`uID`

Strand7 model file ID.

`Calculate`

True to calculate the mass participation for each mode in the analysis.

St7GetNFAModeParticipationCalculate

Returns the state of the mass participation option for natural frequency analysis.

```
long St7GetNFAModeParticipationCalculate(long uID, bool* Calculate)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Calculate`

True if the mass participation is calculated for each mode in the analysis.

St7SetNFAModeParticipationVectors

Assigns the direction vectors and origin, used when calculating mass participation factors for natural frequency analysis.

```
long St7SetNFAModeParticipationVectors(long uID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`Doubles[0..8]`

`[0..2]` – Vector components of the translation axis about which to calculate mode participation.

`[3..5]` – Origin for the calculation of rotational mode participation.

`[6..8]` – Vector components of the rotational axis about which to calculate mode participation.

Usage

All values are in the global XYZ system. If vector components are all zero, participation factors are returned for three global directions. Mode participation factors are recorded in the solver log file.

St7GetNFAModeParticipationVectors

Returns the direction vectors and origin assigned for natural frequency analysis, used when calculating mass participation factors.

```
long St7GetNFAModeParticipationVectors(long uID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Doubles[0..8]`

`[0..2]` – Vector components of the translation axis about which to calculate mode participation.

`[3..5]` – Origin for the calculation of rotational mode participation.

`[6..8]` – Vector components of the rotational axis about which to calculate mode participation.

Usage

All values are in the global XYZ system. If vector components are all zero, participation factors are returned for three global directions. Mode participation factors are recorded in the solver log file and in the result file. Factors in the result file are extracted by the function *St7GetModalResultsNFA*.

Solver – Harmonic Response

St7SetHRARange

Assigns the frequency range for harmonic response analysis.

```
long St7SetHRARange(long uID, long NumSteps, double F1, double F2,
     bool AutoInsert)
```

Input Parameters

`uID`

Strand7 model file ID.

`NumSteps`

Number of steps in the range.

`F1`

Starting frequency (Hz).

`F2`

Finishing frequency (Hz).

`AutoInsert`

True to automatically insert additional steps within the range. This feature is used to ensure that peaks in the frequency response are adequately captured.

St7GetHRARange

Returns the frequency range assigned for harmonic response analysis.

```
long St7GetHRARange(long uID, long* NumSteps, double* F1, double* F2,
     bool* AutoInsert)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumSteps`

Number of steps in the range.

`F1`

Starting frequency (Hz).

`F2`

Finishing frequency (Hz).

`AutoInsert`

True if additional steps are automatically inserted within the range. This feature is used to ensure that peaks in the frequency response are adequately captured.

St7SetHRALoadType

Assigns the load type for harmonic response analysis.

```
long St7SetHRALoadType(long uID, long LType)
```

Input Parameters

`uID`

Strand7 model file ID.

`LType`

One of hlBaseAcc, hlBaseVel, hlBaseDisp or hlAppliedLoad.

St7GetHRALoadType

Returns the load type for harmonic response analysis.

```
long St7GetHRALoadType(long uID, long* LType)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`LType`

One of hlBaseAcc, hlBaseVel, hlBaseDisp or hlAppliedLoad.

St7SetHRAMode

Sets the harmonic mode for harmonic response analysis.

```
long St7SetHRAMode(long uID, long MType)
```

Input Parameters

`uID`

Strand7 model file ID.

`MType`

Harmonic mode; either hmVsFrequency or hmVsTime.

St7GetHRAMode

Returns the harmonic mode for harmonic response analysis.

```
long St7GetHRAMode(long uID, long* MType)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`MType`

Harmonic mode; either hmVsFrequency or hmVsTime.

St7SetHRABaseVector

Assigns the base excitation vector for harmonic response analysis.

```
long St7SetHRABaseVector(long uID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`Doubles[0..2]`

A 3-element array containing the base excitation vector.

St7GetHRABaseVector

Returns the base excitation vector for harmonic response analysis.

```
long St7GetHRABaseVector(long uID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Doubles[0..2]`

A 3-element array containing the base excitation vector.

St7SetHRALoadCase

Assigns a harmonic load case factor, phase angle and frequency to the specified load case. This option is only used when the load type is set to **Applied Load**.

```
long St7SetHRALoadCase(long uID, long CaseNum, long TableID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

`TableID`

Factor vs Frequency/Period table ID, or 0 for none.

`Doubles[0..2]`

A 3-element array containing the load factor, the phase angle (degrees) and the frequency (Hz), for load case `CaseNum`. Note that the frequency is only used in **vs Time** analyses.

St7GetHRALoadCase

Returns the harmonic load case factor, phase angle and frequency assigned to the specified load case. This option is only used when the load type is set to **Applied Load**.

```
long St7GetHRALoadCase(long uID, long CaseNum, long* TableID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

Output Parameters

`TableID`

Factor vs Frequency/Period table ID, or 0 for none.

`Doubles[0..2]`

A 3-element array containing the load factor, the phase angle (degrees) and the frequency (Hz), for load case `CaseNum`. Note that the frequency is only used in **vs Time** analyses.

Solver – Spectral Response

St7SetSRABaseExcitation

Sets the status of the **Base Excitation** setting for Spectral Response analysis.

```
long St7SetSRABaseExcitation(long uID, bool Base)
```

Input Parameters

`uID`

Strand7 model file ID.

`Base`

True to include Base Excitation cases.

St7GetSRABaseExcitation

Returns the status of the **Base Excitation** setting for Spectral Response analysis.

```
long St7GetSRABaseExcitation(long uID, bool* Base)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Base`

True if Base Excitation cases are included.

St7SetSRALoadExcitation

Sets the status of the **Load Excitation** setting for Spectral Response analysis.

```
long St7SetSRALoadExcitation(long uID, bool Load)
```

Input Parameters

`uID`

Strand7 model file ID.

`Load`

True to include Base Excitation cases.

St7GetSRALoadExcitation

Returns the status of the **Load Excitation** setting for Spectral Response analysis.

```
long St7GetSRALoadExcitation(long uID, bool* Load)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Load`

True if Load Excitation cases are included.

St7AddSRALoadCase

Adds a new blank **Load Excitation** case to the Spectral Response analysis load excitation table.

```
long St7AddSRALoadCase(long uID, char* CaseName)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseName`

Spectral Response load excitation case name.

St7InsertSRALoadCase

Inserts a new blank **Load Excitation** case at the specified position within the Spectral Response analysis load excitation table.

```
long St7InsertSRALoadCase(long uID, long SRACase, char* CaseName)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response load excitation case number.

`CaseName`

Spectral Response load excitation case name.

St7DeleteSRALoadCase

Deletes the specified **Load Excitation** case from the Spectral Response analysis load excitation table.

```
long St7DeleteSRALoadCase(long uID, long SRACase)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response load excitation case number.

St7GetNumSRALoadCases

Returns the number of **Load Excitation** cases assigned for Spectral Response analysis.

```
long St7GetNumSRALoadCases(long uID, long* NumCases)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumCases`

Number of Spectral Response load excitation cases.

St7EnableSRALoadCase

Enables the specified **Load Excitation** case for Spectral Response analysis.

```
long St7EnableSRALoadCase(long uID, long SRACase)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response load excitation case number.

St7DisableSRALoadCase

Disables the specified **Load Excitation** case for Spectral Response analysis.

```
long St7DisableSRALoadCase(long uID, long SRACase)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response load excitation case number.

St7GetSRALoadCaseState

Returns the enabled state of the specified **Load Excitation** case for Spectral Response analysis.

```
long St7GetSRALoadCaseState(long uID, long SRACase, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response load excitation case number.

Output Parameters

`Enabled`

True if the case is enabled.

St7SetSRALoadCaseTable

Specifies the table associated with the specified Spectral Response analysis **Load Excitation** case.

```
long St7SetSRALoadCaseTable(long uID, long SRACase, long CaseNum, long TableID)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response load excitation case number.

`CaseNum`

Load case number.

`TableID`

Table ID number, or 0 for none.

St7GetSRALoadCaseTable

Returns the table associated with the specified Spectral Response analysis **Load Excitation** case.

```
long St7GetSRALoadCaseTable(long uID, long SRACase, long CaseNum, long* TableID)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response load excitation case number.

`CaseNum`

Load case number.

Output Parameters

`TableID`

Table ID number, or 0 for none.

St7SetSRALoadCaseName

Sets the name of the specified Spectral Response analysis **Load Excitation** case.

```
long St7SetSRALoadCaseName(long uID, long SRACase, char* CaseName)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response load excitation case number.

`CaseName`

String containing the new name of the load excitation case.

St7GetSRALoadCaseName

Returns the name of the specified Spectral Response analysis **Load Excitation** case.

```
long St7GetSRALoadCaseName(long uID, long SRACase, char* CaseName, long
     MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response load excitation case number.

`MaxStringLen`

Maximum number of characters allocated for CaseName.

Output Parameters

`CaseName`

String containing the name of the load excitation case.

St7AddSRABaseCase

Adds a new **Base Excitation** Spectral Response analysis case.

```
long St7AddSRABaseCase(long uID, char* CaseName)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseName`

Spectral Response base excitation case name.

St7InsertSRABaseCase

Inserts a new **Base Excitation** Spectral Response analysis case at the specified position.

```
long St7InsertSRABaseCase(long uID, long SRACase, char* CaseName)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response base excitation case number.

`CaseName`

Spectral Response base excitation case name.

St7DeleteSRABaseCase

Deletes the specified **Base Excitation** Spectral Response analysis case.

```
long St7DeleteSRABaseCase(long uID, long SRACase)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response base excitation case number.

St7GetNumSRABaseCases

Returns the number of **Base Excitation** cases assigned for Spectral Response Analysis.

```
long St7GetNumSRABaseCases(long uID, long* NumCases)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumCases`

Number of base excitation cases.

St7EnableSRABaseCase

Enables the specified **Base Excitation** case for Spectral Response analysis.

```
long St7EnableSRABaseCase(long uID, long SRACase)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response base excitation case number.

St7DisableSRABaseCase

Disables the specified **Base Excitation** case for Spectral Response analysis.

```
long St7DisableSRABaseCase(long uID, long SRACase)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response base excitation case number.

St7GetSRABaseCaseState

Returns the enabled state of the specified **Base Excitation** case for Spectral Response analysis.

```
long St7GetSRABaseCaseState(long uID, long SRACase, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response base excitation case number.

Output Parameters

`Enabled`

True if the case is enabled.

St7SetSRABaseCaseTable

Sets the table associated with the specified Spectral Response analysis **Base Excitation** case.

```
long St7SetSRABaseCaseTable(long uID, long SRACase, long TableID)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response base excitation case number.

`TableID`

Table ID number, or 0 for none.

St7GetSRABaseCaseTable

Returns the table associated with the specified Spectral Response analysis **Base Excitation** case.

```
long St7GetSRABaseCaseTable(long uID, long SRACase, long* TableID)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response base excitation case number.

Output Parameters

`TableID`

Table ID number, or 0 for none.

St7SetSRABaseCaseFactors

Assigns the components of the direction vector for the specified Spectral Response analysis **Base Excitation** case.

```
long St7SetSRABaseCaseFactors(long uID, long SRACase, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response base excitation case number.

`Doubles[0..2]`

A 3-element array containing the direction vector in the global XYZ system.

St7GetSRABaseCaseFactors

Returns the components of the direction vector assigned to the specified Spectral Response analysis **Base Excitation** case.

```
long St7GetSRABaseCaseFactors(long uID, long SRACase, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response base excitation case number.

Output Parameters

`Doubles[0..2]`

A 3-element array containing the direction vector in the global XYZ system.

St7SetSRABaseCaseType

Sets the modal load type for the specified Spectral Response analysis **Base Excitation** case.

```
long St7SetSRABaseCaseType(long uID, long SRACase, long VectType)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response base excitation case number.

`VectType`

One of slBaseAcc, slBaseVel or slBaseDisp.

St7GetSRABaseCaseType

Returns the modal load type for the specified Spectral Response analysis **Base Excitation** case.

```
long St7GetSRABaseCaseType(long uID, long SRACase, long* VectType)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response base excitation case number.

Output Parameters

`VectType`

One of slBaseAcc, slBaseVel or slBaseDisp.

St7SetSRABaseCaseName

Sets the name of the specified Spectral Response analysis **Base Excitation** case.

```
long St7SetSRABaseCaseName(long uID, long SRACase, char* CaseName)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response base excitation case number.

`CaseName`

String containing the new name of the base excitation case.

St7GetSRABaseCaseName

Returns the name of the specified Spectral Response analysis **Base Excitation** case.

```
long St7GetSRABaseCaseName(long uID, long SRACase, char* CaseName,
    long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRACase`

Spectral Response base excitation case number.

`MaxStringLen`

Maximum number of characters allocated for CaseName.

Output Parameters

`CaseName`

String containing the name of the base excitation case.

St7SetSRAResultSRSS

Sets the state of the **SRSS** result option for Spectral Response analysis.

```
long St7SetSRAResultSRSS(long uID, bool SRSS)
```

Input Parameters

`uID`

Strand7 model file ID.

`SRSS`

True to generate **SRSS** results.

St7GetSRAResultSRSS

Returns the state of the **SRSS** result option for Spectral Response analysis.

```
long St7GetSRAResultSRSS(long uID, bool* SRSS)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`SRSS`

True if **SRSS** results are generated.

St7SetSRAResultCQC

Sets the state of the **CQC** result option for Spectral Response analysis.

```
long St7SetSRAResultCQC(long uID, bool CQC)
```

Input Parameters

`uID`

Strand7 model file ID.

`CQC`

True to generate **CQC** results.

St7GetSRAResultCQC

Returns the state of the **CQC** result option for Spectral Response analysis.

```
long St7GetSRAResultCQC(long uID, bool* CQC)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`CQC`

True if **CQC** results are generated.

St7SetSRAType

Assigns the spectrum type used for Spectral Response analysis.

```
long St7SetSRAType(long uID, long SpectrumType)
```

Input Parameters

`uID`

Strand7 model file ID.

`SpectrumType`

Type of spectrum supplied; either stResponse or stPSD.

St7GetSRAType

Returns the spectrum type used for Spectral Response analysis.

```
long St7GetSRAType(long uID, long* SpectrumType)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`SpectrumType`

Type of spectrum supplied; either stResponse or stPSD.

St7SetSRAResultsSign

Sets the results sign option for Spectral Response analysis.

```
long St7SetSRAResultsSign(long uID, long ResultsSign)
```

Input Parameters

`uID`

Strand7 model file ID.

`ResultsSign`

Results sign; either rsAuto or rsAbsolute.

St7GetSRAResultsSign

Returns the results sign option for Spectral Response analysis.

```
long St7GetSRAResultsSign(long uID, long* ResultsSign)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`ResultsSign`

Results sign; either rsAuto or rsAbsolute.

St7SetAppendSRA

Returns the append/overwrite option for Spectral Response analysis.

```
long St7SetAppendSRA(long uID, bool Append)
```

Input Parameters

`uID`

Strand7 model file ID.

Append

True to append results onto an existing .SRA result file.

False to overwrite an existing result file.

St7GetAppendSRA

Returns the append/overwrite option for Spectral Response analysis.

```
long St7GetAppendSRA(long uID, bool* Append)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

Append

True if results are to be appended onto an existing .SRA result file; False if an existing result file is to be overwritten.

Solver – Linear Transient Dynamic

St7SetLTAInitial

Assigns the initial conditions file to be used for linear transient dynamic analysis.

```
long St7SetLTAInitial(long uID, char* FileName, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the initial conditions file.

`CaseNum`

Result case number within FileName to be used as the initial conditions.

St7GetLTAInitial

Returns the initial conditions file assigned for linear transient dynamic analysis.

```
long St7GetLTAInitial(long uID, char* FileName, long* CaseNum, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxStringLen`

Maximum number of characters allocated for FileName.

Output Parameters

`FileName`

Full path and name for the initial conditions file.

`CaseNum`

Result case number within FileName to be used as the initial conditions.

St7SetLTAMethod

Sets the integration method used for linear transient dynamic analysis.

```
long St7SetLTAMethod(long uID, long Method)
```

Input Parameters

`uID`

Strand7 model file ID.

`Method`

Time integration method; either ltWilson or ltNewmark.

St7GetLTAMethod

Returns the integration method assigned for linear transient dynamic analysis.

```
long St7GetLTAMethod(long uID, long* Method)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Method`

Time integration method; either ltWilson or ltNewmark.

St7SetLTASolutionType

Sets the solution type option for linear transient dynamic analysis.

```
long St7SetLTASolutionType(long uID, long SolutionType)
```

Input Parameters

`uID`

Strand7 model file ID.

`SolutionType`

Solution type; either stFullSystem or stSuperposition.

St7GetLTASolutionType

Returns the solution type option assigned for linear transient dynamic analysis.

```
long St7GetLTASolutionType(long uID, long* SolutionType)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`SolutionType`

Solution type; either stFullSystem or stSuperposition.

Solver – Nonlinear Transient Dynamic

St7SetNTAInitial

Assigns the initial conditions file to be used for nonlinear transient dynamic analysis.

```
long St7SetNTAInitial(long uID, char* FileName, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the initial conditions file.

`CaseNum`

Result case number within FileName to be used as the initial conditions.

St7GetNTAInitial

Returns the initial conditions file assigned for nonlinear transient dynamic analysis.

```
long St7GetNTAInitial(long uID, char* FileName, long* CaseNum, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxStringLen`

Maximum number of characters allocated for FileName

Output Parameters

`FileName`

Full path and name for the initial conditions file.

`CaseNum`

Result case number within FileName to be used as the initial conditions.

Solver – Transient Heat

St7SetTHAInitial

Assigns the initial conditions file to be used for transient heat analysis.

```
long St7SetTHAInitial(long uID, char* FileName, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the initial conditions file.

`CaseNum`

Result case number within FileName to be used as the initial conditions.

St7GetTHAInitial

Returns the initial conditions file assigned for transient heat analysis.

```
long St7GetTHAInitial(long uID, char* FileName, long* CaseNum, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxStringLen`

Maximum number of characters allocated for FileName.

Output Parameters

`FileName`

Full path and name for the initial conditions file.

`CaseNum`

Result case number within FileName to be used as the initial conditions.

St7SetTHATemperatureLoadCase

Assigns the load case that specifies the nodal temperature distribution for transient heat analysis.

```
long St7SetTHATemperatureLoadCase(long uID, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

St7GetTHATemperatureLoadCase

Returns the load case that specifies the nodal temperature distribution for transient heat analysis.

```
long St7GetTHATemperatureLoadCase(long uID, long* CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`CaseNum`

Load case number.

St7SetTHAInitialAttributeOverride

Assigns the state of the override flag to be used with the transient heat analysis initial conditions file.

```
long St7SetTHAInitialAttributeOverride(long uID, bool Active)
```

Input Parameters

`uID`

Strand7 model file ID.

`Active`

True for the initial result file temperatures to override nodal temperature attributes.

St7GetTHAInitialAttributeOverride

Returns the state of the override flag to be used with the transient heat analysis initial conditions file.

```
long St7GetTHAInitialAttributeOverride(long uID, bool* Active)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Active`

True if the initial result file temperatures override nodal temperature attributes.

Solver – Harmonic, Spectral and Linear Transient Dynamic

St7SetModalSuperpositionFile

Assigns the modal superposition file to be used for harmonic response, spectral response and linear transient dynamic analysis with mode superposition.

```
long St7SetModalSuperpositionFile(long uID, char* FileName)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the modal superposition file.

St7GetModalSuperpositionFile

Returns the modal superposition file assigned for harmonic response, spectral response and linear transient dynamic analysis with mode superposition.

```
long St7GetModalSuperpositionFile(long uID, char* FileName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxStringLen`

Maximum number of characters allocated for FileName.

Output Parameters

`FileName`

Full path and name for the modal superposition file.

St7GetNumModesInModalFile

Returns the number of modes found in the modal superposition file.

```
long St7GetNumModesInModalFile(long uID, long* NumModes)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumModes`

Number of modes in file.

Applicability

Applicable to harmonic response analysis and linear transient dynamic analysis with mode superposition.

St7GetNumModesInNFAFile

Returns the number of modes found in a natural frequency result file.

```
long St7GetNumModesInNFAFile(long uID, char* FileName, long* NumModes)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the natural frequency result file.

Output Parameters

`NumModes`

Number of modes in file.

St7EnableMode

Enables the specified mode in the modal superposition file for harmonic response, spectral response and linear transient dynamic analysis with mode superposition.

```
long St7EnableMode(long uID, long ModeNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`ModeNum`

Mode number to enable.

St7DisableMode

Disables the specified mode in the modal superposition file for harmonic response, spectral response and linear transient dynamic analysis with mode superposition.

```
long St7DisableMode(long uID, long ModeNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`ModeNum`

Mode number to disable.

St7GetModeState

Returns the enabled state of the specified mode in the modal superposition file for harmonic response, spectral response and linear transient dynamic analysis with mode superposition.

```
long St7GetModeState(long uID, long ModeNum, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`ModeNum`

Mode number.

Output Parameters

`Enabled`

True if the mode is enabled.

St7SetModeDampingRatio

Sets the modal damping ratio for the specified mode in the modal superposition file.

```
long St7SetModeDampingRatio(long uID, long ModeNum, double Ratio)
```

Input Parameters

`uID`

Strand7 model file ID.

`ModeNum`

Mode number.

`Ratio`

Modal damping ratio.

Applicability

Applicable to harmonic response and linear transient dynamic analysis with mode superposition.

St7GetModeDampingRatio

Returns the modal damping ratio assigned for the specified mode in the modal superposition file.

```
long St7GetModeDampingRatio(long uID, long ModeNum, double* Ratio)
```

Input Parameters

`uID`

Strand7 model file ID.

`ModeNum`

Mode number.

Output Parameters

`Ratio`

Modal damping ratio.

Applicability

Applicable to harmonic response, spectral response and linear transient dynamic analysis with mode superposition.

Solver – Linear and Nonlinear Transient Dynamic

St7SetTransientBaseExcitation

Sets the type of base excitation for linear and nonlinear transient dynamic analysis.

```
long St7SetTransientBaseExcitation(long uID, long BaseType)
```

Input Parameters

`uID`

Strand7 model file ID.

`BaseType`

One of beNone, beAcceleration, beVelocity or beDisplacement.

St7GetTransientBaseExcitation

Returns the type of base excitation for linear and nonlinear transient dynamic analysis.

```
long St7GetTransientBaseExcitation(long uID, long* BaseType)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`BaseType`

One of beNone, beAcceleration, beVelocity or beDisplacement.

St7SetTransientInitialConditionsType

Sets the type of initial conditions to be used for linear and nonlinear transient dynamic analysis.

```
long St7SetTransientInitialConditionsType(long uID, long InitialType)
```

Input Parameters

`uID`

Strand7 model file ID.

`InitialType`

One of icNone, icAppliedVectors, icNodalVelocity or icFromFile.

St7GetTransientInitialConditionsType

Returns the type of initial conditions assigned for linear and nonlinear transient dynamic analysis.

```
long St7GetTransientInitialConditionsType(long uID, long* InitialType)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`InitialType`

One of icNone, icAppliedVectors, icNodalVelocity or icFromFile.

St7SetTransientInitialConditionsVectors

Sets the initial acceleration and velocity vectors used for linear and nonlinear transient dynamic analysis. A uniform acceleration and velocity is applied to all nodes.

```
long St7SetTransientInitialConditionsVectors(long uID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`Doubles[0..5]`

`[0..2]` – Initial acceleration components in the global XYZ system.

`[3..5]` – Initial velocity components in the global XYZ system.

St7GetTransientInitialConditionsVectors

Returns the initial acceleration and velocity vectors assigned for linear and nonlinear transient dynamic analysis. A uniform acceleration and velocity is applied to all nodes.

```
long St7GetTransientInitialConditionsVectors(long uID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Doubles[0..5]`

`[0..2]` – Initial acceleration components in the global XYZ system.

`[3..5]` – Initial velocity components in the global XYZ system.

St7SetTransientInitialConditionsNodalVelocity

Sets the load case that specifies the initial nodal velocity for linear and nonlinear transient dynamic analysis. The initial velocity components are defined by the **Initial Velocity** nodal attribute.

```
long St7SetTransientInitialConditionsNodalVelocity(long uID, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

St7GetTransientInitialConditionsNodalVelocity

Returns the load case that specifies the initial velocity for linear and nonlinear transient dynamic analysis. The initial velocity components are defined by the **Initial Velocity** nodal attribute.

```
long St7GetTransientInitialConditionsNodalVelocity(long uID, long* CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`CaseNum`

Load case number.

St7SetTransientBaseVector

Sets the base excitation vector for linear and nonlinear transient dynamic analysis. The values in the base excitation tables (acceleration, velocity and displacement), are multiplied by the respective base vector component.

```
long St7SetTransientBaseVector(long uID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`Doubles[0..2]`

Components in the global XYZ system.

St7GetTransientBaseVector

Returns the base excitation vector assigned for linear and nonlinear transient dynamic analysis. The values in the base excitation tables (acceleration, velocity and displacement), are multiplied by the respective base vector component.

```
long St7GetTransientBaseVector(long uID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Doubles[0..2]`

Components in the global XYZ system.

St7SetTransientBaseAcceleration

Sets the initial base acceleration for linear and nonlinear transient dynamic analysis. All restrained nodes in the specified model will initially have this acceleration.

```
long St7SetTransientBaseAcceleration(long uID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`Doubles[0..2]`

Base acceleration components in the global XYZ system.

Usage

Transient base acceleration is used as initial conditions for base velocity and base displacement excitation.

St7GetTransientBaseAcceleration

Returns the initial base acceleration assigned for linear and nonlinear transient dynamic analysis. All restrained nodes in the specified model will initially have this acceleration.

```
long St7GetTransientBaseAcceleration(long uID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Doubles[0..2]`

Base acceleration components in the global XYZ system.

Usage

Transient base acceleration is used as initial conditions for base velocity and base displacement excitation.

St7SetTransientBaseVelocity

Sets the initial base velocity for linear and nonlinear transient dynamic analysis. All restrained nodes in the specified model will initially have this velocity.

```
long St7SetTransientBaseVelocity(long uID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`Doubles[0..2]`

Base velocity components in the global XYZ system.

Usage

Transient base velocity is used as initial conditions for base acceleration and base displacement excitation.

St7GetTransientBaseVelocity

Returns the initial base velocity assigned for linear and nonlinear transient dynamic analysis. All restrained nodes in the specified model will initially have this velocity.

```
long St7GetTransientBaseVelocity(long uID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Doubles[0..2]`

Base velocity components in the global XYZ system.

Usage

Transient base velocity is used as initial conditions for base acceleration and base displacement excitation.

St7SetTransientBaseDisplacement

Sets the initial base displacement for linear and nonlinear transient dynamic analysis. All restrained nodes in the specified model will initially have this displacement.

```
long St7SetTransientBaseDisplacement(long uID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`Doubles[0..2]`

Base displacement components in the global XYZ system.

Usage

Transient base displacement is used as initial conditions for base acceleration and base velocity excitation.

St7GetTransientBaseDisplacement

Returns the initial base displacement assigned for linear and nonlinear transient dynamic analysis. All restrained nodes in the specified model will initially have this displacement.

```
long St7GetTransientBaseDisplacement(long uID, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Doubles[0..2]`

Base displacement components in the global XYZ system.

Usage

Transient base displacement is used as initial conditions for base acceleration and base velocity excitation.

St7SetTransientBaseTables

Specifies the time tables to be associated with the base excitation components for linear and nonlinear transient dynamic analysis.

```
long St7SetTransientBaseTables(long uID, long BaseType, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`BaseType`

One of beAcceleration, beVelocity or beDisplacement.

`Integers[0..2]`

ID numbers for tables in the global XYZ directions, or 0 for none; Acceleration vs Time table for beAcceleration, Velocity vs Time table for beVelocity, and Displacement vs Time table for beDisplacement.

St7GetTransientBaseTables

Returns the time tables associated with the base excitation components for linear and nonlinear transient dynamic analysis.

```
long St7GetTransientBaseTables(long uID, long BaseType, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`BaseType`

One of beAcceleration, beVelocity or beDisplacement.

Output Parameters

`Integers[0..2]`

ID numbers for tables in the global XYZ directions, or 0 for none; Acceleration vs Time table for beAcceleration, Velocity vs Time table for beVelocity, and Displacement vs Time table for beDisplacement.

St7AddTransientNodeHistoryCase

Adds a new node history case for linear and nonlinear transient dynamic analysis.

```
long St7AddTransientNodeHistoryCase(long uID, long NodeNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

St7InsertTransientNodeHistoryCase

Inserts a new node history case for linear and nonlinear transient dynamic analysis.

```
long St7InsertTransientNodeHistoryCase(long uID, long Pos, long NodeNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Pos`

Node history case number.

`NodeNum`

Node number.

St7DeleteTransientNodeHistoryCase

Deletes the specified node history case for linear and nonlinear transient dynamic analysis.

```
long St7DeleteTransientNodeHistoryCase(long uID, long Pos)
```

Input Parameters

`uID`

Strand7 model file ID.

`Pos`

Node history case number.

St7GetNumTransientNodeHistoryCases

Returns the number of node history cases assigned for linear and nonlinear transient dynamic analysis.

```
long St7GetNumTransientNodeHistoryCases(long uID, long* NumCases)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumCases`

Number of node history cases.

St7SetTransientNodeHistoryCaseData

Assigns the settings for the specified node history case for linear and nonlinear transient dynamic analysis.

```
long St7SetTransientNodeHistoryCaseData(long uID, long Pos, long NodeNum,
    bool* Logicals)
```

Input Parameters

`uID`

Strand7 model file ID.

`Pos`

Node history case number.

`NodeNum`

Node number.

`Logicals[0..5]`

`[0..2]` – True to include each of the global XYZ nodal result components respectively.

`[3..5]` – True to include displacement, velocity and acceleration results respectively.

St7GetTransientNodeHistoryCaseData

Returns the settings assigned for the specified node history case for linear and nonlinear transient Dynamic analysis.

```
long St7GetTransientNodeHistoryCaseData(long uID, long Pos, long* NodeNum,
     bool* Logicals)
```

Input Parameters

`uID`

Strand7 model file ID.

`Pos`

Node history case number.

Output Parameters

`NodeNum`

Node number.

`Logicals[0..5]`

`[0..2]` – True to include each of the global XYZ nodal result components respectively.

`[3..5]` – True to include displacement, velocity and acceleration results respectively.

Solver – Quasi-static and Nonlinear Transient Dynamic

St7SetTransientTemperatureInputType

Sets the type of temperature data to be used for quasi-static and nonlinear transient dynamic analysis.

```
long St7SetTransientTemperatureInputType(long uID, long InputType)
```

Input Parameters

`uID`

Strand7 model file ID.

`InputType`

Temperature type; either ttFromFile or ttNodalTemp.

St7GetTransientTemperatureInputType

Returns the type of temperature data to be used for quasi-static and nonlinear transient dynamic analysis.

```
long St7GetTransientTemperatureInputType(long uID, long* InputType)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`InputType`

Temperature type; either ttFromFile or ttNodalTemp.

St7SetTransientHeatFile

Assigns the temperature file to be used for quasi-static and nonlinear transient dynamic analysis.

```
long St7SetTransientHeatFile(long uID, char* FileName, double RefTemp)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the temperature file.

`RefTemp`

Reference temperature.

St7GetTransientHeatFile

Returns the temperature file assigned for quasi-static and nonlinear transient dynamic analysis.

```
long St7GetTransientHeatFile(long uID, char* FileName, long MaxStringLen,
     double* RefTemp)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxStringLen`

Maximum number of characters allocated for FileName.

Output Parameters

`FileName`

Full path and name for the temperature file.

`RefTemp`

Reference temperature.

St7SetTransientLoadPositionTable

Assigns a Factor vs Position table that factors loads on the basis of a selected spatial Degree of Freedom (DoF) for the specified load case for quasi-static and nonlinear transient dynamic analysis.

```
long St7SetTransientLoadPositionTable(long uID, long CaseNum, long TableID,
     long UCSId, long Axis)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

`TableID`

ID number for the Factor vs Position table, or 0 for none.

`UCSId`

ID number of the UCS supplying a spatial DoF to factor. UCSId = 1 refers to the global XYZ system.

`Axis`

The UCS DoF used, 1, 2 or 3.

St7GetTransientLoadPositionTable

Returns the Factor vs Position table that factors loads on the basis of a selected spatial Degree of Freedom (DoF) for the specified load case for quasi-static and nonlinear transient dynamic analysis.

```
long St7GetTransientLoadPositionTable(long uID, long CaseNum, long* TableID,
     long* UCSId, long* Axis)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

Output Parameters

`TableID`

ID number for the Factor vs Position table, or 0 for none.

`UCSId`

ID number of the UCS supplying a spatial DoF to factor. UCSId = 1 refers to the global XYZ system.

`Axis`

The UCS DoF used, 1, 2 or 3.

St7SetTransientFreedomPositionTable

Assigns a Factor vs Position table that factors restraint conditions on the basis of a selected spatial Degree of Freedom (DoF) for the specified freedom case for quasi-static and nonlinear transient dynamic analysis.

```
long St7SetTransientFreedomPositionTable(long uID, long CaseNum, long TableID,
     long UCSId, long Axis)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Freedom case number.

`TableID`

ID number for the Factor vs Position table, or 0 for none.

`UCSId`

ID number of the UCS supplying a spatial DoF to factor. UCSId = 1 refers to the global XYZ system.

`Axis`

The UCS DoF used, 1, 2 or 3.

St7GetTransientFreedomPositionTable

Returns the Factor vs Position table that factors restraint conditions on the basis of a selected spatial Degree of Freedom (DoF) for the specified freedom case for quasi-static and nonlinear transient dynamic analysis.

```
long St7GetTransientFreedomPositionTable(long uID, long CaseNum, long* TableID,
     long* UCSId, long* Axis)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Freedom case number.

Output Parameters

`TableID`

ID number for the Factor vs Position table, or 0 for none.

`UCSId`

ID number of the UCS supplying a spatial DoF to factor. UCSId = 1 refers to the global XYZ system.

`Axis`

The UCS DoF used, 1, 2 or 3.

St7GetInitialTemperatureInTHAFile

Returns the temperature used as the initial temperature in a transient heat result file.

```
long St7GetInitialTemperatureInTHAFile(long uID, char* FileName,
     double* InitialTemp)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the transient heat result file.

Output Parameters

`InitialTemp`

The initial temperature.

Solver – Quasi-static and Transient Dynamic

St7EnableTransientLoadCase

Enables the specified load case for quasi-static and transient dynamic analysis.

```
long St7EnableTransientLoadCase(long uID, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

St7DisableTransientLoadCase

Disables the specified load case for quasi-static and transient dynamic analysis.

```
long St7DisableTransientLoadCase(long uID, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

St7GetTransientLoadCaseState

Returns the enabled state of the specified load case for quasi-static and transient dynamic analysis.

```
long St7GetTransientLoadCaseState(long uID, long CaseNum, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

Output Parameters

`Enabled`

True if the specified load case is enabled.

St7EnableTransientFreedomCase

Enables the specified freedom case for quasi-static and transient dynamic analysis.

```
long St7EnableTransientFreedomCase(long uID, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Freedom case number.

St7DisableTransientFreedomCase

Disables the specified freedom case for quasi-static and transient dynamic analysis.

```
long St7DisableTransientFreedomCase(long uID, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Freedom case number.

St7GetTransientFreedomCaseState

Returns the enabled state of the specified freedom case for quasi-static and transient dynamic analysis.

```
long St7GetTransientFreedomCaseState(long uID, long CaseNum, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Freedom case number.

Output Parameters

`Enabled`

True if the specified freedom case is enabled.

St7SetTransientLoadTimeTable

Specifies the Factor vs Time table to be associated with a given load case for quasi-static and transient dynamic analysis.

```
long St7SetTransientLoadTimeTable(long uID, long CaseNum, long TableID,
    bool AddTimeSteps)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

`TableID`

Factor vs Time table ID, or 0 for none.

`AddTimeSteps`

True to **Add Time Steps** at ordinates in the Factor vs Time table.

St7GetTransientLoadTimeTable

Returns the Factor vs Time table associated with the specified load case for quasi-static and transient dynamic analysis.

```
long St7GetTransientLoadTimeTable(long uID, long CaseNum, long* TableID,
    bool* AddTimeSteps)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

Output Parameters

`TableID`

Factor vs Time table ID, or 0 for none.

`AddTimeSteps`

True to **Add Time Steps** at ordinates in the Factor vs Time table.

St7SetTransientFreedomTimeTable

Specifies the Factor vs Time table to be associated with a given freedom case for quasi-static and transient dynamic analysis.

```
long St7SetTransientFreedomTimeTable(long uID, long CaseNum, long TableID,
     bool AddTimeSteps)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Freedom case number.

`TableID`

Factor vs Time table ID, or 0 for none.

`AddTimeSteps`

True to **Add Time Steps** at ordinates in the Factor vs Time table.

St7GetTransientFreedomTimeTable

Returns the Factor vs Time table associated with the specified freedom case for quasi-static and transient dynamic analysis.

```
long St7GetTransientFreedomTimeTable(long uID, long CaseNum, long* TableID,
     bool* AddTimeSteps)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Freedom case number.

Output Parameters

`TableID`

Factor vs Time table ID, or 0 for none.

`AddTimeSteps`

True to **Add Time Steps** at ordinates in the Factor vs Time table.

Solver – Steady Heat and Transient Heat

St7EnableHeatLoadCase

Enables the specified load case such that is included in steady heat and transient heat analysis.

```
long St7EnableHeatLoadCase(long uID, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

St7DisableHeatLoadCase

Disables the specified load case such that it is not included in steady heat and transient heat analysis.

```
long St7DisableHeatLoadCase(long uID, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

St7GetHeatLoadCaseState

Returns the enabled state of the specified load case for steady heat and transient heat analysis.

```
long St7GetHeatLoadCaseState(long uID, long CaseNum, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

Output Parameters

`Enabled`

True if the specified load case is enabled for steady heat or transient heat analysis.

St7SetSolverHeatNonlinear

Sets the state of the nonlinear option for steady heat and transient heat analysis. Models containing radiative boundary conditions or temperature dependent material conditions should use the nonlinear analysis option.

```
long St7SetSolverHeatNonlinear(long uID, bool Nonlinear)
```

Input Parameters

`uID`

Strand7 model file ID.

`Nonlinear`

True to perform nonlinear heat analyses. The nonlinear flag must be active to solve problems incorporating radiative boundary conditions or temperature dependent material properties.

Solver – Natural Frequency and Transient Dynamic

St7EnableNSMassCaseInMassMatrix

Enables the non-structural mass for the specified load case such that it is included in mass matrix for natural frequency and transient dynamic analysis.

```
long St7EnableNSMassCaseInMassMatrix(long uID, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

St7DisableNSMassCaseInMassMatrix

Disables the non-structural mass for the specified load case such that it is not included in mass matrix for natural frequency and transient dynamic analysis.

```
long St7DisableNSMassCaseInMassMatrix(long uID, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

St7GetNSMassCaseInMassMatrixState

Returns whether non-structural mass in the specified load case is enabled for inclusion in the mass matrix for natural frequency and transient dynamic analysis.

```
long St7GetNSMassCaseInMassMatrixState(long uID, long CaseNum, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

Output Parameters

`Enabled`

True if the non-structural mass is enabled for the specified load case.

Solver – Time Stepping

These functions apply to transient dynamic, quasi-static and transient heat analysis.

St7SetNumTimeStepRows

Sets the number of rows used to specify the time integration intervals for the analysis. Each row may have separate time step and integration settings.

```
long St7SetNumTimeStepRows(long uID, long NumRows)
```

Input Parameters

uID

Strand7 model file ID.

NumRows

Number of rows.

St7GetNumTimeStepRows

Returns the number of rows used to specify the time integration interval for the analysis. Each row may have separate time step and integration settings.

```
long St7GetNumTimeStepRows(long uID, long* NumRows)
```

Input Parameters

uID

Strand7 model file ID.

Output Parameters

NumRows

Number of rows.

St7SetTimeStepData

Sets the time step and integration data used for the analysis. The integration data may be specified over multiple rows.

```
long St7SetTimeStepData(long uID, long Row, long NumSteps, long SaveEvery,
    double TimeStep)
```

Input Parameters

uID

Strand7 model file ID.

Row

Integration data row.

`NumSteps`

Total number of time steps in row.

`SaveEvery`

Number of time steps between successive result cases.

`TimeStep`

Time step size.

St7GetTimeStepData

Returns the time step and integration data used for the analysis. The integration data may be specified over multiple rows.

```
long St7GetTimeStepData(long uID, long Row, long* NumSteps, long* SaveEvery,
     double* TimeStep)
```

Input Parameters

`uID`

Strand7 model file ID.

`Row`

Integration data row.

Output Parameters

`NumSteps`

Total number of time steps in row.

`SaveEvery`

Number of time steps between successive result cases.

`TimeStep`

Time step size.

St7SetTimeStepUnit

Sets the units for time stepping for the analysis.

```
long St7SetTimeStepUnit(long uID, long TimeUnit)
```

Input Parameters

`uID`

Strand7 model file ID.

`TimeUnit`

One of tuMilliSec, tuSec, tuMin, tuHour, tuDay.

St7GetTimeStepUnit

Returns the units assigned for time stepping for the analysis.

```
long St7GetTimeStepUnit(long uID, long* TimeUnit)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`TimeUnit`

One of tuMilliSec, tuSec, tuMin, tuHour, tuDay.

Solver – Moving Load

These functions apply to quasi-static, transient dynamic and transient heat analysis.

St7EnableMovingLoad

Enables the specified moving load path for the analysis.

```
long St7EnableMovingLoad(long uID, long LoadPathID)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoadPathID`

Load path ID.

St7DisableMovingLoad

Disables the specified moving load path for the analysis.

```
long St7DisableMovingLoad(long uID, long LoadPathID)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoadPathID`

Load path ID.

St7GetMovingLoadState

Returns the enabled state of the specified moving load path for the analysis.

```
long St7GetMovingLoadState(long uID, long LoadPathID, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoadPathID`

Load path ID.

Output Parameters

`Enabled`

True if the load path is enabled.

St7SetMovingLoadTimeTable

Specifies the Factor vs Time table to be associated with a given load path for the analysis.

```
long St7SetMovingLoadTimeTable(long uID, long LoadPathID, long TableID)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoadPathID`

Load path ID.

`TableID`

Factor vs Time table ID number, or 0 to clear the table.

St7GetMovingLoadTimeTable

Returns the Factor vs Time table associated with a given load path for the analysis.

```
long St7GetMovingLoadTimeTable(long uID, long LoadPathID, long* TableID)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoadPathID`

Load path ID.

Output Parameters

`TableID`

Factor vs Time table ID number, or 0 if no table is assigned.

St7SetMovingLoadAutoDivisions

Sets the option to use load path divisions appropriate for the timesteps for a given load path.

```
long St7SetMovingLoadAutoDivisions(long uID, long LoadPathID, bool Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoadPathID`

Load path ID.

`Enabled`

True to allow the solver to set load path divisions.

St7GetMovingLoadAutoDivisions

Returns the option to use load path divisions appropriate for the timesteps for a given load path.

```
long St7GetMovingLoadAutoDivisions(long uID, long LoadPathID, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoadPathID`

Load path ID.

Output Parameters

`Enabled`

True if the solver is allowed to set load path divisions.

Solver – General

St7SetSolverScheme

Sets the scheme to be used for the solution of the linear system arising from the finite element model.

```
long St7SetSolverScheme(long uID, long Scheme)
```

Input Parameters

`uID`

Strand7 model file ID.

`Scheme`

One of stSkyline, stSparse or stIterativePCG.

St7GetSolverScheme

Returns the scheme assigned for the solution of the linear system arising from the finite element model.

```
long St7GetSolverScheme(long uID, long* Scheme)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Scheme`

One of stSkyline, stSparse or stIterativePCG.

St7SetSolverSort

Sets the node number re-ordering strategy used by the solver.

```
long St7SetSolverSort(long uID, long Sort)
```

Input Parameters

`uID`

Strand7 model file ID.

`Sort`

One of rnNone, rnTree, rnGeometry or rnAMD.

St7GetSolverSort

Returns the node number re-ordering strategy assigned to the solver.

```
long St7GetSolverSort(long uID, long* Sort)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Sort`

One of rnNone, rnTree, rnGeometry or rnAMD.

St7SetSolverTreeStartNumber

Sets the starting node number for the Tree type re-ordering strategy.

```
long St7SetSolverTreeStartNumber(long uID, long Start)
```

Input Parameters

`uID`

Strand7 model file ID.

`Start`

Starting node number.

St7GetSolverTreeStartNumber

Returns the starting node number assigned for the Tree type re-ordering strategy.

```
long St7GetSolverTreeStartNumber(long uID, long* Start)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Start`

Starting node number.

St7SetSolverActiveStage

Sets the active stage for the analysis.

```
long St7SetSolverActiveStage(long uID, long Stage)
```

Input Parameters

`uID`

Strand7 model file ID.

`Stage`

Stage index.

St7GetSolverActiveStage

Returns the active stage assigned for the analysis.

```
long St7GetSolverActiveStage(long uID, long* Stage)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Stage`

Stage index.

St7SetSolverTemperatureDependence

Specifies the type of temperature dependence used in the analysis.

```
long St7SetSolverTemperatureDependence(long uID, long TempType)
```

Input Parameters

`uID`

Strand7 model file ID.

`TempType`

Temperature dependence type; either tdNone or tdCombined.

Applicability

Applicable to quasi-static, nonlinear static and nonlinear transient dynamic analysis.

St7GetSolverTemperatureDependence

Returns the type of temperature dependence used for the analysis.

```
long St7GetSolverTemperatureDependence(long uID, long* TempType)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`TempType`

Temperature dependence type; either tdNone or tdCombined.

Applicability

Applicable to quasi-static, nonlinear static and nonlinear transient dynamic analysis.

St7SetSolverLoadCaseTemperatureDependence

Sets the load case used to specify the temperature dependence for the analysis.

```
long St7SetSolverLoadCaseTemperatureDependence(long uID, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number or 0 for none.

Applicability

Applicable to linear static, load influence, natural frequency and linear transient dynamic analysis.

St7GetSolverLoadCaseTemperatureDependence

Returns the load case assigned to specify the temperature dependence for the analysis.

```
long St7GetSolverLoadCaseTemperatureDependence(long uID, long* CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`CaseNum`

Load case number or 0 for none.

Applicability

Applicable to linear static, load influence, natural frequency and linear transient dynamic analysis.

St7SetSolverLoadCaseCableInertia

Sets the load case that specifies the inertia loads applied to cable elements in the linear solvers. The weight of the cable defines the cable stiffness.

```
long St7SetSolverLoadCaseCableInertia(long uID, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

St7GetSolverLoadCaseCableInertia

Returns the load case that specifies the inertia loads applied to cable elements in the linear solvers. The weight of the cable defines the cable stiffness.

```
long St7GetSolverLoadCaseCableInertia(long uID, long* CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`CaseNum`

Load case number.

St7SetSolverLoadCaseCablePreLoad

Sets the load case that specifies the pre-loads applied to cable elements.

```
long St7SetSolverLoadCaseCablePreLoad(long uID, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

Applicability

Applicable to linear transient dynamic analysis.

St7GetSolverLoadCaseCablePreLoad

Returns the load case that specifies the pre-loads applied to cable elements.

```
long St7GetSolverLoadCaseCablePreLoad(long uID, long* CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`CaseNum`

Load case number.

Applicability

Applicable to linear transient dynamic analysis.

St7SetSolverFreedomCase

Sets the freedom case used for the analysis. Multiple freedom cases may be specified for linear static analysis using the *St7EnableLSALoadCase* function.

```
long St7SetSolverFreedomCase(long uID, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Freedom case number.

St7GetSolverFreedomCase

Returns the freedom case assigned for the analysis. Multiple freedom cases may be specified for linear static analysis using the *St7EnableLSALoadCase* function.

```
long St7GetSolverFreedomCase(long uID, long* CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`CaseNum`

Freedom case number.

St7SetDampingType

Sets the type of damping used for the analysis.

```
long St7SetDampingType(long uID, long DampType)
```

Input Parameters

`uID`

Strand7 model file ID.

`DampType`

One of dtNoDamping, dtRayleighDamping, dtModalDamping or dtViscousDamping.

St7GetDampingType

Returns the type of damping assigned for the analysis.

```
long St7GetDampingType(long uID, long* DampType)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`DampType`

One of dtNoDamping, dtRayleighDamping, dtModalDamping or dtViscousDamping.

St7SetRayleighFactors

Sets the Rayleigh damping factors used for the analysis.

```
long St7SetRayleighFactors(long uID, long RayleighMode, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`RayleighMode`

Type of Rayleigh factors specified; either rmSetFrequencies or rmSetAlphaBeta.

`Doubles[0..5]`

rmSetAlphaBeta:

`[ipRayleighAlpha]` – Alpha.

`[ipRayleighBeta]` – Beta.

rmSetFrequencies:

`[ipRayleighF1]` – Rayleigh damping frequency 1.

`[ipRayleighF2]` – Rayleigh damping frequency 2.

`[ipRayleighR1]` – Rayleigh damping ratio 1.

`[ipRayleighR2]` – Rayleigh damping ratio 2.

Both:

`[ipRayleighDisplayF1]` – Rayleigh damping graph display frequency 1.

`[ipRayleighDisplayF2]` – Rayleigh damping graph display frequency 2.

St7GetRayleighFactors

Returns the Rayleigh damping factors assigned for the analysis.

```
long St7GetRayleighFactors(long uID, long* RayleighMode, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`RayleighMode`

Type of Rayleigh factors specified; either rmSetFrequencies or rmSetAlphaBeta.

`Doubles[0..5]`

rmSetAlphaBeta:

`[ipRayleighAlpha]` – Alpha.

`[ipRayleighBeta]` - Beta.

rmSetFrequencies:

`[ipRayleighF1]` – Rayleigh damping frequency 1.

`[ipRayleighF2]` – Rayleigh damping frequency 2.

`[ipRayleighR1]` – Rayleigh damping ratio 1.

`[ipRayleighR2]` – Rayleigh damping ratio 2.

Both:

`[ipRayleighDisplayF1]` – Rayleigh damping graph display frequency 1.

`[ipRayleighDisplayF2]` – Rayleigh damping graph display frequency 2.

St7SetSoilFluidOptions

Sets the soil/fluid options for the analysis. These parameters are only used for models containing soil or fluid properties.

```
long St7SetSoilFluidOptions(long uID, long CaseNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Gravitational load case number.

`Doubles[0..2]`

`[0]` – Default fluid level.

`[1]` – Fluid mass density per unit volume.

`[2]` – Fluid bulk modulus.

St7GetSoilFluidOptions

Returns the soil/fluid parameters assigned for the analysis. These parameters are only used for models containing soil or fluid properties.

```
long St7GetSoilFluidOptions(long uID, long* CaseNum, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`CaseNum`

Gravitational load case number.

`Doubles[0..2]`

`[0]` – Default fluid level.

`[1]` – Fluid mass density per unit volume.

`[2]` – Fluid bulk modulus.

St7SetSoilAutoDrained

Sets the state of the option to **Set Undrained elements above waterline to Drained**.

```
long St7SetSoilAutoDrained(long uID, bool Active)
```

Input Parameters

`uID`

Strand7 model file ID.

`Active`

True to set Undrained elements above the water line to Drained.

St7GetSoilAutoDrained

Returns the state of the option to **Set Undrained elements above waterline to Drained**.

```
long St7GetSoilAutoDrained(long uID, bool* Active)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Active`

True to set Undrained elements above the water line to Drained.

St7SetSturmCheck

Assigns the state of the **Sturm Check** option, for eigenvalue analyses.

```
long St7SetSturmCheck(long uID, bool DoSturm)
```

Input Parameters

`uID`

Strand7 model file ID.

`DoSturm`

True to enable the **Sturm Check**.

St7GetSturmCheck

Returns the state of the **Sturm Check** option, for eigenvalue analyses.

```
long St7GetSturmCheck(long uID, bool* DoSturm)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`DoSturm`

True if the **Sturm Check** is enabled.

St7SetSolverNonlinearGeometry

Sets the state of the **Nonlinear Geometry** option for nonlinear analyses.

```
long St7SetSolverNonlinearGeometry(long uID, bool NonlinearGeometry)
```

Input Parameters

`uID`

Strand7 model file ID.

`NonlinearGeometry`

True to enable the **Nonlinear Geometry** option.

St7GetSolverNonlinearGeometry

Returns the state assigned for the **Nonlinear Geometry** option for nonlinear analyses.

```
long St7GetSolverNonlinearGeometry(long uID, bool* NonlinearGeometry)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NonlinearGeometry`

True if the **Nonlinear Geometry** option is enabled.

St7SetSolverNonlinearMaterial

Sets the state of the **Nonlinear Material** option for nonlinear analyses.

```
long St7SetSolverNonlinearMaterial(long uID, bool NonlinearMaterial)
```

Input Parameters

`uID`

Strand7 model file ID.

`NonlinearMaterial`

True to enable the **Nonlinear Material** option.

St7GetSolverNonlinearMaterial

Returns the state assigned for the **Nonlinear Material** option for nonlinear analyses.

```
long St7GetSolverNonlinearMaterial(long uID, bool* NonlinearMaterial)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NonlinearMaterial`

True if the **Nonlinear Material** option is enabled.

St7SetSolverCreep

Sets the state of the **Creep** option for creep analyses.

```
long St7SetSolverCreep(long uID, bool Creep)
```

Input Parameters

`uID`

Strand7 model file ID.

`Creep`

True to enable the **Creep** option.

St7GetSolverCreep

Returns the state assigned for the **Creep** option for creep analyses.

```
long St7GetSolverCreep(long uID, bool* Creep)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Creep`

True if the **Creep** option is enabled.

St7SetSolverIncludeKG

Sets the state of the KG option for geometric nonlinear analysis.

```
long St7SetSolverIncludeKG(long uID, bool IncludeKG)
```

Input Parameters

`uID`

Strand7 model file ID.

`IncludeKG`

True to include the geometric stiffness matrix, KG.

St7GetSolverIncludeKG

Returns the state of the KG option for geometric nonlinear analysis.

```
long St7GetSolverIncludeKG(long uID, bool* IncludeKG)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`IncludeKG`

True if the geometric stiffness matrix, KG, is included.

St7SetSolverStressStiffening

Sets the state of the stress stiffening option for natural frequency and linear transient dynamic analyses with initial conditions.

```
long St7SetSolverStressStiffening(long uID, bool AddStressStiffening)
```

Input Parameters

`uID`

Strand7 model file ID.

`AddStressStiffening`

True to include the stress stiffening effects.

St7GetSolverStressStiffening

Returns the state of the stress stiffening option for natural frequency and linear transient dynamic analyses with initial conditions.

```
long St7GetSolverStressStiffening(long uID, bool* AddStressStiffening)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`AddStressStiffening`

True if the stress stiffening effects are included.

St7SetEntityResult

Sets the enabled state for the specified entity result. Only enabled entity results are written to the result file.

```
long St7SetEntityResult(long uID, long Result, bool Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`Result`

See *Solver Options* for additional information.

`Enabled`

Either True or False.

St7GetEntityResult

Returns the enabled state for the specified entity result. Only enabled entity results are written to the result file.

```
long St7GetEntityResult(long uID, long Result, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`Result`

See *Solver Options* for additional information.

Output Parameters

`Enabled`

Either True or False.

St7EnableResultGroup

Enables the specified group results for the analysis. Only element results corresponding to enabled groups are written to the result file.

```
long St7EnableResultGroup(long uID, long GroupID)
```

Input Parameters

`uID`

Strand7 model file ID.

`GroupID`

Group ID number to enable.

St7DisableResultGroup

Disables the specified group results for the analysis. Only element results corresponding to enabled groups are written to the result file.

```
long St7DisableResultGroup(long uID, long GroupID)
```

Input Parameters

`uID`

Strand7 model file ID.

`GroupID`

Group ID number to disable.

St7GetResultGroupState

Returns the enabled state of the group results for the analysis. Only element results corresponding to enabled groups are written to the result file.

```
long St7GetResultGroupState(long uID, long GroupID, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`GroupID`

Group ID.

Output Parameters

`Enabled`

True if the specified group is enabled.

St7EnableResultProperty

Enables the specified property results for the analysis. Only element results corresponding to enabled properties are written to the result file.

```
long St7EnableResultProperty(long uID, long Entity, long PropNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP or ptBRICKPROP.

`PropNum`

Property number.

St7DisableResultProperty

Disables the specified property results for the analysis. Only element results corresponding to enabled properties are written to the result file.

```
long St7DisableResultProperty(long uID, long Entity, long PropNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP or ptBRICKPROP.

`PropNum`

Property number.

St7GetResultPropertyState

Returns the enabled state of the specified property results for the analysis. Only element results corresponding to enabled properties are written to the result file.

```
long St7GetResultPropertyState(long uID, long Entity, long PropNum,
     bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP or ptBRICKPROP.

`PropNum`

Property number.

Output Parameters

`Enabled`

True if the specified property results are enabled.

St7EnableInitialPCGFile

Sets the PCG solver to use the specified result file as initial conditions. The result file name is specified using *St7SetInitialPCGFile*.

```
long St7EnableInitialPCGFile(long uID, long SolverType)
```

Input Parameters

`uID`

Strand7 model file ID.

`SolverType`

Either stLinearStatic or stSteadyHeat.

St7DisableInitialPCGFile

Sets the PCG solver to use the default initial conditions.

```
long St7DisableInitialPCGFile(long uID, long SolverType)
```

Input Parameters

`uID`

Strand7 model file ID.

`SolverType`

Either stLinearStatic or stSteadyHeat.

St7GetInitialPCGFileState

Returns the enabled state assigned for the PCG initial conditions.

```
long St7GetInitialPCGFileState(long uID, long SolverType, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`SolverType`

Either stLinearStatic or stSteadyHeat.

Output Parameters

`Enabled`

True if the starting vector for the PCG solver is obtained from the initial conditions file.

St7SetInitialPCGFile

Assigns the initial conditions file used by the PCG solver.

```
long St7SetInitialPCGFile(long uID, long SolverType, char* FileName)
```

Input Parameters

`uID`

Strand7 model file ID.

`SolverType`

Either stLinearStatic or stSteadyHeat.

`FileName`

Full path and name for the initial conditions file for the PCG solver.

St7GetInitialPCGFile

Returns the initial conditions file assigned to the PCG solver.

```
long St7GetInitialPCGFile(long uID, long SolverType, char* FileName,
     long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`SolverType`

Either stLinearStatic or stSteadyHeat.

`MaxStringLen`

Maximum number of characters allocated for FileName.

Output Parameters

`FileName`

Full path and name for the initial conditions file for the PCG solver.

St7SetResultFileName

Sets the name of the results file for the analysis.

```
long St7SetResultFileName(long uID, char* FileName)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the results file.

St7SetResultLogFileName

Sets the name of the solver log-file for the analysis.

```
long St7SetResultLogFileName(long uID, char* LogName)
```

Input Parameters

`uID`

Strand7 model file ID.

`LogName`

Full path and name of the log-file.

St7SetStaticRestartFile

Sets the name of the static restart file.

```
long St7SetStaticRestartFile(long uID, char* FileName)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the static restart file.

St7GetStaticRestartFile

Returns the name of the static restart file.

```
long St7GetStaticRestartFile(long uID, char* FileName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxStringLen`

Maximum number of characters allocated for FileName.

Output Parameters

`FileName`

Full path and name for the static restart file.

St7SetDynamicRestartFile

Sets the name of the dynamic restart file.

```
long St7SetDynamicRestartFile(long uID, char* FileName)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the dynamic restart file.

St7GetDynamicRestartFile

Returns the name of the dynamic restart file.

```
long St7GetDynamicRestartFile(long uID, char* FileName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxStringLen`

Maximum number of characters allocated for FileName.

Output Parameters

`FileName`

Full path and name for the dynamic restart file.

St7SetQuasiStaticRestartFile

Sets the name of the quasi-static restart file.

```
long St7SetQuasiStaticRestartFile(long uID, char* FileName)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the quasi-static restart file.

St7GetQuasiStaticRestartFile

Returns the name of the quasi-static restart file.

```
long St7GetQuasiStaticRestartFile(long uID, char* FileName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxStringLen`

Maximum number of characters allocated for FileName.

Output Parameters

`FileName`

Full path and name for the quasi-static restart file.

St7SetNodeHistoryFile

Sets the name of the node history file for the analysis.

```
long St7SetNodeHistoryFile(long uID, char* FileName)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the node history file.

St7GetNodeHistoryFile

Returns the name of the node history file for the analysis.

```
long St7GetNodeHistoryFile(long uID, char* FileName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxStringLen`

Maximum number of characters allocated for FileName.

Output Parameters

`FileName`

Full path and name for the node history file.

St7EnableSaveRestart

Enables the **Save restart files** option for the analysis.

```
long St7EnableSaveRestart(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7DisableSaveRestart

Disables the **Save restart files** option for the analysis.

```
long St7DisableSaveRestart(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7EnableSaveLastRestartStep

Enables the **Save only last step** option for the analysis.

```
long St7EnableSaveLastRestartStep(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7DisableSaveLastRestartStep

Disables the **Save only last step** option for the analysis.

```
long St7DisableSaveLastRestartStep(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7SetSolverDefaultsLogical

Sets a series of Boolean parameters for the analysis.

```
long St7SetSolverDefaultsLogical(long uID, long Parameter, bool Value)
```

Input Parameters

`uID`

Strand7 model file ID.

`Parameter`

Solver logical parameter; see *Logical Parameters.*

`Value`

Either True or False.

St7GetSolverDefaultsLogical

Returns the state assigned for a set of Boolean parameters for the analysis.

```
long St7GetSolverDefaultsLogical(long uID, long Parameter, bool* Value)
```

Input Parameters

`uID`

Strand7 model file ID.

`Parameter`

Solver logical parameter; see *Logical Parameters*.

Output Parameters

`Value`

Either True or False.

St7SetSolverDefaultsInteger

Sets the integer solver default values.

```
long St7SetSolverDefaultsInteger(long uID, long Parameter, long Value)
```

Input Parameters

`uID`

Strand7 model file ID.

`Parameter`

Solver integer parameter; see *Integer Parameters*.

`Value`

Integer value.

St7GetSolverDefaultsInteger

Returns the value assigned to the integer solver defaults.

```
long St7GetSolverDefaultsInteger(long uID, long Parameter, long* Value)
```

Input Parameters

`uID`

Strand7 model file ID.

`Parameter`

Solver integer parameter; see *Integer Parameters*.

Output Parameters

`Value`

Integer value.

St7SetSolverDefaultsDouble

Sets the double solver default values.

```
long St7SetSolverDefaultsDouble(long uID, long Parameter, double Value)
```

Input Parameters

`uID`

Strand7 model file ID.

`Parameter`

Solver double parameter; see *Double Parameters*.

`Value`

Double value.

St7GetSolverDefaultsDouble

Returns the value assigned to the double solver defaults.

```
long St7GetSolverDefaultsDouble(long uID, long Parameter, double* Value)
```

Input Parameters

`uID`

Strand7 model file ID.

`Parameter`

Solver double parameter; see *Double Parameters*.

Output Parameters

`Value`

Double value.

Solve

St7RunSolver

Launches the specified Strand7 solver. All Strand7 solvers run as a separate process to the calling application.

```
long St7RunSolver(long uID, long Solver, long Mode, long Wait)
```

Input Parameters

`uID`

Strand7 model file ID.

`Solver`

One of the solver types listed in *Solver Types*.

`Mode`

One of smNormalRun, smNormalCloseRun, smProgressRun or smBackgroundRun. See *Solver Options* for additional information.

`Wait`

Solver execution mode; either btTrue to halt execution of the caller until the solve is complete, or btFalse to pass control back to the caller immediately after the function is called.

St7RunSolverProcess

Launches the specified Strand7 solver and returns the ID number for the new process created. All Strand7 solvers run as a separate process to the calling application.

```
long St7RunSolverProcess(long uID, long Solver, long Mode, long Wait,
     long* ProcessID)
```

Input Parameters

`uID`

Strand7 model file ID.

`Solver`

One of the solver types listed in *Solver Types*.

`Mode`

One of smNormalRun, smNormalCloseRun, smProgressRun or smBackgroundRun. See *Solver Options* for additional information.

`Wait`

Solver execution mode; either btTrue to halt execution of the caller until the solve is complete, or btFalse to pass control back to the caller immediately after the function is called.

Output Parameters

`ProcessID`

ID number for the solver process.

St7CheckSolverRunning

Returns the execution state for the specified solver process.

```
long St7CheckSolverRunning(long ProcessID, bool* IsRunning)
```

Input Parameters

`ProcessID`

Solver process ID.

Output Parameters

`IsRunning`

True if the solver process is currently executing.

St7StopSolverProcess

Stops the execution of the specified solver process.

```
long St7StopSolverProcess(long ProcessID)
```

Input Parameters

`ProcessID`

Solver process ID.

Usage

The function terminates the solver using a brute force approach (equivalent to terminating the process using Task Manager). It is applicable only to the .exe solver when this is executed via *St7RunSolverProcess* with Wait set to btFalse. A solver process terminated using this function may leave behind its temporary files in the ScratchPath folder (see *St7OpenFile*); these will need to be deleted manually.

St7SetSolverWindowPos

Sets the desktop position and size of new solver windows created by *St7RunSolver* and *St7RunSolverProcess*.

```
long St7SetSolverWindowPos(long L, long T, long W, long H)
```

Input Parameters

`L`

Left position of the solver window, measured in pixels, relative to the origin of the desktop.

`T`

Top position of the solver window, measured in pixels, relative to the origin of the desktop.

`W`

Width of the solver window in pixels; ignored if it is less than or equal to zero.

`H`

Height of the solver window in pixels; ignored if it is less than or equal to zero.

Usage

The function is applicable to the following solver run modes: smNormalRun, smNormalCloseRun and smProgressRun. For smProgressRun, the H parameter is ignored; that is, the height cannot be set. To set just the left and/or top positions of the solver window without changing the width or height, set W and/or H to 0. Note that the origin of the desktop might not be (0,0), particularly on systems with more than one screen.

St7ClearSolverWindowPos

Clears the solver window position set by *St7SetSolverWindowPos*. For subsequent executions of the solver, the solver window will appear at the default position and size.

```
long St7ClearSolverWindowPos()
```

St7SetUseSolverDLL

Sets the type of solver to be launched for future solver invocations; either the .exe solver or the .dll solver.

```
long St7SetUseSolverDLL(bool UseDLL)
```

Input Parameters

`UseDLL`

True to use the .dll solver.

False to use the .exe solver.

Usage

The default option of the .exe solver is safer since the solver is isolated from the user's program. In this configuration, launching the solver spawns a new process that runs independently of the process running the user's API program. When run using the .dll solver, the solve is run in the memory space of the user's application.

The advantages of the .dll solver are that the solver can be invoked with less overhead (i.e. it loads faster, which could significantly reduce the total run time for applications requiring a very large number of solver executions), and that more information about a solve's termination can be accessed by the Strand7 API (see

Solver termination error codes).

The disadvantage of the .dll solver is that any abnormal termination of the solver will propogate to the user's application rather than being contained within the solver; this may terminate the user's application.

St7GetUseSolverDLL

Returns the type of solver to be launched for future solver invocations; either the .exe solver or the .dll solver.

```
long St7GetUseSolverDLL(bool* UseDLL)
```

Output Parameters

`UseDLL`

True to use the .dll solver; False to use the .exe solver.

St7SetSolverNumCPU

Sets the number of threads to use for solvers that support parallelisation.

```
long St7SetSolverNumCPU(long uID, long NumCPU)
```

Input Parameters

`uID`

Strand7 model file ID.

`NumCPU`

Number of threads to use.

St7GetSolverNumCPU

Returns the number of threads to use for solvers that support parallelisation.

```
long St7GetSolverNumCPU(long uID, long* NumCPU)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumCPU`

Number of threads to use.

St7SetSolverFontName

Sets the font used in the solver window and log file viewer.

```
long St7SetSolverFontName(long uID, char* FontName)
```

Input Parameters

`uID`

Strand7 model file ID.

`FontName`

String containing the name of the font. Only monospaced fonts may be used here.

St7GetSolverFontName

Returns the font used in the solver window and log file viewer.

```
long St7GetSolverFontName(long uID, char* FontName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxStringLen`

Maximum number of characters allocated for FontName.

Output Parameters

`FontName`

String containing the name of the font.

Results

St7GetNumIterations

Returns the total number of iterations performed by the solver for the result file currently open.

```
long St7GetNumIterations(long uID, long* NumIterations)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumIterations`

Number of iterations.

Applicability

Applicable to stLinearBuckling, stNaturalFrequency, stNonlinearStatic, stQuasiStatic, stNonlinearTransientDynamic, stSteadyHeat and stTransientHeat.

Usage

The number returned is the number of solver iterations performed up to the last saved result case in the result file. If additional iterations are performed after the last saved result case, those iterations are not counted.

If the solver was launched as a restart, the number of iterations performed in the restart is returned, not the cumulative number of iterations.

St7GetResultCaseName

Returns the name of the specified result case in the result file currently open.

```
long St7GetResultCaseName(long uID, long CaseNum, char* CaseName,
     long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`MaxStringLen`

Maximum number of characters allocated for CaseName.

Output Parameters

`CaseName`

Result case name.

Applicability

Applicable to all solver types.

St7GetResultFreedomCaseName

Returns the name of the freedom case in the result file currently open.

```
long St7GetResultFreedomCaseName(long uID, char* CaseName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxStringLen`

Maximum number of characters allocated for CaseName.

Output Parameters

`CaseName`

Result freedom case name.

Applicability

Applicable to all solver types.

St7GetResultCaseStage

Returns the stage associated with the specified result case in the result file currently open.

```
long St7GetResultCaseStage(long uID, long CaseNum, long* Stage)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

Output Parameters

`Stage`

Stage index.

Applicability

Applicable to all solver types.

St7GetResultCaseConvergence

Returns the convergence of the specified result case in the result file currently open.

```
long St7GetResultCaseConvergence(long uID, long CaseNum, bool* Converged)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

Output Parameters

`Converged`

True if the specified result case is converged.

Applicability

Applicable to stLinearStatic, stLinearBuckling, stLoadInfluence, stNonlinearStatic, stQuasiStatic, stNaturalFrequency, stNonlinearTransientDynamic, stSteadyHeat and stTransientHeat.

St7GetModalConvergence

Returns the Eigenvalue and Eigenvector convergence of the specified mode in the modal result file currently open.

```
long St7GetModalConvergence(long uID, long Mode, bool* EigvalConverged,
     bool* EigvectConverged)
```

Input Parameters

`uID`

Strand7 model file ID.

`Mode`

Mode number.

Output Parameters

`EigvalConverged`

True if the Eigenvalue for the specified mode is converged.

`EigvectConverged`

True if the Eigenvector for the specified mode is converged.

Applicability

Applicable to stLinearBuckling and stNaturalFrequency.

St7GetResultCaseTime

Returns the integration time for the specified result case in the result file currently open.

```
long St7GetResultCaseTime(long uID, long CaseNum, double* Time)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

Output Parameters

`Time`

Integration time.

Applicability

Applicable to stQuasiStatic, stHarmonicResponse, stLinearTransientDynamic, stNonlinearTransientDynamic and stTransientHeat.

St7GetResultCaseFactor

Returns a context-dependent factor for the specified result case in the result file currently open.

```
long St7GetResultCaseFactor(long uID, long CaseNum, double* Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

Output Parameters

`Factor`

The value of this output depends on the analysis that produced the results file.

Linear Static Analysis – undefined.

Linear Buckling Analysis – buckling factor for mode CaseNum.

Load Influence Analysis – undefined.

Nonlinear Static Analysis – the fractional substep when CaseNum is a substep, otherwise 0 for complete steps.

Linear Transient Dynamic Analysis – integration time at CaseNum.

Quasi-static Analysis – integration time at CaseNum.

Nonlinear Transient Dynamic Analysis – integration time at CaseNum.

Natural Frequency Analysis – undefined.

Harmonic Response Analysis – frequency (Hz) of applied load for CaseNum, when CaseNum is a harmonic result case. When a harmonic time history is generated, it is the time at CaseNum.

Spectral Response Analysis – returns the natural frequency (Hz) of the mode from which the spectral response arises when CaseNum is not combined:

-1.0 when CaseNum is an SRSS combination,
-2.0 when CaseNum is a CQC combination,
-3.0 when CaseNum has been generated as any other combination.

Steady Heat Analysis – undefined.

Transient Heat Analysis – integration time at CaseNum.

St7GetResultCaseReset

Checks to see if the model has been reset for the specified result case in the result file currently open.

```
long St7GetResultCaseReset(long uID, long CaseNum, bool* Reset)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

Output Parameters

`Reset`

btTrue if **RESET MODEL** has been enabled for the result case.

Applicability

Applicable to stNonlinearStatic.

St7GetResultCaseKineticEnergy

Returns the kinetic energy for the specified result case in the result file currently open.

```
long St7GetResultCaseKineticEnergy(long uID, long CaseNum, double* Energy)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

Output Parameters

`Energy`

Kinetic energy.

Applicability

Applicable to stLinearTransientDynamic and stNonlinearTransientDynamic.

St7GetResultCaseInfluenceVariable

Returns the details of the response variable used to generate the specified result case in the load influence result file currently open.

```
long St7GetResultCaseInfluenceVariable(long uID, long CaseNum, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

Output Parameters

`Integers[0..6]`

`[ipLIAVarLoadCaseNum]` – Load case in which the response variable is defined.

`[ipLIAVarFreedomCaseNum]` – Freedom case used for the influence analysis.

`[ipLIAVarEntity]` – Entity type; one of tyNODE, tyBEAM, tyPLATE or tyBRICK.

`[ipLIAVarEntityNum]` – Entity number.

`[ipLIAVarUCSId]` – ID number of the UCS, if relevant.

`[ipLIAVarType]` – For tyNODE, either rvNodeDisplacement or rvNodeReaction. For tyBEAM, the beam end. For tyPLATE, either rvPlateForce or rvPlateMoment. Not relevant for tyBRICK.

`[ipLIAVarComponent]` – The degree of freedom or axis direction of the response variable.

Applicability

Applicable to stLoadInfluence.

St7GetFrequency

Returns the frequency for the specified result case in the result file currently open.

```
long St7GetFrequency(long uID, long Mode, double* Freq)
```

Input Parameters

`uID`

Strand7 model file ID.

`Mode`

Result case/mode number.

Output Parameters

`Freq`

Mode frequency (Hz).

Applicability

Applicable to stNaturalFrequency and stHarmonicResponse.

St7GetInertiaReliefResults

Returns the inertial results for the specified result case in inertial relief for the specified result case in the result file currently open.

```
long St7GetInertiaReliefResults(long uID, long CaseNum, double* InertiaResult)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

Output Parameters

`InertiaResult[0..11]`

`[ipMassXIRA]` – Mass component in the global X direction.

`[ipMassYIRA]` – Mass component in the global Y direction.

`[ipMassZIRA]` – Mass component in the global Z direction.

`[ipXcIRA]` – Global X ordinate of the centre of mass.

`[ipYcIRA]` – Global Y ordinate of the centre of mass.

`[ipZcIRA]` – Global Z ordinate of the centre of mass.

`[ipAccXIRA]` – Translational acceleration of the centre of mass in the global X direction.

`[ipAccYIRA]` – Translational acceleration of the centre of mass in the global Y direction.

`[ipAccZIRA]` – Translational acceleration of the centre of mass in the global Z direction.

`[ipAngAccXIRA]` – Rotational acceleration about the global X axis (degrees/time^2).

`[ipAngAccYIRA]` – Rotational acceleration about the global Y axis (degrees/time^2).

`[ipAngAccZIRA]` – Rotational acceleration about the global Z axis (degrees/time^2).

Applicability

Applicable to stLinearStatic.

St7GetNumModes

Returns the number of modes included in the modal superposition in the result file currently open.

```
long St7GetNumModes(long uID, long* NumModes)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumModes`

Number of modes.

Applicability

Applicable to stNaturalFrequency and stHarmonicResponse.

St7GetNumSRACases

Returns the number of spectral **Load Excitation** and **Base Excitation** cases in the result file currently open.

```
long St7GetNumSRACases(long uID, long* NumCases)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumCases`

Number of cases.

Applicability

Applicable to stSpectralResponse.

St7GetModalResultsNFA

Returns the modal results for the specified mode in the result file currently open.

`long St7GetModalResultsNFA(long uID, long Mode, double* ModalResult)`

Input Parameters

`uID`

Strand7 model file ID.

`Mode`

Result case/mode number.

Output Parameters

`ModalResult[0..15]`

`[ipFrequencyNFA]` – Mode frequency (Hz).

`[ipModalMassNFA]` – Modal mass.

`[ipModalStiffNFA]` – Modal stiffness.

`[ipModalDampNFA]` – Modal damping.

`[ipModalTMassP1]` – Translational Mass Participation. If a translational direction vector is defined for *St7GetNFAModeParticipationVectors*, the total participation, otherwise the participation in global X.

`[ipModalTMassP2]` – Translational Mass Participation. If a translational direction vector is defined for *St7GetNFAModeParticipationVectors*, zero, otherwise the participation in global Y.

`[ipModalTMassP3]` – Translational Mass Participation. If a translational direction vector is defined for *St7GetNFAModeParticipationVectors*, zero, otherwise the participation in global Z.

`[ipModalRMassP1]` – Rotational Mass Participation. If a rotational direction vector is defined for *St7GetNFAModeParticipationVectors*, the total participation, otherwise the participation about global X.

`[ipModalRMassP2]` – Rotational Mass Participation. If a rotational direction vector is defined for *St7GetNFAModeParticipationVectors*, zero, otherwise the participation about global Y.

`[ipModalRMassP3]` – Rotational Mass Participation. If a rotational direction vector is defined for *St7GetNFAModeParticipationVectors*, zero, otherwise, the participation about global Z.

`[ipModalModeD1]` – Translational Mode Participation. If a translational direction vector is defined for *St7GetNFAModeParticipationVectors*, the total participation, otherwise the participation in global X.

`[ipModalModeD2]` – Translational Mode Participation. If a translational direction vector is defined for *St7GetNFAModeParticipationVectors*, zero, otherwise the participation in global Y.

`[ipModalModeD3]` – Translational Mode Participation. If a translational direction vector is defined for *St7GetNFAModeParticipationVectors*, zero, otherwise the participation in global Z.

`[ipModalModeR1]` – Rotational Mode Participation. If a rotational direction vector is defined for *St7GetNFAModeParticipationVectors*, the total participation, otherwise the participation about global X.

`[ipModalModeR2]` – Rotational Mode Participation. If a rotational direction vector is defined for *St7GetNFAModeParticipationVectors*, zero, otherwise the participation about global Y.

`[ipModalModeR3]` – Rotational Mode Participation. If a rotational direction vector is defined for *St7GetNFAModeParticipationVectors*, the total participation, otherwise the participation about global Z.

Applicability

Applicable to stNaturalFrequency.

St7GetModalResultsNodeDofNFA

Returns the node and degree of freedom used to normalise the Eigenvector to produce the engineering modal mass and stiffness for the specified mode in the result file currently open.

```
long St7GetModalResultsNodeDofNFA(long uID, long Mode, long* NodeNum, long* Dof)
```

Input Parameters

`uID`

Strand7 model file ID.

`Mode`

Result case/mode number.

Output Parameters

`NodeNum`

Node number used for normalisation.

`Dof`

Degree of freedom used for normalisation (an integer from 1 to 6).

Applicability

Applicable to stNaturalFrequency.

St7GetModalResultsHRA

Returns the modal results for the specified result case in the result file currently open.

```
long St7GetModalResultsHRA(long uID, long CaseNum, long Mode,
    double* ModalResult)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Harmonic load case number. For analyses in which multiple load cases do not apply, use 1.

`Mode`

Mode number.

Output Parameters

`ModalResult[0..4]`

`[ipFrequencyHRA]` – Mode frequency (Hz).

`[ipDampRatioHRA]` – Damping Ratio.

`[ipAmplitudeHRA]` – Amplitude.

`[ipPhaseAngleHRA]` – Phase angle (degrees).

`[ipMassPartHRA]` – Mass participation (percentage).

Applicability

Applicable to stHarmonicResponse.

St7GetExcitationTypeSRA

Returns the excitation type for the specified result case in the result file currently open.

```
long St7GetExcitationTypeSRA(long uID, long CaseNum, long* ExcitationType)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Spectral load case number.

Output Parameters

`ExcitationType`

One of slBaseAcc, slBaseVel, slBaseDisp or slAppliedLoad.

Applicability

Applicable to stSpectralResponse.

St7GetModalResultsSRA

Returns the modal results for the specified result case in the result file currently open.

```
long St7GetModalResultsSRA(long uID, long CaseNum, long Mode,
     double* ModalResult)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Spectral load case number.

`Mode`

Mode number.

Output Parameters

`ModalResult[0..5]`

`[ipFrequencySRA]` – Mode frequency (Hz).

`[ipSpectralValueSRA]` – Spectral table value.

`[ipDampRatioSRA]` – Damping ratio.

`[ipExcitationSRA]` – Excitation.

`[ipAmplitudeSRA]` – Amplitude.

`[ipMassPartSRA]` – Mass participation (percentage).

Applicability

Applicable to stSpectralResponse.

St7GetBucklingFactor

Returns the buckling factor for the specified mode in the result file currently open.

```
long St7GetBucklingFactor(long uID, long Mode, double* Fact)
```

Input Parameters

`uID`

Strand7 model file ID.

`Mode`

Result case/mode number.

Output Parameters

`Fact`

Buckling factor.

Applicability

Applicable to stLinearBuckling.

St7GetInitialTemperatureTHA

Returns the temperature used as the initial temperature for the result file currently open.

```
long St7GetInitialTemperatureTHA(long uID, double* InitialTemp)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`InitialTemp`

Initial temperature.

Applicability

Applicable to stTransientHeat.

St7SetReferenceDisplacement

Sets the reference case for extracted displacement results.

```
long St7SetReferenceDisplacement(long uID, long RefCase, bool ApplyToDisplay)
```

Input Parameters

`uID`

Strand7 model file ID.

`RefCase`

One of:

rdNone – absolute displacements;
rdPreviousCase – displacements relative to the previous case; or
A result case number – displacements relative to that specific case.

`ApplyToDisplay`

True to apply the reference displacement to both model window and extracted results.

False to apply the reference displacement only to extracted results.

St7GetElementResultState

Returns element state information for the specified result case.

```
long St7GetElementResultState(long uID, long Entity, long EntityNum,
     long ResultCase, long* State)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK or tyLINK.

`EntityNum`

Entity number.

`ResultCase`

Result case number.

Output Parameters

`State[0..2]`

A 3-element array containing status information.

`[ipResStateActive]` – btTrue if element is active, btFalse otherwise.

`[ipResStateResults]` – btTrue if results are available, btFalse otherwise.

`[ipResStateBirthStage]` – Entity birth stage at this result case.

St7GetNodeResult

Returns the specified nodal result quantity in the global XYZ system.

```
long St7GetNodeResult(long uID, long ResultType, long NodeNum, long ResultCase,
    double* NodeResult)
```

Input Parameters

`uID`

Strand7 model file ID.

`ResultType`

Nodal result quantity; see *Node Results*.

`NodeNum`

Node number.

`ResultCase`

Result case number.

Output Parameters

`NodeResult[0..5]`

A 6-element array containing the nodal results. See *Node Results* for additional information.

St7GetNodeResultEx

Returns the specified nodal result quantity in the global XYZ system. In addition to components, a number of combined results may also be returned, depending on `ResultType`.

```
long St7GetNodeResultEx(long uID, long ResultType, long NodeNum, long ResultCase,
    double* NodeResult)
```

Input Parameters

`uID`

Strand7 model file ID.

`ResultType`

Nodal result quantity; see *Node Results*.

`NodeNum`

Node number.

`ResultCase`

Result case number.

Output Parameters

`NodeResult[0..13]`

A 14-element array containing the nodal results. See *Node Results* for additional information.

St7GetNodeResultUCS

Returns the specified nodal result in a UCS.

```
long St7GetNodeResultUCS(long uID, long ResultType, long UCSId, long NodeNum,
     long ResultCase, double* NodeResult)
```

Input Parameters

`uID`

Strand7 model file ID.

`ResultType`

Nodal result quantity; see *Node Results*.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system. For ResultType=rtNodeReact, UCSId = 0 refers to the coordinate system of the restraint attribute at the node.

`NodeNum`

Node number.

`ResultCase`

Result case number.

Output Parameters

`NodeResult[0..5]`

A 6-element array containing the nodal results. See *Node Results* for additional information.

St7GetNodeResultExUCS

Returns the specified nodal result in a UCS. In addition to components, a number of combined results may also be returned, depending on `ResultType`.

```
long St7GetNodeResultExUCS(long uID, long ResultType, long UCSId, long NodeNum,
     long ResultCase, double* NodeResult)
```

Input Parameters

`uID`

Strand7 model file ID.

`ResultType`

Nodal result quantity; see *Node Results*.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system. For ResultType=rtNodeReact, UCSId = 0 refers to the coordinate system of the restraint attribute at the node.

`NodeNum`

Node number.

`ResultCase`

Result case number.

Output Parameters

`NodeResult[0..13]`

A 14-element array containing the nodal results. See *Node Results* for additional information.

St7SetBeamResultPosMode

Sets the mode of the beam position variable used by the beam result extraction functions.

```
long St7SetBeamResultPosMode(long uID, long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`Mode`

One of bpLength or bpParam, representing beam position in physical length units of the model or as a ratio of element length, respectively.

St7GetBeamResultPosMode

Returns the mode of the beam position variable used by the beam result extraction functions.

```
long St7GetBeamResultPosMode(long uID, long* Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Mode`

One of bpLength or bpParam, representing beam position in physical length units of the model or as a ratio of element length, respectively.

St7GetBeamResultArray

Returns the specified beam result quantity at several stations along the length of the beam. Additional stations may be inserted to ensure that the maximum/minimum results are captured.

```
long St7GetBeamResultArray(long uID, long ResultType, long ResultSubType,
     long BeamNum, long MinStations, long ResultCase, long* NumStations,
     long* NumColumns, double* BeamPos, double* BeamResult)
```

Input Parameters

`uID`

Strand7 model file ID.

`ResultType`

Beam result quantity; see *Beam Results*.

`ResultSubType`

Beam result sub-type; see *Beam Results*.

`BeamNum`

Beam number.

`MinStations`

Minimum number of stations required.

`ResultCase`

Result case number.

Output Parameters

`NumStations`

Number of stations used.

`NumColumns`

Number of result quantities returned at each station.

`BeamPos[0..kMaxBeamResult-1]`

`[0..NumStations-1]` – an array of positions of the beam stations measured along the element from end 1. Positions vary between zero and the length of the element, or between zero and one, depending on the mode set via *St7SetBeamResultPosMode*.

`BeamResult[0..kMaxBeamResult-1]`

`[0..NumStations*NumColumns-1]` – an array containing the beam results at each station.

The results are returned in blocks of length NumColumns with the start of the i^{th} block for the i^{th} station at `BeamResult[(i-1)*NumColumns]`.

See *Beam Results* for additional information.

St7GetBeamResultArrayPos

Returns the specified beam results at a series of positions along the length of the beam.

```
long St7GetBeamResultArrayPos(long uID, long ResultType, long ResultSubType,
    long BeamNum, long ResultCase, long NumStations, double* BeamPos,
    long* NumColumns, double* BeamResult)
```

Input Parameters

`uID`

Strand7 model file ID.

`ResultType`

Beam result quantity; see *Beam Results*.

`ResultSubType`

Beam result sub-type; see *Beam Results*.

`BeamNum`

Beam number.

`ResultCase`

Result case number.

`NumStations`

Number of stations specified.

`BeamPos[0..kMaxBeamResult-1]`

`[0..NumStations-1]` – an array of positions along the beam measured from end 1. Positions vary between zero and the length of the element, or between zero and one, depending on the mode set via *St7SetBeamResultPosMode*.

Output Parameters

`NumColumns`

Number of result quantities returned at each station.

`BeamResult[0..kMaxBeamResult-1]`

`[0..NumStations*NumColumns-1]` – an array containing the beam results at each station.

The results are returned in blocks of length NumColumns with the start of the i^{th} block for the i^{th} station at `BeamResult[(i-1)*NumColumns]`.

See *Beam Results* for additional information.

St7GetBeamResultEndPos

Returns the specified beam result at the beam endpoints.

```
long St7GetBeamResultEndPos(long uID, long ResultType, long ResultSubType,
     long BeamNum, long ResultCase, long* NumColumns, double* BeamResult)
```

Input Parameters

`uID`

Strand7 model file ID.

`ResultType`

Beam result quantity; see *Beam Results*.

`ResultSubType`

Beam result sub-type; see *Beam Results*.

`BeamNum`

Beam number.

`ResultCase`

Result case number.

Output Parameters

`NumColumns`

Number of result quantities returned at each endpoint.

`BeamResult[0..kMaxBeamResult-1]`

`[0..NumColumns-1]` – a block of results for End1.

`[NumColumns..2*NumColumns-1]` – a block of results for End2.

See *Beam Results* for additional information.

St7GetBeamResultSinglePos

Returns the specified beam result at a single position along the length of the beam.

```
long St7GetBeamResultSinglePos(long uID, long ResultType, long ResultSubType,
     long BeamNum, long ResultCase, double BeamPos, long* NumColumns,
     double* BeamResult)
```

Input Parameters

`uID`

Strand7 model file ID.

`ResultType`

Beam result quantity; see *Beam Results*.

`ResultSubType`

Beam result sub-type; see *Beam Results*.

`BeamNum`

Beam number.

`ResultCase`

Result case number.

`BeamPos`

The position along the beam measured from end 1. Positions vary between zero and the length of the element, or between zero and one, depending on the mode set via *St7SetBeamResultPosMode*.

Output Parameters

`NumColumns`

Number of result quantities returned at the specified Position.

`BeamResult[0..kMaxBeamResult-1]`

`[0..NumColumns-1]` – a block of results at the specified Position.

See *Beam Results* for additional information.

St7GetBeamReleaseResult

Returns the release results for the specified beam.

```
long St7GetBeamReleaseResult(long uID, long BeamNum, long ResultCase,
    bool* BeamReleased, double* ReleaseValue)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamNum`

Beam number.

`ResultCase`

Result case number.

Output Parameters

`BeamReleased[0..kMaxBeamRelease-1]`

An array containing the release status for the beam ends according to the beam local axis system for translational and rotational DoF.

Values set to True indicate a release for the corresponding DoF.

`[ipRelEnd1Dir1]` – End 1 translational release in the 1 axis direction.

`[ipRelEnd1Dir2]` – End 1 translational release in the 2 axis direction.

`[ipRelEnd1Dir3]` – End 1 translational release in the 3 axis direction.

`[ipRelEnd1Dir4]` – End 1 rotational release about the 1 axis direction.

`[ipRelEnd1Dir5]` – End 1 rotational release about the 2 axis direction.

`[ipRelEnd1Dir6]` – End 1 rotational release about the 3 axis direction.

`[ipRelEnd2Dir1]` – End 2 translational release in the 1 axis direction.

`[ipRelEnd2Dir2]` – End 2 translational release in the 2 axis direction.

`[ipRelEnd2Dir3]` – End 2 translational release in the 3 axis direction.

`[ipRelEnd2Dir4]` – End 2 rotational release about the 1 axis direction.

`[ipRelEnd2Dir5]` – End 2 rotational release about the 2 axis direction.

`[ipRelEnd2Dir6]` – End 2 rotational release about the 3 axis direction.

`ReleaseValue[0..kMaxBeamRelease-1]`

An array containing the displacement results for the released beam end DoF. The same format as the BeamReleased array is used.

St7GetBeamSectionResult

Returns the section results at the specified point on a beam.

```
long St7GetBeamSectionResult(long uID, long ResultType, long BeamNum,
     long ResultCase, double BeamPos, double x, double y, double* BeamResult)
```

Input Parameters

`uID`

Strand7 model file ID.

`ResultType`

One of rtBeamSectionStress, rtBeamSectionStrain, rtBeamSectionTotalStrain or rtBeamSectionCreepStrain.

`BeamNum`

Beam number.

`ResultCase`

Result case number.

`BeamPos`

The position along the beam measured from end 1. Positions vary between zero and the length of the element, or between zero and one, depending on the mode set via *St7SetBeamResultPosMode*.

`x`

x position on the beam cross section, relative to the centroid, in the local axis system of the beam.

`y`

y position on the beam cross section, relative to the centroid, in the local axis system of the beam.

Output Parameters

`BeamResult[0..12]`

An array containing the beam section results. For rtBeamSectionStress the following are returned:

`[ipFibreStressXY]` – Fibre stress.

`[ipShearStress1XY]` – Shear stress in the 1 direction .

`[ipShearStress2XY]` – Shear stress in the 2 direction.

`[ipMinPrincipalStressXY]` – Minimum principal stress.

`[ipMaxPrincipalStressXY]` – Maximum principal stress.

`[ipAxialStressXY]` – Axial stress.

`[ipBendingStress1XY]` – Stress due to bending in plane 1.

`[ipBendingStress2XY]` – Stress due to bending in plane 2.

`[ipVonMisesStressXY]` – von Mises stress.

`[ipTrescaStressXY]` – Tresca stress.

`[ipTorqueStressXY]` – Shear stress due to torque.

`[ipShearF1ShearStressXY]` – Shear stress due to shear force in plane 1.

`[ipShearF2ShearStressXY]` – Shear stress due to shear force in plane 2.

For rtBeamSectionStrain, rtBeamSectionTotalStrain and rtBeamSectionCreepStrain only the first entry is used:

`[0] –` Fibre strain.

St7GetPlateResultArray

Returns the specified plate results at a series of sample locations on the element.

```
long St7GetPlateResultArray(long uID, long ResultType, long ResultSubType,
     long PlateNum, long ResultCase, long SampleLocation, long Surface,
     long Layer, long* NumPoints, long* NumColumns, double* PlateResult)
```

Input Parameters

`uID`

Strand7 model file ID.

`ResultType`

Plate result quantity; see *Plate Results*.

`ResultSubType`

Plate result sub-type; see *Plate Results*.

`PlateNum`

Plate number.

`ResultCase`

Result case number.

`SampleLocation`

One of spCentroid, spGaussPoints, spNodesAverageNever, spNodesAverageAll or spNodesAverageSame.

`Surface`

One of psPlateMidPlane, psPlateMinusZ or psPlatePlusZ.

`Layer`

Relevant to reinforced concrete or composite results; see *Plate Results*.

Output Parameters

`NumPoints`

Number of sample locations used.

`NumColumns`

Number of result quantities returned at each sample location.

`PlateResult[0..kMaxPlateResult-1]`

`[0..NumPoints*NumColumns-1]` – An array containing the plate results at each sample location.

The results are returned in blocks of length NumColumns, with the start of the i^{th} block for the i^{th} location at `PlateResult[(i-1)*NumColumns]`.

See *Plate Results* for additional information.

St7SetPlateResultMaxJunctionAngle

Sets the maximum junction angle used when calculating averaged plate results.

```
long St7SetPlateResultMaxJunctionAngle(long uID, double MaxJunctionAngle,
     bool Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxJunctionAngle`

Maximum allowable angle between adjacent plate surfaces.

`Enabled`

True to perform the angle check.

False to skip the angle check.

St7GetPlateResultMaxJunctionAngle

Returns the maximum junction angle used when calculating averaged plate results.

```
long St7GetPlateResultMaxJunctionAngle(long uID, double* MaxJunctionAngle,
     bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`MaxJunctionAngle`

Maximum allowable angle between adjacent plate surfaces.

`Enabled`

True to perform the angle check; False to skip the angle check.

St7EnablePlyPropertyResults

Enables a ply when extracting selected composite results. To access the results and for additional information see *St7GetPlateResultArray*.

```
long St7EnablePlyPropertyResults(long uID, long PropNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Ply property number.

St7DisablePlyPropertyResults

Disables a ply when extracting selected composite results. To access the results and for additional information see *St7GetPlateResultArray*.

```
long St7DisablePlyPropertyResults(long uID, long PropNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Ply property number.

St7GetPlyPropertyResultsState

Determines if a ply is enabled for inclusion when extracting selected composite results. To access the results and for additional information see *St7GetPlateResultArray*.

```
long St7GetPlyPropertyResultsState(long uID, long PropNum, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Ply property number.

Output Parameters

`Enabled`

True if the ply property is to be included in the results.

St7GetPlateResultGaussPoints

Returns the position of the result Gauss points for the specified plate.

```
long St7GetPlateResultGaussPoints(long uID, long PlateNum, long ResultCase,
     long* NumGauss, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`ResultCase`

Result case number.

Output Parameters

`NumGauss`

Number of Gauss points.

`Doubles[0..26]`

`[0..3*NumGauss-1]` – An array containing the global XYZ coordinates of the result Gauss points. The positions are returned in blocks of length 3, with the position of the ith point starting at `Doubles[(i-1)*3]`.

St7GetBrickResultArray

Returns the specified brick results at one of a number of sample locations in the element.

```
long St7GetBrickResultArray(long uID, long ResultType, long ResultSubType,
     long BrickNum, long ResultCase, long SampleLocation, long* NumPoints,
     long* NumColumns, double* BrickResult)
```

Input Parameters

`uID`

Strand7 model file ID.

`ResultType`

Brick result quantity; see *Brick Results*.

`ResultSubType`

Brick result sub-type; see *Brick Results*.

`BrickNum`

Brick number.

`ResultCase`

Result case number.

`SampleLocation`

One of spCentroid, spGaussPoints, spNodesAverageNever, spNodesAverageAll or spNodesAverageSame.

Output Parameters

`NumPoints`

Number of sampling points used.

`NumColumns`

Number of result quantities returned at each sample location.

`BrickResult[0..kMaxBrickResult-1]`

`[0..NumPoints*NumColumns-1]` – An array containing the brick results at each sample location.

The results are returned in blocks of length NumColumns, with the start of the i^{th} block for the i^{th} location at `BrickResult[(i-1)*NumColumns]`.

See *Brick Results* for additional information.

St7GetBrickResultGaussPoints

Returns the position of the result Gauss points for the specified brick.

```
long St7GetBrickResultGaussPoints(long uID, long BrickNum, long ResultCase,
     long* NumGauss, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`ResultCase`

Result case number.

Output Parameters

`NumGauss`

Number of Gauss points.

`Doubles[0..80]`

`[0..3*NumGauss-1]` – An array containing the global XYZ coordinates of the result Gauss points. The positions are returned in blocks of length 3, with the position of the i^{th} point starting at `Doubles[(i-1)*3]`.

St7GetLinkResultArray

Returns the specified link result quantity at each node in the link.

```
long St7GetLinkResultArray(long uID, long ResultType, long UCSId, long LinkNum,
     long ResultCase, long* NumPoints, long* NumColumns, double* LinkResult,
     long ArrayDim)
```

Input Parameters

`uID`

Strand7 model file ID.

`ResultType`

One of rtLinkNodeDisp, rtLinkNodeBirthDisp, rtLinkNodeReact, or rtLinkNodeFlux.

`UCSId`

ID number of a UCS frame into which vector results are transformed. UCSId = 1 refers to the global XYZ system.

`LinkNum`

Link number.

`ResultCase`

Result case number.

`ArrayDim`

Size of the array LinkResult.

Output Parameters

`NumPoints`

Number of nodes in the link.

`NumColumns`

Number of result quantities returned at each node.

`LinkResult[0..ArrayDim-1]`

`[0..NumPoints*NumColumns-1]` – an array containing the link results at each node.

The results are returned in blocks of length NumColumns with the start of the i^{th} block for the i^{th} node at `LinkResult[(i-1)*NumColumns]`.

St7GetMultiPointLinkReactionSum

Returns the reaction multi-point link force/moment result for structural solvers.

```
long St7GetMultiPointLinkReactionSum(long uID, long LinkNum, long UCSId,
     long ResultCase, double* Reaction)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`UCSId`

ID number of a UCS frame for the reaction results. UCSId = 1 refers to the global XYZ system. For non-Cartesian systems, the origin of the UCS is the origin of the reaction multi-point link.

`ResultCase`

Result case number.

Output Parameters

`Reaction[0..5]`

`[0..2]` – Force components.

`[3..5]` – Moment components.

St7GetMultiPointLinkNodeReaction

Returns the force/moment results at nodes of the reaction multi-point link for structural solvers.

```
long St7GetMultiPointLinkNodeReaction(long uID, long LinkNum, long UCSId,
     long ResultCase, double* Reaction, long MaxNodes)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`UCSId`

ID number of a UCS frame for the reaction results. UCSId = 1 refers to the global XYZ system. For non-Cartesian systems, the origin of the UCS is the origin of the reaction multi-point link.

`ResultCase`

Result case number.

`MaxNodes`

The maximum number of nodes that can be stored in the Reaction array.

Output Parameters

`Reaction[0..6*MaxNodes-1]`

An array of doubles that stores the force/moment results for each node in the link, up to MaxNodes. For example,

`[0..2]` – Force components for the first node in the link.

`[3..5]` – Moment components for the first node in the link.

`[(i-1)*6..(i-1)*6+2]` – Force components for the ith node in the link.

`[(i-1)*6+3..(i-1)*6+5]` – Moment components for the ith node in the link.

St7GetMultiPointLinkFluxSum

Returns the reaction multi-point link flux result for thermal solvers.

```
long St7GetMultiPointLinkFluxSum(long uID, long LinkNum, long ResultCase,
     double* Flux)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`ResultCase`

Result case number.

Output Parameters

`Flux`

Flux result.

St7GetMultiPointLinkNodeFlux

Returns the flux results at nodes of the reaction multi-point link for thermal solvers.

```
long St7GetMultiPointLinkNodeFlux(long uID, long LinkNum, long ResultCase,
     double* Flux, long MaxNodes)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkNum`

Link number.

`ResultCase`

Result case number.

`MaxNodes`

The maximum number of nodes that can be stored in the Flux array.

Output Parameters

`Flux[0..MaxNodes-1]`

An array of doubles that stores the flux result for each node, up to MaxNodes.

St7GetNodeReactionSum

Returns the force/moment sum of node reactions for structural solvers.

```
long St7GetNodeReactionSum(long uID, long UCSId, long ResultCase, double* Origin,
     long NodeState, double* ReactionSum)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

ID number of a UCS frame for the reaction results. UCSId = 1 refers to the global XYZ system.

`ResultCase`

Result case number.

`Origin[0..2]`

Origin position in global XYZ coordinates.

`NodeState`

Nodes to be included in the sum – either ssSelected or ssUnselected.

Output Parameters

`ReactionSum[0..5]`

`[0..2]` – Force components.

`[3..5]` – Moment components.

St7GetElementNodeForceSum

Returns the force/moment contributions of element node forces at nodes for structural solvers.

```
long St7GetElementNodeForceSum(long uID, long UCSId, long ResultCase,
    double* Origin, long* EntityState, double* ReactionSum)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

ID number of a UCS frame for the reaction results. UCSId = 1 refers to the global XYZ system.

`ResultCase`

Result case number.

`Origin[0..2]`

Origin position in global XYZ coordinates.

`EntityState[0..kMaxEntity-1]`

`[tyNODE]` – Nodes for which the entity node force should be collected – either ssSelected or ssUnselected.

`[tyBEAM]` – Beams that contribute element node force – either ssSelected or ssUnselected.

`[tyPLATE]` – Plates that contribute element node force – either ssSelected or ssUnselected.

`[tyBRICK]` – Bricks that contribute element node force – either ssSelected or ssUnselected.

`[tyLINK]` – Links that contribute element node force – either ssSelected or ssUnselected.

Output Parameters

`ReactionSum[0..5]`

`[0..2]` – Force components.

`[3..5]` – Moment components.

St7GetNodeFluxSum

Returns the node flux sum of nodes for thermal solvers.

```
long St7GetNodeFluxSum(long uID, long ResultCase, long NodeState,
    double* FluxSum)
```

Input Parameters

`uID`

Strand7 model file ID.

`ResultCase`

Result case number.

`NodeState`

Nodes to be included in the sum – either ssSelected or ssUnselected.

Output Parameters

`FluxSum`

Flux.

St7GetElementNodeFluxSum

Returns the flux contributions of elements at nodes for thermal solvers.

```
long St7GetElementNodeFluxSum(long uID, long ResultCase, long* EntityState,
    double* FluxSum)
```

Input Parameters

`uID`

Strand7 model file ID.

`ResultCase`

Result case number.

`EntityState[0..kMaxEntity-1]`

`[tyNODE]` – Nodes for which the entity node flux should be collected – either ssSelected or ssUnselected.

`[tyBEAM]` – Beams that contribute flux – either ssSelected or ssUnselected.

`[tyPLATE]` – Plates that contribute flux – either ssSelected or ssUnselected.

`[tyBRICK]` – Bricks that contribute flux – either ssSelected or ssUnselected.

`[tyLINK]` – Links that contribute flux – either ssSelected or ssUnselected.

Output Parameters

`FluxSum`

Flux.

St7SetResultUserEquation

Assigns a user defined equation for beam, plate or brick results and sets this as the currently active equation. The calculated results can be accessed via the *St7GetBeamResultArray*, *St7GetPlateResultArray* and *St7GetBrickResultArray* functions, respectively.

```
long St7SetResultUserEquation(long uID, long Entity, char* Equation,
     long TrigType)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

`Equation`

User defined equation as a character array. See *User Defined Results* for additional information.

`TrigType`

Type of angle arguments; either auRadian or auDegree.

St7GetResultUserEquation

Returns the currently active user defined equation for beam, plate or brick results. The calculated results can be accessed via the functions *St7GetBeamResultArray*, *St7GetPlateResultArray* and *St7GetBrickResultArray*, respectively.

```
long St7GetResultUserEquation(long uID, long Entity, char* Equation,
     long MaxStringLen, long* TrigType)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

`MaxStringLen`

Maximum number of characters allocated for Equation.

Output Parameters

`Equation`

User defined equation as a character array. See *User Defined Results* for additional information.

`TrigType`

Type of angle arguments; either auRadian or auDegree.

St7StoreResultUserEquation

Stores a user defined equation.

```
long St7StoreResultUserEquation(long uID, long Entity, char* Name,
     char* Equation, long TrigType)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

`Name`

Name of the user defined equation.

`Equation`

User defined equation as a character array. See *User Defined Results* for additional information.

`TrigType`

Type of angle arguments; either auRadian or auDegree.

St7DeleteStoredResultUserEquation

Deletes a stored user defined equation.

```
long St7DeleteStoredResultUserEquation(long uID, long Entity, long Number)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

`Number`

User defined equation number for the specified entity.

St7ReplaceStoredResultUserEquation

Replaces a stored user defined equation.

```
long St7ReplaceStoredResultUserEquation(long uID, long Entity, long Number,
     char* Name, char* Equation, long TrigType)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

`Number`

User defined equation number for the specified entity.

`Name`

Name of the user defined equation.

`Equation`

User defined equation as a character array. See *User Defined Results* for additional information.

`TrigType`

Type of angle arguments; either auRadian or auDegree.

St7RetrieveStoredResultUserEquation

Retrieves a stored user defined equation.

```
long St7RetrieveStoredResultUserEquation(long uID, long Entity, long Number,
     char* Name, char* Equation, long MaxStringLen, long* TrigType)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

`Number`

User defined equation number for the specified entity.

`MaxStringLen`

Maximum number of characters allocated for each of Name and Equation.

Output Parameters

`Name`

Name of the user defined equation.

`Equation`

User defined equation as a character array. See *User Defined Results* for additional information.

`TrigType`

Type of angle arguments; either auRadian or auDegree.

St7GetNumStoredResultUserEquations

Returns the number of stored user defined equations.

```
long St7GetNumStoredResultUserEquations(long uID, long Entity,
     long* NumEquations)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

Output Parameters

`NumEquations`

Number of stored user defined equations for the specified entity.

St7SetStoredResultUserEquation

Makes a stored user defined equation the currently active user defined equation.

```
long St7SetStoredResultUserEquation(long uID, long Entity, long Number)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

`Number`

User defined equation number for the specified entity.

Result Contour File

The results contour functionality offers an alternative workflow to using the functions in *Results.* This workflow is more computationally efficient when a single element-based result quantity is to be extracted from many elements, particularly when the quantity is a nodal quantity extrapolated from Gauss point results (e.g. stress).

St7GeneratePlateContourFile

Generates a plate contour results file containing results of a single quantity for all plates.

```
long St7GeneratePlateContourFile(long uID, long ResultCase, long* Integers,
     long* FileIndex)
```

Input Parameters

`uID`

Strand7 model file ID.

`ResultCase`

Result case.

`Integers[0..7]`

`[ipQuantityRF]` – Plate result quantity; see *Plate Results*. When result quantity is rtPlateUser, the currently active user equation must be assigned via *St7SetResultUserEquation* or *St7SetStoredResultUserEquation* before calling *St7GeneratePlateContourFile*.

`[ipSystemRF]` – Plate result system; see ResultSubType in *Plate Results*.

`[ipComponentRF]` – An index position as specified in *Plate Results*, for example, ipPlateCombVonMises.

`[ipLayerRF]` – Relevant to reinforced concrete or composite results; see *Plate Results*.

`[ipExtrapolateRF]` – eoCentroid for **Centroidal Value**, eoNode for **Nodal Values Extrapolated from Gauss Points** or eoGaussPoint for **Gauss Point Values Placed at Nodes**.

`[ipAverageRF]` – One of aoAlways, aoNever, aoSameProp, aoJumps, aoJumpsN, aoRange or aoSamePropAndStage.

`[ipAbsoluteRF]` – btTrue to return absolute values.

`[ipSubtractSupportRF]` – btTrue to subtract support attribute reactions from node reactions.

Output Parameters

`FileIndex`

File index.

St7GenerateBrickContourFile

Generates a brick contour results file containing results of a single quantity for all bricks.

```
long St7GenerateBrickContourFile(long uID, long ResultCase, long* Integers,
     long* FileIndex)
```

Input Parameters

`uID`

Strand7 model file ID.

`ResultCase`

Result case.

`Integers[0..7]`

`[ipQuantityRF]` – Brick result quantity; see *Brick Results*. When result quantity is rtBrickUser, the currently active user equation must be assigned via *St7SetResultUserEquation* or *St7SetStoredResultUserEquation* before calling *St7GenerateBrickContourFile*.

`[ipSystemRF]` – Brick result system; see ResultSubType in *Brick Results*.

`[ipComponentRF]` – An index position as specified in *Brick Results*. For example, ipBrickCombPrincipal11.

`[ipExtrapolateRF]` – eoCentroid for **Centroidal Value**, eoNode for **Nodal Values Extrapolated from Gauss Points** or eoGaussPoint for **Gauss Point Values Placed at Nodes**.

`[ipAverageRF]` – One of aoAlways, aoNever, aoSameProp, aoJumps, aoJumpsN, aoRange or aoSamePropAndStage.

`[ipAbsoluteRF]` – btTrue to return absolute values.

`[ipSubtractSupportRF]` – btTrue to subtract support attribute reactions from node reactions.

Output Parameters

`FileIndex`

File index.

St7LoadPlateContourFile

Loads a plate contour file.

```
long St7LoadPlateContourFile(long uID, long FileIndex)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileIndex`

File index as returned by *St7GeneratePlateContourFile*.

St7LoadBrickContourFile

Loads a brick contour file.

```
long St7LoadBrickContourFile(long uID, long FileIndex)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileIndex`

File index as returned by *St7GenerateBrickContourFile*.

St7GetPlateContourFileResult

Returns a result from the most recently loaded plate contour file.

```
long St7GetPlateContourFileResult(long uID, long PlateNum, double* PlateResult)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

Output Parameters

`PlateResult[0..kMaxPlateResult-1]`

`[0..26]` – An array containing the plate result at each node and for each of the three planes. Results are returned in three blocks of length 9 and can be accessed as `PlateResult[Surface*9 + Node - 1]` where Surface is one of psPlateMidPlane, psPlateMinusZ or psPlatePlusZ and Node is the node number index from 1 to 9.

St7GetBrickContourFileResult

Returns a result from the most recently loaded brick contour file.

```
long St7GetBrickContourFileResult(long uID, long BrickNum, double* BrickResult)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

Output Parameters

`BrickResult[0..kMaxBrickResult-1]`

`[0..19]` – An array containing the brick result at each node.

Linear Load Case Combinations

Linear load case combination information is stored by the Strand7 model. Secondary result cases are defined by scaling and combining a number of primary result cases. Two tables are associated with each model; one or the other is used to generate combined cases depending on the class of .LSA file being opened:

1. Solver generated .LSA files. These are created by running the Strand7 solver; either from the API or GUI.
2. User generated .LSA files. These are created by the Combine Result Files tool (**CASES/Combine Files**), or with the API *Custom Result Files* functionality.

St7GetNumLSACombinations

Returns the number of linear load case combinations in the specified model for solver-generated .LSA files.

```
long St7GetNumLSACombinations(long uID, long* NumCases)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumCases`

Number of linear load case combinations.

St7SetLSACombinationName

Sets the name of the specified linear load case combination for solver-generated .LSA files.

```
long St7SetLSACombinationName(long uID, long CaseNum, char* CaseName)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case combination number.

`CaseName`

Name of the specified load case combination.

St7GetLSACombinationName

Returns the name of the specified linear load case combination for solver-generated .LSA files.

```
long St7GetLSACombinationName(long uID, long CaseNum, char* CaseName,
     long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case combination number.

`MaxStringLen`

Maximum number of characters allocated for CaseName.

Output Parameters

`CaseName`

Name of the specified load case combination.

St7SetLSACombinationSRAName

Sets the spectral results filename to be used in linear load combination.

```
long St7SetLSACombinationSRAName(long uID, char* FileName)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the spectral results file.

St7GetLSACombinationSRAName

Returns the spectral results filename used in linear load combination.

```
long St7GetLSACombinationSRAName(long uID, char* FileName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxStringLen`

Maximum number of characters allocated for FileName.

Output Parameters

`FileName`

Full path and name for the spectral results file.

St7SetLSACombinationState

Sets the enabled state of the specified linear load case combination for solver-generated .LSA files. Only enabled cases are generated as result cases.

```
long St7SetLSACombinationState(long uID, long CaseNum, bool Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case combination number.

`Enabled`

True to enable the combination.

St7GetLSACombinationState

Returns the enabled state of the specified linear load case combination for solver-generated .LSA files. Only enabled cases are generated as result cases.

```
long St7GetLSACombinationState(long uID, long CaseNum, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case combination number.

`Enabled`

True to enable the combination.

St7AddLSACombination

Adds a new linear load case combination to the specified model for solver-generated .LSA files.

```
long St7AddLSACombination(long uID, char* CaseName)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseName`

Name of load case combination.

St7InsertLSACombination

Inserts a new linear load case combination at the specified position for solver-generated .LSA files.

```
long St7InsertLSACombination(long uID, long Pos, char* CaseName)
```

Input Parameters

`uID`

Strand7 model file ID.

`Pos`

Load case combination number.

`CaseName`

Name of load case combination.

St7DeleteLSACombination

Deletes the specified linear load case combination from the model for solver-generated .LSA files.

```
long St7DeleteLSACombination(long uID, long Pos)
```

Input Parameters

`uID`

Strand7 model file ID.

`Pos`

Load case combination number.

St7SetLSACombinationFactor

Sets the multiplying factor for the specified case in a linear load case combination for solver-generated .LSA files.

```
long St7SetLSACombinationFactor(long uID, long LType, long Pos, long LoadCaseNum,
     long FreedomCaseNum, double Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

`LType`

Either ltLoadCase or ltSpectralCase.

`Pos`

Load case combination number.

`LoadCaseNum`

Load or Spectral case number.

`FreedomCaseNum`

Freedom case number.

`Factor`

Factor value.

St7GetLSACombinationFactor

Returns the multiplying factor for the specified case in a linear load case combination for solver-generated .LSA files.

```
long St7GetLSACombinationFactor(long uID, long LType, long Pos, long LoadCaseNum,
    long FreedomCaseNum, double* Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

`LType`

Either ltLoadCase or ltSpectralCase.

`Pos`

Load case combination number.

`LoadCaseNum`

Load or Spectral case number.

`FreedomCaseNum`

Freedom case number.

Output Parameters

`Factor`

Factor value.

St7GetNumCombinedLSACombinations

Returns the number of linear load case combinations in the specified model for user-generated .LSA files.

```
long St7GetNumCombinedLSACombinations(long uID, long* NumCases)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumCases`

Number of linear load case combinations.

St7SetCombinedLSACombinationName

Sets the name of the specified linear load case combination for user-generated .LSA files.

```
long St7SetCombinedLSACombinationName(long uID, long CaseNum, char* CaseName)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case combination number.

`CaseName`

Name of the specified load case combination.

St7GetCombinedLSACombinationName

Returns the name of the specified linear load case combination for user-generated .LSA files.

```
long St7GetCombinedLSACombinationName(long uID, long CaseNum, char* CaseName,
    long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case combination number.

`MaxStringLen`

Maximum number of characters allocated for CaseName.

Output Parameters

`CaseName`

Name of the specified load case combination.

St7SetCombinedLSACombinationState

Sets the enabled state of the specified linear load case combination for user-generated .LSA files. Only enabled cases are generated as result cases.

```
long St7SetCombinedLSACombinationState(long uID, long CaseNum, bool Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case combination number.

`Enabled`

True to enable the combination.

St7GetCombinedLSACombinationState

Returns the enabled state of the specified linear load case combination for user-generated .LSA files. Only enabled cases are generated as result cases.

```
long St7GetCombinedLSACombinationState(long uID, long CaseNum, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case combination number.

`Enabled`

True if the combination is enabled.

St7AddCombinedLSACombination

Adds a new linear load case combination to the specified model for user-generated .LSA files.

```
long St7AddCombinedLSACombination(long uID, char* CaseName)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseName`

Name of load case combination.

St7InsertCombinedLSACombination

Inserts a new linear load case combination at the specified position for user-generated .LSA files.

```
long St7InsertCombinedLSACombination(long uID, long Pos, char* CaseName)
```

Input Parameters

`uID`

Strand7 model file ID.

`Pos`

Load case combination number.

`CaseName`

Name of load case combination.

St7DeleteCombinedLSACombination

Deletes the specified linear load case combination from the model for user-generated .LSA files.

```
long St7DeleteCombinedLSACombination(long uID, long Pos)
```

Input Parameters

`uID`

Strand7 model file ID.

`Pos`

Load case combination number.

St7SetCombinedLSACombinationFactor

Sets the multiplying factor for the specified case in a linear load case combination for user-generated .LSA files.

```
long St7SetCombinedLSACombinationFactor(long uID, long Pos, long CaseNum,
    double Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

`Pos`

Load case combination number.

`CaseNum`

Primary result case number.

`Factor`

Factor value.

St7GetCombinedLSACombinationFactor

Returns the multiplying factor for the specified case in a linear load case combination for user-generated .LSA files.

```
long St7GetCombinedLSACombinationFactor(long uID, long Pos, long CaseNum,
    double* Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

`Pos`

Load case combination number.

`CaseNum`

Primary result case number.

Output Parameters

`Factor`

Factor value.

Harmonic Time Combination

Functions in this section are equivalent to manipulating **CASES/Harmonic Time** in the GUI. See *Solver – Harmonic Response* for harmonic response configuration prior to the solve being launched.

St7SetHRACombinationLSAName

Sets the linear static results filename to be used with the harmonic time results.

```
long St7SetHRACombinationLSAName(long uID, char* FileName)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the linear static results file.

St7GetHRACombinationLSAName

Returns the linear static results filename to be used with the harmonic time results.

```
long St7GetHRACombinationLSAName(long uID, char* FileName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxStringLen`

Maximum number of characters allocated for FileName.

Output Parameters

`FileName`

Full path and name for the linear static results file.

St7SetHRABaseCombinationFactor

Sets the result factor for the **Base Displacement**, **Base Velocity** or **Base Acceleration** case in harmonic time results.

```
long St7SetHRABaseCombinationFactor(long uID, double Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

`Factor`

Factor value.

St7GetHRABaseCombinationFactor

Returns the result factor for the **Base Displacement**, **Base Velocity** or **Base Acceleration** case in harmonic time results.

```
long St7GetHRABaseCombinationFactor(long uID, double* Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Factor`

Factor value.

St7SetHRACaseCombinationFactor

Sets the result factor for an **Applied Load** case in harmonic time results.

```
long St7SetHRACaseCombinationFactor(long uID, long CaseNum, double Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Case number.

`Factor`

Factor value.

St7GetHRACaseCombinationFactor

Returns the result factor for an **Applied Load** case in harmonic time results.

```
long St7GetHRACaseCombinationFactor(long uID, long CaseNum, double* Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Case number.

Output Parameters

`Factor`

Factor value.

St7SetHRACombinationFactorLSA

Sets the result factor for a linear static result case in harmonic time results.

```
long St7SetHRACombinationFactorLSA(long uID, long LoadCaseNum,
     long FreedomCaseNum, double Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoadCaseNum`

For solver generated linear static result files, the load case number. For Result File Combination generated linear static result files, the result case number.

`FreedomCaseNum`

For solver generated linear static result files, the freedom case number. For Result File Combination generated linear static result files use 0.

`Factor`

Factor value.

St7GetHRACombinationFactorLSA

Returns the result factor for a linear static result case in harmonic time results.

```
long St7GetHRACombinationFactorLSA(long uID, long LoadCaseNum,
     long FreedomCaseNum, double* Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

`LoadCaseNum`

For solver generated linear static result files, the load case number. For Result File Combination generated linear static result files, the result case number.

`FreedomCaseNum`

For solver generated linear static result files, the freedom case number. For Result File Combination generated linear static result files use 0.

Output Parameters

`Factor`

Factor value.

Envelopes

St7GetNumEnvelopes

Returns the number of envelopes for the currently open solution.

```
long St7GetNumEnvelopes(long uID, long* NumLimitEnvelopes,
     long* NumCombinationEnvelopes, long* NumFactorsEnvelopes)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumLimitEnvelopes`

Number of limit envelopes.

`NumCombinationEnvelopes`

Number of combination envelopes.

`NumFactorsEnvelopes`

Number of factors envelopes.

St7GetNumEnvelopesSolver

Returns the number of envelopes for the specified solver and solver mode.

```
long St7GetNumEnvelopesSolver(long uID, long Solver, long SolverMode,
     long* NumLimitEnvelopes, long* NumCombinationEnvelopes,
     long* NumFactorsEnvelopes)
```

Input Parameters

`uID`

Strand7 model file ID.

`Solver`

One of the solver types listed in *Solver Types*.

`SolverMode`

One of smNone, smFreqSolution, smTimeSolution or smTimeMode when Solver is stHarmonicResponse; ignored for all other solvers.

Output Parameters

`NumLimitEnvelopes`

Number of limit envelopes.

`NumCombinationEnvelopes`

Number of combination envelopes.

`NumFactorsEnvelopes`

Number of factors envelopes.

St7AddLimitEnvelope

Adds a new limit envelope for the currently open solution.

```
long St7AddLimitEnvelope(long uID, long EnvType, char* EnvName)
```

Input Parameters

`uID`

Strand7 model file ID.

`EnvType`

One of etLimitEnvelopeAbs, etLimitEnvelopeMin, etLimitEnvelopeMax or etLimitEnvelopeMag.

`EnvName`

Name of the envelope.

St7InsertLimitEnvelope

Inserts a new limit envelope at the specified position for the currently open solution.

```
long St7InsertLimitEnvelope(long uID, long Envelope, long EnvType, char* EnvName)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Limit envelope number.

`EnvType`

One of etLimitEnvelopeAbs, etLimitEnvelopeMin, etLimitEnvelopeMax or etLimitEnvelopeMag.

`EnvName`

Name of the envelope.

St7DeleteLimitEnvelope

Deletes the specified limit envelope for the currently open solution.

```
long St7DeleteLimitEnvelope(long uID, long Envelope)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Limit envelope number.

St7EnableLimitEnvelopeCase

Enables the specified result case in a limit envelope for the currently open solution. Only results from enabled result cases are included in the envelope.

```
long St7EnableLimitEnvelopeCase(long uID, long Envelope, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Limit envelope number.

`CaseNum`

Result case number.

St7DisableLimitEnvelopeCase

Disables the specified result case in a limit envelope for the currently open solution. Only results from enabled result cases are included in the envelope.

```
long St7DisableLimitEnvelopeCase(long uID, long Envelope, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Limit envelope number.

`CaseNum`

Result case number to disable.

St7GetLimitEnvelopeCaseState

Returns the enabled state of the specified result case in a limit envelope for the currently open solution. Only results from enabled result cases are included in the envelope.

```
long St7GetLimitEnvelopeCaseState(long uID, long Envelope, long CaseNum,
     bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Limit envelope number.

`CaseNum`

Result case number.

Output Parameters

`Enabled`

True if the specified result case is enabled.

St7SetLimitEnvelopeData

Assigns the settings for the specified limit envelope for the currently open solution.

```
long St7SetLimitEnvelopeData(long uID, long Envelope, long EnvType,
     char* EnvName)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Limit envelope number.

`EnvType`

One of etLimitEnvelopeAbs, etLimitEnvelopeMin, etLimitEnvelopeMax or etLimitEnvelopeMag.

`EnvName`

Name of the envelope.

St7GetLimitEnvelopeData

Returns the settings assigned to the specified limit envelope for the currently open solution.

```
long St7GetLimitEnvelopeData(long uID, long Envelope, long* EnvType,
     char* EnvName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Limit envelope number.

`MaxStringLen`

Maximum number of characters allocated for EnvName.

Output Parameters

`EnvType`

One of etLimitEnvelopeAbs, etLimitEnvelopeMin, etLimitEnvelopeMax or etLimitEnvelopeMag.

`EnvName`

Name of the envelope.

St7SetEnvelopeAveragingOrder

Sets the averaging option for envelopes.

```
long St7SetEnvelopeAveragingOrder(long uID, long Order)
```

Input Parameters

`uID`

Strand7 model file ID.

`Order`

Order of averaging; either aoAverageThenEnvelope or aoEnvelopeThenAverage.

St7GetEnvelopeAveragingOrder

Returns the averaging option for envelopes.

```
long St7GetEnvelopeAveragingOrder(long uID, long* Order)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Order`

Order of averaging; either aoAverageThenEnvelope or aoEnvelopeThenAverage.

St7SetEnvelopeAdditionalBeamSlices

Sets the option to automatically insert slices to capture maxima and other points of interest in beam force and moment envelopes.

```
long St7SetEnvelopeAdditionalBeamSlices(long uID, bool Additional)
```

Input Parameters

`uID`

Strand7 model file ID.

`Additional`

True to add additional slices.

St7GetEnvelopeAdditionalBeamSlices

Returns the option to automatically insert slices to capture maxima and other points of interest in beam force and moment envelopes.

```
long St7GetEnvelopeAdditionalBeamSlices(long uID, bool* Additional)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Additional`

True to add additional slices.

St7AddCombinationEnvelope

Adds a new combination envelope for the currently open solution.

```
long St7AddCombinationEnvelope(long uID, long EnvType, char* EnvName)
```

Input Parameters

`uID`

Strand7 model file ID.

`EnvType`

Combination envelope type; either etCombEnvelopeMin or etCombEnvelopeMax.

`EnvName`

Name of the envelope.

St7InsertCombinationEnvelope

Inserts a new combination envelope at the specified position for the currently open solution.

```
long St7InsertCombinationEnvelope(long uID, long Envelope, long EnvType,
     char* EnvName)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Combination envelope number.

`EnvType`

Combination envelope type; either etCombEnvelopeMin or etCombEnvelopeMax.

`EnvName`

Name of the envelope.

St7DeleteCombinationEnvelope

Deletes the specified combination envelope for the currently open solution.

```
long St7DeleteCombinationEnvelope(long uID, long Envelope)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Combination envelope number.

St7SetCombinationEnvelopeCase

Sets the state of the specified results case in a combination envelope for the currently open solution.

```
long St7SetCombinationEnvelopeCase(long uID, long Envelope, long CaseNum,
     long State)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Combination envelope number.

`CaseNum`

Result case number.

`State`

One of esCombEnvelopeOn, esCombEnvelopeOff or esCombEnvelopeCheck.

St7GetCombinationEnvelopeCase

Returns the state of the specified result case in a combination envelope for the currently open solution.

```
long St7GetCombinationEnvelopeCase(long uID, long Envelope, long CaseNum,
     long* State)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Combination envelope number.

`CaseNum`

Result case number.

Output Parameters

`State`

One of esCombEnvelopeOn, esCombEnvelopeOff or esCombEnvelopeCheck.

St7SetCombinationEnvelopeData

Assigns the settings for the specified combination envelope for the currently open solution.

```
long St7SetCombinationEnvelopeData(long uID, long Envelope, long EnvType,
     char* EnvName)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Combination envelope number.

`EnvType`

Combination envelope type; either etCombEnvelopeMin or etCombEnvelopeMax.

`EnvName`

Name of the envelope.

St7GetCombinationEnvelopeData

Returns the settings assigned to the specified combination envelope for the currently open solution.

```
long St7GetCombinationEnvelopeData(long uID, long Envelope, long* EnvType,
     char* EnvName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Combination envelope number.

`MaxStringLen`

Maximum number of characters allocated for EnvName.

Output Parameters

`EnvType`

Combination envelope type; either etCombEnvelopeMin or etCombEnvelopeMax.

`EnvName`

Name of the envelope.

St7AddFactorsEnvelope

Adds a new factors envelope to the specified model for the currently open solution.

```
long St7AddFactorsEnvelope(long uID, long EnvType, char* EnvName)
```

Input Parameters

`uID`

Strand7 model file ID.

`EnvType`

Factors envelope type; either etFactEnvelopeMin or etFactEnvelopeMax.

`EnvName`

Name of the envelope.

St7InsertFactorsEnvelope

Inserts a new factors envelope at the specified position for the currently open solution.

```
long St7InsertFactorsEnvelope(long uID, long Envelope, long EnvType,
     char* EnvName)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Factors envelope number.

`EnvType`

Factors envelope type; either etFactEnvelopeMin or etFactEnvelopeMax.

`EnvName`

Name of the envelope.

St7DeleteFactorsEnvelope

Deletes the specified factors envelope for the currently open solution.

```
long St7DeleteFactorsEnvelope(long uID, long Envelope)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Factors envelope number.

St7SetFactorsEnvelopeData

Assigns the settings for the specified factors envelope for the currently open solution.

```
long St7SetFactorsEnvelopeData(long uID, long Envelope, long EnvType,
     char* EnvName)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Factors envelope number.

`EnvType`

Factors envelope type; either etFactEnvelopeMin or etFactEnvelopeMax.

`EnvName`

Name of the envelope.

St7GetFactorsEnvelopeData

Returns the settings assigned to the specified factors envelope for the currently open solution.

```
long St7GetFactorsEnvelopeData(long uID, long Envelope, long* EnvType,
    char* EnvName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Factors envelope number.

`MaxStringLen`

Maximum number of characters allocated for EnvName.

Output Parameters

`EnvType`

Factors envelope type; either etFactEnvelopeMin or etFactEnvelopeMax.

`EnvName`

Name of the envelope.

St7AddFactorsEnvelopeCase

Adds a new result case dependency to the specified factors envelope for the currently open solution.

```
long St7AddFactorsEnvelopeCase(long uID, long Envelope)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Factors envelope number.

St7InsertFactorsEnvelopeCase

Inserts a new result case dependency at the specified position in a factors envelope for the currently open solution.

```
long St7InsertFactorsEnvelopeCase(long uID, long Envelope, long Pos)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Factors envelope number.

`Pos`

New factors envelope case number.

St7DeleteFactorsEnvelopeCase

Deletes the specified result case dependency for a factors envelope for the currently open solution.

```
long St7DeleteFactorsEnvelopeCase(long uID, long Envelope, long Pos)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Factors envelope number.

`Pos`

Factors envelope case number.

St7SetFactorsEnvelopeCaseData

Assigns the settings for the specified factors envelope case for the currently open solution.

```
long St7SetFactorsEnvelopeCaseData(long uID, long Envelope, long Pos,
     long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Factors envelope number.

`Pos`

Factors envelope case number.

`Integers[0..1]`

A 2-element array containing the result case number and set number for the factors envelope case respectively.

`Doubles[0..1]`

A 2-element array containing the **Factor1** and **Factor2** values for the factors envelope case.

St7GetFactorsEnvelopeCaseData

Returns the settings assigned to the specified factors envelope case for the currently open solution.

```
long St7GetFactorsEnvelopeCaseData(long uID, long Envelope, long Pos,
     long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Factors envelope number.

`Pos`

Factors envelope case number.

Output Parameters

`Integers[0..1]`

A 2-element array containing the result case number and set number for the factors envelope case respectively.

`Doubles[0..1]`

A 2-element array containing the **Factor1** and **Factor2** values for the factors envelope case.

St7GetNumFactorsEnvelopeCases

Returns the number of cases (rows) included in the specified factors envelope for the currently open solution.

```
long St7GetNumFactorsEnvelopeCases(long uID, long Envelope, long* NumCases)
```

Input Parameters

`uID`

Strand7 model file ID.

`Envelope`

Factors envelope number.

Output Parameters

`NumCases`

Number of cases in the envelope.

St7AddFactorsEnvelopeSet

Adds a new set to the specified factors envelope for the currently open solution.

```
long St7AddFactorsEnvelopeSet(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7InsertFactorsEnvelopeSet

Inserts a new set at the specified position for a factors envelope for the currently open solution.

```
long St7InsertFactorsEnvelopeSet(long uID, long Pos)
```

Input Parameters

`uID`

Strand7 model file ID.

`Pos`

New set position.

St7DeleteFactorsEnvelopeSet

Deletes the specified set from a factors envelope for the currently open solution.

```
long St7DeleteFactorsEnvelopeSet(long uID, long Pos)
```

Input Parameters

`uID`

Strand7 model file ID.

`Pos`

Set position.

St7GetNumFactorsEnvelopeSets

Returns the number of sets assigned to the specified factors envelope for the currently open solution.

```
long St7GetNumFactorsEnvelopeSets(long uID, long* NumSets)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumSets`

Number of sets in the envelope.

St7SetFactorsEnvelopeSetData

Assigns the settings for the specified set in a factors envelope for the currently open solution.

```
long St7SetFactorsEnvelopeSetData(long uID, long Pos, long SetType,
     char* SetName, char* SetGroup)
```

Input Parameters

`uID`

Strand7 model file ID.

`Pos`

Set position.

`SetType`

Type of set; either stExclusiveOR or stExclusiveAND.

`SetName`

Name of the set.

`SetGroup`

Group identifier for set.

St7GetFactorsEnvelopeSetData

Returns the settings assigned to the specified set in a factors envelope for the currently open solution.

```
long St7GetFactorsEnvelopeSetData(long uID, long Pos, long* SetType,
     char* SetName, char* SetGroup, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`Pos`

Set position.

`MaxStringLen`

Maximum number of characters allocated for SetName.

Output Parameters

`SetType`

Type of set; either stExclusiveOR or stExclusiveAND.

`SetName`

Name of the set.

`SetGroup`

Group identifier for set.

Result File Combination

St7SetResultFileCombTargetFileName

Sets the name of the target file produced when forming a combined result file.

```
long St7SetResultFileCombTargetFileName(long uID, char* FileName)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the target file.

St7GetResultFileCombTargetFileName

Returns the name of the target file produced when forming a combined result file.

```
long St7GetResultFileCombTargetFileName(long uID, char* FileName,
     long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxStringLen`

Maximum number of characters allocated for FileName.

Output Parameters

`FileName`

Full path and name for the target file.

St7AddResultFileCombFileName

Adds a new file to the current results file combination.

```
long St7AddResultFileCombFileName(long uID, char* FileName)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the new result file.

St7DeleteResultFileCombFileName

Deletes the specified file from the current results file combination.

```
long St7DeleteResultFileCombFileName(long uID, long FileNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileNum`

File number in current combination.

St7SetResultFileCombFileName

Sets the name of the specified file in the current result file combination.

```
long St7SetResultFileCombFileName(long uID, long FileNum, char* FileName)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileNum`

File number in the current combination.

`FileName`

Full path and name for the result file.

St7GetResultFileCombFileName

Returns the name of the specified file in the current results file combination.

```
long St7GetResultFileCombFileName(long uID, long FileNum, char* FileName,
     long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileNum`

File number in the current combination.

`MaxStringLen`

Maximum number of characters allocated for FileName.

Output Parameters

`FileName`

Full path and name for the file.

St7AddResultFileCombCase

Adds a new case to the current result file combination.

```
long St7AddResultFileCombCase(long uID, char* CaseName)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseName`

New combination case name.

St7DeleteResultFileCombCase

Deletes the specified case from the current result file combination.

```
long St7DeleteResultFileCombCase(long uID, long Pos)
```

Input Parameters

`uID`

Strand7 model file ID.

`Pos`

Result case in combined file.

St7SetResultFileCombCaseData

Sets the combination data for a combined result case in the specified result file combination.

```
long St7SetResultFileCombCaseData(long uID, long FileNum, long Pos, long CaseNum,
    double Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileNum`

Combination file number.

`Pos`

Result case in combined file.

`CaseNum`

Result case.

`Factor`

Combination factor.

St7GetResultFileCombCaseData

Returns the combination data assigned to a combined result case in the specified result file combination.

```
long St7GetResultFileCombCaseData(long uID, long FileNum, long Pos,
     long* CaseNum, double* Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileNum`

Combination file number.

`Pos`

Result case in combined file.

Output Parameters

`CaseNum`

Result case.

`Factor`

Combination factor.

St7SetResultFileCombCaseName

Sets the name of a combined result case in the specified result file combination.

```
long St7SetResultFileCombCaseName(long uID, long Pos, char* CaseName)
```

Input Parameters

`uID`

Strand7 model file ID.

`Pos`

Result case in combined file.

`CaseName`

Combined result case name.

St7GetResultFileCombCaseName

Returns the name assigned to a combined result case in the specified result file combination.

```
long St7GetResultFileCombCaseName(long uID, long Pos, char* CaseName,
     long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`Pos`

Result case in combined file.

`MaxStringLen`

Maximum number of characters allocated for CaseName.

Output Parameters

`CaseName`

Combined result case name.

St7GenerateResultFileComb

Generates the combined result file using the specified method.

```
long St7GenerateResultFileComb(long uID, long Method)
```

Input Parameters

`uID`

Strand7 model file ID.

`Method`

Combination method; either rfCombFactors or rfCombSRSS.

St7RetrieveResultFileComb

Retrieves the combination components from a previously generated result file.

```
long St7RetrieveResultFileComb(long uID, char* FileName)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the result file.

Harmonic Time History

St7GenerateHRATimeHistory

Generates the time history response for the specified model based on a harmonic response analysis. An associated harmonic response result file must currently be open.

```
long St7GenerateHRATimeHistory(long uID, double StartTime, double EndTime,
     long NumSteps, long* WarningCode)
```

Input Parameters

`uID`

Strand7 model file ID.

`StartTime`

Start time for the time history integration.

`EndTime`

End time for the time history integration.

`NumSteps`

Number of steps used for the time history integration.

Output Parameters

`WarningCode`

Either wcHarmonicCombineNoWarning if the operation was successful, or wcHarmonicCombineInvalidLSA if the time history was generated but the specified linear static file was invalid.

St7ClearHRATimeHistory

Clears the time history response for the specified model based on a harmonic response analysis. An associated harmonic response result file must currently be open.

```
long St7ClearHRATimeHistory(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

Influence Combinations

St7SetInfluenceFileName

Sets the name of the load influence result file used for combinations.

```
long St7SetInfluenceFileName(long uID, char* FileName)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name of load influence result file name.

St7GetInfluenceFileName

Returns the name of the load influence result file used for combinations.

```
long St7GetInfluenceFileName(long uID, char* FileName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxStringLen`

Maximum number of characters allocated for CaseName.

Output Parameters

`FileName`

Full path and name of load influence result file name.

St7GetNumInfluenceVariables

Returns the number of load influence variables available in the current result file.

```
long St7GetNumInfluenceVariables(long uID, long* NumVariables)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumVariables`

Number of available load influence variables.

St7GetNumInfluenceMultiVariableCases

Returns the number of load influence variables available in the current result file.

```
long St7GetNumInfluenceMultiVariableCases(long uID, long* NumMultiVariableCases)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumMultiVariableCases`

Number of multi-variable load influence cases defined.

St7GetInfluenceVariable

Returns information about a load influence variable.

```
long St7GetInfluenceVariable(long uID, long VariableID, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`VariableID`

Load influence variable number.

Output Parameters

`Integers[0..6]`

`[ipLIAVarLoadCaseNum]` – Load case in which the response variable is defined.

`[ipLIAVarFreedomCaseNum]` – Freedom case used for the influence analysis.

`[ipLIAVarEntity]` – Entity type; one of tyNODE, tyBEAM, tyPLATE or tyBRICK.

`[ipLIAVarEntityNum]` – Entity number.

`[ipLIAVarUCSId]` – ID number of the UCS, if relevant.

`[ipLIAVarType]` – For tyNODE, either rvNodeDisplacement or rvNodeReaction. For tyBEAM, the beam end. For tyPLATE, either rvPlateForce or rvPlateMoment. Not relevant for tyBRICK.

`[ipLIAVarComponent]` – The degree of freedom or axis direction of the response variable.

St7SetInfluenceMinVariableState

Sets the enabled state of the specified single-variable minimum load influence combination.

```
long St7SetInfluenceMinVariableState(long uID, long MinVariableID, bool Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`MinVariableID`

ID of the load influence variable.

`Enabled`

True to enable the variable.

St7GetInfluenceMinVariableState

Returns the enabled state of the specified single-variable minimum load influence combination.

```
long St7GetInfluenceMinVariableState(long uID, long MinVariableID, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`MinVariableID`

ID of the load influence variable.

Output Parameters

`Enabled`

True if the variable is enabled.

St7SetInfluenceMaxVariableState

Sets the enabled state of the specified single-variable maximum load influence combination.

```
long St7SetInfluenceMaxVariableState(long uID, long MaxVariableID, bool Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxVariableID`

ID of the load influence variable.

`Enabled`

True to enable the variable.

St7GetInfluenceMaxVariableState

Returns the enabled state of the specified single-variable maximum load influence combination.

```
long St7GetInfluenceMaxVariableState(long uID, long MaxVariableID, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxVariableID`

ID of the load influence variable.

Output Parameters

`Enabled`

True if the variable is enabled.

St7SetInfluenceMultiVariableState

Set the enabled state of the specified variable within a multi-variable load influence combination case.

```
long St7SetInfluenceMultiVariableState(long uID, long MultiVariableID,
     long MultiVariableCaseID, bool Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`MultiVariableID`

ID of the load influence variable.

`MultiVariableCaseID`

ID of the load influence multi-variable combination case.

`Enabled`

True to enable the variable in the case.

St7GetInfluenceMultiVariableState

Returns the enabled state of the specified variable within a multi-variable load influence combination case.

```
long St7GetInfluenceMultiVariableState(long uID, long MultiVariableID,
     long MultiVariableCaseID, bool* Enabled)
```

Input Parameters

`uID`

Strand7 model file ID.

`MultiVariableID`

ID of the load influence variable.

`MultiVariableCaseID`

ID of the load influence multi-variable combination case.

Output Parameters

`Enabled`

True if the variable is enabled in the case.

St7SetInfluenceMultiVariableType

Sets the type of a multi-variable load influence combination case.

```
long St7SetInfluenceMultiVariableType(long uID, long MultiVariableCaseID,
     long MultiVariableType)
```

Input Parameters

`uID`

Strand7 model file ID.

`MultiVariableCaseID`

ID of the load influence multi-variable combination case.

`MultiVariableType`

Either icInfluenceMin or icInfluenceMax.

St7GetInfluenceMultiVariableType

Returns the type of a multi-variable load influence combination case.

```
long St7GetInfluenceMultiVariableType(long uID, long MultiVariableCaseID,
     long* MultiVariableType)
```

Input Parameters

`uID`

Strand7 model file ID.

`MultiVariableCaseID`

ID of the load influence multi-variable combination case.

Output Parameters

`MultiVariableType`

Either icInfluenceMin or icInfluenceMax.

St7AddInfluenceMultiVariableCase

Adds a new multi-variable load influence combination case.

```
long St7AddInfluenceMultiVariableCase(long uID, long MultiVariableType,
     char* MultiVariableName)
```

Input Parameters

`uID`

Strand7 model file ID.

`MultiVariableType`

Either icInfluenceMin or icInfluenceMax.

`MultiVariableName`

Name of the load influence multi-variable combination case.

St7DeleteInfluenceMultiVariableCase

Deletes a multi-variable load influence combination case.

```
long St7DeleteInfluenceMultiVariableCase(long uID, long MultiVariableCaseID)
```

Input Parameters

`uID`

Strand7 model file ID.

`MultiVariableCaseID`

ID of the load influence multi-variable combination case.

St7SetInfluenceMultiVariableName

Sets the name of a multi-variable load influence combination case.

```
long St7SetInfluenceMultiVariableName(long uID, long MultiVariableCaseID,
     char* MultiVariableName)
```

Input Parameters

`uID`

Strand7 model file ID.

`MultiVariableCaseID`

ID of the load influence multi-variable combination case.

`MultiVariableName`

Name of the load influence multi-variable combination case.

St7GetInfluenceMultiVariableName

Returns the name of a multi-variable load influence combination case.

```
long St7GetInfluenceMultiVariableName(long uID, long MultiVariableCaseID,
     char* MultiVariableName, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`MultiVariableCaseID`

ID of the load influence multi-variable combination case.

`MaxStringLen`

The maximum number of characters allocated for MultiVariableName.

Output Parameters

`MultiVariableName`

Name of the load influence multi-variable combination case.

St7SetInfluenceGroupState

Includes or excludes a group from load influence combinations.

```
long St7SetInfluenceGroupState(long uID, long GroupID, bool Included)
```

Input Parameters

`uID`

Strand7 model file ID.

`GroupID`

Group ID.

`Included`

True if the group is to be included.

St7GetInfluenceGroupState

Returns the included state of a group from load influence combinations.

```
long St7GetInfluenceGroupState(long uID, long GroupID, bool* Included)
```

Input Parameters

`uID`

Strand7 model file ID.

`GroupID`

Group ID.

Output Parameters

`Included`

True if the group is included.

St7SetInfluencePropertyState

Includes or excludes a property from load influence combinations.

```
long St7SetInfluencePropertyState(long uID, long Entity, long PropNum,
    bool Included)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP or ptBRICKPROP.

`PropNum`

Property number.

`Included`

True if the property is to be included.

St7GetInfluencePropertyState

Returns the included status of a property from load influence combinations.

```
long St7GetInfluencePropertyState(long uID, long Entity, long PropNum,
    bool* Included)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of ptBEAMPROP, ptPLATEPROP or ptBRICKPROP.

`PropNum`

Property number.

Output Parameters

`Included`

True if the property is included.

St7SetInfluenceCombinationOptions

Sets the load influence combination generation options.

```
long St7SetInfluenceCombinationOptions(long uID, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Integers[0..4]`

Components to include in the names of load cases generated by *St7GenerateInfluenceCases*.

`[ipInfCaseLabel]` – btTrue to include the label, or btFalse to exclude it.

`[ipInfCaseVariable]` – btTrue to include the variable, or btFalse to exclude it.

`[ipInfCaseLoadCase]` – btTrue to include the load case, or btFalse to exclude it.

`[ipInfCaseFreedomCase]` – btTrue to include the freedom case, or btFalse to exclude it.

`[ipInfCaseResponseType]` – btTrue to include the response type, or btFalse to exclude it.

St7GetInfluenceCombinationOptions

Returns the load influence combination generation options.

```
long St7GetInfluenceCombinationOptions(long uID, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Integers[0..4]`

Components to include in the names of load cases generated by *St7GenerateInfluenceCases*.

`[ipInfCaseLabel]` – btTrue to include the label, or btFalse to exclude it.

`[ipInfCaseVariable]` – btTrue to include the variable, or btFalse to exclude it.

`[ipInfCaseLoadCase]` – btTrue to include the load case, or btFalse to exclude it.

`[ipInfCaseFreedomCase]` – btTrue to include the freedom case, or btFalse to exclude it.

`[ipInfCaseResponseType]` – btTrue to include the response type, or btFalse to exclude it.

St7GenerateInfluenceCases

Generates load influence combination load cases.

```
long St7GenerateInfluenceCases(long uID, bool RemoveExisting, bool AllowStop,
     bool WriteLog, long Mode, long* NumCasesDeleted, long* NumCasesGenerated,
     long* WarningCode)
```

Input Parameters

`uID`

Strand7 model file ID.

`RemoveExisting`

Remove pre-existing load influence combination cases.

`AllowStop`

True to permit the user to stop the generation of load influence combination cases.

`WriteLog`

True to write log file output of combination case generation. If True, full filename with the path can be accessed using *St7GetGlobalStringValue*.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

Output Parameters

`NumCasesDeleted`

Number of pre-existing cases deleted.

`NumCasesGenerated`

Number of load cases generated.

`WarningCode`

One of wcInfluenceNoWarning to indicate success, wcInfluenceUserTerminated to indicate the combinations are incomplete due to user termination, or wcInfluenceRanOutOfAttributeID to indicate the supply of unique attribute IDs has been exhausted.

Dependencies

`Influence Combination Options`

Generated load case names controlled by *St7SetInfluenceCombinationOptions*.

Custom Result Files

St7NewResFile

Creates a new custom result file.

```
long St7NewResFile(long uID, char* FileName, long ResultType)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the new custom result file.

`ResultType`

One of stLinearStatic, stLinearBuckling, stNonlinearStatic, stQuasiStatic, stNaturalFrequency, stLinearTransientDynamic, stNonlinearTransientDynamic, stSteadyHeat, stTransientHeat.

St7OpenResFile

Opens the specified custom result file.

```
long St7OpenResFile(long uID, char* FileName)
```

Input Parameters

`uID`

Strand7 model file ID.

`FileName`

Full path and name for the custom result file.

St7CloseResFile

Closes the open custom result file.

```
long St7CloseResFile(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7SetResFileDescription

Sets the description for the specified custom result file.

```
long St7SetResFileDescription(long uID, char* Name)
```

Input Parameters

`uID`

Strand7 model file ID.

`Name`

Custom result file description.

St7GetResFileDescription

Returns the description of the specified custom result file.

```
long St7GetResFileDescription(long uID, char* Name, long MaxStringLen)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxStringLen`

Maximum number of characters allocated for Name.

Output Parameters

`Name`

Custom result file description.

St7SetResFileNumCases

Sets the number of result cases in the open custom result file.

```
long St7SetResFileNumCases(long uID, long NumCases)
```

Input Parameters

`uID`

Strand7 model file ID.

`NumCases`

Number of result cases in the custom result file.

St7SetResFileCaseName

Sets the name of the specified result case in the custom result file.

```
long St7SetResFileCaseName(long uID, long CaseNum, char* CaseName)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`CaseName`

Result case name.

St7AssociateResFileCase

Associates load and freedom cases with the specified result case in the custom result file.

```
long St7AssociateResFileCase(long uID, long CaseNum, long LoadCase,
     long FreedomCase)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`LoadCase`

Load case number.

`FreedomCase`

Freedom case number for linear static custom result file.

Applicability

Applicable to linear static and steady heat custom result files.

St7AssociateResFileStage

Associates a stage with the specified result case in the custom result file.

```
long St7AssociateResFileStage(long uID, long CaseNum, long Stage)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`Stage`

Stage index.

St7AssociateResFileNSMassCase

Associates the non-structural mass from a particular load case with a custom result file; this is relevant to natural frequency results. The association is required for correct determination of mass participation if the custom natural frequency result file is used in a conventional spectral response analysis, and the frequencies and mode shapes represented in the custom natural frequency result file depend on non-structural mass attributes.

```
long St7AssociateResFileNSMassCase(long uID, long CaseNum, double Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

`Factor`

Scaling factor for load case to be associated, generally 1.0.

Applicability

Applicable to natural frequency custom result files.

St7SetResFileFreedomCase

Assigns the freedom case associated with the result file.

```
long St7SetResFileFreedomCase(long uID, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Freedom case number.

Applicability

Applicable to linear static, linear buckling and natural frequency custom result files. When applied to a linear static result file, the function sets the default freedom case for result cases that have not been explicitly assigned a freedom case using *St7AssociateResFileCase*.

St7GetResFileFreedomCase

Returns the freedom case associated with the custom result file.

```
long St7GetResFileFreedomCase(long uID, long* CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`CaseNum`

Freedom case number.

Applicability

Applicable to linear buckling and natural frequency custom result files.

St7SetResFileMode

Sets the frequency or buckling factor for the specified result case in the custom result file.

```
long St7SetResFileMode(long uID, long CaseNum, double Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`Mode`

Mode frequency (Hz) for natural frequency results, or buckling factor for linear buckling results.

St7GetResFileMode

Returns the frequency or buckling factor for the specified result case in the custom result file.

```
long St7GetResFileMode(long uID, long CaseNum, double* Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

Output Parameters

`Mode`

Mode frequency (Hz) for natural frequency results, or buckling factor for linear buckling results.

St7SetResFileTime

Sets the integration time for the specified result case in the custom result file; units are seconds.

```
long St7SetResFileTime(long uID, long CaseNum, double Time)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`Time`

Integration time in seconds.

St7GetResFileTime

Returns the integration time assigned to the specified result case in the custom result file; units are seconds.

```
long St7GetResFileTime(long uID, long CaseNum, double* Time)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

Output Parameters

`Time`

Integration time in seconds.

St7SetResFileTimeUnit

Sets the time units displayed in the specified custom result file. Note that this setting does not affect the time input to *St7SetResFileTime*, which is always in seconds.

```
long St7SetResFileTimeUnit(long uID, long TimeUnit)
```

Input Parameters

`uID`

Strand7 model file ID.

`TimeUnit`

One of tuMilliSec, tuSec, tuMin, tuHour or tuDay.

St7GetResFileTimeUnit

Returns the time units displayed in the specified custom result file. Note that this setting does not affect the time output by *St7GetResFileTime*, which is always in seconds.

```
long St7GetResFileTimeUnit(long uID, long* TimeUnit)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`TimeUnit`

One of tuMilliSec, tuSec, tuMin, tuHour or tuDay.

St7SetResFileQuantity

Includes the specified result quantity in a given result case of the custom result file.

```
long St7SetResFileQuantity(long uID, long CaseNum, long Entity, long Quantity)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`Entity`

One of tyNODE, tyBEAM, tyPLATE or tyBRICK.

`Quantity`

Result quantity. See *Custom Results* for additional information.

St7ClearResFileQuantity

Removes the specified result quantity from a given result case of the custom result file.

```
long St7ClearResFileQuantity(long uID, long CaseNum, long Entity, long Quantity)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`Entity`

One of tyNODE, tyBEAM, tyPLATE or tyBRICK.

`Quantity`

Result quantity. See *Custom Results* for additional information.

St7GetResFileQuantityState

Returns the included/excluded state of an element result quantity in a given result case of the custom result file.

```
long St7GetResFileQuantityState(long uID, long CaseNum, long Entity,
     long Quantity, bool* Included)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`Entity`

One of tyNODE, tyBEAM, tyPLATE or tyBRICK.

`Quantity`

Result quantity. See *Custom Results* for additional information.

Output Parameters

`Included`

True if the specified entity result quantity is included in the given result case of the custom result file.

St7GetResFileUnits

Returns the units in which result data must be provided to *St7SetResFileNodeResult*, *St7SetResFileBeamResult*, *St7SetResFilePlateResult, St7SetResFilePlatePressureResult* and *St7SetResFileBrickResult*.

```
long St7GetResFileUnits(long uID, long* Units)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Units[0..kLastUnit-1]`

`[ipLENGTHU]` – one of luMETRE, luCENTIMETRE, luMILLIMETRE, luFOOT or luINCH;
measuring length in metres, centimetres, millimetres, feet or inches respectively.

`[ipFORCEU]` – one of fuNEWTON, fuKILONEWTON, fuMEGANEWTON, fuKILOFORCE, fuPOUNDFORCE, fuTONNEFORCE or fuKIPFORCE;

measuring force in newtons, kilonewtons, meganewtons, kilograms-force, pounds-force, tonnes-force or kilopounds-force respectively.

`[ipSTRESSU]` – one of suPASCAL, suKILOPASCAL, suMEGAPASCAL, suKSCm, suPSI, suKSI or suPSF; measuring stress in units of pascals, kilopascals, megapascals, kilograms-force per square centimetre, pounds per square inch, kilopounds per square inch, or pounds per square foot respectively.

`[ipMASSU]` – one of muKILOGRAM, muTONNE, muGRAM, muPOUND or muSLUG;
measuring mass in units of kilograms, tonnes, grams, pounds or slugs respectively.

`[ipTEMPERU]` – one of tuCELSIUS, tuFAHRENHEIT, tuKELVIN or tuRANKINE;
measuring temperature in units of Celsius, Fahrenheit, Kelvin or Rankine respectively.

`[ipENERGYU]` – one of euJOULE, euKILOJOULE, euBTU, euFTLBF or euCALORIE;
measuring energy in units of joules, kilojoules, British thermal units, foot pounds-force or calories respectively.

St7SetResFileNodeResult

Sets the specified nodal result quantities for a given node and result case in the custom result file.

```
long St7SetResFileNodeResult(long uID, long CaseNum, long NodeNum, long Quantity,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`NodeNum`

Node number.

`Quantity`

One of rtNodeDisp, rtNodeVel, rtNodeAcc, rtNodeReact, rtNodeTemp or rtNodeFlux.

`Doubles[0..5]`

An array defining the specified nodal result quantity. See *Custom Results* for additional information.

St7GetResFileNodeResult

Returns the specified nodal result quantities for a given node and result case in the custom result file.

```
long St7GetResFileNodeResult(long uID, long CaseNum, long NodeNum, long Quantity,
    double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`NodeNum`

Node number.

`Quantity`

One of rtNodeDisp, rtNodeVel, rtNodeAcc, rtNodeReact, rtNodeTemp or rtNodeFlux.

Output Parameters

`Doubles[0..5]`

An array defining the specified nodal result quantity. See *Custom Results* for additional information.

St7SetResFileBeamStations

Sets the number of result stations used to store beam results for the specified result case in the custom result file.

```
long St7SetResFileBeamStations(long uID, long CaseNum, long Stations)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`Stations`

Number of result stations along the length of beam elements in the specified result case.

Note that only two stations (corresponding to the beam ends) are permitted for heat transfer results.

Usage

This function should be called once per result case in the custom result file, not once per beam element. The set value applies to all beam elements in the model for the specified result case.

St7GetResFileBeamStations

Returns the number of result stations used to store beam results for the specified result case in the custom result file.

```
long St7GetResFileBeamStations(long uID, long CaseNum, long* Stations)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

Output Parameters

`Stations`

Number of result stations along the length of beam elements in the specified result case of the custom result file.

St7SetResFileBeamResult

Sets the specified beam result quantities for a given beam element and result case in the custom result file.

```
long St7SetResFileBeamResult(long uID, long CaseNum, long BeamNum, long Quantity,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`BeamNum`

Beam number.

`Quantity`

One of rtBeamForce, rtBeamAllStrain, rtBeamNodeReact, rtBeamFlux or rtBeamNodeFlux.

`Doubles[..]`

An array defining the specified beam result quantity at each station along the beam. See *Custom Results* for additional information.

St7GetResFileBeamResult

Returns the specified beam result quantities for a given beam element and result case in the custom result file.

```
long St7GetResFileBeamResult(long uID, long CaseNum, long BeamNum, long Quantity,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`BeamNum`

Beam number.

`Quantity`

One of rtBeamForce, rtBeamAllStrain, rtBeamNodeReact, rtBeamFlux or rtBeamNodeFlux.

Output Parameters

`Doubles[..]`

An array defining the specified beam result quantity at each station along the beam. See *Custom Results* for additional information.

St7SetResFileBeamReleaseResult

Sets the specified beam end release quantities for a given beam element and result case in the custom result file. Note that for beam end release results to be stored, *St7SetResFileQuantity* must have been called with either rtBeamExtraResults, or at least with rtBeamForce or rtBeamAllStrain.

```
long St7SetResFileBeamReleaseResult(long uID, long CaseNum, long BeamNum,
     bool* BeamReleased, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`BeamNum`

Beam number.

`BeamReleased[0..kMaxBeamRelease-1]`

An array containing the release status for the beam ends according to the local beam axis system for translational and rotational DoF. See *Custom Results* for additional information.

`Doubles[0..kMaxBeamRelease-1]`

An array containing the displacement results for the released beam end DoF. See *Custom Results* for additional information.

St7GetResFileBeamReleaseResult

Returns the specified beam end release quantities for a given beam element and result case in the custom result file.

```
long St7GetResFileBeamReleaseResult(long uID, long CaseNum, long BeamNum,
     bool* BeamReleased, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`BeamNum`

Beam number.

Output Parameters

`BeamReleased[0..kMaxBeamRelease-1]`

An array containing the release status for the beam ends according to the local beam axis system for translational and rotational DoF. See *Custom Results* for additional information.

`Doubles[0..kMaxBeamRelease-1]`

An array containing the displacement results for the released beam end DoF. See *Custom Results* for additional information.

St7SetResFilePlateResult

Sets the specified plate result quantities for a given plate element and result case in the custom result file.

```
long St7SetResFilePlateResult(long uID, long CaseNum, long PlateNum,
     long Quantity, bool NonlinearMaterial, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`PlateNum`

Plate number.

`Quantity`

One of rtPlateStress, rtPlateStrain, rtPlateNodeReact, rtPlateFlux or rtPlateNodeFlux.

`NonlinearMaterial`

True if the results correspond to a material nonlinear analysis. This option can be set only for result files of type stNonlinearStatic, stQuasiStatic and stNonlinearTransientDynamic.

`Doubles[..]`

An array defining the specified plate result quantity at each Gauss point or node on the plate, depending on the quantity. See *Custom Results* for additional information.

St7GetResFilePlateResult

Returns the specified plate result quantities for a given plate element and result case in the custom result file.

```
long St7GetResFilePlateResult(long uID, long CaseNum, long PlateNum,
     long Quantity, bool* NonlinearMaterial, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`PlateNum`

Plate number.

`Quantity`

One of rtPlateStress, rtPlateStrain, rtPlateNodeReact, rtPlateFlux or rtPlateNodeFlux.

Output Parameters

`NonlinearMaterial`

True if the results correspond to a material nonlinear analysis. This option applies only for result files of type stNonlinearStatic, stQuasiStatic and stNonlinearTransientDynamic.

`Doubles[..]`

An array defining the specified plate result quantity at each Gauss point or node on the plate, depending on the quantity. See *Custom Results* for additional information.

St7SetResFilePlatePressureResult

Sets the applied normal pressure load for a given plate element and result case in the custom result file.

```
long St7SetResFilePlatePressureResult(long uID, long CaseNum, long PlateNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`PlateNum`

Plate number.

Output Parameters

`Doubles[0..1]`

`[0]` – Normal face pressure over the -z surface of the plate.

`[1]` – Normal face pressure over the +z surface of the plate.

Positive pressure is directed from the relevant surface into the plate. See *Custom Results* for additional information.

St7GetResFilePlatePressureResult

Returns the applied normal pressure load for a given plate element and result case in the custom result file.

```
long St7GetResFilePlatePressureResult(long uID, long CaseNum, long PlateNum,
     double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`PlateNum`

Plate number.

Output Parameters

`Doubles[0..1]`

`[0]` – Normal face pressure over the -z surface of the plate.

`[1]` – Normal face pressure over the +z surface of the plate.

Positive pressure is directed from the relevant surface into the plate. See *Custom Results* for additional information.

St7SetResFileBrickResult

Sets the specified brick result quantities for a given brick element and result case in the custom result file.

```
long St7SetResFileBrickResult(long uID, long CaseNum, long BrickNum,
     long Quantity, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`BrickNum`

Brick number.

`Quantity`

One of rtBrickStress, rtBrickStrain, rtBrickNodeReact, rtBrickFlux or rtBrickNodeFlux.

`Doubles[..]`

An array defining the specified brick result quantity at each Gauss point or node on the brick, depending on the quantity. See *Custom Results* for additional information.

St7GetResFileBrickResult

Returns the specified brick result quantities for a given brick element and result case in the custom result file.

```
long St7GetResFileBrickResult(long uID, long CaseNum, long BrickNum,
     long Quantity, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Result case number.

`BrickNum`

Brick number.

`Quantity`

One of rtBrickStress, rtBrickStrain, rtBrickNodeReact, rtBrickFlux or rtBrickNodeFlux.

Output Parameters

`Doubles[..]`

An array defining the specified brick result quantity at each Gauss point or node on the brick, depending on the quantity. See *Custom Results* for additional information.

Clipboard

Clipboard, in this section, refers to the internal Strand7 clipboard, rather than the standard Windows clipboard. The Strand7 clipboard is used for copying/pasting entities from the model database; this functionality is accessible via the VISUAL tab in the Strand7 GUI. The functions in this chapter allow you cut, copy and paste entities within a single model or between models. As with the GUI, the Strand7 clipboard allows you to transfer entities between currently open models, rather than to external programs. To copy model window graphics to the Windows clipboard for access by other applications see *St7ExportImageToClipboard*.

St7SetPasteOptions

Sets paste options that will be applied by subsequent paste operations.

```
long St7SetPasteOptions(long uID, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Integers[0..7]`

`[ipPasteCases]` – Either poCasesInOrder or poCasesMatchNames.

`[ipPasteProperties]` – One of poPropertiesUsePropertyID, poPropertiesMatchExisting or poPropertiesCreateNew.

`[ipPasteLoadPaths]` – Either poLoadPathUseTemplateID or poLoadPathCreateNew.

`[ipPasteAttributes]` – btTrue to **Paste Attributes**.

`[ipPasteGroups]` – btTrue to **Match Group Names**.

`[ipPasteSets]` – btTrue to **Match Set Names**.

`[ipPasteGlobals]` – btTrue to **Merge Load Case Globals**.

`[ipPasteTables]` – btTrue to **Paste tables**.

St7CutToSt7Clipboard

Cuts the selected entities from the model to the clipboard.

```
long St7CutToSt7Clipboard(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

St7CopyToSt7Clipboard

Copies the selected entities from the model to the clipboard.

```
long St7CopyToSt7Clipboard(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

St7PasteFromSt7ClipboardByIncrements

Pastes the contents of the clipboard (source) into the model (target), similarly to pasting using the **Adjust** tab in the GUI.

```
long St7PasteFromSt7ClipboardByIncrements(long uID, double* Rotation,
    double* Translation, double Scaling)
```

Input Parameters

`uID`

Strand7 model file ID.

`Rotation[0..2]`

Rotation angles (degrees) about the X, Y and Z global axes of the source, applied to the entities being pasted.

`Translation[0..2]`

Translation increments in the length units of the target, in the X, Y and Z global axes of the target, applied to the entities being pasted.

`Scaling`

Factor that scales the size of the entities being pasted. Scaling is centered at the origin of the global axes of the source.

Dependencies

`Paste Options`

Assigned using *St7SetPasteOptions*.

St7PasteFromSt7ClipboardByAnchors

Pastes the contents of the clipboard (source) into the model (target), similarly to pasting via the **Anchors** and **Adjust** tabs in the GUI.

```
long St7PasteFromSt7ClipboardByAnchors(long uID, long* SourceAnchorType,
     long* SourceAnchorID, long* TargetAnchorType, long* TargetAnchorID,
     double* Rotation, double* Translation, double Scaling)
```

Input Parameters

`uID`

Strand7 model file ID.

`SourceAnchorType[0..2]`

Entity type defining each source anchor point; must be either tyNODE or tyVERTEX.

`SourceAnchorID[0..2]`

Entity number defining each source anchor point. The way the three points are used to define the axis system is explained in the Note below.

`TargetAnchorType[0..2]`

Entity type defining each target anchor point; must be either tyNODE or tyVERTEX.

`TargetAnchorID[0..2]`

Entity number defining each target anchor point. The way the three points are used to define the axis system is explained in the Note below.

`Rotation[0..2]`

Angles (degrees) around the X, Y and Z anchor axes of the source, applied to the entities being pasted. Rotation is applied after the source anchor to target anchor transformation.

`Translation[0..2]`

Translation increments in the length units of the target, in the X, Y and Z global axes of the target, applied to the entities being pasted. Translation is applied after the source anchor to target anchor transformation.

`Scaling`

Factor that scales the size of the entities being pasted. Scaling is centered at the origin of the anchor axes of the source and is applied after the source anchor to target anchor transformation.

Note

If all three anchor numbers are non-zero, the anchor axis system is defined as follows:

- The first point defines the origin;
- The vector from the first point to the second point defines the X axis;
- The third point defines the XY plane;
- The Z axis is normal to the XY plane;
- The Y axis is defined as the cross product of the Z and X axes.

If the first number is non-zero and the other two are both zero, the anchor axis system remains parallel to the global axis system, but its origin is located at the first point.

Dependencies

`Paste Options`

Assigned using *St7SetPasteOptions*.

Tools – Line Definition

Functions in this section create API line definitions. These lines are referenced by other functions, for example, to represent an extrusion target. API line definitions do not correspond to beams in the model, are not visible to the user and no longer exist once the model is closed.

St7DefineLineN2

Creates a new line definition based on two nodes.

```
long St7DefineLineN2(long uID, long NodeNum1, long NodeNum2, long* LineID)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum1`

Start node.

`NodeNum2`

End node.

Output Parameters

`LineID`

Line identifier.

St7DefineLineV2

Creates a new line definition based on two vertices.

```
long St7DefineLineV2(long uID, long VertexNum1, long VertexNum2, long* LineID)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum1`

Start vertex.

`VertexNum2`

End vertex.

Output Parameters

`LineID`

Line identifier.

St7DefineLineNV

Creates a new line definition based on one node and one vertex.

```
long St7DefineLineNV(long uID, long NodeNum, long VertexNum, bool Reversed,
     long* LineID)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Node number.

`VertexNum`

Vertex number.

`Reversed`

If True, line direction is oriented from the vertex to the node.

If False, line direction is oriented from the node to the vertex.

Output Parameters

`LineID`

Line identifier.

St7DefineLineP2

Creates a new line definition using two points in the global XYZ system.

```
long St7DefineLineP2(long uID, double* P1, double* P2, long* LineID)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Start point in global XYZ coordinates.

`P2[0..2]`

End point in global XYZ coordinates.

Output Parameters

`LineID`

Line identifier.

Tools – Plane Definition

Functions in this section create API plane definitions. These planes are referenced by other functions, for example, to represent an extrusion target. API plane definitions do not correspond to plates or coordinate systems in the model, are not visible to the user and no longer exist once the model is closed.

St7DefinePlaneGlobalN

Creates a new plane definition by locating one of the three global XYZ planes at a node.

```
long St7DefinePlaneGlobalN(long uID, long NodeNum, long Plane, long* PlaneID)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Number of a node that lies on the plane being defined.

`Plane`

Plane number, 1, 2 or 3 for the XY, YZ or ZX plane respectively.

Output Parameters

`PlaneID`

Plane identifier.

St7DefinePlaneGlobalV

Creates a new plane definition by locating one of the three global XYZ planes at a vertex.

```
long St7DefinePlaneGlobalV(long uID, long VertexNum, long Plane, long* PlaneID)
```

Input Parameters

`uID`

Strand7 model file ID.

`VertexNum`

Number of a vertex that lies on the plane being defined.

`Plane`

Plane number, 1, 2 or 3 for the XY, YZ or ZX plane respectively.

Output Parameters

`PlaneID`

Plane identifier.

St7DefinePlaneP3

Creates a new plane definition using three points in the global XYZ system.

```
long St7DefinePlaneP3(long uID, double* P1, double* P2, double* P3,
     long* PlaneID)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Point 1 in global XYZ coordinates.

`P2[0..2]`

Point 2 in global XYZ coordinates.

`P3[0..2]`

Point 3 in global XYZ coordinates.

Output Parameters

`PlaneID`

Plane identifier.

St7DefinePlaneUCS

Creates a new plane definition from an existing UCS.

```
long St7DefinePlaneUCS(long uID, long UCSId, long UCSPlane, long* PlaneID)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

UCS ID.

`UCSPlane`

Plane of UCS; one of 1, 2 or 3.

Output Parameters

`PlaneID`

Plane identifier.

Tools – Projection Direction

Functions in this section set the state of the projection direction option, as referenced by later function calls. The state persists until the model is closed, or the state is changed by a subsequent call. The default projection direction, which is used if none of the functions in this chapter have been called, is **Target Normal**.

St7SetProjectDirectionAsSource

Sets the mode of projection to **Source (Plate) Normal** for subsequent uses of projection tools.

```
long St7SetProjectDirectionAsSource(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7SetProjectDirectionAsTarget

Sets the mode of projection to **Target Normal** for subsequent uses of projection tools.

```
long St7SetProjectDirectionAsTarget(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7SetProjectDirectionAsConical

Sets the mode of projection to **Conical** for subsequent uses of projection tools.

```
long St7SetProjectDirectionAsConical(long uID, double* Apex)
```

Input Parameters

`uID`

Strand7 model file ID.

`Apex[0..2]`

Apex of cone.

St7SetProjectDirectionAsParallel

Sets the mode of projection to **Parallel** for subsequent uses of projection tools.

```
long St7SetProjectDirectionAsParallel(long uID, double* P1, double* P2)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Origin point of projection direction.

`P2[0..2]`

Destination point of projection direction.

Tools – General State

Functions in this section set a variety of states as referenced by later function calls. The states persists until the model is closed.

St7SetPropertyIncrement

Sets the **Property Increment** for copy operations.

```
long St7SetPropertyIncrement(long uID, long PropInc)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropInc`

Property increment.

St7SetKeepSelect

Sets the **Keep Selection** flag for applicable tools operations.

```
long St7SetKeepSelect(long uID, bool KeepSelect)
```

Input Parameters

`uID`

Strand7 model file ID.

`KeepSelect`

Keep selected flag.

St7SetCopyFlags

Sets the state of the copy options for applicable tools operations.

```
long St7SetCopyFlags(long uID, long* Flags)
```

Input Parameters

`uID`

Strand7 model file ID.

`Flags[0..5]`

`[ipCopyNodeVertexAttributes]` – either btTrue or btFalse; setting for **Copy Node/Vertex Attributes**.

`[ipCopyElementFaceAttributes]` – either btTrue or btFalse; setting for **Copy Element/Face Attributes**.

`[ipIncrementStringID]` – either btTrue or btFalse; setting for **Increment String Group ID**.

`[ipIncrementClusterID]` – either btTrue or btFalse; setting for **Increment Cluster ID**.

`[ipCreateNewGroup]` – either btTrue or btFalse; setting for **Create New Group for Copy**.

`[ipCopyAxisUCS]` – either btTrue or btFalse; setting for **Create New UCS for Element Axes**.

St7SetExtrudeFlags

Sets the state of the extrude options for applicable tools operations.

```
long St7SetExtrudeFlags(long uID, long* Flags)
```

Input Parameters

`uID`

Strand7 model file ID.

`Flags[0..0]`

`[ipExtrudePlateEdgeAttributes]` – either btTrue or btFalse; setting for **Extrude Plate Edge Attributes**.

St7SetExtrudeTargets

Sets the parameters for the Target options for the extrusion tools.

```
long St7SetExtrudeTargets(long uID, long* Targets)
```

Input Parameters

`uID`

Strand7 model file ID.

`Targets[0..6]`

`[ipExtrudeNodeTarget]` – One of etBeam2, etBeam3, etMasterSlaveLink, etPinnedLink, etRigidLink or etShrinkLink.

`[ipExtrudeNodeTargetOption]` – Additional information about the node extrusion target, depending on the setting of `[ipExtrudeNodeTarget]`.

etBeam2, etBeam3	– Beam property number.
etMasterSlaveLink	– Bitmask of flags for the constrained degrees of freedom.
etPinnedLink	– Not used.
etRigidLink	– Plane of action; one of rlPlaneXYZ, rlPlaneXY, rlPlaneYZ or rlPlaneZX.
etShrinkLink	– Bitmask of flags for the constrained degrees of freedom.

`[ipExtrudeNodeTargetUCS]` – UCS of extruded links. If `[ipExtrudeNodeTargetOption]` is etMasterSlaveLink, any UCS can be used. If `[ipExtrudeNodeTargetOption]` is etRigidLink, a Cartesian UCS must be used.

`[ipExtrudeLinksAsMPL]` – Used by *St7ExtrudeByProjectionToPoint* and *St7ExtrudeByProjectionToAveragePoint* when `[ipExtrudeNodeTargetOption]` is one of etMasterSlaveLink, etPinnedLink or etRigidLink.

btTrue – A single MPL is generated.

btFalse – Multiple 2-node links are generated.

`[ipExtrudeBeamTarget]` – One of etPlateQuad4, etPlateQuad8 or etPlateQuad9.

`[ipExtrudePlateTarget]` – Additional option for the extrusion of plates.

ptFaceAsBrick – Plates are extruded into bricks.

ptEdgeAsPlate – Plate edges are extruded into plates.

`[ipExtrudeShrinkFreedomCase]` – Nonlinear control freedom case when `[ipExtrudeNodeTargetOption]` is etShrinkLink.

St7SetSourceAction

Sets the state of the **Source** option for applicable tools.

```
long St7SetSourceAction(long uID, long SourceAction)
```

Input Parameters

`uID`

Strand7 model file ID.

`SourceAction`

One of saLeave, saDelete, saCopy or saMove.

St7SetPLTarget

Sets the type of entities created by the points and lines tools.

```
long St7SetPLTarget(long uID, long Target, long PropNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Target`

One of plNode, plBeam2, or plBeam3.

`PropNum`

Property number of created beams.

St7DefineEntityCollection

Creates an entity collection from the currently selected entities.

```
long St7DefineEntityCollection(long uID, long* CollectionID)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`CollectionID`

Entity collection identifier.

Dependencies

`Selection`

Elements can be selected using functions in *Entity Selection*.

Tools – Copy

St7CopyByIncrement

Copies selected entities by increment.

```
long St7CopyByIncrement(long uID, double* DXYZ, long UCSId, long NumCopies)
```

Input Parameters

`uID`

Strand7 model file ID.

`DXYZ[0..2]`

Increment.

`UCSId`

UCS in which increment has been provided.

`NumCopies`

Number of copies.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Property Increment`

Assigned using *St7SetPropertyIncrement*.

`Copy Flags`

Assigned using *St7SetCopyFlags*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7CopyByRotation

Copies selected entities by rotation.

```
long St7CopyByRotation(long uID, long UCSId, long Axis, double Angle,
     double* Origin, long NumCopies)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

UCS in which axis of rotation is defined.

`Axis`

Axis of rotation; one of 1, 2 or 3.

`Angle`

Rotation increment (degrees).

`Origin[0..2]`

Rotation origin.

`NumCopies`

Number of copies.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Property Increment`

Assigned using *St7SetPropertyIncrement*.

`Copy Flags`

Assigned using *St7SetCopyFlags*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7CopyByProjectionToLine

Copies selected entities by projection to a predefined line.

```
long St7CopyByProjectionToLine(long uID, long LineID, bool EquiSpace)
```

Input Parameters

`uID`

Strand7 model file ID.

`LineID`

The ID of a line to which the selected entities will be projected. Functions in *Tools – Line Definition* are used to define a line and return the LineID.

`EquiSpace`

Equal spacing between nodes projected to the line.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Property Increment`

Assigned using *St7SetPropertyIncrement*.

`Direction`

Assigned by the most recent call to a function in *Tools – Projection Direction*.

`Copy Flags`

Assigned using *St7SetCopyFlags*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7CopyByProjectionToPlane

Copies selected entities by projection to a predefined plane.

```
long St7CopyByProjectionToPlane(long uID, long PlaneID)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlaneID`

The ID of a plane to which the selected entities will be projected. Functions in *Tools – Plane Definition* are used to define a plane and return the PlaneID.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Property Increment`

Assigned using *St7SetPropertyIncrement*.

`Direction`

Assigned by the most recent call to a function in *Tools – Projection Direction*.

`Copy Flags`

Assigned using *St7SetCopyFlags*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7CopyByProjectionToUCS

Copies selected entities by projection to the specified ordinate on a coordinate system.

```
long St7CopyByProjectionToUCS(long uID, long UCSId, long UCSPlane,
     double Ordinate)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

UCS onto which projection will occur. Cartesian, Cylindrical and Spherical coordinate systems can be used.

`UCSPlane`

Plane of the UCS onto which projection will occur; either 1, 2 or 3 for planes XY, YZ or ZX of Cartesian coordinate systems. Not applicable to Cylindrical and Spherical coordinate systems.

`Ordinate`

For Cartesian systems: The ordinate that locates the plane. For example, the Y ordinate if a ZX plane is used as the projection target.

For Cylindrical and Spherical systems: Radius.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Property Increment`

Assigned using *St7SetPropertyIncrement*.

`Direction`

Assigned by the most recent call to a function in *Tools – Projection Direction*.

`Copy Flags`

Assigned using *St7SetCopyFlags*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7CopyByProjectionToEntityFace

Copies selected entities by projection to the faces of predefined entities.

```
long St7CopyByProjectionToEntityFace(long uID, long CollectionID)
```

Input Parameters

`uID`

Strand7 model file ID.

`CollectionID`

The ID of a collection of entities returned by a previous call to *St7DefineEntityCollection*.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Property Increment`

Assigned using *St7SetPropertyIncrement*.

`Direction`

Assigned by the most recent call to a function in *Tools – Projection Direction*.

`Copy Flags`

Assigned using *St7SetCopyFlags*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7CopyByThickness

Copies selected entities by thickness.

```
long St7CopyByThickness(long uID, double Thickness, long BeamDir,
     long PlateSurface, long FaceSurface, bool UsePlateThickness,
     bool UseFaceThickness)
```

Input Parameters

`uID`

Strand7 model file ID.

`Thickness`

Distance to copy.

`BeamDir`

Direction to copy beams in their principal axis system; either 1 or 2.

`PlateSurface`

Direction to copy plates if UsePlateThickness is True. Either psPlateMinusZ or psPlatePlusZ.

`FaceSurface`

Direction to copy geometry faces if UseFaceThickness is True. Either psPlateMinusZ or psPlatePlusZ.

`UsePlateThickness`

True to base the increment on the thickness of the plate.

`UseFaceThickness`

True to base the increment on the thickness of the geometry faces.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Property Increment`

Assigned using *St7SetPropertyIncrement*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7CopyByMirror

Copies selected entities by reflecting them over a predefined plane.

```
long St7CopyByMirror(long uID, long PlaneID)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlaneID`

The ID of a plane over which the selected entities will be mirrored. Functions in *Tools – Plane Definition* are used to define a plane and return the PlaneID.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Property Increment`

Assigned using *St7SetPropertyIncrement*.

`Copy Flags`

Assigned using *St7SetCopyFlags*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7CopyToAbsolute

Copies selected entities to a fixed ordinate in a given axis of a coordinate system.

```
long St7CopyToAbsolute(long uID, double Value, long UCSId, long Axis)
```

Input Parameters

`uID`

Strand7 model file ID.

`Value`

Value of ordinate adjusted by copy operation.

`UCSId`

The coordinate system into which the copy will occur.

`Axis`

The axis in the coordinate system; one of 1, 2 or 3.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Property Increment`

Assigned using *St7SetPropertyIncrement*.

`Copy Flags`

Assigned using *St7SetCopyFlags*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

Tools – Move

St7MoveByIncrement

Moves selected entities by increment.

```
long St7MoveByIncrement(long uID, double* DXYZ, long UCSId)
```

Input Parameters

`uID`

Strand7 model file ID.

`DXYZ[0..2]`

Increment.

`UCSId`

UCS in which increment has been provided.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7MoveByRotation

Moves selected entities by rotation.

```
long St7MoveByRotation(long uID, long UCSId, long Axis, double Angle,
    double* Origin)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

UCS in which axis of rotation is defined.

`Axis`

Axis of rotation; one of 1, 2 or 3.

`Angle`

Rotation increment (degrees).

`Origin[0..2]`

Rotation origin.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7MoveByProjectionToLine

Moves selected entities by projection to a predefined line.

```
long St7MoveByProjectionToLine(long uID, long LineID, bool EquiSpace)
```

Input Parameters

`uID`

Strand7 model file ID.

`LineID`

The ID of a line to which the selected entities will be projected. Functions in *Tools – Line Definition* are used to define a line and return the LineID.

`EquiSpace`

Equal spacing between nodes projected to the line.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Direction`

Assigned by the most recent call to a function in *Tools – Projection Direction*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7MoveByProjectionToPlane

Moves selected entities by projection to a predefined plane.

```
long St7MoveByProjectionToPlane(long uID, long PlaneID)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlaneID`

The ID of a plane to which the selected entities will be projected. Functions in *Tools – Plane Definition* are used to define a plane and return the PlaneID.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Direction`

Assigned by the most recent call to a function in *Tools – Projection Direction*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7MoveByProjectionToUCS

Moves selected entities by projection to the specified ordinate on a coordinate system.

```
long St7MoveByProjectionToUCS(long uID, long UCSId, long UCSPlane,
     double Ordinate)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

UCS onto which projection will occur. Cartesian, Cylindrical and Spherical coordinate systems can be used.

`UCSPlane`

Plane of the UCS onto which projection will occur; either 1, 2 or 3 for planes XY, YZ or ZX of Cartesian coordinate systems. Not applicable to Cylindrical and Spherical coordinate systems.

`Ordinate`

For Cartesian systems: The ordinate that locates the plane. For example, the Y ordinate if a ZX plane is used as the projection target.

For Cylindrical and Spherical systems: Radius.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Direction`

Assigned by the most recent call to a function in *Tools – Projection Direction*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7MoveByProjectionToEntityFace

Moves selected entities by projection to the faces of predefined entities.

```
long St7MoveByProjectionToEntityFace(long uID, long CollectionID)
```

Input Parameters

`uID`

Strand7 model file ID.

`CollectionID`

The ID of a collection of entities returned by a previous call to *St7DefineEntityCollection*.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Direction`

Assigned by the most recent call to a function in *Tools – Projection Direction*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7MoveByThickness

Moves selected entities by thickness.

```
long St7MoveByThickness(long uID, double Thickness, long BeamDir,
     long PlateSurface, long FaceSurface, bool UsePlateThickness,
     bool UseFaceThickness)
```

Input Parameters

`uID`

Strand7 model file ID.

`Thickness`

Distance to move.

`BeamDir`

Direction to move beams in their principal axis system; either 1 or 2.

`PlateSurface`

Direction to move plates if UsePlateThickness is True. Either psPlateMinusZ or psPlatePlusZ.

`FaceSurface`

Direction to move geometry faces if UseFaceThickness is True. Either psPlateMinusZ or psPlatePlusZ.

`UsePlateThickness`

True to base the increment on the thickness of the plate.

`UseFaceThickness`

True to base the increment on the thickness of the geometry faces.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7MoveByMirror

Moves selected entities by reflecting them over a predefined plane.

```
long St7MoveByMirror(long uID, long PlaneID)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlaneID`

The ID of a plane over which the selected entities will be mirrored. Functions in *Tools – Plane Definition* are used to define a plane and return the PlaneID.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7MoveBySkew

Moves selected entities by a prescribed skew amount.

```
long St7MoveBySkew(long uID, double* Origin, double* Skew, long Axis)
```

Input Parameters

`uID`

Strand7 model file ID.

`Origin[0..2]`

Coordinates of the skew origin, in the global XYZ system.

`Skew[0..2]`

Amount of skew in global X, Y and Z respectively.

`Axis`

The axis in the global XYZ system to be skewed. Either 1, 2 or 3.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7MoveToAbsolute

Moves selected entities to a fixed ordinate in a given axis of a coordinate system.

```
long St7MoveToAbsolute(long uID, double Value, long UCSId, long Axis)
```

Input Parameters

`uID`

Strand7 model file ID.

`Value`

Value of ordinate adjusted by move operation.

`UCSId`

The coordinate system into which the move will occur.

`Axis`

The axis in the coordinate system; one of 1, 2 or 3.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7MoveToUCSIntersection

Moves selected entities to the intersection of two coordinate systems.

```
long St7MoveToUCSIntersection(long uID, long UCSId1, long UCSId2,
     double Ordinate1, double Ordinate2)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId1`

ID of first coordinate system.

`UCSId2`

ID of second coordinate system.

`Ordinate1`

Ordinate in first coordinate system – if UCSId1 is Cartesian this is a Z ordinate. Otherwise, it is the radius.

`Ordinate2`

Ordinate in second coordinate system – if UCSId2 is Cartesian this is a Z ordinate. Otherwise, it is the radius.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7MoveToOriginByPoint

Moves selected entities by shifting a reference point to the origin of a coordinate system.

```
long St7MoveToOriginByPoint(long uID, long UCSId, double* Point)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

The coordinate system specifying the minimum point.

`Point[0..2]`

The reference point that will be located at the origin.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7MoveToOriginMinXYZ

Moves the minimum XYZ point of selected entities to the origin of a coordinate system.

```
long St7MoveToOriginMinXYZ(long uID, long UCSId)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

The coordinate system specifying the minimum point.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7MoveToPlane

Aligns a model with the specified coordinate system.

```
long St7MoveToPlane(long uID, long SourcePlaneID, long TargetPlaneID)
```

Input Parameters

`uID`

Strand7 model file ID.

`SourcePlaneID`

Source PlaneID. Functions in *Tools – Plane Definition* are used to define a plane and return the PlaneID.

`TargetPlaneID`

Target PlaneID. Functions in *Tools – Plane Definition* are used to define a plane and return the PlaneID.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

Tools – Extrude

St7ExtrudeByIncrement

Extrudes selected entities by increment.

```
long St7ExtrudeByIncrement(long uID, double* DXYZ, long UCSId, long NumCopies)
```

Input Parameters

`uID`

Strand7 model file ID.

`DXYZ[0..2]`

Increment.

`UCSId`

UCS in which increment has been provided.

`NumCopies`

Number of repeats.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Property Increment`

Assigned using *St7SetPropertyIncrement*.

`Source Action`

Assigned using *St7SetSourceAction*.

`Extrusion Targets`

Assigned using *St7SetExtrudeTargets*.

`Extrusion Flags`

Assigned using *St7SetExtrudeFlags*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

`Default Group`

Target group for nodes extruded into beams is specified by *St7SetDefaultGroupID*; for other targets the group is inherited.

St7ExtrudeByRotation

Extrudes selected entities by rotation.

```
long St7ExtrudeByRotation(long uID, long UCSId, long Axis, double Angle,
     double* Origin, long NumCopies)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

UCS in which axis of rotation is defined.

`Axis`

Axis of rotation; one of 1, 2 or 3.

`Angle`

Rotation increment (degrees).

`Origin[0..2]`

Rotation origin.

`NumCopies`

Number of repeats.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Property Increment`

Assigned using *St7SetPropertyIncrement*.

`Source Action`

Assigned using *St7SetSourceAction*.

`Extrusion Targets`

Assigned using *St7SetExtrudeTargets*.

`Extrusion Flags`

Assigned using *St7SetExtrudeFlags*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

`Default Group`

Target group for nodes extruded into beams is specified by *St7SetDefaultGroupID*; for other targets the group is inherited.

St7ExtrudeByProjectionToPoint

Extrudes selected entities by projection to a point specified by coordinates.

```
long St7ExtrudeByProjectionToPoint(long uID, double* Point)
```

Input Parameters

`uID`

Strand7 model file ID.

`Point[0..2]`

The point to which the extrusion will occur, in the global XYZ system.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Property Increment`

Assigned using *St7SetPropertyIncrement*.

`Source Action`

Assigned using *St7SetSourceAction*.

`Extrusion Targets`

Assigned using *St7SetExtrudeTargets*.

`Extrusion Flags`

Assigned using *St7SetExtrudeFlags*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

`Default Group`

Target group for nodes extruded into beams is specified by *St7SetDefaultGroupID*; for other targets the group is inherited.

St7ExtrudeByProjectionToAveragePoint

Extrudes selected entities by projection to a point. The point is calculated as the average of all selected nodes and all nodes in the set of selected elements.

```
long St7ExtrudeByProjectionToAveragePoint(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Property Increment`

Assigned using *St7SetPropertyIncrement*.

`Source Action`

Assigned using *St7SetSourceAction*.

`Extrusion Targets`

Assigned using *St7SetExtrudeTargets*.

`Extrusion Flags`

Assigned using *St7SetExtrudeFlags*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

`Default Group`

Target group for nodes extruded into beams is specified by *St7SetDefaultGroupID*; for other targets the group is inherited.

St7ExtrudeByProjectionToLine

Extrudes selected entities by projection to a predefined line.

```
long St7ExtrudeByProjectionToLine(long uID, long LineID, bool EquiSpace)
```

Input Parameters

`uID`

Strand7 model file ID.

`LineID`

The ID of a line to which the selected entities will be projected. Functions in *Tools – Line Definition* are used to define a line and return the LineID.

`EquiSpace`

True to produce equal spacing between nodes projected to the line.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Property Increment`

Assigned using *St7SetPropertyIncrement*.

`Direction`

Assigned by the most recent call to a function in *Tools – Projection Direction*.

`Source Action`

Assigned using *St7SetSourceAction*.

`Extrusion Targets`

Assigned using *St7SetExtrudeTargets*.

`Extrusion Flags`

Assigned using *St7SetExtrudeFlags*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

`Default Group`

Target group for nodes extruded into beams is specified by *St7SetDefaultGroupID*; for other targets the group is inherited.

St7ExtrudeByProjectionToPlane

Extrudes selected entities by projection to a predefined plane.

```
long St7ExtrudeByProjectionToPlane(long uID, long PlaneID)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlaneID`

The ID of a plane to which the selected entities will be projected. Functions in *Tools – Plane Definition* are used to define a plane and return the PlaneID.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Property Increment`

Assigned using *St7SetPropertyIncrement*.

`Direction`

Assigned by the most recent call to a function in *Tools – Projection Direction*.

`Source Action`

Assigned using *St7SetSourceAction*.

`Extrusion Targets`

Assigned using *St7SetExtrudeTargets*.

`Extrusion Flags`

Assigned using *St7SetExtrudeFlags*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

`Default Group`

Target group for nodes extruded into beams is specified by *St7SetDefaultGroupID*; for other targets the group is inherited.

St7ExtrudeByProjectionToUCS

Extrudes selected entities by projection to the specified ordinate on a coordinate system.

```
long St7ExtrudeByProjectionToUCS(long uID, long UCSId, long UCSPlane,
     double Ordinate)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

UCS onto which projection will occur. Cartesian, Cylindrical and Spherical coordinate systems can be used.

`UCSPlane`

Plane of the UCS onto which projection will occur; either 1, 2 or 3 for planes XY, YZ or ZX of Cartesian coordinate systems. Not applicable to Cylindrical and Spherical coordinate systems.

`Ordinate`

For Cartesian systems: The ordinate that locates the plane. For example, the Y ordinate if a ZX plane is used as the projection target.

For Cylindrical and Spherical systems: Radius.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Property Increment`

Assigned using *St7SetPropertyIncrement*.

`Direction`

Assigned by the most recent call to a function in *Tools – Projection Direction*.

`Source Action`

Assigned using *St7SetSourceAction*.

`Extrusion Targets`

Assigned using *St7SetExtrudeTargets*.

`Extrusion Flags`

Assigned using *St7SetExtrudeFlags*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

`Default Group`

Target group for nodes extruded into beams is specified by *St7SetDefaultGroupID*; for other targets the group is inherited.

St7ExtrudeByProjectionToEntityFace

Extrudes selected entities by projection to the faces of predefined entities.

```
long St7ExtrudeByProjectionToEntityFace(long uID, long CollectionID)
```

Input Parameters

`uID`

Strand7 model file ID.

`CollectionID`

The ID of a collection of entities returned by a previous call to *St7DefineEntityCollection*.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Property Increment`

Assigned using *St7SetPropertyIncrement*.

`Direction`

Assigned by the most recent call to a function in *Tools – Projection Direction*.

`Source Action`

Assigned using *St7SetSourceAction*.

`Extrusion Targets`

Assigned using *St7SetExtrudeTargets*.

`Extrusion Flags`

Assigned using *St7SetExtrudeFlags*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

`Default Group`

Target group for nodes extruded into beams is specified by *St7SetDefaultGroupID*; for other targets the group is inherited.

St7ExtrudeByThickness

Extrudes selected entities by thickness.

```
long St7ExtrudeByThickness(long uID, double Thickness, long BeamDir,
     long PlateSurface, bool UsePlateThickness, bool SourceMidPlane)
```

Input Parameters

`uID`

Strand7 model file ID.

`Thickness`

Distance to extrude.

`BeamDir`

Direction to extrude beams in their principal axis system; either 1 or 2.

`PlateSurface`

Direction to extrude plates if UsePlateThickness is True. Either psPlateMinusZ or psPlatePlusZ.

`UsePlateThickness`

True to base the increment on the thickness of the plate.

`SourceMidPlane`

True to extrude half the thickness on each side, producing a single element centred on the source's mid-plane.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Property Increment`

Assigned using *St7SetPropertyIncrement*.

`Extrusion Targets`

Assigned using *St7SetExtrudeTargets*.

`Extrusion Flags`

Assigned using *St7SetExtrudeFlags*.

`Source Action`

Assigned using *St7SetSourceAction*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7ExtrudeByLine

Extrudes selected entities along a line.

```
long St7ExtrudeByLine(long uID, long CollectionID, long Divisions,
     long Direction, double RotationAngle, double RadialScale)
```

Input Parameters

`uID`

Strand7 model file ID.

`CollectionID`

The ID of a collection of entities returned by a previous call to *St7DefineEntityCollection* that defines the line of extrusion. The collection must contain beams that are joined end-to-end.

`Divisions`

The number of evenly spaces divisions along the line, or 0 to divide the extrusions at the nodes as defined in the line definition.

`Direction`

Either ldAuto to extrude along the line starting from the end closest to the selected elements, or ldReversed to extrude the other direction.

`RotationAngle`

The source entities are progressively rotated about the axis of the line of extrusion such that the total rotation at the end of the extrusion equals the specified rotation angle (degrees). A zero rotation angle produces a parallel extrusion whereas a non-zero rotation angle produces a helical extrusion.

`RadialScale`

The source entities are progressively scaled radially with respect to the axis of the line of extrusion such that at the end of the extrusion they are scaled radially by the specified radial scale. A radial scale of 1.0 produces a parallel extrution whereas a non-unit radial scale produces a tapered extrusion.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Property Increment`

Assigned using *St7SetPropertyIncrement*.

`Source Action`

Assigned using *St7SetSourceAction*.

`Extrusion Targets`

Assigned using *St7SetExtrudeTargets*.

`Extrusion Flags`

Assigned using *St7SetExtrudeFlags*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

`Default Group`

Target group for nodes extruded into beams is specified by *St7SetDefaultGroupID*; for other targets the group is inherited.

St7ExtrudeToAbsolute

Extrudes selected entities to a fixed ordinate in a given axis of a coordinate system.

```
long St7ExtrudeToAbsolute(long uID, double Value, long UCSId, long Axis)
```

Input Parameters

`uID`

Strand7 model file ID.

`Value`

Value of ordinate adjusted by extrude operation.

`UCSId`

The coordinate system into which the extrusion will occur.

`Axis`

The axis in the coordinate system; one of 1, 2 or 3.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Property Increment`

Assigned using *St7SetPropertyIncrement*.

`Source Action`

Assigned using *St7SetSourceAction*.

`Extrusion Targets`

Assigned using *St7SetExtrudeTargets*.

`Extrusion Flags`

Assigned using *St7SetExtrudeFlags*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

`Default Group`

Target group for nodes extruded into beams is specified by *St7SetDefaultGroupID*; for other targets the group is inherited.

Tools – Scale

St7ScaleByCartesianUCS

Scales selected entities by a Cartesian coordinate system.

```
long St7ScaleByCartesianUCS(long uID, long UCSId, long ScaleAbout,
     double* Factors, double* Point)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

The Cartesian coordinate system in which to scale.

`ScaleAbout`

One of saMiddle, saOrigin or saPoint.

`Factors[0..2]`

Scaling factors in X, Y and Z respectively.

`Point[0..2]`

Global Cartesian coordinates of point about which to scale. Only applicable if ScaleAbout is saPoint.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7ScaleByCylindricalUCS

Scales selected entities by a cylindrical coordinate system.

```
long St7ScaleByCylindricalUCS(long uID, long UCSId, long ScaleAbout,
     double* Factors, double* Point, double AngularCentre)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

The cylindrical coordinate system in which to scale.

`ScaleAbout`

One of saMiddle, saOrigin or saPoint.

`Factors[0..2]`

Scaling factors in X, Y and Z respectively.

`Point[0..2]`

Global Cartesian coordinates of point about which to scale. Only applicable if ScaleAbout is saPoint.

`AngularCentre`

Angular origin about which to scale the Theta ordinates (degrees).

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7ScaleBySphericalUCS

Scales the radius of selected entities in a spherical coordinate system.

```
long St7ScaleBySphericalUCS(long uID, long UCSId, double Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

The spherical coordinate system in which to scale.

`Factor`

Factor by which to scale the radius.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7ScaleByToroidalUCS

Scales the radius of selected entities in a toroidal coordinate system.

```
long St7ScaleByToroidalUCS(long uID, long UCSId, double Factor)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

The toroidal system in which to scale.

`Factor`

Factor by which to scale the radius.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7ScaleByTaper

Scales the radius of selected node and elements in a spherical coordinate system.

```
long St7ScaleByTaper(long uID, long UCSId, long LineID, long Axis, double Scale1,
     double Scale2)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

The ID of a Cartesian coordinate system in which to scale.

`LineID`

The ID of a line along which the selected nodes and elements will be tapered. Functions in *Tools – Line Definition* are used to define a line and return the LineID.

`Axis`

Axis to taper: 1, 2 or 3 for X, Y or Z.

`Scale1`

Taper factor at start of line.

`Scale2`

Taper factor at end of line.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

Tools – Geometry

St7GraftEdgesToFaces

Grafts edges to geometry faces.

```
long St7GraftEdgesToFaces(long uID, long DistanceType, double Distance)
```

Input Parameters

`uID`

Strand7 model file ID.

`DistanceType`

Specifies how the Distance is to be interpreted; either ztRelative or ztAbsolute.

`Distance`

The graft tolerance. Edges that are closer to geometry faces than this tolerance will be grafted onto them.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

Global Parameters

`ivSeamsAdded, ivIntersectionsFound, ivFacesChanged, ivFacesCreated, ivFacesFailed.`

St7IntersectEdges

Intersects the edges of geometry faces placing a vertex at the intersection point.

```
long St7IntersectEdges(long uID, long DistanceType, double Distance,
     bool SplitFaces)
```

Input Parameters

`uID`

Strand7 model file ID.

`DistanceType`

Specifies how the Distance is to be interpreted; either ztRelative or ztAbsolute.

`Distance`

The edge tolerance. Edges that are closer than this tolerance will be split.

`SplitFaces`

True to split faces along their intersection.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

Global Parameters

`ivSeamsAdded, ivIntersectionsFound, ivFacesChanged, ivFacesCreated, ivFacesFailed.`

St7MorphEdges

Morphs the selected edges of geometry faces.

```
long St7MorphEdges(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

Global Parameters

`ivEdgesMorphed.`

St7SubdivideEdges

Divides selected edges into the specified number of equal segments by inserting evenly spaced vertices.

```
long St7SubdivideEdges(long uID, long Divisions, long VertexType)
```

Input Parameters

`uID`

Strand7 model file ID.

`Divisions`

Number of segments to divide edges into.

`VertexType`

Either vtFree or vtFixed.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7SplitFaceByVertices

Splits a geometry face between pairs of vertices.

```
long St7SplitFaceByVertices(long uID, long NumVertexSets, long* VertexSetData)
```

Input Parameters

`uID`

Strand7 model file ID.

`NumVertexSets`

Number of vertex pairs to perform split operations.

`VertexSetData[0..3*NumVertexSets-1]`

For each of the `iVertexSet` vertex pairs to split,

`[3*iVertexSet]` – First vertex.

`[3*iVertexSet+1]` – Second vertex.

`[3*iVertexSet+2]` – Split direction. Use 0 for the shortest distance split, or a higher number to select an alternative for faces where multiple splits are possible.

Dependencies

`Keep Selected`

Assigned using *St7SetKeepSelect*.

Global Parameters

`ivSeamsAdded, ivIntersectionsFound, ivFacesChanged, ivFacesCreated, ivFacesFailed.`

St7SplitFaceByPlane

Splits a geometry face at a plane.

```
long St7SplitFaceByPlane(long uID, long PlaneID, long NumCutFaces,
     long NumRepeats, double Increment)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlaneID`

The ID of a plane at which to cut faces. Functions in *Tools – Plane Definition* are used to define a plane and return the PlaneID.

`NumCutFaces`

Number of faces to create at the cut. Either 0, 1 or 2. If two faces are requested, the generated faces will have opposing normals.

`NumRepeats`

Number of repeated cuts to make using parallel planes offset from that specified by PlaneID.

`Increment`

Offset increment for repeated planes, in the positive Z direction from the plane specified by PlaneID.

Dependencies

`Keep Selected`

Assigned using *St7SetKeepSelect*.

`Default Group`

Target group for the cut faces is specified by *St7SetDefaultGroupID*.

Global Parameters

`ivSeamsAdded, ivIntersectionsFound, ivFacesChanged, ivFacesCreated, ivFacesFailed.`

St7FaceFromPlate

Converts one or more plate elements to geometric faces.

```
long St7FaceFromPlate(long uID, bool NodeAttribToVertices,
     bool PlateAttribToFaces, bool CircularFaceEdges)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeAttribToVertices`

True to pass on **Node Attributes to Vertices**.

`PlateAttribToFaces`

True to pass on **Plate Attributes to Faces**.

`CircularFaceEdges`

True to create **Circular Face Edges**, False to create **Quadratic Face Edges**. Note that circular face edges are inferred from mid-side nodes for singly curved plates only.

Dependencies

`Selection`

Plates can be selected using functions in *Entity Selection*.

`Source Action`

Erase or keep plates, depending on *St7SetSourceAction*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

Global Parameters

`ivFacesCreated, ivTessellationsFailed.`

St7FaceFromBeamPolygon

Converts one or more beam polygons to geometric faces.

```
long St7FaceFromBeamPolygon(long uID, long FaceNum, long PropNum, double EdgeTol,
    bool BeamPropAsLoop, bool BeamGroupAsLoop)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number of target face, or 0 to project to average plane.

`PropNum`

Plate property number for new faces.

`EdgeTol`

Angular tolerance between adjacent polygon edges (degrees). Edges within tolerance may be smoothed via curve fitting.

`BeamPropAsLoop`

True to operate on each beam property separately.

`BeamGroupAsLoop`

True to operate on each group separately.

Dependencies

`Selection`

Beams can be selected using functions in *Entity Selection*.

`Source Action`

Erase or keep beams, depending on *St7SetSourceAction*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

`Default Group`

Target group for the faces is specified by *St7SetDefaultGroupID*.

Global Parameters

`ivFacesCreated, ivTessellationsFailed.`

St7FaceFromCavity

Converts one or more geometry face cavity loops to geometric faces.

```
long St7FaceFromCavity(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

Dependencies

`Selection`

Cavity loops can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

Global Parameters

`ivFacesCreated, ivTessellationsFailed, ivLoopsDeleted.`

St7MidPlaneThinSolids

Extracts a mid-plane geometry from thin solids defined by a collection of faces.

```
long St7MidPlaneThinSolids(long uID, double NormalsTol)
```

Input Parameters

`uID`

Strand7 model file ID.

`NormalsTol`

Angular tolerance (degrees) – connected faces with angles between normals below this tolerance are considered as being part of the same mid-planable surface. The edge of the mid-planed surface occurs where a normal between adjacent faces exceeds this value. The normals are checked only at the vertices of the edge.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Source Action`

Erase or keep the faces that define the solid, depending on *St7SetSourceAction*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

Global Parameters

`ivFacesCreated, ivFacesDeleted`

St7RebuildFaces

Redefines the parametric space of NURBS geometry faces to improve mesh quality using parameters determined by Strand7.

```
long St7RebuildFaces(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

Global Parameters

`ivFacesChanged, ivFacesFailed.`

St7RebuildFacesUV

Redefines the parametric space of NURBS geometry faces to improve mesh quality using specified parameters.

```
long St7RebuildFacesUV(long uID, long DegreeU, long DegreeV, long ControlPointsU,
    long ControlPointsV)
```

Input Parameters

`uID`

Strand7 model file ID.

`DegreeU`

Degree of the surface defining polynomial in the u direction.

`DegreeV`

Degree of the surface defining polynomial in the v direction.

`ControlPointsU`

Number of control points in the u direction.

`ControlPointsV`

Number of control points in the v direction.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

Global Parameters

`ivFacesChanged, ivFacesFailed.`

St7ConvertToNURBS

Converts the selected geometry faces to NURBS.

```
long St7ConvertToNURBS(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

Global Parameters

`ivFacesChanged, ivFacesFailed.`

St7DeleteCavityLoops

Delete all the cavity loops within the selected faces.

```
long St7DeleteCavityLoops(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

Dependencies

`Selection`

Faces can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

Global Parameters

`ivFacesChanged, ivLoopsDeleted.`

St7DetachFaces

Detach selected geometry faces.

```
long St7DetachFaces(long uID, long DetachMode)
```

Input Parameters

`uID`

Strand7 model file ID.

`DetachMode`

Either dmDetachIndividual for **Individual faces**, dmDetachAsCluster for **Clusters of faces** or dmDetachGroups for **Clusters of groups**.

Dependencies

`Selection`

Faces can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

Global Parameters

`ivFacesChanged.`

Tools – Mesh

St7SurfaceMesh

Performs a surface meshing operation based on the geometry included in the specified Strand7 model.

```
long St7SurfaceMesh(long uID, long* Integers, double* Doubles, long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`Integers[0..10]`

`[ipSurfaceMeshMode]` – Meshing mode; either mmAuto or mmCustom.

`[ipSurfaceMeshSizeMode]` – Mesh size option; either smPercentage or smAbsolute.

`[ipSurfaceMeshTargetNodes]` – Number of nodes in target element; one of 3, 4, 6 or 8.

`[ipSurfaceMeshTargetPropertyID]` – Element property definition; one of -1 to use the face property, 0 to use the face number or >0 to use a constant property.

`[ipSurfaceMeshAutoCreateProperties]` – Create properties as needed; either btTrue or btFalse.

`[ipSurfaceMeshMinEdgesPerCircle]` – Minimum number of edges per circular edge.

`[ipSurfaceMeshApplyTransitioning]` – Apply edge transitioning when placing boundary nodes; either btTrue or btFalse.

`[ipSurfaceMeshApplySurfaceCurvature]` – Consider surface curvature when placing boundary nodes; either btTrue or btFalse.

`[ipSurfaceMeshAllowUserStop]` – Allow the user to terminate the meshing process; either btTrue or btFalse.

`[ipSurfaceMeshConsiderNearVertex]` – Allow automesher to base element size on vertices near to, but not on, a surface; either btTrue or btFalse.

`[ipSurfaceMeshSelectedFaces]` – btTrue to mesh only the selected faces, otherwise all faces are meshed.

`Doubles[0..3]`

`[ipSurfaceMeshSize]` – Mesh size, scaled based on `Integers[ipSurfaceMeshSizeMode]`.

`[ipSurfaceMeshLengthRatio]` – Maximum allowable ratio between the largest and smallest edge on each face.

`[ipSurfaceMeshMaximumIncrease]` – Rate of increase in edge length between neighbouring elements, in the range 0 to 1.

`[ipSurfaceMeshOnEdgesLongerThan]` – Minimum curve length on which the **Min Edges per Circle** parameter is to be used.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

Dependencies

`Selection`

Faces can be selected using functions in *Entity Selection*. Only applicable if Integers[ipSurfaceMeshSelectedFaces] is btTrue.

Global Parameters

`ivFacesMeshed, ivFacesPartiallyMeshed, ivFacesNotMeshed.`

St7SolidTetMesh

Performs a solid meshing operation based on the surface mesh definitions included in the specified Strand7 model. Surface mesh definitions can be created using the *St7SurfaceMesh* function, but may also be created by other means.

```
long St7SolidTetMesh(long uID, long* Integers, long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`Integers[0..12]`

`[ipTetraMeshSize]` – Mesh size control for internal elements; one of msFine, msMedium or msCoarse.

`[ipTetraMeshProperty]` – Brick property number.

`[ipTetraMeshInc]` – Brick property number increment for separate solid parts.

`[ipTetraMesh10]` – Mesh using Tetra10 elements; either btTrue or btFalse.

`[ipTetraMeshGroupsAsSolids]` – Mesh groups as solids; either btTrue or btFalse.

`[ipTetraMeshSmooth]` – Smooth tetra elements after meshing; either btTrue or btFalse.

`[ipTetraMeshAutoCreateProperties]` – Create brick properties as needed; either btTrue or btFalse.

`[ipTetraMeshDeletePlates]` – Delete surface plates after meshing; either btTrue or btFalse.

`[ipTetraMeshMultiBodyOption]` – Action when multiple bodies are detected; one of mbCancelMeshing, mbCavity or mbSeparateSolids.

`[ipTetraMeshAllowUserStop]` – Allow the user to terminate the meshing process; either btTrue or btFalse.

`[ipTetraMeshCheckSelfIntersect]` – Check for initial self intersections in the surface plate mesh; either btTrue or btFalse.

`[ipTetraMeshZipOption]` – Zip option for adjacent solids; one of tmAutoZipMultipleSolids, tmUserZipMultipleSolids or tmDontZipMultipleSolids.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

Global Parameters

`ivSolidsMeshed, ivSolidsPartiallyMeshed, ivSolidsNotMeshed.`

St7DirectSolidTetMesh

Performs a solid meshing operation based on the geometry included in the specified Strand7 model. Effectively, the function combines the functionality provided by a call to *St7SurfaceMesh* followed by a call *St7SolidTetMesh* to produce a solid mesh directly from the geometry, bypassing the explicit generation of surface plate elements.

```
long St7DirectSolidTetMesh(long uID, long* Integers, double* Doubles, long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`Integers[0..12]`

`[ipDirectTetraMeshMode]` – Meshing mode; either mmAuto or mmCustom.

`[ipDirectTetraMeshSizeMode]` – Mesh size option; either smPercentage or smAbsolute.

`[ipDirectTetraMinEdgesPerCircle]` – Minimum number of edges per circular edge.

`[ipDirectTetraApplyTransitioning]` – Apply edge transitioning when placing boundary nodes; either btTrue or btFalse.

`[ipDirectTetraApplySurfaceCurvature]` – Consider surface curvature when placing boundary nodes; either btTrue or btFalse.

`[ipDirectTetraAllowUserStop]` – Allow the user to terminate the meshing process; either btTrue or btFalse.

`[ipDirectTetraConsiderNearVertex]` – Allow automesher to base element size on vertices near to, but not on, a surface; either btTrue or btFalse.

`[ipDirectTetraMeshSelectedGroups]` – btTrue to mesh only the groups with at least one selected face, otherwise all groups are meshed.

`[ipDirectTetraMeshSize]` – Mesh size control for internal elements; one of msFine, msMedium or msCoarse.

`[ipDirectTetraMesh10]` – Mesh using Tetra10 elements; either btTrue or btFalse.

`[ipDirectTetraMeshSmooth]` – Smooth tetra elements after meshing; either btTrue or btFalse.

`[ipDirectTetraAutoCreateProperties]` – Create brick properties as needed; either btTrue or btFalse.

`[ipDirectMeshZipOption]` – Zip option for adjacent solids; one of tmAutoZipMultipleSolids, tmUserZipMultipleSolids or tmDontZipMultipleSolids.

`Doubles[0..3]`

`[ipSurfaceMeshSize]` – Mesh size, scaled based on `Integers[ipDirectTetraMeshSizeMode]`.

`[ipSurfaceMeshLengthRatio]` – Maximum allowable ratio between the largest and smallest edge on each face.

`[ipSurfaceMeshMaximumIncrease]` – Rate of increase in edge length between neighbouring elements, in the range 0 to 1.

`[ipSurfaceMeshOnEdgesLongerThan]` – Minimum curve length on which the **Min Edges per Circle** parameter is to be used.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

Dependencies

`Selection`

Faces can be selected using functions in *Entity Selection*. Only applicable if Integers[ipDirectTetraMeshSelectedGroups] is btTrue.

Global Parameters

`ivFacesMeshed, ivFacesPartiallyMeshed, ivFacesNotMeshed, ivSolidsMeshed, ivSolidsPartiallyMeshed, ivSolidsNotMeshed.`

St7MeshFromLoops

Performs a surface meshing operation on a single face. The definition of this face is specified explicitly using the array inputs for this function.

```
long St7MeshFromLoops(long uID, long* Integers, double* Doubles, long* Loops,
     double* Points, long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`Integers[0..3]`

`[ipMeshTargetNodes]` – Number of nodes in the target element; one of 3, 4, 6 or 8.

`[ipMeshTargetPropertyID]` – Plate property number for new elements.

`[ipMeshUCSID]` – UCS ID number onto which the polygon is projected.

`[ipMeshGroupID]` – Group ID number for new elements.

`Doubles[0..0]`

`[ipMeshPositionUCS]` – Location of the elements on the UCS 3 axis.

`Loops[..]`

`[0]` – the total number of loops in the polygon.

`[1]` – the number of points in the first loop in the polygon. This loop is always the outer loop.

`[2..1+Loops[1]]` – a list of point indices defining the first loop.

`[2+Loops[1]]` – the number of points in the second loop of the polygon.

Then recursively, where `Loops[k]` contains the number of points in the i^{th} loop,

`[k+1..k+Loops[k]]` – contains a list of point indices defining the i^{th} loop

`[k+Loops[k]+1]` contains the number of points in the $(i+1)^{th}$ loop.

`Points[..]`

A list of the XY coordinates for the polygon points, with the X and Y coordinates stored contiguously.

`[2*j-2]` – the X coordinate of point j.

`[2*j-1]` – the Y coordinate of point j.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

St7Subdivide

Subdivides selected elements.

```
long St7Subdivide(long uID, long DivsA, long DivsB, long DivsC, long PlateTarget,
     long BrickTarget)
```

Input Parameters

`uID`

Strand7 model file ID.

`DivsA`

Number of subdivisions in the A direction. Applies to beams, plates and bricks.

`DivsB`

Number of subdivisions in the B direction. Applies to plates and bricks.

`DivsC`

Number of subdivisions in the C direction. Applies to bricks.

`PlateTarget`

One of stPlateTri3, stPlateTri6, stPlateQuad4, stPlateQuad8, stPlateQuad9, stPlateSource, stPlateTri or stPlateQuad.

`BrickTarget`

One of stBrickTetra4, stBrickTetra10, stBrickWedge6, stBrickWedge15, stBrickHexa8, stBrickHexa16, stBrickHexa20, stBrickSource, stBrickTetra, stBrickWedge or stBrickHexa.

St7Grade

Grades selected elements. See *Grade Types* for additional information.

```
long St7Grade(long uID, long GradeType, double GradeRatio)
```

Input Parameters

`uID`

Strand7 model file ID.

`GradeType`

One of gt1x2Grade, gt1x2TriGrade, gt1x3Grade, gt2x3Grade, gt2x3TriGrade, gtQuarterQuadGrade, gtQuarterCircleCut, gtQuarterAnnulusCut, gtFullQuarterCircleCut, gtTriGrade2, gtTriGrade1, gtTriGrade3, gt2x4Grade, gtBrickCornerGrade, gtQuadTriGrade1, gtTriGrade5, gtQuadCutOut, gtTriGrade4, gtFullQuarterCircleGrade or gtQuadGradeTri.

`GradeRatio`

Ratio of the grade for applicable grade types.

Dependencies

`Selection`

Elements can be selected using functions in *Entity Selection*.

St7CutElementsByLine

Cuts selected beams and plates along a line.

```
long St7CutElementsByLine(long uID, long LineID, long EdgeTol, long BeamPropNum,
     long PlatePropNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`LineID`

The ID of a line at which the selected elements will be cut. Functions in *Tools – Line Definition* are used to define a line and return the LineID.

`EdgeTol`

Tolerance value from 0 to 40, to avoid creating nearly collapsed elements. Defines how close a cut can be to the end of an edge.

`BeamPropNum`

The property number of beams created on cut plate edges, or a negative number to not create beams.

`PlatePropNum`

The property number of plates created on cut brick faces edges, or a negative number to not create plates.

Dependencies

`Selection`

Beam and plate elements can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7CutElementsByPlane

Cuts selected beams and plates through a plane.

```
long St7CutElementsByPlane(long uID, long PlaneID, long EdgeTol,
     long BeamPropNum, long PlatePropNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlaneID`

The ID of a plane at which to cut beams and plates. Functions in *Tools – Plane Definition* are used to define a plane and return the PlaneID.

`EdgeTol`

Tolerance value from 0 to 40, to avoid creating nearly collapsed elements. Defines how close a cut can be to the end of an edge.

`BeamPropNum`

The property number of beams created on cut plate edges, or a negative number to not create beams.

`PlatePropNum`

The property number of plates created on cut brick faces edges, or a negative number to not create plates.

Dependencies

`Selection`

Beam and plate elements can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7CutElementsByUCS

Cuts selected beams and plates at a given radius.

```
long St7CutElementsByUCS(long uID, long UCSId, long EdgeTol, long BeamPropNum,
     long PlatePropNum, double Radius)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

The ID of a cylindrical or spherical UCS in which to cut the beams and plates.

`EdgeTol`

Tolerance value from 0 to 40, to avoid creating nearly collapsed elements. Defines how close a cut can be to the end of an edge.

`BeamPropNum`

The property number of beams created on cut plate edges, or a negative number to not create beams.

`PlatePropNum`

The property number of plates created on cut brick faces edges, or a negative number to not create plates.

`Radius`

The radius at which to cut elements.

Dependencies

`Selection`

Beam and plate elements can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7SplitBeams

Splits the selected beams by a defined ratio.

```
long St7SplitBeams(long uID, double SplitRatio, long SplitType)
```

Input Parameters

`uID`

Strand7 model file ID.

`SplitRatio`

Ratio of the split between ends 1 and 2, from 0.0 to 1.0.

`SplitType`

Property ID for the split portion of the beam. Set -1 to assign the ID of the parent beam.

Dependencies

`Selection`

Beams can be selected using functions in *Entity Selection*.

St7SubdivideBeams

Subdivides the selected beams so they are as close as possible to the target length.

```
long St7SubdivideBeams(long uID, double Length)
```

Input Parameters

`uID`

Strand7 model file ID.

`Length`

Target length of beams.

Dependencies

`Selection`

Beams can be selected using functions in *Entity Selection*.

St7InterpolateBeamSections

Transitions selected beams from one property to another by interpolating section dimensions. Creates new beam properties if required.

```
long St7InterpolateBeamSections(long uID, long PropNum1, long PropNum2,
     long Divisions)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum1`

Property number at beam end 1.

`PropNum2`

Property number at beam end 2.

`Divisions`

Number of divisions.

Dependencies

`Selection`

Beams can be selected using functions in *Entity Selection*.

St7IntersectBeamsAndLinks

Intersects selected beams and links with selected beams, plate edges, brick edges and links. Nodes are created at found intersections, and the selected beams and links can optionally be split where these intersections are found.

```
long St7IntersectBeamsAndLinks(long uID, double MaxGap, double MinAngle,
     bool SplitBeams, bool SplitLinks, bool ConsiderEdgeMidsideNode)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxGap`

Distance between beams or plate edges to look for an intersection.

`MinAngle`

Nearly parallel beams are not intersected; up to the angular tolerance specified here (degrees).

`SplitBeams`

True to split beams at interestions.

`SplitLinks`

True to split links at interestions. Master-Slave, Pinned, Rigid, Shrink and Two-Point links can be split. Other link types can be used to find intersection points but will not themselves be split.

`ConsiderEdgeMidsideNode`

True to consider the midside node of quadratic edges.

Dependencies

`Selection`

Beams can be selected using functions in *Entity Selection*.

St7LoftBeams

Generates cross beams and/or surfaces of plates across a set of lofting planes.

```
long St7LoftBeams(long uID, long CrossBeamPropNum, long PlatePropNum,
     long NumSteps, long NumSubSteps, bool MakeCrossBeams, bool MakePlates)
```

Input Parameters

`uID`

Strand7 model file ID.

`CrossBeamPropNum`

Beam property number for cross beams.

`PlatePropNum`

Plate property number for plates.

`NumSteps`

Number of steps in which to divide lofting beams.

`NumSubSteps`

Number of lofting plates to put in each step.

`MakeCrossBeams`

True to create lofting cross beams.

`MakePlates`

True to create lofting plates.

Dependencies

`Selection`

Beams can be selected using functions in *Entity Selection*.

St7SliceOnPlane

Cuts selected beams that pass through a plane, and inserts beams on plate edges that lie on the plane.

```
long St7SliceOnPlane(long uID, long PlaneID, long PropNum, long NumRepeats,
     double Increment, bool DoBeams, bool DoPlates)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlaneID`

The ID of a plane at which to slice. Functions in *Tools – Plane Definition* are used to define a plane and return the PlaneID.

`PropNum`

Property number of beam, if beams are being generated.

`NumRepeats`

Number of slices to make.

`Increment`

Distance between each slice.

`DoBeams`

True to cut selected beams if they pass through the slice plane.

`DoPlates`

True to create beams on selected plates that pass through the cutting plane.

Dependencies

`Selection`

Beams can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7FilletPlates

Replaces a corner between selected plates with filleted Quad8 elements.

```
long St7FilletPlates(long uID, double Radius, bool StitchPlates)
```

Input Parameters

`uID`

Strand7 model file ID.

`Radius`

Radius of the fillet to apply.

`StitchPlates`

True to modify select plates so they connect to the fillet plates.

Dependencies

`Selection`

Beams can be selected using functions in *Entity Selection*.

St7MidPlanePlateProjection

Generates a midplane mesh between two surfaces of selected plates. Two separate sets of plates should be selected prior to this call.

```
long St7MidPlanePlateProjection(long uID, long PlateNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

The number of a plate on one of the selected sets; this is the master plate. The mid-plane mesh will be generated by projecting the set of plates containing the master plate, halfway towards the other set of plates.

Dependencies

`Selection`

Plates can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7RepairTri3Mesh

Repairs meshes of selected Tri3 plate elements by removing sliver and slit elements that have internal angles less than a set angular tolerance. The tool is typically used to improve the geometry of imported STL files, particularly when these need to be solid automeshed into tetrahedral elements.

```
long St7RepairTri3Mesh(long uID, double MaxAngle)
```

Input Parameters

`uID`

Strand7 model file ID.

`MaxAngle`

The maximum angle (degrees) allowed at the corners of a Tri3 element. Elements with at least one angle below this value will be collapsed and removed. Surrounding elements will then be adjusted and/or split to maintain the closed topology of the mesh.

Dependencies

`Selection`

Plates can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7DetachElements

Detaches the selected cluster of elements from the rest of the mesh by duplicating nodes at the boundary and optionally connecting the duplicated nodes with links or beams.

```
long St7DetachElements(long uID, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Integers[0..5]`

[ipDetachConnectionType] – One of ctNone, ctMasterSlaveLink or ctBeam2.

[ipDetachUCSId] – UCSId of the coordinate system in which Master-Slave links are defined. Only applicable if [ipDetachConnectionType] is ctMasterSlaveLink.

[ipDetachDoFBits] – Bitmask of flags for six degrees of freedom if Master-Slave links are created. For example, to couple DX, DY and RZ, set ConstraintBits = 1+2+32 = 35.

[ipDetachPropNum] – Property number of beams created at boundary.

[ipDetachUseElementGroup] – btTrue to assign the connecting links or beams to the same group as the detached element; btFalse to assign the connecting links or beams to the default group.

[ipDetachDetectCorners] – btTrue to automatically detect unselected plate corners and unselected brick edges within the selected edges and faces, respectively, and detach those as well. This option is necessary because a single corner of a triangle (for example) cannot be selected (only plate edges and brick faces can be selected).

Dependencies

Selection

Elements can be selected using functions in *Entity Selection*.

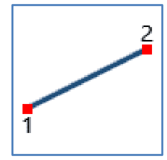
Keep Selected

Assigned using *St7SetKeepSelect*.

Tools – Create

St7PLLine2

Generates a line of nodes or beam elements on a straight line between the two end points, P1 and P2.




```
long St7PLLine2(long uID, double* P1, double* P2, long NumSteps)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Start of line in global XYZ coordinates.

`P2[0..2]`

End of line in global XYZ coordinates.

`NumSteps`

Number of steps.

Dependencies

`Points and Lines Target`

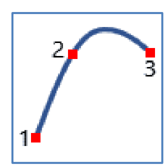
Target beam type and property assigned using *St7SetPLTarget*.

`Default Group`

Target group for the beams is specified by *St7SetDefaultGroupID*.

St7PLParabola3

Generates a line of nodes or beam elements on a parabola passing through three points, P1, P2 and P3.




```
long St7PLParabola3(long uID, double* P1, double* P2, double* P3, long NumSteps)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Point 1 in global XYZ coordinates.

`P2[0..2]`

Point 2 in global XYZ coordinates.

`P3[0..2]`

Point 3 in global XYZ coordinates.

`NumSteps`

Number of steps.

Dependencies

`Points and Lines Target`

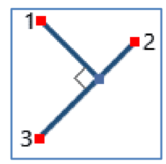
Target beam type and property assigned using *St7SetPLTarget*.

`Default Group`

Target group for the beams is specified by *St7SetDefaultGroupID*.

St7PLNormal3

Generates two perpendicular lines. A straight line is created between P2-P3. The second line is created from P1 to a new node on P2-P3 such that the second line is perpendicular to P2-P3.




```
long St7PLNormal3(long uID, double* P1, double* P2, double* P3)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Point 1 in global XYZ coordinates.

`P2[0..2]`

Point 2 in global XYZ coordinates.

`P3[0..2]`

Point 3 in global XYZ coordinates.

Dependencies

`Points and Lines Target`

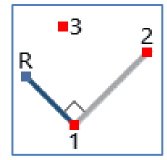
Target beam type and property assigned using *St7SetPLTarget*.

`Default Group`

Target group for the beams is specified by *St7SetDefaultGroupID*.

St7PLNormal3R

Finds the centre of a circle with radius R such that a straight line between P1 and P2 is tangent to the circle and touches the circle at P1. P3 is necessary to define the plane of the circle. The line between P1 and the centre of the circle will be normal to the line between P1 and P2.




```
long St7PLNormal3R(long uID, double* P1, double* P2, double* P3, double Radius,
     long NumSteps)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Point 1 in global XYZ coordinates.

`P2[0..2]`

Point 2 in global XYZ coordinates.

`P3[0..2]`

Point 3 in global XYZ coordinates.

`Radius`

Radius.

`NumSteps`

Number of steps.

Dependencies

`Points and Lines Target`

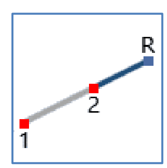
Target beam type and property assigned using *St7SetPLTarget*.

`Default Group`

Target group for the beams is specified by *St7SetDefaultGroupID*.

St7PLExtend2R

Extends a line by the distance R and generates a node at the new end of the line. The line is defined by two points, P1 and P2.



```
long St7PLExtend2R(long uID, double* P1, double* P2, double Radius,
     long NumSteps)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Point 1 in global XYZ coordinates.

`P2[0..2]`

Point 2 in global XYZ coordinates.

`Radius`

Radius.

`NumSteps`

Number of steps.

Dependencies

`Points and Lines Target`

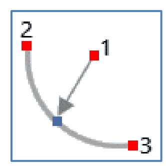
Target beam type and property assigned using *St7SetPLTarget*.

`Default Group`

Target group for the beams is specified by *St7SetDefaultGroupID*.

St7PLAverage2

Creates a point such that its coordinates are the average of P1 and P2 in the prescribed coordinate system.




```
long St7PLAverage2(long uID, double* P1, double* P2, long UCSId)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Point 1 in global XYZ coordinates.

`P2[0..2]`

Point 2 in global XYZ coordinates.

`UCSId`

Coordinate system ID.

Dependencies

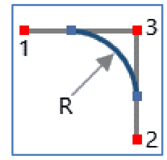
`Points and Lines Target`

Target beam type and property assigned using *St7SetPLTarget*.

`Default Group`

Target group for the beams is specified by *St7SetDefaultGroupID*.

St7PLFillet3R



Generates a line of nodes or beam elements along a circular arc. The arc is located such that the two intersecting straight lines are tangents to the arc. The arc forms a fillet between the lines. The lines are defined using three points. P3 is the intersection of the two lines. Two other points are also required, one on each of the two lines. The three nodes defining the lines may not be collinear.

```
long St7PLFillet3R(long uID, double* P1, double* P2, double* P3, double Radius,
     long NumSteps)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Point 1 in global XYZ coordinates.

`P2[0..2]`

Point 2 in global XYZ coordinates.

`P3[0..2]`

Point 3 in global XYZ coordinates.

`Radius`

Fillet radius.

`NumSteps`

Number of steps.

Dependencies

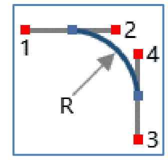
`Points and Lines Target`

Target beam type and property assigned using *St7SetPLTarget*.

`Default Group`

Target group for the beams is specified by *St7SetDefaultGroupID*.

St7PLFillet4R



Generates a line of nodes or beam elements along a circular arc. The arc is located such that the two intersecting straight lines are tangents to the arc. The arc forms a fillet between the lines. The lines are defined using four points. Points P1 and P2 define the first line and points P3 and P4 define the second line. The four points must be co-planar.

```
long St7PLFillet4R(long uID, double* P1, double* P2, double* P3, double* P4,
     double Radius, long NumSteps)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Point 1 in global XYZ coordinates.

`P2[0..2]`

Point 2 in global XYZ coordinates.

`P3[0..2]`

Point 3 in global XYZ coordinates.

`P4[0..2]`

Point 4 in global XYZ coordinates.

`Radius`

Fillet radius.

`NumSteps`

Number of steps.

Dependencies

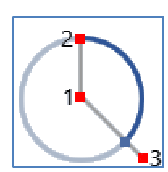
`Points and Lines Target`

Target beam type and property assigned using *St7SetPLTarget*.

`Default Group`

Target group for the beams is specified by *St7SetDefaultGroupID*.

St7PLCircleO3



Generates a line of nodes and/or beam elements around the circumference of a circle. P1 defines the centre of the circle, P2 is a point on the circumference of the circle and P3 is used to define the plane of the circle.

```
long St7PLCircleO3(long uID, double* P1, double* P2, double* P3, long NumSteps,
     bool FullCircle)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Point 1 in global XYZ coordinates.

`P2[0..2]`

Point 2 in global XYZ coordinates.

`P3[0..2]`

Point 3 in global XYZ coordinates.

`NumSteps`

Number of steps.

`FullCircle`

True to span the full 360°.

False to span the angle between P2 and P3.

Dependencies

`Points and Lines Target`

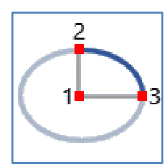
Target beam type and property assigned using *St7SetPLTarget*.

`Default Group`

Target group for the beams is specified by *St7SetDefaultGroupID*.

St7PLEllipseO3

Generates a line of nodes and/or beam elements around an ellipse. The ellipse is centred at P1 and passes through P2 and P3.




```
long St7PLEllipseO3(long uID, double* P1, double* P2, double* P3, long NumSteps,
     bool FullCircle)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Point 1 in global XYZ coordinates.

`P2[0..2]`

Point 2 in global XYZ coordinates.

`P3[0..2]`

Point 3 in global XYZ coordinates.

`NumSteps`

Number of steps.

`FullCircle`

True to span the full 360°.

False to span the angle between P2 and P3.

Dependencies

`Points and Lines Target`

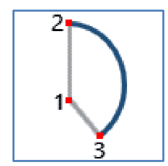
Target beam type and property assigned using *St7SetPLTarget*.

`Default Group`

Target group for the beams is specified by *St7SetDefaultGroupID*.

St7PLCurve3

Generates a line of beam elements or nodes around a variable radius curve. The curve is centred at P1. The radius spirals from P2 to P3. It is similar to an ellipse.




```
long St7PLCurve3(long uID, double* P1, double* P2, double* P3, long NumSteps)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Point 1 in global XYZ coordinates.

`P2[0..2]`

Point 2 in global XYZ coordinates.

`P3[0..2]`

Point 3 in global XYZ coordinates.

`NumSteps`

Number of steps.

Dependencies

`Points and Lines Target`

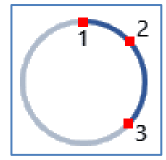
Target beam type and property assigned using *St7SetPLTarget*.

`Default Group`

Target group for the beams is specified by *St7SetDefaultGroupID*.

St7PLCircleC3

Generates a line of nodes and/or beam elements around the circumference of a circle. The circle is defined by three points on the circumference.




```
long St7PLCircleC3(long uID, double* P1, double* P2, double* P3, long NumSteps,
    bool FullCircle)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Point 1 in global XYZ coordinates.

`P2[0..2]`

Point 2 in global XYZ coordinates.

`P3[0..2]`

Point 3 in global XYZ coordinates.

`NumSteps`

Number of steps.

`FullCircle`

True to span the full 360°.

False to span the angle between P2 and P3.

Dependencies

`Points and Lines Target`

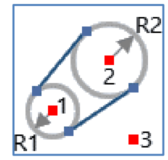
Target beam type and property assigned using *St7SetPLTarget*.

`Default Group`

Target group for the beams is specified by *St7SetDefaultGroupID*.

St7PLCirclesTangent3R

Two circles are defined and the two lines that are tangent to both circles are found. Four new nodes are generated, two on each of the circles (at the intersection between the tangent lines and the circles). The two circles are defined by centres P1 and P2, and radii R1 and R2 respectively. P3 is used to define the plane of the two circles.




```
long St7PLCirclesTangent3R(long uID, double* P1, double* P2, double* P3,
     double R1, double R2)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Point 1 in global XYZ coordinates.

`P2[0..2]`

Point 2 in global XYZ coordinates.

`P3[0..2]`

Point 3 in global XYZ coordinates.

`R1`

Radius of circle centred at P1.

`R2`

Radius of circle centred at P2.

Dependencies

`Points and Lines Target`

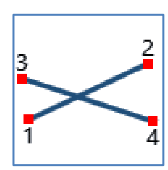
Target beam type and property assigned using *St7SetPLTarget*.

`Default Group`

Target group for the beams is specified by *St7SetDefaultGroupID*.

St7PLIntersect4

Generates a new node at the intersection of two lines. Each of the two lines is defined by two points, P1-P2 for line 1 and P3-P4 for line 2. All four nodes must lie on the same plane. If this is not the case then the message "Intersection not found" will be displayed.



```
long St7PLIntersect4(long uID, double* P1, double* P2, double* P3, double* P4)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Point 1 in global XYZ coordinates.

`P2[0..2]`

Point 2 in global XYZ coordinates.

`P3[0..2]`

Point 3 in global XYZ coordinates.

`P4[0..2]`

Point 4 in global XYZ coordinates.

Dependencies

`Points and Lines Target`

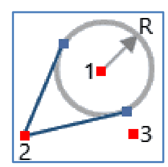
Target beam type and property assigned using *St7SetPLTarget*.

`Default Group`

Target group for the beams is specified by *St7SetDefaultGroupID*.

St7PLCircleTangent3R

Locates the intersection between a circe and a tangent. The circle is defined by a single point, P1, and a radius R1. The line is assumed to be tangent to the circle and to pass through P2. P3 is used to define the plane of the circle.




```
long St7PLCircleTangent3R(long uID, double* P1, double* P2, double* P3,
     double Radius)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Point 1 in global XYZ coordinates.

`P2[0..2]`

Point 2 in global XYZ coordinates.

`P3[0..2]`

Point 3 in global XYZ coordinates.

`Radius`

Radius of circle centred at P1.

Dependencies

`Points and Lines Target`

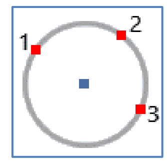
Target beam type and property assigned using *St7SetPLTarget*.

`Default Group`

Target group for the beams is specified by *St7SetDefaultGroupID*.

St7PLCircleCentre3

Finds the centre of a circle. A new node is generated at this location. The circle is defined by three points on the circumference.




```
long St7PLCircleCentre3(long uID, double* P1, double* P2, double* P3)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Point 1 in global XYZ coordinates.

`P2[0..2]`

Point 2 in global XYZ coordinates.

`P3[0..2]`

Point 3 in global XYZ coordinates.

Dependencies

`Points and Lines Target`

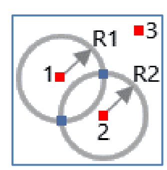
Target beam type and property assigned using *St7SetPLTarget*.

`Default Group`

Target group for the beams is specified by *St7SetDefaultGroupID*.

St7PLCirclesIntersect3R

Generates two new nodes at the points where two overlapping circles intersect. The centre of each circle is defined using P1 and P2. The radius of each circle is defined by R1 and R2 respectively. P3 is used to define the plane of the two circles. Note that in the special case where the points P1 and P2 are R1+R2 apart, only one point is generated.



```
long St7PLCirclesIntersect3R(long uID, double* P1, double* P2, double* P3,
    double R1, double R2)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Point 1 in global XYZ coordinates.

`P2[0..2]`

Point 2 in global XYZ coordinates.

`P3[0..2]`

Point 3 in global XYZ coordinates.

`R1`

Radius of circle centred at P1.

`R2`

Radius of circle centred at P2.

Dependencies

`Points and Lines Target`

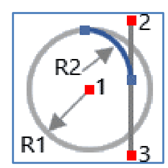
Target beam type and property assigned using *St7SetPLTarget*.

`Default Group`

Target group for the beams is specified by *St7SetDefaultGroupID*.

St7PLCircleLineInnerFillet3R

Generates a circular arc fillet between a straight line and a circle. The circle is defined by P1, at the centre and a radius R1. The line is defined by P2 and P3. The radius of the fillet is specified as R2.




```
long St7PLCircleLineInnerFillet3R(long uID, double* P1, double* P2, double* P3,
    double R1, double R2, long NumSteps, bool FullCircle)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Point 1 in global XYZ coordinates.

`P2[0..2]`

Point 2 in global XYZ coordinates.

`P3[0..2]`

Point 3 in global XYZ coordinates.

`R1`

Radius of circle centred at P1.

`R2`

Fillet radius.

`NumSteps`

Number of steps.

`FullCircle`

True to span the full 360°.

False to span the angle between the circle and line tangents.

Dependencies

`Points and Lines Target`

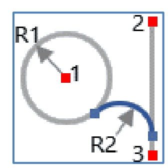
Target beam type and property assigned using *St7SetPLTarget*.

`Default Group`

Target group for the beams is specified by *St7SetDefaultGroupID*.

St7PLCircleLineOuterFillet3R

Generates a circular arc fillet (or full circle) between a straight line and a circle. The radius of the arc can be set. The main circle is defined by point P1 and radius R1. The straight line is defined by points P2-P3. The arc has fillet radius R2. Note that R1 must be greater than R2.




```
long St7PLCircleLineOuterFillet3R(long uID, double* P1, double* P2, double* P3,
     double R1, double R2, long NumSteps, bool FullCircle)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Point 1 in global XYZ coordinates.

`P2[0..2]`

Point 2 in global XYZ coordinates.

`P3[0..2]`

Point 3 in global XYZ coordinates.

`R1`

Radius of circle centred at P1.

`R2`

Fillet radius.

`NumSteps`

Number of steps.

`FullCircle`

True to span the full 360°.

False to span the angle between the circle and line tangents.

Dependencies

`Points and Lines Target`

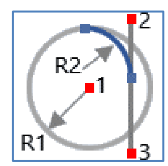
Target beam type and property assigned using *St7SetPLTarget*.

`Default Group`

Target group for the beams is specified by *St7SetDefaultGroupID*.

St7PLCircleLineIntersect3

Generates two nodes at the intersection between a circle and a straight line (except in the case where the line is a tangent to the circle in which case only one point is generated). The circle is defined by a point at the centre, P1, and a radius R. The line is defined by two points, P2 and P3. If one of these points lies inside the circle, the line is extended to find the second point of intersection.




```
long St7PLCircleLineIntersect3(long uID, double* P1, double* P2, double* P3,
     double Radius)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Point 1 in global XYZ coordinates.

`P2[0..2]`

Point 2 in global XYZ coordinates.

`P3[0..2]`

Point 3 in global XYZ coordinates.

`Radius`

Radius of circle.

Dependencies

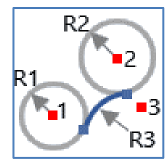
`Points and Lines Target`

Target beam type and property assigned using *St7SetPLTarget*.

`Default Group`

Target group for the beams is specified by *St7SetDefaultGroupID*.

St7PLCirclesFillet3R



Generates a line of beams or nodes on a circular fillet between two circles. The fillet touches each of the two circles at a single tangent point. The two circles are defined by centres P1 and P2 and radii R1 and R2 respectively. A third point, P3, is used to define the plane of the circles. The radius of the fillet is specified as R3.

Note that this tool will not produce any results if one of the circles is completely enclosed by the other or if the circles are further than 2 x R3 apart.

```
long St7PLCirclesFillet3R(long uID, double* P1, double* P2, double* P3,
     double R1, double R2, double R3, long NumSteps, bool FullCircle)
```

Input Parameters

`uID`

Strand7 model file ID.

`P1[0..2]`

Point 1 in global XYZ coordinates.

`P2[0..2]`

Point 2 in global XYZ coordinates.

`P3[0..2]`

Point 3 in global XYZ coordinates.

`R1`

Radius of circle centred at P1.

`R2`

Radius of circle centred at P2.

`R3`

Fillet radius.

`NumSteps`

Number of steps.

`FullCircle`

True to span the full 360°.

False to span the angle between the circle and line tangents.

Dependencies

`Points and Lines Target`

Target beam type and property assigned using *St7SetPLTarget*.

`Default Group`

Target group for the beams is specified by *St7SetDefaultGroupID*.

St7CreateRigidLinkCluster

Creates a rigid link cluster between selected nodes, such that they act as a rigid body.

```
long St7CreateRigidLinkCluster(long uID, long UCSId, long Axis, long NodeNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

The Cartesian coordinate system in which the rigid plane is prescribed. Not relevant if Axis is rlPlaneXYZ.

`Axis`

One of rlPlaneXYZ, rlPlaneXY, rlPlaneYZ or rlPlaneZX.

`NodeNum`

Slave node number, or 0 to create a new slave node at the average position of the selected nodes.

Dependencies

`Selection`

Nodes can be selected using functions in *Entity Selection*.

`Default Group`

Target group for the links is specified by *St7SetDefaultGroupID*.

St7CreatePinnedLinkCluster

Creates a pinned link cluster between selected nodes.

```
long St7CreatePinnedLinkCluster(long uID, long NodeNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Slave node number, or 0 to create a new slave node at the average position of the selected nodes.

Dependencies

`Selection`

Nodes can be selected using functions in *Entity Selection*.

`Default Group`

Target group for the links is specified by *St7SetDefaultGroupID*.

St7CreateMasterSlaveLinkCluster

Creates a master-slave link cluster between selected nodes.

```
long St7CreateMasterSlaveLinkCluster(long uID, long UCSId, long DoFBits,
     long NodeNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

The coordinate system in which the degrees of freedom are defined.

`DoFBits`

Bitmask of flags for six degrees of freedom if Master-Slave links are created. For example, to couple DX, DY and RZ, set ConstraintBits = 1+2+32 = 35.

`NodeNum`

Slave node number, or 0 to create a new slave node at the average position of the selected nodes.

Dependencies

`Selection`

Nodes can be selected using functions in *Entity Selection*.

`Default Group`

Target group for the link is specified by *St7SetDefaultGroupID*.

St7CreateSectorSymmetryLinkCluster

Creates a sector-symmetry links that couple nodes on angular planes.

```
long St7CreateSectorSymmetryLinkCluster(long uID, long Axis, double Plane1,
    double Plane2, double RadialTol, double AngularTol)
```

Input Parameters

`uID`

Strand7 model file ID.

`Axis`

Either 1, 2 or 3 for the global X, Y or Z axis, respectively.

`Plane1`

Angle of first plane (degrees).

`Plane2`

Angle of second plane (degrees).

`RadialTol`

Radial tolerance.

`AngularTol`

Angular tolerance.

Dependencies

`Default Group`

Target group for the links is specified by *St7SetDefaultGroupID*.

St7CreateInterpolatedMultiPointLink

Creates an interpolated multi-point link cluster between selected nodes.

```
long St7CreateInterpolatedMultiPointLink(long uID, long Couple, long NodeNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Couple`

One of cpTranslational, cpRotational or cpBoth.

`NodeNum`

Slave node number, or 0 to create a new slave node at the average position of the selected nodes.

Dependencies

`Selection`

Nodes can be selected using functions in *Entity Selection*.

`Default Group`

Target group for the link is specified by *St7SetDefaultGroupID*.

St7CreateMasterSlaveMultiPointLink

Creates a master-slave multi-point link.

```
long St7CreateMasterSlaveMultiPointLink(long uID, long UCSId, long DoFBits,
    long NodeNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

The coordinate system in which the degrees of freedom are defined.

`DoFBits`

Bitmask of flags for six degrees of freedom if Master-Slave links are created. For example, to couple DX, DY and RZ, set ConstraintBits = 1+2+32 = 35.

`NodeNum`

Slave node number, or 0 to create a new slave node at the average position of the selected nodes.

Dependencies

`Selection`

Nodes can be selected using functions in *Entity Selection*.

`Default Group`

Target group for the links is specified by *St7SetDefaultGroupID*.

St7CreatePinnedMultiPointLink

Creates a pinned multi-point link.

```
long St7CreatePinnedMultiPointLink(long uID, long NodeNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`NodeNum`

Slave node number, or 0 to create a new slave node at the average position of the selected nodes.

Dependencies

`Selection`

Nodes can be selected using functions in *Entity Selection*.

`Default Group`

Target group for the links is specified by *St7SetDefaultGroupID*.

St7CreateRigidMultiPointLink

Creates a rigid multi-point link.

```
long St7CreateRigidMultiPointLink(long uID, long UCSId, long Axis, long NodeNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

The Cartesian coordinate system in which the rigid plane is prescribed. Not relevant if Axis is rlPlaneXYZ.

`Axis`

One of rlPlaneXYZ, rlPlaneXY, rlPlaneYZ or rlPlaneZX.

`NodeNum`

Slave node number, or 0 to create a new slave node at the average position of the selected nodes.

Dependencies

`Selection`

Nodes can be selected using functions in *Entity Selection*.

`Default Group`

Target group for the links is specified by *St7SetDefaultGroupID*.

St7CreateReactionMultiPointLink

Creates a reaction multi-point link.

```
long St7CreateReactionMultiPointLink(long uID, long SetNum, long OriginCode,
    double* Origin)
```

Input Parameters

`uID`

Strand7 model file ID.

`SetNum`

Element set number defining the entities that contribute to the reaction multi-point link cluster.

`OriginCode`

One of ocUseOrigin (to use the specified origin); ocUseNodeAverage (to use the average position of the selected nodes); or any node number (to place the origin at the position of the specified node).

`Origin[0..2]`

Location of origin in global XYZ coordinates, about which moments are calculated.

Dependencies

`Selection`

Nodes can be selected using functions in *Entity Selection*.

`Default Group`

Target group for the links is specified by *St7SetDefaultGroupID*.

St7CreateBeamsOnElementEdges

Creates beams on selected plate edges and brick faces.

```
long St7CreateBeamsOnElementEdges(long uID, long PropNum, long QuadraticAs,
     long BasedOn, double FacetAngle, bool FreeEdges, bool TJunctions,
     bool PropBoundary, bool GroupBoundary, bool InternalBricks)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Property number of created beams.

`QuadraticAs`

One of eeSplit, eeIgnoreMid or eeBeam3.

`BasedOn`

Either beBasedOnModel or beBasedOnSelected.

`FacetAngle`

Angle (degrees).

`FreeEdges`

True to apply to free edges.

`TJunctions`

True to apply to T-junctions.

`PropBoundary`

True to apply property boundaries.

`GroupBoundary`

True to apply to group boundaries.

`InternalBricks`

True to apply to internal brick edges. If FacetAngle=0, DoTJunctions=True and DoInternalBricks=True, a beam will be created on every edge of every selected brick.

Dependencies

`Selection`

Plate edges and brick faces can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7CreateBeamsOnGeometryEdges

Creates beams on selected geometry edges.

```
long St7CreateBeamsOnGeometryEdges(long uID, long PropNum, long GeometryAs)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Property number of created beams.

`GeometryAs`

Either geBeam2 or geBeam3.

Dependencies

`Selection`

Geometry edges can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7CreatePlatesOnBricks

Creates plates on selected brick faces.

```
long St7CreatePlatesOnBricks(long uID, bool FreeFacesOnly, bool AllowDuplicates)
```

Input Parameters

`uID`

Strand7 model file ID.

`FreeFacesOnly`

True to create plates on selected free brick faces.

False to create plates on all selected brick faces.

`AllowDuplicates`

If True, two overlapping plates will be created when two selected brick faces share nodes. If False, only a single plate is created when two selected brick faces share nodes.

Relevant only when FreeFacesOnly=False.

Dependencies

`Selection`

Brick faces can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7CreateEntityUCS

Creates User Coordinate Systems on selected entities.

```
long St7CreateEntityUCS(long uID, long CurvedPipeAxis, long BeamAxis,
     long OriginLocation, bool OriginNode)
```

Input Parameters

`uID`

Strand7 model file ID.

`CurvedPipeAxis`

Target coordinate system type for curved pipe elements; either puCylindrical or puCartesian.

`BeamAxis`

Target coordinate system type for non pipe elements; either buPrincipal or buLocal.

`OriginLocation`

One of ulAtMin, ulAtMax or ulAtMean to locate the coordinate system at the minimum corner, maximum corner or centre of the entity UCS bounding box.

`OriginNode`

True to create a node at the origin of the UCS.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7CreateLoadPatches

Creates Load Patch plate elements on selected beam frameworks.

```
long St7CreateLoadPatches(long uID, double PlaneTol, bool TriangularLoad,
     bool UseBeamGroup)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlaneTol`

Angular tolerance (degrees); the tool ignores adjacent beam segments that exceed this tolerance.

`TriangularLoad`

True to apply triangular loads on load patch polygons with five or more edges.

False to apply a constant load on load patch polygons with five or more edges.

`UseBeamGroup`

True to assign plate elements to the same group as their surrounding beam elements.

False to assign plate elements to the default group.

Dependencies

`Selection`

Beams can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7CreateAttachments

Generates attachment links based on the pre-defined attachment attributes.

```
long St7CreateAttachments(long uID, long BrickTarget, double AngleDelta,
     bool DeleteExisting)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickTarget`

Either ktFreeFaces to attach to free (exposed) brick faces, ktAllFaces to attach to all brick faces (exposed and internal), or ktInsideBricks to create internal brick attachments.

`AngleDelta`

Angular tolerance (degrees), used when calculating attachment directions.

`DeleteExisting`

True to delete existing attachments links before attaching.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

Global Parameters

`ivAttachmentsCreated, ivAttachmentsFailed.`

St7CreateCartesianSymmetryRestraints

Locates nodes on the three global XYZ symmetry planes (that is, X=0, Y=0 or Z=0) and assigns appropriate symmetry restraints to them.

```
long St7CreateCartesianSymmetryRestraints(long uID, long FreedomCase)
```

Input Parameters

`uID`

Strand7 model file ID.

`FreedomCase`

Freedom case number.

St7CreateCylindricalSymmetryRestraints

Locates nodes on the two angular planes specified and assigns appropriate symmetry restraints to them.

```
long St7CreateCylindricalSymmetryRestraints(long uID, long Axis,
     long FreedomCase, double Theta1, double Theta2, double AngularTol)
```

Input Parameters

`uID`

Strand7 model file ID.

`Axis`

Axis of the global XYZ system about which angular restraints will be applied.

`FreedomCase`

Freedom case number.

`Theta1`

Angle of first symmetry plane (degrees).

`Theta2`

Angle of second symmetry plane (degrees).

`AngularTol`

Angular tolerance.

Tools – Merge

St7MergeElementPairs

Merges pairs of selected adjacent elements.

```
long St7MergeElementPairs(long uID, bool Quadratic)
```

Input Parameters

`uID`

Strand7 model file ID.

`Quadratic`

True to merge plate and brick pairs into quadratic elements.

False to merge plate and brick pairs into linear elements.

Dependencies

`Selection`

Elements can be selected using functions in *Entity Selection*.

St7MergeLineOfBeams

Merges lines of beams.

```
long St7MergeLineOfBeams(long uID, double AngleTol, long AngleMode)
```

Input Parameters

`uID`

Strand7 model file ID.

`AngleTol`

Angular tolerance, above which beams will not be merged.

`AngleMode`

Either mbStatic or mbDynamic.

Dependencies

`Selection`

Beams can be selected using functions in *Entity Selection*.

St7MergeTriToQuad

Merges triangular plates into quadrilateral plates.

```
long St7MergeTriToQuad(long uID, double MinInternalAngle,
     double MaxInternalAngle, double MaxNormalAngle)
```

Input Parameters

`uID`

Strand7 model file ID.

`MinInternalAngle`

Minimum internal angle (degrees); if a potential quad plate has an internal angle less than this, it will not be generated.

`MaxInternalAngle`

Maximum internal angle (degrees); if a potential quad plate has an internal angle larger than this, it will not be generated.

`MaxNormalAngle`

Maximum angular difference between tri plate normals; if the angle between normals of adjacent triangles exceed this value, they will not be merged into a single quad plate.

Dependencies

`Selection`

Plates can be selected using functions in *Entity Selection*.

Tools – Convert

St7ConvertBeamsToLinks

Converts selected beams into links.

```
long St7ConvertBeamsToLinks(long uID, long LinkType, long LinkOption,
     long CaseID)
```

Input Parameters

`uID`

Strand7 model file ID.

`LinkType`

One of ltMasterSlaveLink, ltPinnedLink, ltRigidLink or ltShrinkLink.

`LinkOption`

Additional information about the link, depending on the setting of LinkType:

ltMasterSlaveLink	– Bitmask of flags for the constrained degrees of freedom
ltPinnedLink	– Not used
ltRigidLink	– Plane of action; one of rlPlaneXYZ, rlPlaneXY, rlPlaneYZ or rlPlaneZX
ltShrinkLink	– Bitmask of flags for the constrained degrees of freedom

`CaseID`

For LinkType either ltMasterSlaveLink or ltRigidLink, the UCS for the created links.

For LinkType ltShrinkLink, the freedom case that controls the shrink in a nonlinear analysis.

Dependencies

`Selection`

Beams can be selected using functions in *Entity Selection*.

`Source Action`

Assigned using *St7SetSourceAction*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7ConvertLinksToBeams

Converts selected links into beams.

```
long St7ConvertLinksToBeams(long uID, long PropNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`PropNum`

Target beam property number.

Dependencies

`Selection`

Links can be selected using functions in *Entity Selection*.

`Source Action`

Assigned using *St7SetSourceAction*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7ConvertBeamOffsetsToRigidLinks

Moves selected offset beams to their spatial locations, sets the offset attributes to zero, and connects the beams to their original nodes via rigid links.

```
long St7ConvertBeamOffsetsToRigidLinks(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

Dependencies

`Selection`

Beams can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7ConvertPatchLoads

Converts load patches in the specified load case to distributed beam loads.

```
long St7ConvertPatchLoads(long uID, long CaseNum, bool Overwrite)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

`Overwrite`

True to overwrite the existing beam loads.

False to add to the existing beam loads.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

Global Parameters

`ivPlateEdgesAssigned, ivPlateEdgesNotFullyAssigned.`

St7CheckPatchLoads

Simulates the conversion of load patches to distributed beam loads in the specified load case without actually generating the beam loads. This allows the developer to determine which load patches can successfully transfer loads and which may require adjustment. This can be done by examining the global parameters to check for load patch plate edges that remain selected after the function has been called; edges that remain selected are those at which the load could not be transferred to beam elements.

```
long St7CheckPatchLoads(long uID, long CaseNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`CaseNum`

Load case number.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

Global Parameters

`ivPlateEdgesAssigned, ivPlateEdgesNotFullyAssigned.`

St7ConvertLoadPathsToLoadCases

Converts selected load paths to load cases.

```
long St7ConvertLoadPathsToLoadCases(long uID, bool PointForces,
     bool DistributedForces, bool HeatSources)
```

Input Parameters

`uID`

Strand7 model file ID.

`PointForces`

True to generate point forces.

`DistributedForces`

True to generate distributed forces.

`HeatSources`

True to generate heat sources.

Dependencies

`Selection`

Load paths can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7ConvertBeamPolygonsToPlates

Converts selected beam polygons into plates. Beam polygons can have three or four sides.

```
long St7ConvertBeamPolygonsToPlates(long uID, double MinInternalAngle,
    double MaxInternalAngle, double MaxNormalAngle, bool CreateQuad4)
```

Input Parameters

`uID`

Strand7 model file ID.

`MinInternalAngle`

Minimum internal angle for a Quad4 element (otherwise, two Tri3 elements are created).

`MaxInternalAngle`

Maximum internal angle for a Quad4 element (otherwise, two Tri3 elements are created).

`MaxNormalAngle`

Maximum normal angle between two adjacent Tri3 elements to be merged to a Quad4 (otherwise, two Tri3 elements are created).

`CreateQuad4`

True to create Quad4 elements if the angular criteria are met.

Dependencies

`Selection`

Beams can be selected using functions in *Entity Selection*.

`Source Action`

Assigned using *St7SetSourceAction*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

Tools – Adjust

St7AdjustMidsideNodes

Adjusts the mid-side nodes of selected elements.

```
long St7AdjustMidsideNodes(long uID, bool MakeStraight)
```

Input Parameters

`uID`

Strand7 model file ID.

`MakeStraight`

True to move the mid-side node to the middle of the circle defined by the three original node positions.

False to move the mid-side node to the Cartesian average position.

Dependencies

`Selection`

Plates and bricks can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7SmoothPlates

Smooths the selected plate elements.

```
long St7SmoothPlates(long uID, long UCSId, bool SmoothBoundary)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

The coordinate system in which to smooth the plate edges.

`SmoothBoundary`

True to allow the smoothing function to move boundary nodes.

Dependencies

`Selection`

Plates can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7ReorderNodesTree

Renumber mesh nodes using the Tree algorithm.

```
long St7ReorderNodesTree(long uID, long StartNodeNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`StartNodeNum`

Starting node number.

St7ReorderNodesGeometry

Renumber mesh nodes using the Geometry algorithm.

```
long St7ReorderNodesGeometry(long uID, double* DXYZ)
```

Input Parameters

`uID`

Strand7 model file ID.

`DXYZ[0..2]`

Direction vector in the global XYZ system.

St7ReorderNodesAMD

Renumber mesh nodes using the AMD algorithm.

```
long St7ReorderNodesAMD(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7CorrectAttachmentLinkGroups

Place attachment links in the same groups as the elements to which they attach, as required for a staged analysis.

```
long St7CorrectAttachmentLinkGroups(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

Dependencies

`Selection`

Links can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7TrimMultiPointLinks

Removes free nodes from selected multi-point links.

```
long St7TrimMultiPointLinks(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

Dependencies

`Selection`

Links can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*

Tools – Align

St7BeamOffsetsByCrossSection

Assign offsets to selected beams to shift the node location to some point on the cross section.

```
long St7BeamOffsetsByCrossSection(long uID, long* Offsets)
```

Input Parameters

`uID`

Strand7 model file ID.

`Offsets[0..11]`

Assign the desired offset by setting the value in the index position corresponding to each beam cross section type.

Index positions are ipCircularSection, ipSquareSection, ipCSection, ipISection, ipTSection, ipLSection, ipZSection, ipBXSSection, ipTrapezoidalSection, ipTriangularSection, ipCruciformSection and ipUndefinedSection.

Cross section offset values are soNoChange, soTopLeft, soTopMid, soTopRight, soMidLeft, soGeometricCenter, soMidRight, soBottomLeft, soBottomMid, soBottomRight, soCentroid or soShearCenter.

St7AlignBeamAxesToUCS

Aligns the local axes of selected beams to the axes of the specified UCS.

```
long St7AlignBeamAxesToUCS(long uID, long BeamAxis, long BeamAxisType,
     long UCSAxis, long UCSId, double AngleTol, bool KeepEndAttributeLocation)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamAxis`

The 1, 2 or 3 local beam axis to align.

`BeamAxisType`

Local beam axis type – axBeamLocal or axBeamPrincipal.

`UCSAxis`

The 1, 2 or 3 UCS axis to align with. Valid values include ±1, ±2 or ±3.

`UCSId`

ID number of the UCS supplying the UCSAxis to align with. UCSId = 1 refers to the global XYZ system.

`AngleTol`

Angular tolerance providing the range over which the tools operates, from 0 and 90 degrees.

`KeepEndAttributeLocation`

If a beam element is flipped, exchange the end attributes such that the attributes stay in the same physical position. Relevant if BeamAxis=3.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7AlignBeamAxesToPlate

Aligns the selected beams with the plates they are connected to.

```
long St7AlignBeamAxesToPlate(long uID, long BeamAxis, long BeamAxisType,
     bool PositiveDir)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamAxis`

The 1 or 2 local beam axis to align.

`BeamAxisType`

Local beam axis type – axBeamLocal or axBeamPrincipal.

`PositiveDir`

True to align in the same direction as the normal to the notional surface.

Dependencies

`Selection`

Beams can be selected using functions in *Entity Selection*. Plates do not need to be selected – Strand7 determines the relevant plates based on the connectivity.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7AlignBeamAxesToFramework

Aligns the selected beams with a notional surface that spans the beam framework.

```
long St7AlignBeamAxesToFramework(long uID, long BeamAxis, long BeamAxisType,
     bool PositiveDir)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamAxis`

The 1 or 2 local beam axis to align.

`BeamAxisType`

Local beam axis type – axBeamLocal or axBeamPrincipal.

`PositiveDir`

True to align in the same direction as the normal to the notional surface.

Dependencies

`Selection`

Beams can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7AlignBeam3AxisByConnection

Aligns selected beams such that they have a consistently aligned three axis.

```
long St7AlignBeam3AxisByConnection(long uID, bool KeepEndAttributeLocation)
```

Input Parameters

`uID`

Strand7 model file ID.

`KeepEndAttributeLocation`

If a beam element is flipped, exchange the end attributes such that the attributes stay in the same physical position.

Dependencies

`Selection`

Beams can be selected using functions in *Entity Selection*.

St7AlignBeamAxisToVector

Aligns an axis of the selected beams with a given vector.

```
long St7AlignBeamAxisToVector(long uID, long BeamAxis, long BeamAxisType,
    double AngleTol, double* Vector)
```

Input Parameters

`uID`

Strand7 model file ID.

`BeamAxis`

The 1 or 2 local beam axis to align.

`BeamAxisType`

Local beam axis type – axBeamLocal or axBeamPrincipal.

`AngleTol`

Angular tolerance providing the range over which the tools operates, from 0 and 90 degrees.

`Vector[0..2]`

Vector to align beams to in the global XYZ system.

Dependencies

`Selection`

Beams can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7RemoveBeamReferenceNode

Converts selected Beam3 elements into Beam2 elements with Principal Axis Angle attributes to provide the equivalent orientation.

```
long St7RemoveBeamReferenceNode(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

Dependencies

`Selection`

Beams can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7PlateOffsetByThickness

Offsets selected plate elements to one of the plate surfaces, based on the plate thickness.

```
long St7PlateOffsetByThickness(long uID, long Surface)
```

Input Parameters

`uID`

Strand7 model file ID.

`Surface`

One of psPlateMidPlane, psPlateMinusZ or psPlatePlusZ.

Dependencies

`Selection`

Plates can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7AlignPlateAxesToUCS

Aligns the local axes of selected plates with the axis of the specified UCS.

```
long St7AlignPlateAxesToUCS(long uID, long PlateAxis, long UCSAxis, long UCSId,
    double AngleTol)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateAxis`

The local x, y or z plate axis to align.

`UCSAxis`

The 1, 2 or 3 UCS axis to align with.

`UCSId`

ID number of the UCS supplying the UCSAxis to align with. UCSId = 1 refers to the global XYZ system.

`AngleTol`

Angular tolerance providing the range over which the tools operates, from 0 and 90 degrees.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7AlignPlateRCDirectionsToUCS

Aligns the concrete reinforcement attribute of the selected plates.

```
long St7AlignPlateRCDirectionsToUCS(long uID, long RCLayers, long UCSAxis,
     long UCSId, double AngleTol)
```

Input Parameters

`uID`

Strand7 model file ID.

`RCLayers`

Either raLayer13 for layers 1 and 3, or raLayer24 for layers 2 and 4.

`UCSAxis`

Either 1, 2 or 3 to specify coordinate system axis.

`UCSId`

Coordinate system identifier.

`AngleTol`

Angular tolerance.

Dependencies

`Selection`

Plates can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7AlignPlateAxesByConnection

Aligns selected plates such that they have a consistently aligned local axes.

```
long St7AlignPlateAxesByConnection(long uID, long PlateNum, double MaxShearAngle)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Master plate element on which the alignment is based.

`MaxShearAngle`

Maximum shearing angle (degrees) to accept when double curvature leads to conflicting specifications of plate local axes x and y.

Dependencies

`Selection`

Plates can be selected using functions in *Entity Selection*.

St7AlignPlateNormalByConnection

Aligns the normals on selected and connected plate elements such that their normals are consistent with the master element. The master element must be connected to the selected elements.

```
long St7AlignPlateNormalByConnection(long uID, long PlateNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Master element number.

Dependencies

`Selection`

Plates can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7RotatePlateConnections

Rotates the selected plate elements by reordering the nodes. This effectively swaps the A and B directions in *St7Subdivide*.

```
long St7RotatePlateConnections(long uID, bool Clockwise)
```

Input Parameters

`uID`

Strand7 model file ID.

`Clockwise`

Rotation direction about the local z axis of the plate. True for clockwise; False for anti-clockwise.

Dependencies

`Selection`

Plates can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7AlignFaceNormalByConnection

Aligns the normals of selected faces such that they have consistent normal, based on a master face. The master face must be connected to the selected faces.

```
long St7AlignFaceNormalByConnection(long uID, long FaceNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Master face number.

Dependencies

`Selection`

Faces can be selected using functions in *Entity Selection*.

St7InvertPathNormal

Inverts the normal direction of selected load paths. The ends of the path are not flipped.

```
long St7InvertPathNormal(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

Dependencies

`Selection`

Load paths can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

St7FlipEntity

Flips the orientation of selected entities.

```
long St7FlipEntity(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

Dependencies

`Selection`

Entities can be selected using functions in *Entity Selection*.

`Keep Selected`

Assigned using *St7SetKeepSelect*.

Tools – Utility

St7InsituStress

Generates the soil in-situ stress distribution.

```
long St7InsituStress(long uID, long Mode, long Wait, long* Integers,
     double* Doubles, long* ProcessID, long* WarningCode)
```

Input Parameters

`uID`

Strand7 model file ID.

`Mode`

One of smNormalRun, smNormalCloseRun, smProgressRun or smBackgroundRun. See *Solver Options* for additional information.

`Wait`

Solver execution mode; either btTrue to halt execution of the caller until the solve is complete, or btFalse to pass control back to the caller immediately after the function is called.

`Integers[0..8]`

`[ipInsituGravityCase]` – Load case in which gravity is defined.

`[ipInsituFreedomCase]` – Freedom case.

`[ipInsituStageIndex]` – Stage at which to calculate the in-situ stress, or 0 to have all groups active.

`[ipInsituUseExisting]` – btTrue to use the existing in-situ stress distribution as the initial conditions.

`[ipInsituReplaceK0]` – Calculate horizontal stress ratio (K0) from stress results.

`[ipInsituMaxIterations]` – Iteration limit used by the solver.

`[ipInsituAllowIterations]` – btTrue to allow the solver to add iterations.

`[ipInsituSolverScheme]` – one of stSkyline, stSparse or stIterativePCG.

`[ipInsituMatrixSort]` – one of rnNone, rnTree, rnGeometry or rnAMD.

`Doubles[0..1]`

`[ipInsituDefaultFluidLevel]` – Default fluid level.

`[ipInsituDefaultFluidDensity]` – Default fluid mass density per unit volume.

Output Parameters

`ProcessID`

ID number for the solver process. Applicable if Wait is btFalse.

`WarningCode`

One of wcInsituNoWarning, wcInsituUnconverged or wcInsituTensileStress.

Utility Functions

St7TransformToUCS

Transforms a position specified in the global XYZ system to a User Coordinate System (UCS).

```
long St7TransformToUCS(long uID, long UCSId, double* XYZ)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

Input-Output Parameters

`XYZ[0..2]`

The position as a 3-element array. The global coordinates to be transformed are input in this array. The function call then returns the transformed UCS coordinates in this array in 123 DoF order.

St7TransformToXYZ

Transforms a position specified in an arbitrary User Coordinate System (UCS) to the global XYZ system.

```
long St7TransformToXYZ(long uID, long UCSId, double* XYZ)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

Input-Output Parameters

`XYZ[0..2]`

The position as a 3-element array. The UCS coordinates to be transformed are input in this array. The function call then returns the transformed global coordinates in this array in 123 DoF order.

St7VectorTransformToUCS

Transforms a vector specified in the global XYZ system to an arbitrary User Coordinate System (UCS) frame, at a given position in space.

```
long St7VectorTransformToUCS(long uID, long UCSId, double* Position,
     double* VXYZ)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Position[0..2]`

The position of the transformation in global coordinates as a 3-element array. The UCS directions into which VXYZ will be transformed are evaluated at this point.

Input-Output Parameters

`VXYZ[0..2]`

The vector defined as a 3-element array. The vector to be transformed is input in this array, defined in the global XYZ system. The function call then returns the transformed vector in this array in 123 DoF order, defined in the UCS frame.

St7VectorTransformToXYZ

Transforms a vector specified in a User Coordinate System (UCS) frame to the global XYZ system, at a given position in space.

```
long St7VectorTransformToXYZ(long uID, long UCSId, double* Position,
     double* VXYZ)
```

Input Parameters

`uID`

Strand7 model file ID.

`UCSId`

ID number of the specified UCS. UCSId = 1 refers to the global XYZ system.

`Position[0..2]`

The position in global XYZ coordinates as a 3-element array. The UCS directions into which VXYZ will be transformed are evaluated at this point.

Input-Output Parameters

`VXYZ[0..2]`

The vector defined as a 3-element array. The vector to be transformed is input in this array, defined in the UCS frame. The function call then returns the transformed vector in this array in 123 DoF order, defined in the global XYZ system.

St7GetElementData

Returns element specific geometric data for a given element.

```
long St7GetElementData(long uID, long Entity, long EntityNum, long ResultCase,
     double* EltData)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK or tyLINK.

`EntityNum`

Entity number.

`ResultCase`

Result case number – if greater than 0, it must be a valid result case; if less than 1, it is ignored.

Output Parameters

`EltData`

Geometric data for the specified element:

tyBEAM – Beam length; for curved pipe elements, the curved length; for cable elements, the cable length.

tyPLATE – Plate area; for curved offset plates, the area considers the offset.

tyBRICK – Brick volume.

tyLINK – For links with two nodes, the distance between the nodes; for links with more than two nodes, the sum of the distances between the slave node and each master node; for attachment links, the distance between the node and the attachment point on the target element.

Usage

If a result file is open, EltData is based on the birth coordinates of the element for the specified result case (for a staged analysis, birth coordinates can depend on the result case). If a result file is not open, or ResultCase is 0, EltData is based on the initial position of the nodes.

St7GetElementDataGNL

Returns element specific geometric data for a given element for geometrically nonlinear results.

```
long St7GetElementDataGNL(long uID, long Entity, long EntityNum, long ResultCase,
     double* EltData)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK or tyLINK.

`EntityNum`

Entity number.

`ResultCase`

Result case number.

Output Parameters

`EltData`

Geometric data for the specified element:

tyBEAM – Beam deformed length; for curved pipe elements, the curved undeformed length; for cable elements, the deformed cable length.

tyPLATE – Plate deformed area; for curved offset plates, the area considers the offset.

tyBRICK – Brick deformed volume.

tyLINK – For links with two nodes, the distance between the displaced nodes; for links with more than two nodes, the sum of the distances between the displaced slave node and each displaced master node; for attachment links, the distance between the displaced node and the attachment point on the displaced target element.

Usage

A result file that includes geometric nonlinearity must be open when calling this function. The data is calculated based on the element deformation at an absolute displacement scale of 1 for the specified result case.

St7GetElementDataDeformed

Returns element specific geometric data for a given element for a result file with displacements.

```
long St7GetElementDataDeformed(long uID, long Entity, long EntityNum, long
    ResultCase, double DispScale, double* EltData)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK or tyLINK.

`EntityNum`

Entity number.

`ResultCase`

Result case number.

`DispScale`

The absolute scale to be applied to the node displacements.

Output Parameters

`EltData`

Geometric data for the specified element:

tyBEAM – Distance between the two displaced nodes, irrespective of beam type.

tyPLATE – Plate deformed area; for curved offset plates, the area considers the offset.

tyBRICK – Brick deformed volume.

tyLINK – For links with two nodes, the distance between the displaced nodes; for links with more than two nodes, the sum of the distances between the displaced slave node and each displaced master node; for attachment links, the distance between the displaced node and the attachment point on the displaced target element.

Usage

A result file that produces displacements must be open when calling this function. The data is calculated based on the scaled element deformation for the requested result case.

St7GetPlateEdgeLengths

Returns the edge lengths of the specified plate element.

```
long St7GetPlateEdgeLengths(long uID, long PlateNum, long ResultCase,
    double* Lengths)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`ResultCase`

Result case number – if greater than 0, it must be a valid result case; if less than 1, it is ignored.

Output Parameters

`Lengths[0..3]`

Up to four edge lengths.

Usage

If a result file is open, Lengths is based on the birth coordinates of the element for the specified result case (for a staged analysis, birth coordinates can depend on the result case). If a result file is not open, or ResultCase is 0, Lengths is based on the initial position of the nodes.

St7GetPlateEdgeLengthsGNL

Returns the edge lengths of the specified plate element for geometrically nonlinear results.

```
long St7GetPlateEdgeLengthsGNL(long uID, long PlateNum, long ResultCase,
     double* Lengths)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`ResultCase`

Result case number.

Output Parameters

`Lengths[0..3]`

Up to four edge lengths.

Usage

A result file that includes geometric nonlinearity must be open when calling this function. The lengths are calculated based on the element deformation at an absolute displacement scale of 1 for the specified result case.

St7GetPlateEdgeLengthsDeformed

Returns the edge lengths of the specified plate element for a result file with displacements.

```
long St7GetPlateEdgeLengthsDeformed(long uID, long PlateNum, long ResultCase,
     double DispScale, double* Lengths)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`ResultCase`

Result case number.

`DispScale`

The absolute scale to be applied to the node displacements.

Output Parameters

`Lengths[0..3]`

Up to four edge lengths.

Usage

A result file that produces displacements must be open when calling this function. The lengths are then calculated based on the scaled element deformation for the requested result case.

St7GetBrickFaceAreas

Returns the face areas of the specified brick element.

```
long St7GetBrickFaceAreas(long uID, long BrickNum, long ResultCase,
     double* Areas)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`ResultCase`

Result case number – if greater than 0, it must be a valid result case; if less than 1, it is ignored.

Output Parameters

`Areas[0..5]`

Up to six face areas.

Usage

If a result file is open, Areas is based on the birth coordinates of the element for the specified result case (for a staged analysis, birth coordinates can depend on the result case). If a result file is not open, or ResultCase is 0, Areas is based on the initial position of the nodes.

St7GetBrickFaceAreasGNL

Returns the face areas of the specified brick element for geometrically nonlinear results.

```
long St7GetBrickFaceAreasGNL(long uID, long BrickNum, long ResultCase,
     double* Areas)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`ResultCase`

Result case number.

Output Parameters

`Areas[0..5]`

Up to six face areas.

Usage

A result file that includes geometric nonlinearity must be open when calling this function. The areas are calculated based on the element deformation at an absolute displacement scale of 1 for the specified result case.

St7GetBrickFaceAreasDeformed

Returns the face areas of the specified brick element for a result file with displacements.

```
long St7GetBrickFaceAreasDeformed(long uID, long BrickNum, long ResultCase,
     double DispScale, double* Areas)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`ResultCase`

Result case number.

`DispScale`

The absolute scale to be applied to the node displacements.

Output Parameters

`Areas[0..5]`

Up to six face areas.

Usage

A result file that produces displacements must be open when calling this function. The areas are calculated based on the scaled element deformation for the requested result case.

St7GetElementCentroid

Returns the coordinates of the geometric centroid of the specified element in its undeformed position.

```
long St7GetElementCentroid(long uID, long Entity, long EntityNum,
     long FaceEdgeNum, double* XYZ)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

`EntityNum`

Entity number.

`FaceEdgeNum`

Local face or edge number; one of 0, 1, 2, 3 or 4 for tyPLATE or 0, 1, 2, 3, 4, 5 or 6 for tyBRICK. Enter 0 to return the centroid of the whole element. See *Element Connections* for additional information.

Output Parameters

`XYZ[0..2]`

The global Cartesian coordinates of the centroid as a 3-element array.

St7GetElementCentroidAtBirth

Returns the coordinates of the geometric centroid of the specified element in its birth position when a result file is open.

```
long St7GetElementCentroidAtBirth(long uID, long Entity, long EntityNum,
     long FaceEdgeNum, long ResultCase, double* XYZ)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

`EntityNum`

Entity number.

`FaceEdgeNum`

Local face or edge number; one of 0, 1, 2, 3 or 4 for tyPLATE or 0, 1, 2, 3, 4, 5 or 6 for tyBRICK. Enter 0 to return the centroid of the whole element. See *Element Connections* for additional information.

`ResultCase`

Result case number.

Output Parameters

`XYZ[0..2]`

The global Cartesian coordinates of the centroid as a 3-element array.

St7GetElementCoordinatesAtBirth

Returns the coordinates of the nodes of the specified element in its birth position when a result file is open.

```
long St7GetElementCoordinatesAtBirth(long uID, long Entity, long EntityNum,
    long ResultCase, double* XYZ)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE or tyBRICK.

`EntityNum`

Entity number.

`ResultCase`

Result case number.

Output Parameters

`XYZ[0..3*N-1]`

The global Cartesian coordinates of N nodes as an array. The first three values in the array refer to the first node, the next three to the second node, and so on. The array must be long enough to cater for the number of nodes on the element. The maximum array length will be 60 to cater for a 20-node brick element.

Usage

The birth coordinates will be the same as the initial coordinates for an element that is not morphed. For a morphed element, the birth coordinates can change from result case to result case depending on the staging history of the element.

St7GetPlateUV

Returns the local plate uv coordinates corresponding to a global XYZ position. The XYZ position should be located approximately on the surface of the element.

```
long St7GetPlateUV(long uID, long PlateNum, double* XYZ, double* UV)
```

Input Parameters

`uID`

Strand7 model file ID.

`PlateNum`

Plate number.

`XYZ[0..2]`

A 3-element array containing the global XYZ coordinates of the point.

Output Parameters

`UV[0..1]`

A 2-element array containing the local uv plate coordinates. See *Plate Local Coordinates* for additional information.

St7GetBrickUVW

Returns the local brick uvw coordinates corresponding to a global XYZ position. The XYZ position should be located approximately within the brick.

```
long St7GetBrickUVW(long uID, long BrickNum, double* XYZ, double* UVW)
```

Input Parameters

`uID`

Strand7 model file ID.

`BrickNum`

Brick number.

`XYZ[0..2]`

A 3-element array containing the global XYZ coordinates of the point.

Output Parameters

`UVW[0..2]`

A 3-element array containing the local brick uvw coordinates. See *Brick Local Coordinates* for additional information.

St7PlateHullVolume

Returns the volume enclosed by the selected plates. If multiple enclosed volumes are present in the selection the sum of all volumes is returned. The selected plates must form a closed boundary surface around the region(s) for which the volume is to be calculated.

```
long St7PlateHullVolume(long uID, long ResultCase, double* Volume)
```

Input Parameters

`uID`

Strand7 model file ID.

`ResultCase`

Either the result case number, or 0 for the undeformed hull volume.

Output Parameters

`Volume`

The enclosed volume.

Dependencies

`Selection`

Plates can be selected using functions in *Entity Selection*.

St7GenerateAdjacencyList

Returns an index to an adjacency list that can be used to efficiently find the elements connected to a node.

```
long St7GenerateAdjacencyList(long uID, long* Integers, long* AdjacencyIndex)
```

Input Parameters

`uID`

Strand7 model file ID.

`Integers[0..6]`

`[ipAdjIncludeBeams]` – Include beam elements in the adjacency list; either btTrue or btFalse.

`[ipAdjIncludePlates]` – Include plate elements in the adjacency list; either btTrue or btFalse.

`[ipAdjIncludeBricks]` – Include brick elements in the adjacency list; either btTrue or btFalse.

`[ipAdjIncludeLinks]` – Include links in the adjacency list; either btTrue or btFalse.

`[ipAdjIncludeSelected]` – Include selected elements in the adjacency list; either btTrue or btFalse.

`[ipAdjIncludeUnselected]` – Include unselected elements in the adjacency list; either btTrue or btFalse.

`[ipAdjIgnoreBeamRefN]` – Ignore beam reference nodes in the adjacency list; either btTrue or btFalse.

Output Parameters

`AdjacencyIndex`

Index pointer to a new adjacency list.

Usage

Any number of adjacency lists can be generated for a model. Once generated, adjacency lists remain static. If the model is changed, the adjacency list is not automatically updated or invalidated; data can still be extracted from the list, even though it may no longer reflect the model. To update an adjacency list, it must be freed using *St7FreeAdjacencyList* and a new one generated using *St7GenerateAdjacencyList*. Adjacency lists are automatically freed when the model is closed.

St7GetNumElementsAtNode

Returns the number of elements connected to a node, based on a previously generated adjacency list.

```
long St7GetNumElementsAtNode(long uID, long AdjacencyIndex, long NodeNum,
     long* NumElements)
```

Input Parameters

`uID`

Strand7 model file ID.

`AdjacencyIndex`

Index pointer to a previously generated adjacency list.

`NodeNum`

Node number.

Output Parameters

`NumElements`

The number of elements connected to the node.

St7GetElementsAtNode

Returns the number and type of elements connected to a node based on a previously generated adjacency list.

```
long St7GetElementsAtNode(long uID, long AdjacencyIndex, long NodeNum,
     long* EntityNums, long* EntityTypes, long ArrayDim)
```

Input Parameters

`uID`

Strand7 model file ID.

`AdjacencyIndex`

Index pointer to a previously generated adjacency list.

`NodeNum`

Node number.

`ArrayDim`

Size of the arrays EntityNums and EntityTypes.

Output Parameters

`EntityNums[0..ArrayDim-1]`

`[0..MIN(ArrayDim-1,NumElements)]` – an array containing the element numbers for elements connected to NodeNum. The maximum number of values returned in the array will be the minimum of ArrayDim and the NumElements result returned by a call to *St7GetNumElementsAtNode*. If EntityNums[0] is returned as 0, no elements are connected to the node. If ArrayDim is greater than NumElements, EntityNums[NumElements] will be returned as 0 to signify the end of the list.

`EntityTypes[0..ArrayDim-1]`

`[0..MIN(ArrayDim-1,NumElements)]` – an array containing the element type (tyBEAM, tyPLATE, tyBRICK or tyLINK) for elements connected to NodeNum. The maximum number of values returned in the array will be the minimum of ArrayDim and the NumElements result returned by a call to *St7GetNumElementsAtNode*. If EntityTypes[0] is returned as 0, no elements are connected to the node. If ArrayDim is greater than NumElements, EntityTypes[NumElements] will be returned as 0 to signify the end of the list.

St7FreeAdjacencyList

Frees the specified adjacency list.

```
long St7FreeAdjacencyList(long uID, long AdjacencyIndex)
```

Input Parameters

`uID`

Strand7 model file ID.

`AdjacencyIndex`

An index pointer to a previously generated adjacency list.

Options

St7SetToolOptions

Sets the tool options for the specified model.

```
long St7SetToolOptions(long uID, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`Integers[0..21]`

`[ipToolOptsElementTolType]` – Element tolerance type; either ztAbsolute or ztRelative.

`[ipToolOptsGeometryAccuracyType]` – Geometry accuracy type; either ztAbsolute or ztRelative.

`[ipToolOptsGeometryFeatureType]` – Geometry feature length type; either ztAbsolute or ztRelative.

`[ipToolOptsZipMesh]` – Mesh zipping; one of zmAsNeeded, zmOnSave or zmOnRequest.

`[ipToolOptsNodeCoordinate]` – New node coordinates; one of ncAverage, ncLowerNode, ncHigherNode or ncSelectedNode.

`[ipToolOptsNodeAttributeKeep]` – Attribute keep; one of naLower, naHigher or naAccumulate.

`[ipToolOptsAllowZeroLengthLinks]` – Allow zero length links; either btTrue or btFalse.

`[ipToolOptsAllowZeroLengthBeams]` – Allow zero length beams; either btTrue or btFalse.

`[ipToolOptsSubdivideBeams]` – Subdivide only normal beams; either btTrue or btFalse.

`[ipToolOptsInterpSideAttachments]` – Interpolate beam side attachments when subdividing; either btTrue or btFalse.

`[ipToolOptsCompatibleTriangle]` – Compatible triangle faces; either btTrue or btFalse.

`[ipToolOptsAdjustMidsideNodes]` – Automatically adjust midside nodes when subdividing; either btTrue or btFalse.

`[ipToolOptsEvaluateFormulas]` – Automatically re-evaluate formulas when subdividing; either btTrue or btFalse.

`[ipToolOptsPlateAxisAlign]` – Axis alignment; either paCentroid or paCurvilinear.

`[ipToolOptsWedgeSubdivision]` – Wedge subdivision option; either wsUseAB or wsUseAC.

`[ipToolOptsCopyMode]` – Copy mode; either cmRoot or cmSibling.

`[ipToolOptsAutoCreateProperties]` – Auto create new properties; either btTrue or btFalse.

`[ipToolOptsInsertMPLNodes]` – Add/remove nodes on Multi-Point links; either btTrue or btFalse.

`[ipToolOptsConsiderDroopedCables]` – Subdivide drooped cables based on the catenary shape; either btTrue or btFalse.

`[ipToolOptsConsiderBeam3]` – Consider the third node of Beam3 elements; either btTrue or btFalse.

`Doubles[0..2]`

`[ipToolOptsElementTol]` – Element zip tolerance.

`[ipToolOptsGeometryAccuracy]` – Geometry accuracy.

`[ipToolOptsGeometryFeatureLength]` – Geometry feature length.

St7GetToolOptions

Returns the tool options assigned to the specified model.

```
long St7GetToolOptions(long uID, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Integers[0..21]`

`[ipToolOptsElementTolType]` – Element tolerance type; either ztAbsolute or ztRelative.

`[ipToolOptsGeometryAccuracyType]` – Geometry accuracy type; either ztAbsolute or ztRelative.

`[ipToolOptsGeometryFeatureType]` – Geometry feature length type; either ztAbsolute or ztRelative.

`[ipToolOptsZipMesh]` – Mesh zipping; one of zmAsNeeded, zmOnSave or zmOnRequest.

`[ipToolOptsNodeCoordinate]` – New node coordinates; one of ncAverage, ncLowerNode, ncHigherNode or ncSelectedNode.

`[ipToolOptsNodeAttributeKeep]` – Attribute keep; one of naLower, naHigher or naAccumulate.

`[ipToolOptsAllowZeroLengthLinks]` – Allow zero length links; either btTrue or btFalse.

`[ipToolOptsAllowZeroLengthBeams]` – Allow zero length beams; either btTrue or btFalse.

`[ipToolOptsSubdivideBeams]` – Subdivide only normal beams; either btTrue or btFalse.

`[ipToolOptsInterpSideAttachments]` – Interpolate beam side attachments when subdividing; either btTrue or btFalse.

`[ipToolOptsCompatibleTriangle]` – Compatible triangle faces; either btTrue or btFalse.

`[ipToolOptsAdjustMidsideNodes]` – Automatically adjust midside nodes when subdividing; either btTrue or btFalse.

`[ipToolOptsEvaluateFormulas]` – Automatically re-evaluate formulas when subdividing; either btTrue or btFalse.

[ipToolOptsPlateAxisAlign] – Axis alignment; either paCentroid or paCurvilinear.

[ipToolOptsWedgeSubdivision] – Wedge subdivision option; either wsUseAB or wsUseAC.

[ipToolOptsCopyMode] – Copy mode; either cmRoot or cmSibling.

[ipToolOptsAutoCreateProperties] – Auto create new properties; either btTrue or btFalse.

[ipToolOptsInsertMPLNodes] – Add/remove nodes on Multi-Point links; either btTrue or btFalse.

[ipToolOptsConsiderDroopedCables] – Subdivide drooped cables based on the catenary shape; either btTrue or btFalse.

[ipToolOptsConsiderBeam3] – Consider the third node of Beam3 elements; either btTrue or btFalse.

`Doubles[0..2]`

[ipToolOptsElementTol] – Element zip tolerance.

[ipToolOptsGeometryAccuracy] – Geometry accuracy.

[ipToolOptsGeometryFeatureLength] – Geometry feature length.

St7SetCleanMeshOptions

Specifies the settings used by *St7CleanMesh*.

```
long St7SetCleanMeshOptions(long uID, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

`Integers[0..19]`

[ipMeshToleranceType] – Tolerance type; either ztAbsolute or ztRelative.

[ipActOnWholeModel] – Clean whole model; either btTrue or btFalse.

[ipZipNodes] – Clean nodes; either btTrue or btFalse.

[ipRemoveDuplicateElements] – Remove duplicate elements; either btTrue or btFalse.

[ipFixElementConnectivity] – Repair element connectivity; either btTrue or btFalse.

[ipDeleteFreeNodes] – Delete unconnected nodes; either btTrue or btFalse.

[ipDeleteInvalidElements] – Delete invalid elements; either btTrue or btFalse.

[ipPackStringGroupIDs] – Pack beam string group IDs; either btTrue or btFalse.

[ipDoBeams] – Act on beam elements; either btTrue or btFalse.

[ipDoPlates] – Act on plate elements; either btTrue or btFalse.

[ipDoBricks] – Act on bricks; either btTrue or btFalse.

[ipDoLinks] – Act on links; either btTrue or btFalse.

[ipZeroLengthLinks] – Allow zero length links; either btTrue or btFalse.

[ipZeroLengthBeams] – Allow zero length beams; either btTrue or btFalse.

[ipNodeAttributeKeep] – Keep attributes from nodes; one of naLower, naHigher or naAccumulate.

[ipNodeCoordinates] – Move nodes; one of ncAverage, ncLowerNode, ncHigherNode or ncSelectedNode.

[ipAllowDifferentProps] – Allow duplicate elements of different properties; either btTrue or btFalse.

[ipAllowDifferentGroups] – Allow duplicate elements of different groups; either btTrue or btFalse.

[ipAllowDifferentBeamOffset] – Allow duplicate beam elements with different offset; either btTrue or btFalse.

[ipAllowDifferentPlateOffset] – Allow duplicate plate elements with different offset; either btTrue or btFalse.

Doubles[0..0]

[ipMeshTolerance] – Zip tolerance, scaled based on Integers[ipMeshToleranceType].

St7GetCleanMeshOptions

Returns the current settings used by *St7CleanMesh*.

```
long St7GetCleanMeshOptions(long uID, long* Integers, double* Doubles)
```

Input Parameters

uID

Strand7 model file ID.

Output Parameters

Integers[0..19]

[ipMeshToleranceType] – Tolerance type; either ztAbsolute or ztRelative.

[ipActOnWholeModel] – Clean whole model; either btTrue or btFalse.

[ipZipNodes] – Clean nodes; either btTrue or btFalse.

[ipRemoveDuplicateElements] – Remove duplicate elements; either btTrue or btFalse.

[ipFixElementConnectivity] – Repair element connectivity; either btTrue or btFalse.

[ipDeleteFreeNodes] – Delete unconnected nodes; either btTrue or btFalse.

[ipDeleteInvalidElements] – Delete invalid elements; either btTrue or btFalse.

[ipPackStringGroupIDs] – Pack beam string group IDs; either btTrue or btFalse.

[ipDoBeams] – Act on beam elements; either btTrue or btFalse.

[ipDoPlates] – Act on plate elements; either btTrue or btFalse.

[ipDoBricks] – Act on bricks; either btTrue or btFalse.

[ipDoLinks] – Act on links; either btTrue or btFalse.

[ipZeroLengthLinks] – Allow zero length links; either btTrue or btFalse.

[ipZeroLengthBeams] – Allow zero length beams; either btTrue or btFalse.

[ipNodeAttributeKeep] – Keep attributes from nodes; either naLower, naHigher or naAccumulate.

[ipNodeCoordinates] – Move nodes; one of ncAverage, ncLowerNode, ncHigherNode or ncSelectedNode.

[ipAllowDifferentProps] – Allow duplicate elements of different properties; either btTrue or btFalse.

[ipAllowDifferentGroups] – Allow duplicate elements of different groups; either btTrue or btFalse.

[ipAllowDifferentBeamOffset] – Allow duplicate beam elements with different offset; either btTrue or btFalse.

[ipAllowDifferentPlateOffset] – Allow duplicate plate elements with different offset; either btTrue or btFalse.

Doubles[0..0]

[ipMeshTolerance] – Zip tolerance, scaled based on Integers[ipMeshToleranceType].

St7SetCleanGeometryOptions

Specifies the settings used when performing subsequent geometry clean operations.

```
long St7SetCleanGeometryOptions(long uID, long* Integers, double* Doubles)
```

Input Parameters

uID

Strand7 model file ID.

Integers[0..4]

[ipGeometryFeatureType] – Feature tolerance type; either ztRelative or ztAbsolute.

[ipGeometryActOnWholeModel] – If btTrue, perform clean on whole model. If btFalse, perform clean on selected faces.

[ipGeometryFreeEdgesOnly] – Act on free edges only; either btTrue or btFalse.

[ipGeometryDuplicateFaces] – Duplicate face operation; one of dfLeaveAll, dfLeaveOne, dfLeaveNone.

[ipGeometryWithinGroups] – If btTrue, geometry zipping occurs between faces in the same group. If btFalse, geometry faces in different groups can also be zipped together.

Doubles[0..1]

[ipGeometryFeatureLength] – Geometry feature length.

`[ipGeometryEdgeMergeAngle]` – Merging angle for adjacent edges.

St7GetCleanGeometryOptions

Retrieves the current settings used when performing a geometry clean operation.

```
long St7GetCleanGeometryOptions(long uID, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Integers[0..4]`

`[ipGeometryFeatureType]` – Feature tolerance type; either ztRelative or ztAbsolute.

`[ipGeometryActOnWholeModel]` – If btTrue, perform clean on whole model. If btFalse, perform clean on selected faces.

`[ipGeometryFreeEdgesOnly]` – Act on free edges only; either btTrue or btFalse.

`[ipGeometryDuplicateFaces]` – Duplicate face operation; one of dfLeaveAll, dfLeaveOne, dfLeaveNone.

`[ipGeometryWithinGroups]` – If btTrue, geometry zipping occurs between faces in the same group. If btFalse, geometry faces in different groups can also be zipped together.

`Doubles[0..1]`

`[ipGeometryFeatureLength]` – Geometry feature length.

`[ipGeometryEdgeMergeAngle]` – Merging angle for adjacent edges.

St7SetResultOptions

Sets the **Results Options** for the specified model.

```
long St7SetResultOptions(long uID, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

`Integers[0..8]`

`[ipResOptsRotationUnit]` – Rotation units for model window output; either auRadian or auDegree. By default this setting is ignored by the Strand7 API; see *St7EnableModelRotationUnit*.

`[ipResOptsStrainUnit]` – Strain unit; one of suUnit, suPercent or suMicro. By default this setting is ignored by the Strand7 API; see *St7EnableModelStrainUnit*.

`[ipResOptsAddGNLDisp]` – btTrue to add initial GNL displacements.

`[ipResOptsOffsetDisp]` – btTrue to consider elements offset in displacement results.

`[ipResOptsNFADisp]` – Either dmUnitModalMass or dmEngModalMass.

`[ipResOptsReactionLinkGNL]` – btTrue to move reaction multi-point link origin by average node displacement of connected node, or btFalse to use the unmodified user-specified origin.

`[ipResOptsBaseDisp]` – Displacement results to present for transient analyses, bmRelative for **Relative to Base** or bmTotal for **Total**.

`[ipResOptsBaseVel]` – Velocity results to present for transient analyses, bmRelative for **Relative to Base** or bmTotal for **Total**.

`[ipResOptsBaseAcc]` – Acceleration results to present for transient analyses, bmRelative for **Relative to Base** or bmTotal for **Total**.

St7GetResultOptions

Returns the **Results Options** for the specified model.

```
long St7GetResultOptions(long uID, long* Integers)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Integers[0..8]`

`[ipResOptsRotationUnit]` – Rotation units for model window output; either auRadian or auDegree. By default this setting is ignored by the Strand7 API; see *St7EnableModelRotationUnit.*

`[ipResOptsStrainUnit]` – Strain unit; one of suUnit, suPercent or suMicro. By default this setting is ignored by the Strand7 API; see *St7EnableModelStrainUnit*.

`[ipResOptsAddGNLDisp]` – btTrue to add initial GNL displacements.

`[ipResOptsOffsetDisp]` – btTrue to consider elements offset in displacement results.

`[ipResOptsNFADisp]` – Either dmUnitModalMass or dmEngModalMass.

`[ipResOptsReactionLinkGNL]` – btTrue to move reaction multi-point link origin by average node displacement of connected node, or btFalse to use the unmodified user-specified origin.

`[ipResOptsBaseDisp]` – Displacement results to present for transient analyses, bmRelative for **Relative to Base** or bmTotal for **Total**.

`[ipResOptsBaseVel]` – Velocity results to present for transient analyses, bmRelative for **Relative to Base** or bmTotal for **Total**.

`[ipResOptsBaseAcc]` – Acceleration results to present for transient analyses, bmRelative for **Relative to Base** or bmTotal for **Total**.

St7GetInsituStressOptions

Returns the options used in the last execution of *St7InsituStress.*

```
long St7GetInsituStressOptions(long uID, long* Integers, double* Doubles)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`Integers[0..8]`

`[ipInsituGravityCase]` – Load case in which gravity is defined.

`[ipInsituFreedomCase]` – Freedom case.

`[ipInsituStageIndex]` – Stage at which to calculate the in-situ stress, or 0 to have all groups active.

`[ipInsituUseExisting]` – btTrue to use the existing in-situ stress distribution as the initial conditions.

`[ipInsituReplaceK0]` – Calculate horizontal stress ratio (K0) from stress results.

`[ipInsituMaxIterations]` – Iteration limit used by the solver.

`[ipInsituAllowIterations]` – btTrue to allow the solver to add iterations.

`[ipInsituSolverScheme]` – one of stSkyline, stSparse or stIterativePCG.

`[ipInsituMatrixSort]` – one of rnNone, rnTree, rnGeometry or rnAMD.

`Doubles[0..1]`

`[ipInsituDefaultFluidLevel]` – Default fluid level.

`[ipInsituDefaultFluidDensity]` – Default fluid mass density per unit volume.

Cleaning

St7DeleteUnusedNodes

Deletes nodes that are not referenced by the connectivity of any element.

```
long St7DeleteUnusedNodes(long uID, long* NumDeleted)
```

Input Parameters

`uID`

Strand7 model file ID.

Output Parameters

`NumDeleted`

Number of unused nodes deleted.

St7InvalidateElement

Marks the specified element as invalid to be subsequently removed using the *St7DeleteInvalidElements* function.

```
long St7InvalidateElement(long uID, long Entity, long EntityNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK or tyLINK.

`EntityNum`

Entity number.

St7DeleteInvalidElements

Deletes all elements marked as invalid from a Strand7 model.

```
long St7DeleteInvalidElements(long uID, long Entity, long* NumDeleted)
```

Input Parameters

`uID`

Strand7 model file ID.

`Entity`

One of tyBEAM, tyPLATE, tyBRICK or tyLINK.

Output Parameters

`NumDeleted`

Number of entities deleted.

St7InvalidateGeometryFaceCavityLoopID

Marks the specified cavity loop as invalid for subsequent deletion using the *St7DeleteInvalidGeometry* function. This function uses the loop ID number to identify the appropriate loop.

```
long St7InvalidateGeometryFaceCavityLoopID(long uID, long FaceNum, long LoopNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`LoopNum`

The ID number of the loop to be marked for deletion.

St7InvalidateGeometryFace

Marks the specified geometry face as invalid for subsequent deletion using the *St7DeleteInvalidGeometry* function.

```
long St7InvalidateGeometryFace(long uID, long FaceNum)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number to invalidate.

St7InvalidateGeometryFaceCavityLoopIndex

Marks the specified cavity loop as invalid for subsequent deletion using the *St7DeleteInvalidGeometry* function. This function uses the loop index number to identify the appropriate loop.

```
long St7InvalidateGeometryFaceCavityLoopIndex(long uID, long FaceNum,
    long LoopIndex)
```

Input Parameters

`uID`

Strand7 model file ID.

`FaceNum`

Face number.

`LoopIndex`

The index number of the loop to be marked for deletion.

St7DeleteInvalidGeometry

Deletes all invalid faces in the specified model.

```
long St7DeleteInvalidGeometry(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

St7CleanMesh

Performs a mesh cleaning operation on the Strand7 model using the current settings.

```
long St7CleanMesh(long uID)
```

Input Parameters

`uID`

Strand7 model file ID.

Dependencies

`Clean Mesh Options`

Assigned using *St7SetCleanMeshOptions*.

St7CleanGeometry

Performs a geometry clean operation on the geometry included in the specified Strand7 model. Cleaning the geometry can be used to improve geometry definitions and is typically recommended before subsequent geometry or meshing operations are conducted. The operation consists of a number of different stages including: vertex and curve zipping, duplicate face processing, curve and surface refitting and morphing.

```
long St7CleanGeometry(long uID, long* ChangesMade, long Mode)
```

Input Parameters

`uID`

Strand7 model file ID.

`Mode`

Controls the display of a progress bar; either ieQuietRun or ieProgressRun.

Output Parameters

`ChangesMade`

btTrue or btFalse.

Dependencies

`Clean Geometry Data`

Assigned using *St7GetCleanGeometryOptions*.

Global

St7GetGlobalLogicalValue

Returns a global logical variable, which is internal to the Strand7 application.

```
long St7GetGlobalLogicalValue(long Index, bool* Value)
```

Input Parameters

`Index`

One of the following global integer indexes.

Index	Notes
`lvFormulaParseError`	An error occurred evaluating an attribute formula.

Output Parameters

`Value`

Global boolean state variable.

St7GetGlobalIntegerValue

Returns a global integer state variable, which is internal to the Strand7 application.

```
long St7GetGlobalIntegerValue(long Index, long* Value)
```

Input Parameters

`Index`

One of the following global integer indexes.

Index	Notes
`ivAttachmentsCreated`	The number of successful attachments created by *St7CreateAttachments*.
`ivAttachmentsFailed`	The number of unsuccessful attachments attempted by *St7CreateAttachments.*
`ivAttributesApplied`	
`ivBeamsChanged`	
`ivBeamsCollapsed`	
`ivBeamsCreated`	

ivBeamsDeleted

ivBeamsFailed

ivBeamsMoved

ivBeamsSplit

ivBeamsSubdivided

ivBricksChanged

ivBricksCollapsed

ivBricksCreated

ivBricksDeleted

ivBricksFailed

ivBricksGraded

ivBricksMoved

ivBricksSplit

ivBricksSubdivided

ivClipboardBeams

ivClipboardBricks

ivClipboardFaces

ivClipboardLinks

ivClipboardLoadPaths

ivClipboardNodes

ivClipboardPlates

ivClipboardVertices

ivDuplicateBeamsDeleted

ivDuplicateBricksDeleted

`ivDuplicateLinksDeleted`	
`ivDuplicatePlatesDeleted`	
`ivEdgesMorphed`	The number of edges morphed.
`ivEdgesSubdivided`	
`ivFacesChanged`	The number of faces modified by a geometry operation.
`ivFacesCreated`	The number of faces created by a geometry operation.
`ivFacesDeleted`	
`ivFacesFailed`	The number of faces that could not be processed by a geometry operation.
`ivFacesMeshed`	
`ivFacesMoved`	
`ivFacesNotMeshed`	
`ivFacesPartiallyMeshed`	
`ivFilletsCreated`	
`ivFilletsFailed`	
`ivIntersectionsFound`	The number of edges intersected by an intersection operation.
`ivLinksChanged`	
`ivLinksCollapsed`	
`ivLinksCreated`	
`ivLinksDeleted`	
`ivLinksMoved`	
`ivLoadCasesCreated`	
`ivLoadPathsChanged`	
`ivLoadPathsCreated`	

`ivLoadPathsMoved`	
`ivLoftSeriesFound`	
`ivLoopsDeleted`	The number of deleted loops.
`ivNodesCreated`	
`ivNodesDeleted`	
`ivNodesMoved`	
`ivPatchPlatesCreated`	
`ivPlateEdgesAssigned`	The numbers of plate edges assigned by *St7ConvertPatchLoads*.
`ivPlateEdgesNotFullyAssigned`	The number of plate edges not fully assigned by *St7ConvertPatchLoads*.
`ivPlatesChanged`	
`ivPlatesCollapsed`	
`ivPlatesCreated`	
`ivPlatesDeleted`	
`ivPlatesFailed`	
`ivPlatesGraded`	
`ivPlatesMoved`	
`ivPlatesSplit`	
`ivPlatesSubdivided`	
`ivSeamsAdded`	The number of seams added by a geometry operation.
`ivSolidsMeshed`	
`ivSolidsNotMeshed`	
`ivSolidsPartiallyMeshed`	

`ivSolverTerminationCode`	The termination code of the most recent solve if the .dll solver is in use. See *Solver termination error codes* for possible return values.
`ivStringGroupsPacked`	
`ivTessellationsFailed`	The number of geometry faces that could not be rendered.
`ivUCSCreated`	The number of User Coordinate Systems (UCS) created.

Output Parameters

`Value`

Global integer state variable.

St7GetGlobalStringValue

Returns a global string, which is internal to the Strand7 application.

```
long St7GetGlobalStringValue(long Index, char* Value, long MaxStringLen)
```

Input Parameters

`Index`

Index	Notes
`svInfluenceCombinationLog`	Full filename with path of the load influence combination log file generated by calling *St7GenerateInfluenceCases*.

`MaxStringLen`

Amount of space allocated to Value.

Output Parameters

`Value`

Global string state variable.

St7ClearGlobalLogicalValues

Resets all the global logical values returned by *St7GetGlobalLogicalValue* to False.

```
long St7ClearGlobalLogicalValues()
```

St7ClearGlobalIntegerValues

Resets all the global integer values returned by *St7GetGlobalIntegerValue* to zero.

```
long St7ClearGlobalIntegerValues()
```

St7ClearGlobalStringValues

Resets all the global string values returned by *St7GetGlobalStringValue* to empty strings.

```
long St7ClearGlobalStringValues()
```

Helper Functions

St7RGBToColour

Generates the Strand7 24-bit colour representation for a colour defined by its RGB components. See also *RGB Colours*.

```
long St7RGBToColour(double Red, double Green, double Blue, long* Colour)
```

Input Parameters

`Red`

A value between 0.0 and 1.0 representing the red component of the colour.

`Green`

A value between 0.0 and 1.0 representing the green component of the colour.

`Blue`

A value between 0.0 and 1.0 representing the blue component of the colour.

Output Parameters

`Colour`

The 24-bit colour value as a long integer.

St7ColourToRGB

Extracts the RGB components from a Strand7 24-bit colour representation. See also *RGB Colours*.

```
long St7ColourToRGB(long Colour, double* Red, double* Green, double* Blue)
```

Input Parameters

`Colour`

The 24-bit colour value as a long integer.

Output Parameters

`Red`

A value between 0.0 and 1.0 representing the red component of the colour.

`Green`

A value between 0.0 and 1.0 representing the green component of the colour.

`Blue`

A value between 0.0 and 1.0 representing the blue component of the colour.

St7GetNumElementResultGaussPoints

Returns the number of Gauss points used to store result quantities for the specified entity type.

```
long St7GetNumElementResultGaussPoints(long Entity, long NumNodes,
     long* NumGauss)
```

Input Parameters

`Entity`

Either tyPLATE or tyBRICK.

`NumNodes`

Number of nodes for the element type.

Output Parameters

`NumGauss`

Number of result Gauss points.

St7ConvertElementResultNodeToGaussPoint

Converts element nodal results to Gauss point results via interpolation. It is important to specify the un-averaged nodal quantities in order to capture the true element Gauss point values.

```
long St7ConvertElementResultNodeToGaussPoint(long Entity, long NumNodes,
     long NumColumns, double* NodeDoubles, long* NumGauss, double* GaussDoubles)
```

Input Parameters

`Entity`

Either tyPLATE or tyBRICK.

`NumNodes`

Number of nodes in the element.

`NumColumns`

Number of result quantities contained in the NodeDoubles array.

`NodeDoubles[0..NumNodes*NumColumns-1]`

An array containing the nodal result quantities, arranged in NumNodes blocks of length NumColumns. The start of the i^{th} block, relating to the i^{th} node in the element's definition, is at `NodeDoubles[(i-1)*NumColumns]`.

Output Parameters

`NumGauss`

Number of result Gauss points for the element; a maximum of 9 for tyPLATE, or 27 for tyBRICK.

`GaussDoubles[0..NumGauss*NumColumns-1]`

An array containing the interpolated Gauss point result quantities, arranged in NumGauss blocks of length NumColumns. The start of the i^{th} block, relating to the i^{th} Gauss point in the element's definition, is at `GaussDoubles[(i-1)*NumColumns]`.

Type Definitions

Strand7 API functions all pass either base types (e.g. single byte booleans, four byte integers, eight byte floats, etc.), or arrays of these base types. The base types are defined (at some level) by all programming languages. The array types must be defined explicitly for languages such as Pascal. For other languages, a C-style pointer to the array is sufficient, provided that the array is of the same base type and is at least as long as the Pascal types indicated below. If the array allocates less space than these types, an access violation can occur in the Strand7 API function call.

Note that by default, both C and Pascal array indices are zero-based – this convention is followed below, where constants are defined to indicate array length. Frequently, the array size is defined by other constants.

Pascal Type	Base Type	Range
`Array2Doubles`	`Double`	`[0..1]`
`Array2Longint`	`Longint`	`[0..1]`
`Array3Doubles`	`Double`	`[0..2]`
`Array3Longint`	`Longint`	`[0..2]`
`Array4Doubles`	`Double`	`[0..3]`
`Array6Doubles`	`Double`	`[0..5]`
`Array6Longint`	`Longint`	`[0..5]`
`Array10Doubles`	`Double`	`[0..9]`
`Array12Doubles`	`Double`	`[0..11]`
`Array60Doubles`	`Double`	`[0..59]`
`AttributeDoublesArray`	`Double`	`[0..kMaxAttributeDoubles-1]`
`AttributeLogicalsArray`	`Boolean`	`[0..kMaxAttributeLogicals-1]`
`AttributeLongintArray`	`Longint`	`[0..kMaxAttributeLongint-1]`
`BeamDispArray`	`Double`	`[0..kMaxDisp-1]`
`BeamGeometryArrayBGL`	`Double`	`[0..kMaxBGLDimensions-1]`
`BeamReleasedArray`	`Boolean`	`[0..kMaxBeamRelease-1]`
`BeamReleaseDoublesArray`	`Double`	`[0..kMaxBeamRelease-1]`
`BeamResultArray`	`Double`	`[0..kMaxBeamResult-1]`
`BeamSectionArray`	`Double`	`[0..kNumBeamSectionData-1]`
`BrickResultArray`	`Double`	`[0..kMaxBrickResult-1]`
`CharString`	`AnsiChar`	`[0..kMaxStrLen]`
`ConnectionArray`	`Longint`	`[0..kMaxElementNode]`
`EntityStateArray`	`Longint`	`[0..kMaxEntity-1]`
`EntityTotalsArray`	`Longint`	`[0..kMaxEntityTotals-1]`
`FreedomCaseDefaultsArray`	`Longint`	`[0..5]`

LoadCaseDefaultsArray	Double	[0..12]
MaterialArray	Double	[0..kNumMaterialData-1]
ModalResultsArray	Double	[0..15]
NodeResultArray	Double	[0..5]
NodeResultExArray	Double	[0..13]
PlateResultArray	Double	[0..kMaxPlateResult-1]
SeismicCaseDefaultsArray	Double	[0..9]
UCSDoublesArray	Double	[0..kMaxUCSDoubles-1]
UnitsArray	Longint	[0..kLastUnit-1]

Error Codes

The following lists the error codes that may be encountered when using the Strand7 API. The error codes are of three varieties:

- Non-solver error codes, prefixed by ERR7_ and interrogated by *St7GetAPIErrorString*.
- These error codes are returned by general function calls.
- Solver setup error codes generated during the solver initialisation phase, prefixed by SE_ and interrogated by *St7GetSolverErrorString*. These error codes are returned by calls to *St7RunSolver* or *St7RunSolverProcess*.
- Solver termination error codes. If the .dll version of the solver is in use these can be accessed after a solve has terminated abnormally using *St7GetGlobalIntegerValue* with Index=ivSolverTerminationCode.

Non-solver error codes

Error Code	Additional Information
`ERR7_AnimationDimensionsTooLarge`	
`ERR7_AnimationDimensionsTooSmall`	
`ERR7_AnimationHandleOutOfRange`	
`ERR7_AnimationNotRunning`	
`ERR7_APIAlreadyInitialised`	Returned by functions that must be called before *St7Init*. An example is the function *St7SetIconSize*.
`ERR7_APINotInitialised`	*St7Init* has not been called successfully.
`ERR7_APIModuleNotLicensed`	
`ERR7_AutoMesherModuleNotLicensed`	
`ERR7_BeamIsNotBXS`	
`ERR7_CannotCalculateBXSData`	
`ERR7_CannotCombResFiles`	
`ERR7_CannotCommunicate`	Returned when Strand7 is running in NETWORK mode but cannot communicate with the NetHASP licence manager.
`ERR7_CannotEditSolverFiles`	The specified result file was created by the solver, and cannot be edited.
`ERR7_CannotFindNetworkLock`	Returned by *St7Init* when Strand7 is running in NETWORK mode but cannot find the NETWORK hardware lock.
`ERR7_CannotFindStandaloneLock`	Returned by *St7Init* when Strand7 is running in STANDALONE mode but cannot find the STANDALONE hardware lock.
`ERR7_CannotInitialiseDirectX`	Windowing function cannot initialise the DirectX graphics engine.

ERR7_CannotMakeBXS	
ERR7_CannotMorphEdges	
ERR7_CannotOpenResultFile	
ERR7_CannotReadBXS	
ERR7_CannotReadFile	An I/O error occurred reading the file.
ERR7_CannotReadImportFile	
ERR7_CannotSaveFile	
ERR7_CannotSaveImageFile	
ERR7_CannotSaveIniFile	Could not save user configuration settings .ini file.
ERR7_CannotSetWindowParent	Cannot set the specified model window parent.
ERR7_CannotWriteExportFile	
ERR7_CantDoWithModalWindows	Operation cannot be performed when modal dialogs are open.
ERR7_CantGenerateFillet	
ERR7_CaseNameAlreadyExists	Case names must uniquely identify load and freedom cases.
ERR7_CavityFluidNotConstBulk	
ERR7_CavityFluidNotIdealGas	
ERR7_CombinationDoesNotExist	
ERR7_CommentDoesNotExist	
ERR7_CompositesModuleNotLicensed	
ERR7_ContourFileNotLoaded	
ERR7_CouldNotCreateModelWindow	
ERR7_CouldNotDestroyModelWindow	
ERR7_CouldNotSaveAnimationFile	
ERR7_CouldNotShowModelWindow	
ERR7_CreepIDAlreadyExists	
ERR7_DataNotFound	The requested output (e.g. attribute information) does not exist.
ERR7_DynamicsSolverModuleNotLicensed	
ERR7_EquationDoesNotExist	
ERR7_ErrorCreatingImage	Unable to create image, for example, due to insufficient RAM.
ERR7_ExceededMaxNumColumns	
ERR7_ExceededMaxNumCombEnvelopes	

`ERR7_ExceededMaxNumCombResFiles`	The maximum number of results files for combination is exceeded.
`ERR7_ExceededMaxNumEnvelopeSets`	
`ERR7_ExceededMaxNumFactorsEnvelopes`	
`ERR7_ExceededMaxNumLimitEnvelopes`	
`ERR7_ExceededMaxNumLoadPathTemplates`	
`ERR7_ExceededMaxNumLoadPathVehicles`	
`ERR7_ExceededMaxNumNodeHistory`	
`ERR7_ExceededMaxNumPlies`	
`ERR7_ExceededMaxNumRows`	
`ERR7_ExceededMaxNumSpectralCases`	
`ERR7_ExceededMaxNumStages`	
`ERR7_ExceededResultCase`	The result case input exceeds the total number of result cases.
`ERR7_ExceededTotal`	The entity number input exceeds the total number of such entities.
`ERR7_FileAlreadyOpen`	
`ERR7_FileIsNewer`	The file that *St7OpenFile* has attempted to open was created by a later version of Strand7 than the loaded Strand7 API.
`ERR7_FileIsProtected`	
`ERR7_FileNotFound`	
`ERR7_FileNotOpen`	The Strand7 file identified by uID is not open.
`ERR7_FileNotSt7`	
`ERR7_FilesStillOpen`	
`ERR7_FreeEdgesFound`	
`ERR7_FunctionalityNotAvailable`	The requested function is not available in this instance of the API.
`ERR7_FunctionNotSupported`	The requested function has been retired in the loaded version of the API, and has no effect on its arguments or the state of the API (with the exception of re-setting *St7GetLastError*).
`ERR7_GroupIdDoesNotExist`	
`ERR7_IncompatibleCriterionCombination`	The combination of elastic/plastic behaviour with this stress criterion is unsupported.
`ERR7_IncompatibleMaterialCombination`	The specified material is not supported by this plate type.

ERR7_IncompatibleResultFile	The opened result file does not support the requested result.
ERR7_IncompatibleSections	
ERR7_IncompatibleTableType	The specified property does not set this table type.
ERR7_IncrementDoesNotExist	
ERR7_InsituCalculationFailed	Returned by *St7InsituStress.*
ERR7_InsufficientFrames	
ERR7_IntersectionNotFound	
ERR7_InvalidAlphaTempType	
ERR7_InvalidAnimationFile	
ERR7_InvalidAnimationMode	
ERR7_InvalidAnimationType	
ERR7_InvalidAnsysArrayStatus	
ERR7_InvalidAnsysEndReleaseOption	
ERR7_InvalidAnsysExportFormat	
ERR7_InvalidAnsysExportUnits	
ERR7_InvalidAnsysImportFormat	
ERR7_InvalidArcLength	
ERR7_InvalidAttachConnectionType	
ERR7_InvalidAttachmentDirection	
ERR7_InvalidAttachmentType	
ERR7_InvalidAttachPartsParams	
ERR7_InvalidAttributeSetting	
ERR7_InvalidAttributeType	
ERR7_InvalidAveragingOption	
ERR7_InvalidAxis	
ERR7_InvalidAxisSystem	
ERR7_InvalidBackgroundMode	
ERR7_InvalidBaseExcitationType	
ERR7_InvalidBeamAxisType	
ERR7_InvalidBeamDir	
ERR7_InvalidBeamEnd	
ERR7_InvalidBeamExtrudeTarget	
ERR7_InvalidBeamLoadType	
ERR7_InvalidBeamPosition	The position indicated on the beam is invalid.

ERR7_InvalidBeamSectionType	
ERR7_InvalidBeamType	
ERR7_InvalidBGLData	
ERR7_InvalidBrickFace	
ERR7_InvalidBrickIntegrationPoints	The number of Gauss points specified in a given direction is invalid (maximum of 3).
ERR7_InvalidCementHardeningType	
ERR7_InvalidCollectionID	
ERR7_InvalidCombEnvelope	The specified combination envelope is not valid.
ERR7_InvalidCombEnvelopeAccType	The combination envelope accumulation type is not valid.
ERR7_InvalidCombEnvelopeType	
ERR7_InvalidCombinationCaseNumber	
ERR7_InvalidCombResFile	The specified result file for combination is invalid.
ERR7_InvalidComponent	
ERR7_InvalidConnectionType	
ERR7_InvalidContactSubType	
ERR7_InvalidContactType	
ERR7_InvalidContactYieldType	
ERR7_InvalidContourFileIndex	
ERR7_InvalidContourIndex	
ERR7_InvalidCoupleType	The specified couple type for the coupling/multi-point link is invalid.
ERR7_InvalidCreepFunctionType	The specified creep function/chain type is not valid.
ERR7_InvalidCreepHardeningLaw	
ERR7_InvalidCreepID	
ERR7_InvalidCreepLaw	
ERR7_InvalidCreepShrinkageType	
ERR7_InvalidCreepViscoChainRow	
ERR7_InvalidCurvedPipesAsOption	
ERR7_InvalidCutoffType	
ERR7_InvalidDampingType	
ERR7_InvalidDefaultsMode	
ERR7_InvalidDetachMode	
ERR7_InvalidDiagramAxis	
ERR7_InvalidDigits	

ERR7_InvalidDirection	
ERR7_InvalidDirectionVector	
ERR7_InvalidDisplayOptionsPath	The configuration file folder is not valid.
ERR7_InvalidDivisionParameters	
ERR7_InvalidDivisions	
ERR7_InvalidDivisionTargets	
ERR7_InvalidDLLsPresent	SlvPanel.dll and/or St6List.dll are incompatible with St7API.dll.
ERR7_InvalidDrawParameters	Invalid entity display parameters.
ERR7_InvalidDuplicateFaceType	
ERR7_InvalidDXFBeamOption	
ERR7_InvalidDXFPlateOption	
ERR7_InvalidEdgeTolerance	
ERR7_InvalidEntity	The specified entity type is invalid.
ERR7_InvalidEntityContourFileType	
ERR7_InvalidEntityID	
ERR7_InvalidEntityNodes	The number of nodes specified for the entity is invalid.
ERR7_InvalidEntityNumber	The entity number is not in the correct range.
ERR7_InvalidEntitySet	
ERR7_InvalidEnvelopeSet	
ERR7_InvalidEnvelopeSetType	
ERR7_InvalidErrorCode	An invalid error code was passed to *St7GetAPIErrorString* or *St7GetSolverErrorString*.
ERR7_InvalidExponentFormat	
ERR7_InvalidExportParameters	
ERR7_InvalidFaceFromBeamPolygonParameters	
ERR7_InvalidFaceSurface	
ERR7_InvalidFactorsEnvelope	
ERR7_InvalidFactorsEnvelopeType	
ERR7_InvalidFileName	
ERR7_InvalidFilePath	
ERR7_InvalidFileUnit	The Strand7 file uID is invalid. The valid range is 1 to 32 inclusive.
ERR7_InvalidFontName	
ERR7_InvalidFreedomCase	

`ERR7_InvalidFreedomCaseType`	
`ERR7_InvalidFrequencyType`	
`ERR7_InvalidGeometryAsOption`	
`ERR7_InvalidGeometryCavityLoop`	
`ERR7_InvalidGeometryEdgeType`	
`ERR7_InvalidGeometryFormatProtocol`	
`ERR7_InvalidGradeRatio`	
`ERR7_InvalidGradeType`	
`ERR7_InvalidGravityDirection`	
`ERR7_InvalidHardeningType`	
`ERR7_InvalidHarmonicLoadType`	
`ERR7_InvalidHRAMode`	
`ERR7_InvalidImageDimensions`	
`ERR7_InvalidImageType`	
`ERR7_InvalidImportExportMode`	
`ERR7_InvalidIndex`	
`ERR7_InvalidInfluenceFile`	
`ERR7_InvalidInitialCaseNumber`	The result case number is not valid for this result file.
`ERR7_InvalidInitialConditionsType`	
`ERR7_InvalidInitialFile`	The specified result file for initial conditions is invalid.
`ERR7_InvalidInsituRunMode`	Returned by *St7InsituStress.*
`ERR7_InvalidIterationNumber`	
`ERR7_InvalidK0Expression`	
`ERR7_InvalidK1Expression`	
`ERR7_InvalidLaminateID`	
`ERR7_InvalidLayoutID`	
`ERR7_InvalidLength`	
`ERR7_InvalidLibraryID`	
`ERR7_InvalidLibraryItemID`	
`ERR7_InvalidLibraryItemName`	
`ERR7_InvalidLibraryName`	
`ERR7_InvalidLibraryPath`	
`ERR7_InvalidLibraryType`	
`ERR7_InvalidLimitEnvelope`	
`ERR7_InvalidLimitEnvelopeType`	

`ERR7_InvalidLineDefinition`	
`ERR7_InvalidLineID`	
`ERR7_InvalidLinePoints`	
`ERR7_InvalidLinkData`	
`ERR7_InvalidLinkTarget`	
`ERR7_InvalidLinkType`	
`ERR7_InvalidLoadCase`	
`ERR7_InvalidLoadCaseFilePath`	The ANSYS load case file folder is not valid.
`ERR7_InvalidLoadCaseType`	
`ERR7_InvalidLoadID`	
`ERR7_InvalidLoadPath`	
`ERR7_InvalidLoadPathID`	
`ERR7_InvalidLoadPathLane`	
`ERR7_InvalidLoadPathLaneFactorType`	
`ERR7_InvalidLoadPathShape`	
`ERR7_InvalidLoadPathSurface`	
`ERR7_InvalidLoadPathTemplateID`	
`ERR7_InvalidLoadPathVehicle`	
`ERR7_InvalidLoadPathVehicleInstance`	
`ERR7_InvalidLTAMethod`	The specified linear transient solver method is invalid.
`ERR7_InvalidLTASolutionType`	
`ERR7_InvalidMarkerLineThickness`	
`ERR7_InvalidMarkerSize`	
`ERR7_InvalidMarkerStyle`	
`ERR7_InvalidMarkerType`	
`ERR7_InvalidMaterialType`	
`ERR7_InvalidMatrixType`	
`ERR7_InvalidMeshPositionOnUCS`	
`ERR7_InvalidMirrorOption`	
`ERR7_InvalidMobilityType`	
`ERR7_InvalidModalFile`	The specified modal results file is invalid.
`ERR7_InvalidModalLoadType`	
`ERR7_InvalidModalNodeReactType`	
`ERR7_InvalidModeNumber`	
`ERR7_InvalidModType`	The specified time-dependent modulus type is invalid.

ERR7_InvalidMultiPointLink	
ERR7_InvalidMultiPointType	
ERR7_InvalidMultiVariableCaseID	
ERR7_InvalidMultiVariableType	
ERR7_InvalidName	The specified name is not valid.
ERR7_InvalidNodeCoordinateKeepType	
ERR7_InvalidNodeExtrudeTarget	
ERR7_InvalidNumBeamStations	
ERR7_InvalidNumberOfEntries	The specified number of table entries is invalid.
ERR7_InvalidNumCopies	
ERR7_InvalidNumCutFaces	
ERR7_InvalidNumericStyle	
ERR7_InvalidNumLayers	The number of plate integration layers is invalid (less than 1 or greater than 100).
ERR7_InvalidNumMeshingLoops	
ERR7_InvalidNumModes	
ERR7_InvalidNumPathDivs	
ERR7_InvalidNumRepeats	
ERR7_InvalidNumSteps	
ERR7_InvalidOption	
ERR7_InvalidOriginMethod	
ERR7_InvalidP1P2	
ERR7_InvalidP1P2P3	
ERR7_InvalidP1P2P3P4	
ERR7_InvalidParameters	One or more parameters passed to the function are invalid.
ERR7_InvalidPasteOption	
ERR7_InvalidPatchType	The specified load patch type is invalid.
ERR7_InvalidPatchTypeForPlate	The specified plate does not support this load patch type.
ERR7_InvalidPathDefinition	
ERR7_InvalidPlane	
ERR7_InvalidPlaneID	
ERR7_InvalidPlanePoints	
ERR7_InvalidPlateEdge	
ERR7_InvalidPlateSurface	

`ERR7_InvalidPlateType`	
`ERR7_InvalidPLTarget`	
`ERR7_InvalidPositionTableAxis`	The specified axis for use with a Factor vs Position table is invalid.
`ERR7_InvalidPositionType`	
`ERR7_InvalidPreLoadType`	
`ERR7_InvalidProcessingMode`	
`ERR7_InvalidProjectFlag`	The projection type relating to the application of global pressure attributes is invalid. It should be one of ppNone, ppProjResultant or ppProjComponents.
`ERR7_InvalidProjectionDirection`	
`ERR7_InvalidPropertyNumber`	
`ERR7_InvalidQuadraticAsOption`	
`ERR7_InvalidR1R2`	
`ERR7_InvalidR2`	
`ERR7_InvalidRadius`	
`ERR7_InvalidRayleighMode`	
`ERR7_InvalidRCLayers`	
`ERR7_InvalidReferenceNode`	
`ERR7_InvalidRegionalSettings`	The system settings under **Region and Language** are invalid.
`ERR7_InvalidResOptsBaseMode`	
`ERR7_InvalidResOptsNFADisp`	
`ERR7_InvalidResOptsReactionLinkGNL`	
`ERR7_InvalidResOptsRotationUnit`	
`ERR7_InvalidResOptsStrainUnit`	
`ERR7_InvalidResponseType`	
`ERR7_InvalidResponseVariable`	
`ERR7_InvalidResultCase`	
`ERR7_InvalidResultFile`	
`ERR7_InvalidResultQuantity`	
`ERR7_InvalidResultsSign`	The specified results sign for spectral results is invalid.
`ERR7_InvalidResultSubQuantity`	
`ERR7_InvalidResultType`	
`ERR7_InvalidRigidPlane`	The specified plane of action for the rigid link is invalid.

ERR7_InvalidRubberModel	
ERR7_InvalidScaleAbout	
ERR7_InvalidScratchPath	
ERR7_InvalidSectionParameters	
ERR7_InvalidSectionPosition	The specified position does not lie on the beam cross section.
ERR7_InvalidSectionProperties	
ERR7_InvalidSegmentsPerCircle	
ERR7_InvalidSeismicCase	
ERR7_InvalidSelectionEndEdgeFace	
ERR7_InvalidSolverMode	
ERR7_InvalidSolverParameter	
ERR7_InvalidSolverPath	
ERR7_InvalidSolverScheme	The solver storage scheme is not valid.
ERR7_InvalidSolverType	The specified solver is either unknown or not valid.
ERR7_InvalidSortMethod	
ERR7_InvalidSortOption	
ERR7_InvalidSourceAction	
ERR7_InvalidSpectralCase	
ERR7_InvalidSpectrumType	
ERR7_InvalidSplitData	
ERR7_InvalidSplitRatio	
ERR7_InvalidSt7ExportFormat	
ERR7_InvalidStaadCountryCodeOption	
ERR7_InvalidStaadForceUnit	
ERR7_InvalidStaadLengthUnit	
ERR7_InvalidStartEndTimes	
ERR7_InvalidSTLBeamOption	
ERR7_InvalidSTLFileFormat	
ERR7_InvalidSTLGroupingOption	
ERR7_InvalidSTLPlateOption	
ERR7_InvalidStringID	The specified string group ID is invalid.
ERR7_InvalidSurfaceMeshTargetType	
ERR7_InvalidTableID	
ERR7_InvalidTableName	

ERR7_InvalidTableRow	
ERR7_InvalidTableSetting	
ERR7_InvalidTableType	
ERR7_InvalidTaperAxis	
ERR7_InvalidTaperRatio	
ERR7_InvalidTaperType	
ERR7_InvalidTempDependenceType	The setting for property temperature dependence is invalid.
ERR7_InvalidTemperatureType	
ERR7_InvalidTimeRow	
ERR7_InvalidTimeUnit	
ERR7_InvalidTolerance	
ERR7_InvalidToleranceType	
ERR7_InvalidToolOptsCopyOptions	
ERR7_InvalidToolOptsSubdivideOptions	
ERR7_InvalidToolOptsZipOptions	
ERR7_InvalidTransientTempType	The transient temperature specification is invalid.
ERR7_InvalidTrigType	
ERR7_InvalidUCSID	
ERR7_InvalidUCSIndex	
ERR7_InvalidUCSType	
ERR7_InvalidUnits	
ERR7_InvalidUserEquation	
ERR7_InvalidUVPos	
ERR7_InvalidVectorComponents	
ERR7_InvalidVertexMeshSize	
ERR7_InvalidVertexType	
ERR7_InvalidWindowDimensions	
ERR7_InvalidWindowMode	
ERR7_LaminateIDAlreadyExists	
ERR7_LaminateNameAlreadyExists	Laminate names must be unique.
ERR7_LayoutIDAlreadyExists	
ERR7_LinkNotAttachment	
ERR7_LinkNotCoupling	
ERR7_LinkNotMasterSlave	

`ERR7_LinkNotMultiPoint`	
`ERR7_LinkNotPinned`	
`ERR7_LinkNotRigid`	
`ERR7_LinkNotSectorSymmetry`	
`ERR7_LinkNotShrink`	
`ERR7_LinkNotTwoPoint`	
`ERR7_LoadPathIDAlreadyExists`	
`ERR7_LoadPathTemplateIDAlreadyExists`	
`ERR7_LoginExceeded`	Returned by *St7Init* when the maximum number of licences are in use.
`ERR7_MarkerNotFound`	
`ERR7_MaterialIsUserDefined`	Data is set/get that is irrelevant for the user-defined beam type.
`ERR7_MaterialNotAnisotropic`	
`ERR7_MaterialNotFluid`	
`ERR7_MaterialNotIsotropic`	
`ERR7_MaterialNotLaminate`	
`ERR7_MaterialNotOrthotropic`	
`ERR7_MaterialNotRubber`	
`ERR7_MaterialNotSoil`	
`ERR7_MaterialNotUserDefined`	
`ERR7_MeshingErrors`	The automesher has generated an error.
`ERR7_ModelMixesAxiNonAxi`	
`ERR7_ModelWindowWasNotCreated`	The function requires a model window be opened by *St7CreateModelWindow*.
`ERR7_ModelWindowWasNotShowing`	
`ERR7_MovingLoadModuleNotLicensed`	
`ERR7_NoActiveResponseVariables`	Returned by *St7GenerateInfluenceCases.*
`ERR7_NodeHistoryDoesNotExist`	
`ERR7_NoElementsOnLoadPaths`	Returned by *St7GenerateInfluenceCases.*
`ERR7_NoError`	No error occurred in the API call. `ERR7_NoError = 0`.
`ERR7_NoInfluenceCombinationsDefined`	Returned by *St7GenerateInfluenceCases.*
`ERR7_NoLoadPathsFound`	Returned by *St7GenerateInfluenceCases.*
`ERR7_NoMultiVariableInfluenceCases`	
`ERR7_NonlinearSolverModuleNotLicensed`	

`ERR7_NoPlateElements`	The model contains no plate elements, which are necessary to define a BXS.
`ERR7_NoResponsesFound`	Returned by *St7GenerateInfluenceCases.*
`ERR7_NoSoilElementsFound`	Returned by *St7InsituStress.*
`ERR7_NotFrequencyTable`	
`ERR7_NothingSelected`	
`ERR7_OnlyOneFreedomCase`	A model must contain at least one freedom case, it cannot be deleted.
`ERR7_OnlyOneLoadCase`	A model must contain at least one load case, it cannot be deleted.
`ERR7_OperationFailed`	
`ERR7_OperationUserTerminated`	Returned by *St7InsituStress.*
`ERR7_PlateDoesNotHaveLayers`	The plate property does not require integration layers.
`ERR7_PlateDoesNotHaveThickness`	
`ERR7_PlyDoesNotExist`	
`ERR7_PropertyAlreadyExists`	
`ERR7_PropertyNotBeam`	The specified beam property type is not a flexural beam.
`ERR7_PropertyNotCable`	
`ERR7_PropertyNotConnectionBeam`	
`ERR7_PropertyNotCutOffBar`	
`ERR7_PropertyNotPipe`	
`ERR7_PropertyNotPointContact`	
`ERR7_PropertyNotSpring`	
`ERR7_PropertyNotTruss`	
`ERR7_PropertyNotUserDefinedBeam`	
`ERR7_PseudoTimeNotDefined`	
`ERR7_RayleighNotApplicable`	
`ERR7_RCModuleNotLicensed`	
`ERR7_ReducedAnimation`	Insufficient memory for complete animation.
`ERR7_ResFileAlreadyOpen`	Custom result file is already open.
`ERR7_ResFileAssociationNotAllowed`	Load and freedom case association is not supported by this custom result file type.
`ERR7_ResFileCantClearQuantity`	Cannot clear the custom result quantity.
`ERR7_ResFileCantSave`	Cannot save the custom result file.

`ERR7_ResFileContainsNoElements`	The custom result file contains no elements.
`ERR7_ResFileContainsNoNodes`	The custom result file contains no nodes.
`ERR7_ResFileDoesNotHaveEntity`	No such entity on which to define the specified custom result exists.
`ERR7_ResFileIncompatibleQuantity`	The specified quantity is not compatible with the custom result file.
`ERR7_ResFileInvalidCase`	The case number is invalid for the custom result file.
`ERR7_ResFileInvalidNumCases`	
`ERR7_ResFileInvalidQuantity`	
`ERR7_ResFileNotOpen`	Custom result file is not open.
`ERR7_ResFileQuantityNotExist`	
`ERR7_ResFileUnsupportedType`	Custom result file type is unsupported.
`ERR7_ResultCaseNotInertiaRelief`	
`ERR7_ResultFileIsOpen`	Result files must be closed with *St7CloseResultFile* for legitimate usage.
`ERR7_ResultFileNotOpen`	A result file must be opened with *St7OpenResultFile* for legitimate usage.
`ERR7_ResultIsNotAvailable`	
`ERR7_ResultQuantityNotAvailable`	The requested result was not calculated or stored in the open result file.
`ERR7_SectionCannotBeMirrored`	
`ERR7_SectionNotBGL`	
`ERR7_SoilTypeNotCC`	The specified soil type of the property is not Cam-Clay.
`ERR7_SoilTypeNotDC`	The specified soil type of the property is not Duncan-Chang.
`ERR7_SoilTypeNotDP`	The specified property type does not use a Drucker-Prager soil material model.
`ERR7_SoilTypeNotLS`	The specified property does not use a linear elastic soil material model.
`ERR7_SoilTypeNotMC`	The specified property does not use a Mohr-Coulomb soil material model.
`ERR7_SolverStillRunning`	
`ERR7_SparseSolverModuleNotLicensed`	
`ERR7_StageDoesNotExist`	
`ERR7_TableDoesNotExist`	
`ERR7_TableNameAlreadyExists`	

ERR7_TableTypeIsNotTimeBased	
ERR7_TJunctionsFound	
ERR7_TooManyAnimations	
ERR7_TooManyBeamStations	
ERR7_UCSIDAlreadyExists	
ERR7_UCSMustBeDifferent	
ERR7_UnexpectedSolverTermination	The solve was not able to run to expected completion. If the .dll solver is in use (that is, *St7SetUseSolverDLL* has been called with UseDLL=True), *St7GetGlobalIntegerValue* can be used with ivSolverTerminationCode to access the specific solver error code.
ERR7_UnknownError	
ERR7_UnknownFileType	
ERR7_UnknownProperty	
ERR7_UnknownResultLocation	The input argument indicating position or location on the element is not recognised.
ERR7_UnknownResultType	The input argument ResultType is not recognised.
ERR7_UnknownSubType	The input argument ResultSubType is not recognised.
ERR7_UnknownSurfaceLocation	The input argument Surface, for plates, is not recognised.
ERR7_UnknownTitle	The input argument TitleType is not recognised.
ERR7_YieldNotMCDP	The yield criterion is not Mohr Coulomb or Drucker Prager.

Solver error codes

Error Code	**Additional Information**
SE_ActiveStageHasNoIncrements	At least one load increment must be defined for every active stage.
SE_AttachmentsInWrongGroup	One or more attachment links are active in stages where their targets are inactive.
SE_BadTaperData	
SE_BeamPoissonOutOfRange	
SE_BeamPropertiesMayHaveChanged	
SE_BeamRequiresPoisson	
SE_CableRequiresGNL	
SE_CableRequiresNonlinearSolver	
SE_CannotAppendToFile	

SE_CannotConvertAttachmentLink	Attachment link is not valid as it generates a singular matrix.
SE_CannotConvertInterpMultiPoint	Multi-point link generated a singular matrix.
SE_CannotFindSolver	
SE_CannotOverwriteFile	
SE_CannotReadRestartFile	
SE_CannotReadWriteScratchPath	The scratch path does not have sufficient read/write access to allow the solver to run.
SE_CannotWriteToLogFile	
SE_CannotWriteToResultFile	
SE_CompositesModuleNotLicensed	
SE_ConcreteCreepMNL	Concrete creep and material stress-strain tables cannot be considered together.
SE_CQCRequiresDamping	
SE_CreepTimeTooShort	
SE_DuplicateLinks	
SE_ElementUsesInvalidProperty	
SE_HarmonicFactorsAllZero	
SE_HaveLinearCables	
SE_InactiveCavityControlCase	
SE_IncompatibleRestartFile	
SE_IncompatibleRestartUnits	
SE_InitialConditionsNotValid	
SE_InitialSolutionFileIsBad	
SE_InsufficientRestartFileSteps	The restart file contains fewer result cases than the requested restart case.
SE_InvalidBrickCohesionValue	
SE_InvalidBrickShrinkageDefinition	
SE_InvalidCavityFluidDefinition	
SE_InvalidDirectionVector	
SE_InvalidElement	
SE_InvalidElements	
SE_InvalidFrequencyRange	
SE_InvalidGravityCase	The load case selected as the soil/fluid gravity case is not valid.
SE_InvalidInitialFile	
SE_InvalidInitialTemperatureFile	

SE_InvalidLaminateID	
SE_InvalidLink	
SE_InvalidMaterialNonlinearString	For material nonlinearity, all elements in a string group must use the same property set.
SE_InvalidPlateCohesionValue	
SE_InvalidPlateShrinkageDefinition	
SE_InvalidPlateVariableRequested	Plate(s) have one or more invalid response variables assigned.
SE_InvalidPreTensionOnString	A string group with variable pre-tension was found.
SE_InvalidRayleighFactors	
SE_InvalidRestartFile	
SE_InvalidSolverResultFile	
SE_InvalidStringGroupDefinition	
SE_InvalidTimeStep	
SE_InvalidUserBrickCreepDefinition	
SE_InvalidUserPlateCreepDefinition	
SE_LinksHaveNoFreedomCase	
SE_LoadIncrementsNotDefined	
SE_MissingInsituStress	
SE_ModelMixesAxiNonAxi	The model mixes axisymmetric elements with non-axisymmetric elements.
SE_MoreLoadIncrementsNeeded	
SE_MovingLoadModuleNotLicensed	
SE_NeedElementNodeForce	
SE_NeedNodeTempNTASolver	Table Type nodal temperatures are not supported by the linear transient dynamic solver.
SE_NeedNonlinearHeatSolver	
SE_NeedTemperatureDependence	The model contains temperature dependent creep data, but temperature dependence has not been set.
SE_NeedTemperatureTables	The model contains temperature dependent material properties, which are ignored by the current solver settings.
SE_NoBeamProperties	
SE_NoBrickProperties	
SE_NoFreedomCaseInIncrements	No freedom case is set in the load increments tables.
SE_NoFreedomCaseSelected	
SE_NoLoadCaseSelected	
SE_NoLoadTablesDefined	

SE_NoModesIncluded	
SE_NoMovingLoadPathsInCases	No load paths were found in the selected load cases.
SE_NonlinearSolverRequired	
SE_NoNodes	
SE_NoPlateProperties	
SE_NoResponseVariablesDefined	
SE_NoSpectralResultsSelected	
SE_NoTimeStepsSaved	
SE_NoVelocityDataInInitialFile	
SE_RubberRequiresGNL	
SE_ShearPanelMustBeQuad4	
SE_SingleShotRestartFile	The restart file contains only the last saved result case.
SE_SingularBrickMatrix	
SE_SingularPlateMatrix	
SE_SkylineUsesBadSort	The Skyline scheme usually works best with the Tree and Geometry node orderings.
SE_SoilRequiresMNL SE_SpectralBaseExcitationsAllZero SE_SpectralCasesNotDefined SE_SpectralExcitationsAllZero	
SE_SpectralLoadExcitationsAllZero	
SE_StagedSolutionFileNotFound	The file used in the initial staged analysis cannot be found or is invalid.
SE_StagingHasChanged	Stage definitions in the initial file are not compatible with the current stage definitions in the model.
SE_StringOrderHasChanged	The string elements defined in the model are not compatible with those in the restart file.
SE_TableNotFound	
SE_TemperatureDependenceCaseNotSet	
SE_TensileInsituBrickStress	
SE_TensileInsituPlateStress	
SE_UnknownException	
SE_ZeroLengthRigidLinkGenerated	

Solver termination error codes

If the .dll solver is used (that is, *St7SetUseSolverDLL* has been called with UseDLL=True) and the solver terminates unexpectedly, *St7GetGlobalIntegerValue* with Index=ivSolverTerminationCode can be used to access an error code. Possible error codes related to the finite element problem are listed in the help topic **Solver Messages: Error Messages**. For example, a solver termination code of 4 corresponds to "`*ERROR[   4]:Global stiffness matrix is singular`", which also appears in the solver log file.

The following additional termination codes can also be returned. These are related to the platform on which Strand7 is running or other causes unrelated to the specific finite element problem.

Error Code	Additional Information
`ST_Abnormal`	The solver terminated due to an unknown error.
`ST_CreateLog`	The log file cannot be created for a new analysis.
`ST_Internal`	The solver terminated due to an internal error – please forward the model and solver log file to support@strand7.com for assistance if this is encountered.
`ST_MemError`	A request for memory allocation has failed – this usually means that although the requested amount of memory is available, a contiguous block of the requested size is not.
`ST_NoDisk`	Insufficient disk space is available to run or continue running the analysis. Under normal conditions this error will not occur because a lack of disk space produces an alert message to the user, to enable the clearing of additional space to continue the analysis.
`ST_NoError`	The solver completed without error.
`ST_NoLicence`	The solver terminated due to loss of licence.
`ST_NoRam`	Insufficient memory is available to run or continue running the analysis.
`ST_OpenLog`	The log file cannot be opened for an analysis that is appending to existing results.
`ST_Scratch`	The solver reads and writes a number of temporary (scratch) files during the execution, until the end of the analysis when the model result file (e.g. .LSA for linear static analysis) is generated and all the scratch files are automatically deleted. If one of the scratch files is inaccessible at the start or becomes inaccessible part way through an analysis, this error will be generated. Under normal conditions, this should never happen because all scratch files required for an analysis are locked as soon as the solver starts and remain locked until solver completion. However, this error could occur when launching the batch solver – the scratch files for the batch solver are usually prepared well before the solver is launched, and therefore there is no guarantee that they will all still exist, intact, at the time the batch solver runs.
`ST_UserStop`	The solve was stopped by the user.
`ST_WriteLog`	Cannot write to the log file – usually this is because the log file size has reached a limit for a text file on the system (typically 2 GB).

Coordinate System Conventions

All coordinate systems in Strand7 define a right-hand set of locally orthogonal axes, $\mathbf{i}_1$, $\mathbf{i}_2$ and $\mathbf{i}_3$, with reference to the global XYZ system.

- The translational degrees of freedom in these directions are generically termed the 123 degrees of freedom (DoF) and are always listed in the same order.
- Rotational degrees of freedom are defined by the right-hand rule about the $\mathbf{i}_1$, $\mathbf{i}_2$ and $\mathbf{i}_3$ axes. These degrees of freedom are listed in this order after the 123 components – collectively they are referred to as the 123456 DoF.

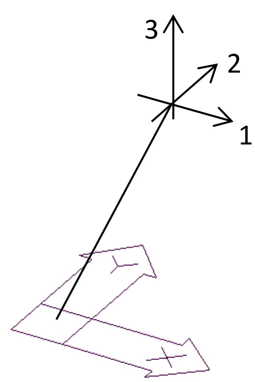
When a Coordinate System ID is passed by the Strand7 API, the global XYZ system is always identified as ID = 1. Subsequent indices ID > 1 then identify User Coordinate Systems (UCS) that have been defined. Note that the ID numbers need not be contiguous and may be retrieved by their index using *St7GetUCSID*.

UCS Types

Strand7 supports four UCS (User Coordinate System) types: Cartesian, cylindrical, spherical and toroidal. The integer values and types (as defined in the include and header files) are shown below.

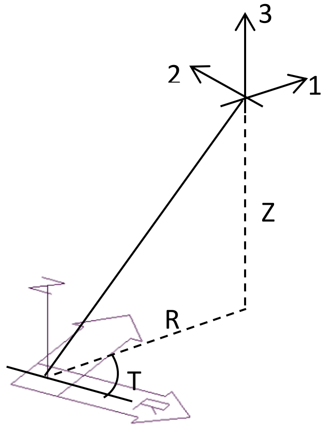
Cartesian coordinate system

```
csCartesian
```



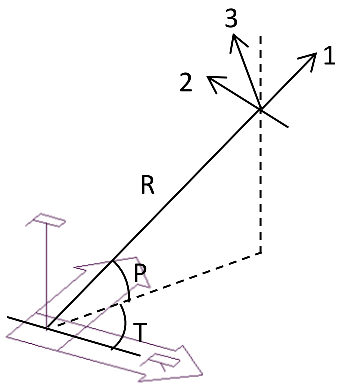
Cylindrical coordinate system

`csCylindrical`



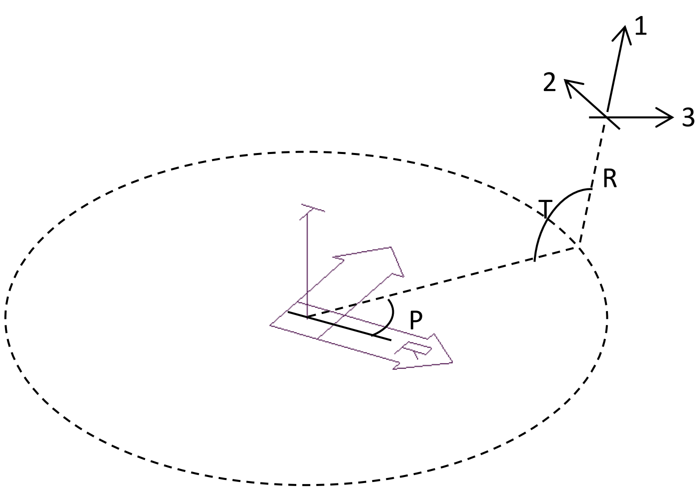
Spherical coordinate system

`csSpherical`



Toroidal coordinate system

`csToroidal`



UCS Doubles Array

The UCSDoublesArray vector contains the definition of the UCS, including the origin point, and two points defining a plane. In the case of a toroidal system, an additional value defines the major radius of the torus. The following includes the positional information for the UCSDoublesArray vector.

`[0..2]`	Origin point in global XYZ coordinates.
`[3..5]`	First plane point in global XYZ coordinates.
`[6..8]`	Second plane point in global XYZ coordinates.
`[9]`	Toroidal radius.

Element Connections

The `ConnectionArray` vector is used to determine the nodal connections of an element. It is used for all element types ranging from Beam2 to Brick20 elements.

The first position in the ConnectionArray vector, `ConnectionArray[0]`, holds the number of nodes in the element. Positions thereafter hold the ordered nodal connections. For a Beam2 element, `ConnectionArray[0..2]` is filled, with `ConnectionArray[0] = 2`, `ConnectionArray[1] = Node1` and `ConnectionArray[2] = Node2` respectively. A Brick20 element will use the entire vector in a similar fashion.

Refer to *Beam Local Coordinates*, *Plate Local Coordinates* and *Brick Local Coordinates* for element node connection sequences for all element types.

Beam Local Coordinates

Each beam element in Strand7 possesses a node numbering scheme that defines the default orientation of the principal coordinate system of the beam, denoted 1-2-3. The principal coordinate system is a right-handed coordinate system defined by the beam properties such that over the cross section *S* in the 1-2 plane

$$I_{12} = \int_S x_1 x_2 \, dA = 0 .$$

Note that this is generally a rotation away from the native x-y directions in which the beam cross section is defined. The x-y directions (with a z-direction completing the right-hand coordinate system) define the local beam axis system.

The default orientation for Beam2 elements is defined by:

- $\mathbf{i}_3$ – is the unit vector directed from Node 1 to Node 2.
- $\mathbf{i}_2$ – is the unit vector arising from $\mathbf{i}_2 = \mathbf{Z} \times \mathbf{i}_3$ where $\mathbf{Z}$ is the unit vector in the global Z-direction
- $\mathbf{i}_1$ – completes the right handed system such that $\mathbf{i}_1 \times \mathbf{i}_2 = \mathbf{i}_3$

The default orientation for Beam3 elements is defined by:

- $\mathbf{i}_3$ – is the unit vector directed from Node 1 to Node 2.
- $\mathbf{i}_2$ – is the unit vector perpendicular to $\mathbf{i}_3$, lying in the plane defined by Nodes 1, 2 and 3, directed towards Node 3.
- $\mathbf{i}_1$ – completes the right handed system such that $\mathbf{i}_1 \times \mathbf{i}_2 = \mathbf{i}_3$

The principal axes (hence the beam itself) may be rotated about the 3-axis from the default orientation using *St7SetBeamReferenceAngle1*.

The principal axes can be interrogated using *St7GetBeamAxisSystemInitial*, *St7GetBeamAxisSystemBirth* and *St7GetBeamAxisSystemGNL*.

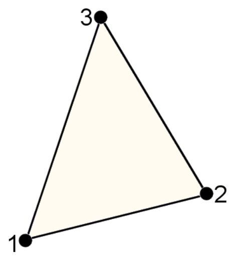
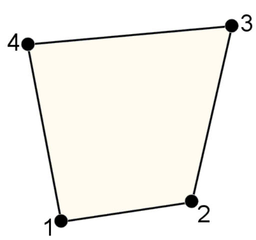
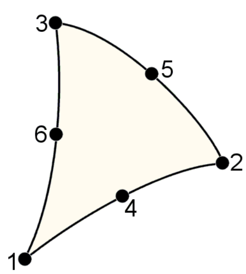
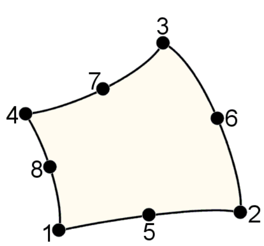
Beam End Numbering

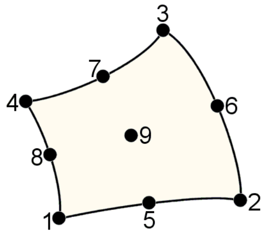
Attributes may also be applied to a particular end of a beam element. End 1 is defined as the end occurring at Node 1, and End 2 is defined as the end occurring at Node 2. Where l is the distance in the 3-direction from Node 1 along the beam, and L is the length of the beam, the relative length position is defined as l/L.

Plate Local Coordinates

Intrinsic Coordinate System

Each plate element in Strand7 possesses a node and edge numbering scheme, and a set of intrinsic coordinates uv that parameterise its extent in the global XYZ system. The intrinsic coordinates are defined by the node numbering scheme, summarised in the table below. They may be queried using *St7GetPlateUV*.

Element Type	**Nodal Intrinsic Coordinates**	**Edges**
 Tri3	**Node 1**: (0, 0) **Node 2**: (1, 0) **Node 3**: (0, 1)	Edge 1: **1**-**2** Edge 2: **2**-**3** Edge 3: **3**-**1**
 Quad4	**Node 1**: (-1, -1) **Node 2**: (1, -1) **Node 3**: (1, 1) **Node 4**: (-1, 1)	Edge 1: **1**-**2** Edge 2: **2**-**3** Edge 3: **3**-**4** Edge 4: **4**-**1**
 Tri6	**Node 1**: (0, 0) **Node 2**: (1, 0) **Node 3**: (0, 1) Node 4: (0.5, 0) Node 5: (0.5, 0.5) Node 6: (0, 0.5)	Edge 1: **1**-4-**2** Edge 2: **2**-5-**3** Edge 3: **3**-6-**1**
 Quad8	**Node 1**: (-1, -1) **Node 2**: (1, -1) **Node 3**: (1, 1) **Node 4**: (-1, 1) Node 5: (0, -1) Node 6: (1, 0) Node 7: (0, 1) Node 8: (-1, 0)	Edge 1: **1**-5-**2** Edge 2: **2**-6-**3** Edge 3: **3**-7-**4** Edge 4: **4**-8-**1**

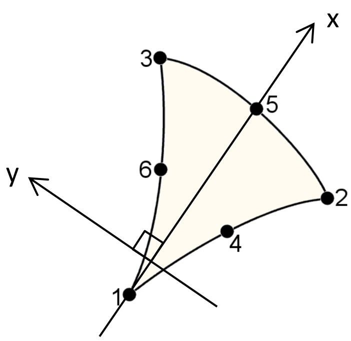
 Quad9	**Node 1**: (-1, -1) **Node 2**: (1, -1) **Node 3**: (1, 1) **Node 4**: (-1, 1) Node 5: (0, -1) Node 6: (1, 0) Node 7: (0, 1) Node 8: (-1, 0) Node 9: (0, 0)	Edge 1: **1**-5-**2** Edge 2: **2**-6-**3** Edge 3: **3**-7-**4** Edge 4: **4**-8-**1**

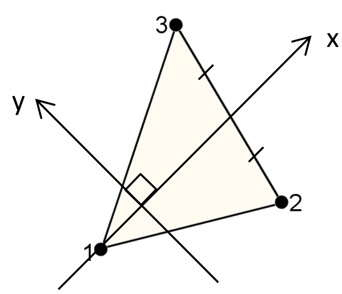
Local Axis System

The intrinsic coordinate system is generally curvilinear and non-orthogonal. An orthogonal local axis system is also defined for plates for the purpose of applying directional attributes and material properties.

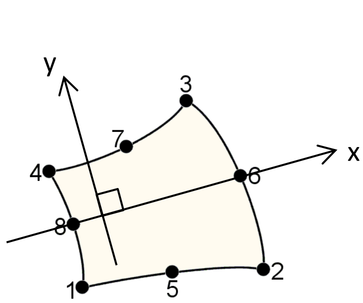
The local axis system is a right-handed coordinate system defined such that the x-y plane lies in the median plane of the plate. The median plane of the plate is that plane that minimises the sum of squared perpendicular distances to it (calculated by a principal components analysis).

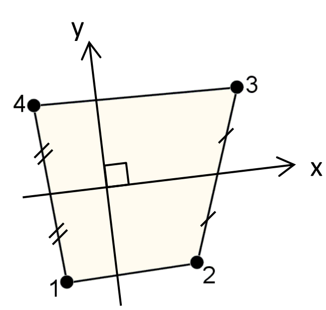
By default the local axis system is aligned relative to the plate nodes such that for triangular elements:





and for quadrilateral elements:





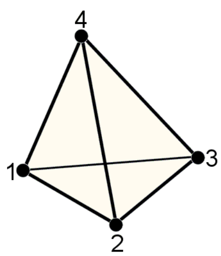
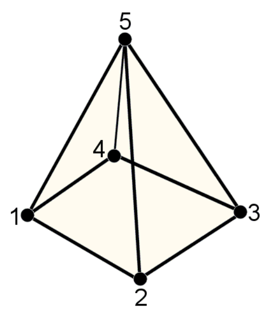
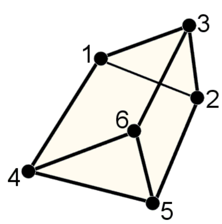
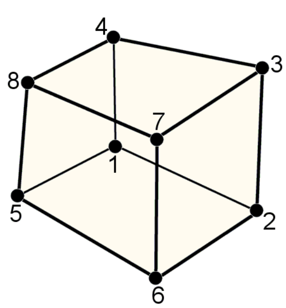
The local x-y axes may be rotated about the local z axis using *St7SetPlateXAngle1*. The local z axis is invariant, and completes the right-hand coordinate system.

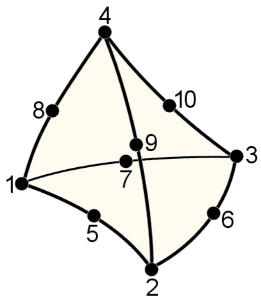
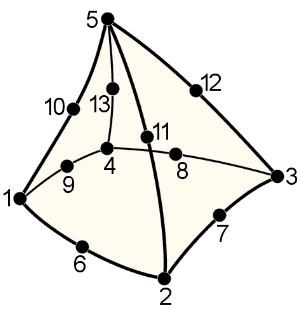
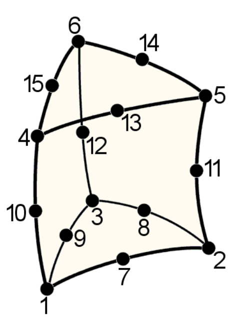
The axis system for a plate element can be interrogated using *St7GetPlateAxisSystemInitial*, *St7GetPlateAxisSystemBirth* and *St7GetPlateAxisSystemGNL*.

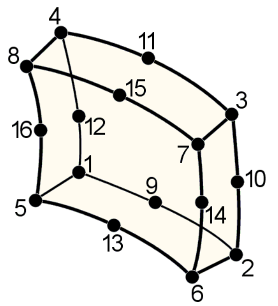
Brick Local Coordinates

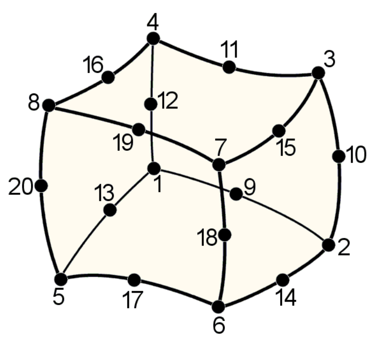
Intrinsic Coordinate System

Each brick element in Strand7 possesses a node and face numbering scheme, and a set of intrinsic coordinates uvw that parameterise its extent in the global XYZ system. The intrinsic coordinates are defined by the node numbering scheme, summarised in the table below. They may be queried using *St7GetBrickUVW*.

Element Type	**Nodal Intrinsic Coordinates**	**Faces**
 Tet4	**Node 1**: (0, 0, 0) **Node 2**: (1, 0, 0) **Node 3**: (0, 1, 0) **Node 4**: (0, 0, 1)	Face 1: **1-2-3** Face 2: **4-1-3** Face 3: **4-2-1** Face 4: **4-3-2**
 Pyra5	**Node 1**: (-1, -1, 0) **Node 2**: (1, -1, 0) **Node 3**: (1, 1, 0) **Node 4**: (-1, 1, 0) **Node 5**: (0, 0, 1)	Face 1: **1-2-3-4** Face 2: **5-1-4** Face 3: **5-2-1** Face 4: **5-3-2** Face 5: **5-4-3**
 Wedge6	**Node 1**: (0, 0, -1) **Node 2**: (1, 0, -1) **Node 3**: (0, 1, -1) **Node 4**: (0, 0, 1) **Node 5**: (1, 0, 1) **Node 6**: (0, 1, 1)	Face 1: **1-2-3** Face 2: **5-2-1**-4 Face 3: **4-6-5** Face 4: **4-1-3-6** Face 5: **6-3-2-5**
 Hex8	**Node 1**: (-1, -1, -1) **Node 2**: (1, -1, -1) **Node 3**: (1, 1, -1) **Node 4**: (-1, 1, -1) **Node 5**: (-1, -1, 1) **Node 6**: (1, -1, 1) **Node 7**: (1, 1, 1) **Node 8**: (-1, 1, 1)	Face 1: **1-2-3-4** Face 2: **7-3-2-6** Face 3: **6-5-8-7** Face 4: **5-1-4-8** Face 5: **8-4-3-7** Face 6: **6-2-1-5**

 Tet10	**Node 1**: (0, 0, 0) **Node 2**: (1, 0, 0) **Node 3**: (0, 1, 0) **Node 4**: (0, 0, 1) Node 5: (0.5, 0, 0) Node 6: (0.5, 0.5, 0) Node 7: (0, 0.5, 0) Node 8: (0, 0, 0.5) Node 9: (0.5, 0, 0.5) Node 10: (0, 0.5, 0.5)	Face 1: **1**-5-**2**-6-**3**-7 Face 2: **4**-8-**1**-7-**3**-10 Face 3: **4**-9-**2**-5-**1**-8 Face 4: **4**-10-**3**-6-**2**-9
 Pyra13	**Node 1**: (-1, -1, 0) **Node 2**: (1, -1, 0) **Node 3**: (1, 1, 0) **Node 4**: (-1, 1, 0) **Node 5**: (0, 0, 1) Node 6: (0, -1, 0) Node 7: (1, 0, 0) Node 8: (0, 1, 0) Node 9: (-1, 0, 0) Node 10: (-0.5, -0.5, 0.5) Node 11: (0.5, -0.5, 0.5) Node 12: (0.5, 0.5, 0.5) Node 13: (-0.5, 0.5, 0.5)	Face 1: **1**-6-**2**-7-**3**-8-**4**-9 Face 2: **5**-10-**1**-9-**4**-13 Face 3: **5**-11-**2**-6-**1**-10 Face 4: **5**-12-**3**-7-**2**-11 Face 5: **5**-13-**4**-8-**3**-12
 Wedge15	**Node 1**: (0, 0, -1) **Node 2**: (1, 0, -1) **Node 3**: (0, 1, -1) **Node 4**: (0, 0, 1) **Node 5**: (1, 0, 1) **Node 6**: (0, 1, 1) Node 7: (0.5, 0, -1) Node 8: (0.5, 0.5, -1) Node 9: (0, 0.5, -1) Node 10: (0, 0, 0) Node 11: (1, 0, 0) Node 12: (0, 1, 0) Node 13: (0.5, 0, 1) Node 14: (0.5, 0.5, 1) Node 15: (0, 0.5, 1)	Face 1: **1**-7-**2**-8-**3**-9 Face 2: **5**-11-**2**-7-**1**-10-**4**-13 Face 3: **4**-15-**6**-14-**5**-13 Face 4: **4**-10-**1**-9-**3**-12-**6**-15 Face 5: **6**-12-**3**-8-**2**-11-**5**-14

<table>
<tr>
<td>


Hex16
</td>
<td>
Node 1: (-1, -1, -1)
Node 2: (1, -1, -1)
Node 3: (1, 1, -1)
Node 4: (-1, 1, -1)
Node 5: (-1, -1, 1)
Node 6: (1, -1, 1)
Node 7: (1, 1, 1)
Node 8: (-1, 1, 1)
Node 9: (0, -1, -1)
Node 10: (1, 0, -1)
Node 11: (0, 1, -1)
Node 12: (-1, 0, -1)
Node 13: (0, -1, 1)
Node 14: (1, 0, 1)
Node 15: (0, 1, 1)
Node 16: (-1, 0, 1)
</td>
<td>
Face 1: 1-9-2-10-3-11-4-12
Face 2: 7-3-10-2-6-14
Face 3: 6-13-5-16-8-15-7-14
Face 4: 5-1-12-4-8-16
Face 5: 8-4-11-3-7-15
Face 6: 6-2-9-1-5-13
</td>
</tr>
<tr>
<td>


Hex20
</td>
<td>
Node 1: (-1, -1, -1)
Node 2: (1, -1, -1)
Node 3: (1, 1, -1)
Node 4: (-1, 1, -1)
Node 5: (-1, -1, 1)
Node 6: (1, -1, 1)
Node 7: (1, 1, 1)
Node 8: (-1, 1, 1)
Node 9: (0, -1, -1)
Node 10: (1, 0, -1)
Node 11: (0, 1, -1)
Node 12: (-1, 0, -1)
Node 13: (-1, -1, 0)
Node 14: (1, -1, 0)
Node 15: (1, 1, 0)
Node 16: (-1, 1, 0)
Node 17: (0, -1, 1)
Node 18: (1, 0, 1)
Node 19: (0, 1, 1)
Node 20: (-1, 0, 1)
</td>
<td>
Face 1: 1-9-2-10-3-11-4-12
Face 2: 7-15-3-10-2-14-6-18
Face 3: 6-17-5-20-8-19-7-18
Face 4: 5-13-1-12-4-16-8-20
Face 5: 8-16-4-11-3-15-7-19
Face 6: 6-14-2-9-1-13-5-17
</td>
</tr>
</table>

Local Axis System

The intrinsic coordinate system is generally curvilinear and non-orthogonal. An orthogonal local axis system is also defined for bricks for the purpose of applying directional attributes and material properties.

By default the local axis system is aligned with the global XYZ system, but may be realigned to any other UCS using *St7SetBrickLocalAxes1*.

Face Axis System

Some attributes are defined with reference to a coordinate system defined on a given face of a brick. The face axis system is oriented with respect to the nodes in the face's definition (listed above) such that it coincides with the default local axis system of a plate element with the same nodal definition.

The face axis system for a face on a particular brick element can be interrogated using *St7GetBrickFaceAxisSystemInitial*, *St7GetBrickFaceAxisSystemBirth* and *St7GetBrickFaceAxisSystemGNL*.

User Defined Material Matrix

User defined material matrices are supplied and returned with the translation, rotation and coupling matrices in a single array. The format applies to the functions *St7SetUserBeamData*, *St7GetUserBeamData*, *St7SetPlateUserDefinedMaterial* and *St7GetPlateUserDefinedMaterial*.

$$\begin{bmatrix} \textbf{Translation} & \textbf{Coupling} \\ & \textbf{Rotation} \end{bmatrix}$$

The indices into the array Doubles[] are as follows:

$$\begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 \\ & 6 & 7 & 8 & 9 & 10 \\ & & 11 & 12 & 13 & 14 \\ & & & 15 & 16 & 17 \\ & & & & 18 & 19 \\ & & & & & 20 \end{bmatrix}$$

Attribute Types

Attribute types are identified by an integer constant as defined in the header files. Individual functions are provided to set and get attribute data, the *Usage*

> A positive property number signifies that the element is re-born at the switched stage (i.e., its birth stage becomes the stage index associated with the property switch). A negative PropID signifies that the element changes its property type but its birth stage is inherited from the current element (i.e., the Inherited Birth Stage option is set).

St7DeleteAttribute function can be used to delete attributes, and the *St7SetAttributeDisplay* and *St7GetAttributeDisplay* functions control the display of attributes in the model window.

Attribute instances are uniquely identified by the arguments Entity, EntityNum, AttributeType, LocalID, Axis, CaseNum, and ID. The entity type, element number and attribute type arguments are required for all attributes. The local ID, axis, case number and ID arguments are dependent on the attribute type. When these arguments are not required they are ignored.

Node Attributes

Attribute	Constant	ipAttrLocal	ipAttrAxis	ipAttrCase	ipAttrID
Acceleration	`aoNodeAcceleration`			Load Case	
Damping	`aoDamping`			Freedom Case	
Force	`aoForce`			Load Case	
Heat Source	`aoNodeHeatSource`			Load Case	
Initial Velocity	`aoNodeVelocity`			Load Case	
Moment	`aoMoment`			Load Case	
Non-Structural Mass	`aoNSMass`			Load Case	ID (1 - 192)
Response Variable	`aoNodeInfluence`		rvNodeDisplacement or rvNodeReaction	Load Case	
Restraint	`aoRestraint`			Freedom Case	
Rotational Mass	`aoMRotation`				
Rotational Stiffness	`aoKRotation`			Freedom Case	
Temperature	`aoTemperature`			Load Case	
Translational Mass	`aoMTranslation`				
Translational Stiffness	`aoKTranslation`			Freedom Case	

Beam Attributes

Attribute	Constant	ipAttrLocal	ipAttrAxis	ipAttrCase	ipAttrID
Angle	`aoBeamAngle`				
Cable Free-Length	`aoCableFreeLength`				
Connection UCS	`aoBeamConnectionUCS`	End (1 - 2)			
Convection	`aoBeamConvection`	End (1 - 2)		Load Case	
Creep Loading Age	`aoBeamCreepLoadingAge`				
End Attachment	`aoBeamEndAttachment`	End (1 - 2)			
Global Distributed Force	`aoBeamDLG`		Local Axis (1 - 3)	Load Case	ID (1 - 192)
Global Point Force	`aoBeamCFG`			Load Case	ID (1 - 192)
Global Point Moment	`aoBeamCMG`			Load Case	ID (1 - 192)
Heat Flux	`aoBeamFlux`	End (1 - 2)		Load Case	
Heat Source	`aoBeamHeatSource`			Load Case	
Non-Structural Mass	`aoBeamNSMass`			Load Case	ID (1 - 192)
Offset	`aoBeamOffset`				
Pipe Pressure	`aoPipePressure`			Load Case	
Pipe Radius	`aoBeamRadius`				
Pipe Temperature	`aoPipeTemperature`			Load Case	
Pre-Curvature	`aoBeamPreCurvature`			Load Case	
Pre-Load	`aoBeamPreTension`			Load Case	
Principal Distributed Force	`aoBeamDLL`		Local Axis (1 - 3)	Load Case	ID (1 - 192)
Principal Distributed Moment	`aoBeamDML`		Local Axis (1 - 3)	Load Case	ID (1 - 192)
Principal Point Force	`aoBeamCFL`			Load Case	ID (1 - 192)
Principal Point Moment	`aoBeamCML`			Load Case	ID (1 - 192)
Radiation	`aoBeamRadiation`	End (1 - 2)		Load Case	
Response Variable	`aoBeamInfluence`	End (1 - 2)		Load Case	
Rotational End Release	`aoBeamREndRelease`	End (1 - 2)			
Side Attachment	`aoBeamSideAttachment`	End (1 - 2)	adPlus1, adPlus2, adMinus1 or adMinus2		
Stage Property	`aoBeamStageProperty`				Stage ID, or 0 for all stages.
Stiffness/Mass Factor	`aoBeamSectionFactor`				
String Group	`aoBeamStringGroup`				
Support	`aoBeamSupport`		adPlus1, adPlus2, adMinus1 or adMinus2	Freedom Case	
Taper	`aoBeamTaper`		axLocalX or axLocalY		
Temperature Gradient	`aoBeamTempGradient`			Load Case	

Translational End Release	aoBeamTEndRelease	End (1 - 2)			

Plate Attributes

Attribute	Constant	ipAttrLocal	ipAttrAxis	ipAttrCase	ipAttrID
Axis Angle	aoPlateAngle				
Cavity Fluid Layout	aoPlateCavity	psPlateMinusZ or psPlatePlusZ			
Concrete Reinforcement Layout	aoPlateReinforcement				
Creep Loading Age	aoPlateCreepLoadingAge				
Edge Attachment	aoPlateEdgeAttachment	Edge (1 - 4)	adPlanar, adPlusZ or adMinusZ		
Edge Convection	aoPlateEdgeConvection	Edge (1 - 4)		Load Case	
Edge Global Pressure	aoPlateEdgeGlobalPressure	Edge (1 - 4)		Load Case	
Edge Normal Pressure	aoPlateEdgeNormalPressure	Edge (1 - 4)		Load Case	
Edge Radiation	aoPlateEdgeRadiation	Edge (1 - 4)		Load Case	
Edge Release	aoPlateEdgeRelease	Edge (1 - 4)			
Edge Shear Stress	aoPlateEdgeShear	Edge (1 - 4)		Load Case	
Edge Support	aoPlateEdgeSupport	Edge (1 - 4)		Freedom Case	
Edge Transverse Shear Stress	aoPlateEdgeTransverseShear	Edge (1 - 4)		Load Case	
Face Attachment	aoPlateFaceAttachment	psPlateMinusZ or psPlatePlusZ			
Face Convection	aoPlateFaceConvection	psPlateMinusZ or psPlatePlusZ		Load Case	
Face Radiation	aoPlateFaceRadiation	psPlateMinusZ or psPlatePlusZ		Load Case	
Face Support	aoPlateFaceSupport	psPlateMinusZ or psPlatePlusZ		Freedom Case	
Global Pressure	aoPlateGlobalPressure	psPlateMinusZ or psPlatePlusZ		Load Case	
Heat Flux	aoPlateFlux	Edge (1 - 4)		Load Case	
Heat Source	aoPlateHeatSource			Load Case	
Load Patch	aoLoadPatch				
Non-Structural Mass	aoPlateNSMass			Load Case	ID (1 - 192)
Normal Pressure	aoPlateFacePressure			Load Case	
Offset	aoPlateOffset				
Point Force	aoPlatePointForce			Load Case	ID (1 - 192)
Point Moment	aoPlatePointMoment			Load Case	ID (1 - 192)
Pre-Curvature	aoPlatePreCurvature			Load Case	

Pre-Load	aoPlatePreLoad			Load Case	
Response Variable	aoPlateInfluence		rvPlateForce or rvPlateMoment	Load Case	
Shear Stress	aoPlateFaceShear			Load Case	
Soil In-situ Ratio	aoPlateSoilRatio				
Soil In-situ Stress	aoPlateSoilStress				
Stage Property	aoPlateStageProperty				Stage ID, or 0 for all stages.
Stiffness/Mass Factor	aoPlateSectionFactor				
Temperature Gradient	aoPlateTempGradient			Load Case	
Thickness	aoPlateThickness				

Brick Attributes

Attribute	Constant	ipAttrLocal	ipAttrAxis	ipAttrCase	ipAttrID
Cavity Fluid Layout	aoBrickCavity	Face (1 - 6)			
Convection	aoBrickConvection	Face (1 - 6)		Load Case	
Creep Loading Age	aoBrickCreepLoadingAge				
Face Attachment	aoBrickFaceAttachment	Face (1 - 6)			
Face Global Pressure	aoBrickGlobalPressure	Face (1 - 6)		Load Case	
Face Normal Pressure	aoBrickPressure	Face (1 - 6)		Load Case	
Face Shear Stress	aoBrickShear	Face (1 - 6)		Load Case	
Face Support	aoBrickFaceFoundation	Face (1 - 6)		Freedom Case	
Heat Flux	aoBrickFlux	Face (1 - 6)		Load Case	
Heat Source	aoBrickHeatSource			Load Case	
Local Axes	aoBrickLocalAxes				
Non-Structural Mass	aoBrickNSMass	Face (1 - 6)		Load Case	ID (1 - 192)
Point Force	aoBrickPointForce	Face (1 - 6)		Load Case	ID (1 - 192)
Pre-Load	aoBrickPreLoad			Load Case	
Radiation	aoBrickRadiation	Face (1 - 6)		Load Case	
Response Variable	aoBrickInfluence			Load Case	
Soil In-situ Ratio	aoBrickSoilRatio				
Soil In-situ Stress	aoBrickSoilStress				
Stage Property	aoBrickStageProperty				Stage ID, or 0 for all stages.

Path Attributes

Attribute	Constant	ipAttrLocal	ipAttrAxis	ipAttrCase	ipAttrID
Distributed Force	aoPathDistributedForce				
Heat Source	aoPathHeatSource				
Point Force	aoPathPointForce				

Vertex Attributes

Attribute	Constant	ipAttrLocal	ipAttrAxis	ipAttrCase	ipAttrID
Damping	aoDamping			Freedom Case	
Force	aoForce			Load Case	
Heat Source	aoNodeHeatSource			Load Case	
Mesh Size	aoVertexMeshSize				
Moment	aoMoment			Load Case	
Non-Structural Mass	aoNSMass			Load Case	ID (1 - 192)
Restraint	aoRestraint			Freedom Case	
Rotational Mass	aoMRotation				
Rotational Stiffness	aoKRotation			Freedom Case	
Temperature	aoTemperature			Load Case	
Translational Mass	aoMTranslation				
Translational Stiffness	aoKTranslation			Freedom Case	

Coedge Attributes

Attribute	Constant	ipAttrLocal	ipAttrAxis	ipAttrCase	ipAttrID
Edge Attachment	aoPlateEdgeAttachment		adPlanar, adPlusZ or adMinusZ		
Edge Convection	aoPlateEdgeConvection			Load Case	
Edge Global Pressure	aoPlateEdgeGlobalPressure			Load Case	
Edge Heat Flux	aoPlateFlux			Load Case	
Edge Normal Pressure	aoPlateEdgeNormalPressure			Load Case	
Edge Radiation	aoPlateEdgeRadiation			Load Case	
Edge Release	aoPlateEdgeRelease				
Edge Shear Stress	aoPlateEdgeShear			Load Case	
Edge Support	aoPlateEdgeSupport			Freedom Case	
Edge Transverse Shear Stress	aoPlateEdgeTransverseShear			Load Case	

Face Attributes

Attribute	Constant	ipAttrLocal	ipAttrAxis	ipAttrCase	ipAttrID
Face Attachment	`aoPlateFaceAttachment`	psPlateMinusZ or psPlatePlusZ			
Face Convection	`aoPlateFaceConvection`	psPlateMinusZ or psPlatePlusZ		Load Case	
Face Radiation	`aoPlateFaceRadiation`	psPlateMinusZ or psPlatePlusZ		Load Case	
Face Support	`aoPlateFaceSupport`	psPlateMinusZ or psPlatePlusZ		Freedom Case	
Global Pressure	`aoPlateGlobalPressure`	psPlateMinusZ or psPlatePlusZ		Load Case	
Heat Source	`aoPlateHeatSource`			Load Case	
Non-Structural Mass	`aoPlateNSMass`			Load Case	ID (1 - 192)
Normal Pressure	`aoPlateFacePressure`			Load Case	
Offset	`aoPlateOffset`				
Temperature Gradient	`aoPlateTempGradient`			Load Case	
Thickness	`aoPlateThickness`				

Beam Cross Section Shapes

Beam cross section shape parameters (dimensions) for standard sections are manipulated through the arrays of doubles for *St7SetBeamSectionGeometry* and *St7GetBeamSectionGeometry*, as well as the index positions ipD1 to ipT3 on *St7GetBeamPropertyData*. The interpretation of these values depends on the cross section type, as presented in the table below. Not all indices are relevant to all section types – blank entries are set as 0.0.

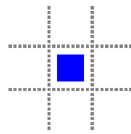
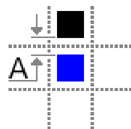
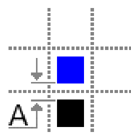
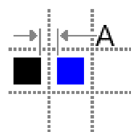
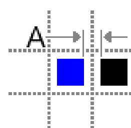
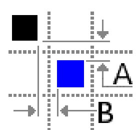
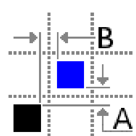
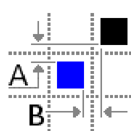
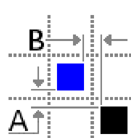
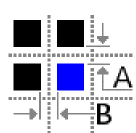
Standard Cross Section	**D1**	**D2**	**D3**	**T1**	**T2**	**T3**	**Shape**
bsCircularHollow	D			T			
bsCircularSolid	D						
bsCruciform	B	D		T1	T2		
bsISection	B1	B2	D	T1	T2	T3	
bsLipChannel	B	D	L	T1	T2	T3	
bsLSection	B	D		T1	T2		
bsSquareHollow	B	D		T1	T2		
bsSquareSolid	B	D					
bsTopHatChannel	B	D	L	T1	T2	T3	
bsTrapezoidHollow	B1	B2	D	T1	T2		
bsTrapezoidSolid	B1	B2	D				

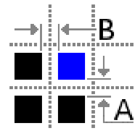
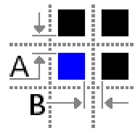
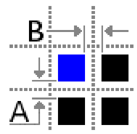
bsTriangleHollow	B	D		T1	T2		
bsTriangleSolid	B	D					
bsTSection	B	D	L	T1	T2	T3	
bsZSection	B1	B2	D	T1	T2	T3	

Additionally, the section can be bsBXSSection, bsBGLSection or bsNullSection (when it has not been defined).

Beam Cross Section Mirror Options

The standard beam cross section shapes, as defined in *Beam Cross Section Shape*, with the exception of bsCircularHollow and bsCircularSolid, support the mirror options defined in the table below.

MirrorType	Number of Sections	Illustration
`mtNone`	1	
`mtTop`	2	
`mtBot`	2	
`mtLeft`	2	
`mtRight`	2	
`mtLeftTopOnly`	2	
`mtLeftBotOnly`	2	
`mtRightTopOnly`	2	
`mtRightBotOnly`	2	
`mtLeftAndTop`	4	

`mtLeftAndBot`	4	
`mtRightAndTop`	4	
`mtRightAndBot`	4	

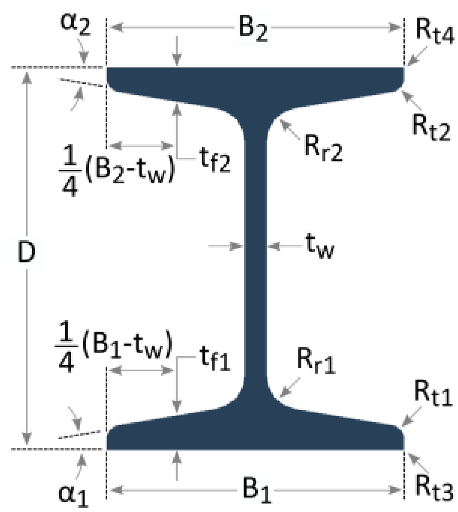
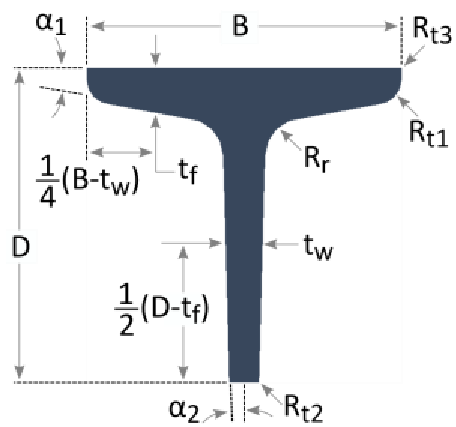
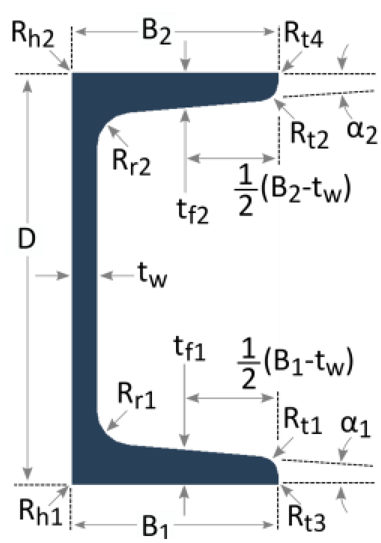
Beam Geometry Library (BGL) Cross Section Shapes

Beam cross section shape parameters (dimensions) for BGL cross sections are available from the individual beam property data via *St7GetBeamSectionGeometryBGL*, and from the BGL libraries via *St7GetLibraryBeamSectionGeometryBGL.*

The dimensions are returned in the array of doubles `Dimensions[0..kMaxBGLDimensions-1].` The interpretation of the values in the array depends on the cross section type as presented in the table below. Lengths are in the length unit of the model. Angles are in degrees. Not all values are relevant to all section types – blank entries are set to 0.0.

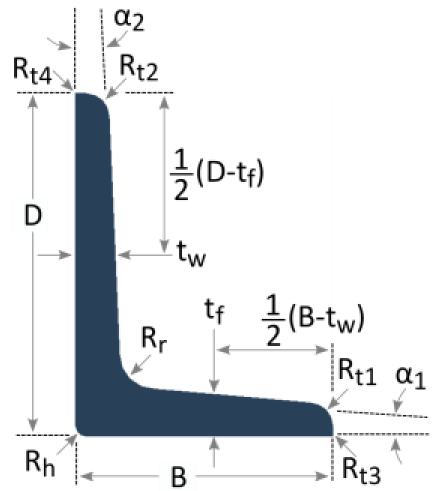
	I Section	**T Section**	**Channel**	**Angle**	**Hollow Rectangle**	**Bulb Flat**
Name Constant	bgISection	bgTSection	bgChannel	bgAngle	bgRectangularHollow	bgBulbFlat
Array Index	**Dimension**					
0	D	D	D	D	D	D
1	B_1	B	B_1	B	B	B
2	B_2	t_w	B_2	t_w	t_w	t
3	t_w	t_f	t_w	t_f	t_f	R_r
4	t_{f1}	R_r	t_{f1}	R_r	R_i	R_h
5	t_{f2}	R_{t1}	t_{f2}	R_h	R_o	R_{t1}
6	R_{r1}	R_{t2}	R_{r1}	R_{t1}		R_{t2}
7	R_{r2}	R_{t3}	R_{r2}	R_{t2}		R_{t3}
8	R_{t1}	α_1	R_{h1}	R_{t3}		α
9	R_{t2}	α_2	R_{h2}	R_{t4}		
10	R_{t3}		R_{t1}	α_1		
11	R_{t4}		R_{t2}	α_2		
12	α_1		R_{t3}			
13	α_2		R_{t4}			
14			α_1			
15			α_2			

The following table illustrates the dimensions for each cross section type.

`bgISection`	
`bgTSection`	
`bgChannel`	

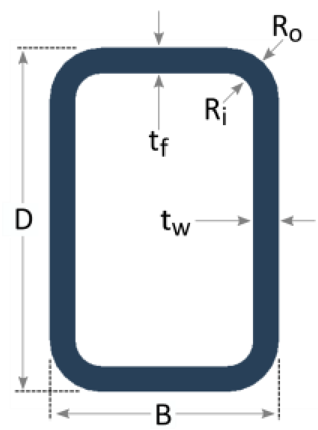
<table>
<tr><td>

`bgAngle`

</td><td>


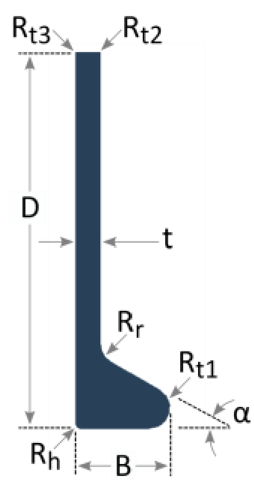
</td></tr>
<tr><td>

`bgRectangularHollow`

</td><td>


</td></tr>
<tr><td>

`bgBulbFlat`

</td><td>


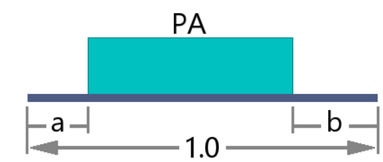
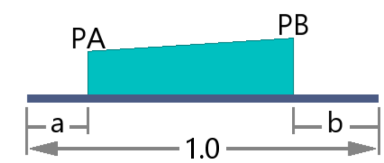
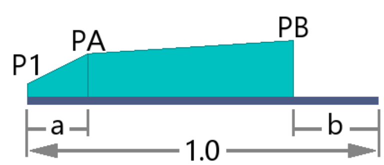
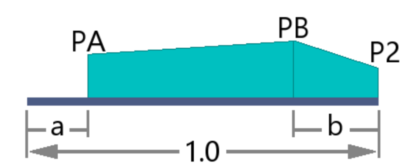
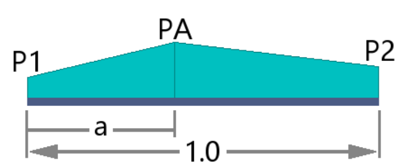
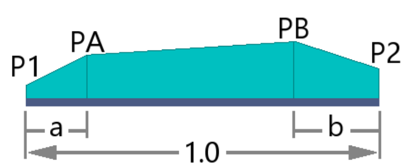
</td></tr>
</table>

Beam Distributed Load Types

There are six beam distribution types in Strand7, used by the following API functions:

- *St7SetBeamDistributedForcePrincipal6ID*
- *St7SetBeamDistributedForceGlobal6ID*
- *St7SetBeamDistributedMomentPrincipal6ID*
- *St7SetBeamNSMass10ID*
- *St7GetBeamDistributedForcePrincipal6ID*
- *St7GetBeamDistributedForceGlobal6ID*
- *St7GetBeamDistributedMomentPrincipal6ID*
- *St7GetBeamNSMass10ID*.

In these functions, the distribution type is specified by the parameter DLType, which may set one of the constants listed below.

<table>
<tr><th>DLType</th><th>Distribution</th></tr>
<tr><td><code>dlConstant</code></td><td>
</td></tr>
<tr><td><code>dlLinear</code></td><td>
</td></tr>
<tr><td><code>dlThreePoint0</code></td><td>
</td></tr>
<tr><td><code>dlThreePoint1</code></td><td>
</td></tr>
<tr><td><code>dlTriangular</code></td><td>
</td></tr>
<tr><td><code>dlTrapezoidal</code></td><td>
</td></tr>
</table>

The above schematics are for the distributed force attribute, but the layout also applies to the distributed moment and distributed mass attributes.

A Doubles array is also passed to these functions, in which the variables identifed in the schematics are set.

The values PA, PB, P1, P2, a and b are found at:

```
Doubles[0] = PA
Doubles[1] = PB
Doubles[2] = P1
Doubles[3] = P2
Doubles[4] = a
Doubles[5] = b
```

Link Types

Strand7 defines a number of different link types, and they are identified in the API by the following constants, denoted as LinkType in the function parameters.

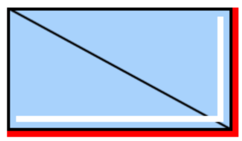
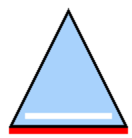
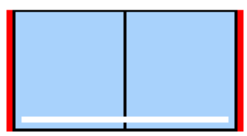
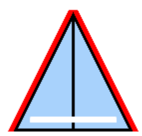
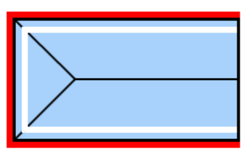
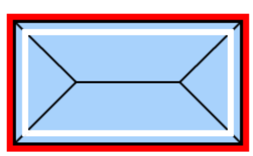
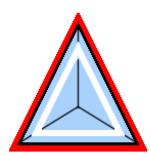
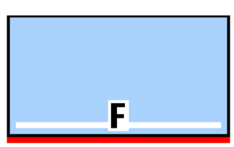
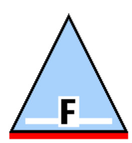
LinkType	Description
`ltAttachmentLink`	Attachment link – attaches a node to an element by constraining the degrees of freedom of the node to the degrees of freedom of the nodes of the element, according to the shape functions of the element.
`ltCouplingLink`	Coupling link – constrains the degrees of freedom of a node as a linear combination of the degrees of freedom of two other nodes.
`ltInterpolatedMultiPointLink`	Interpolated MPL – constrains the degrees of freedom of a node to be a combination of the degrees of freedom of a cluster of other nodes, using a least squares interpolation.
`ltMasterSlaveLink`	Master-slave link – constrains any of the degrees of freedom at two nodes to act as one.
`ltMasterSlaveMultiPointLink`	Master-slave MPL - constrains any of the degrees of freedom at a cluster of nodes to act as one.
`ltPinnedLink`	Pinned link – for the structural solvers, enforces no relative axial movement between the two nodes on the link; for the heat solvers constrains both nodes to have the same temperature.
`ltPinnedMultiPointLink`	Pinned MPL – for the structural solvers, enforces no relative axial movement between a nominated slave node and all nodes in a cluster of connected nodes; for the heat solvers constrains all nodes in the cluster to have the same temperature as the slave node.
`ltReactionMultiPointLink`	Reaction MPL – a special link used to produce reaction summations (forces/moments in the structural solvers and flux in the heat solvers) at a point in space from the reactions at nodes/elements in the cluster.
`ltRigidLink`	Rigid link – for the structural solvers, enforces no relative movement between the two nodes on the link except for a rigid body movement; for the heat solvers constrains both nodes to have the same temperature.
`ltRigidMultiPointLink`	Rigid MPL – for the structural solvers, enforces no relative movement between all nodes in a cluster except for a rigid body movement of the cluster as a whole; for the heat solvers constrains all nodes in the cluster to have the same temperature as the slave node.
`ltSectorSymmetryLink`	Sector-symmetry link – for the structural solvers, enforces rotational symmetry between two nodes on respective planes according to a cylindrical axis system; for the heat solvers constrains both nodes to have the same temperature.
`ltShrinkLink`	Shrink link – for the structural solvers, enforces displacements such that the sum of all the displacements at the two nodes equal zero; for the heat solvers constrains both nodes to have the same temperature.

`ltTwoPointLink`	Two-point link – allows for any user-defined linear relationship between the degrees of freedom at two nodes.
`ltUserDefinedMultiPointLink`	User-defined MPL – allows for any user-defined linear relationship between the degrees of freedom at any number of nodes.

Load Patch Types

There are six load patch types in Strand7. These are referenced via the PatchType parameter in the API functions *St7SetPlateLoadPatch4* and *St7GetPlateLoadPatch4*.

The parameter PatchType in these functions specifies the load patch type, as illustrated below.

PatchType	Load Distribution	Description
`ptAngleSplit`		Quadrilateral plates apply triangular load to two adjacent edges.
`ptAuto1`	 	Quadrilateral plates apply rectangular load to one edge. Triangular plates apply triangular load to one edge.
`ptAuto2`	 	Quadrilateral plates apply rectangular load to two opposite edges. Triangular plates apply triangular load to two opposite edges.
`ptAuto3`		Quadrilateral plates apply trapezoidal load to two opposite edges and triangular load to the intermediate edge.
`ptAuto4`	 	Quadrilateral plates apply trapezoidal load on the long edges and triangular load on the short edges. For a square quadrilateral, all loads are triangular. Triangular plates apply triangular load to all edges.
`ptManual*` *factors **F** in *Doubles* array	 	Both quadrilateral and triangular plates apply scaled rectangular load to the selected edges.

In the images:

- the whilte lines indicate the edges of the plate element to be selected to apply the specified type;
- the red lines indicate the edges where beam elements are required to receive the load;
- the black lines indicate the approximate shape of the load distribution applied to beam elements on the edges.

The type ptManual requires a set of weights specified by the parameter Doubles:

```
Doubles[0] = Edge 1 weight
Doubles[1] = Edge 2 weight
Doubles[2] = Edge 3 weight
Doubles[3] = Edge 4 weight
```

All types except ptAuto4 require the specification of edges via the parameter EdgeBits to define the distribution, or in the case of ptManual, to define the edges on which to apply the weights. The four least significant bits in the four byte integer EdgeBits correspond to the four (possible) edges of the load patch.

Writing EdgeBits in binary form it is seen that:

```
Edge 1 : b'00000000 00000000 00000000 00000001' = 1
Edge 2 : b'00000000 00000000 00000000 00000010' = 2
Edge 3 : b'00000000 00000000 00000000 00000100' = 4
Edge 4 : b'00000000 00000000 00000000 00001000' = 8
```

To define two or more edges, the binary values are combined using a 'logical or' operation. For example, to define edges 2 and 3, the following two binary values are combined using a logical or operation:

```
b'00000000 00000000 00000000 00000010'
b'00000000 00000000 00000000 00000100'
```

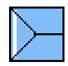
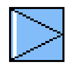
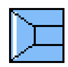
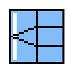
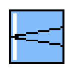
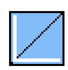
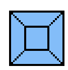
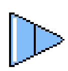
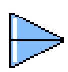
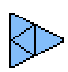
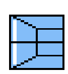
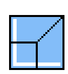
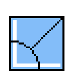
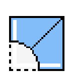
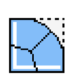
which produces the following binary value:

```
b'00000000 00000000 00000000 00000110' = 6.
```

That is, EdgeBits=6 selects edge 2 and edge 3 at the same time.

Grade Types

The GradeType parameter provided to *St7Grade* determines grade applied to the selected plates and bricks. Either whole elements, edges or brick faces should be selected as shown in the following table.

GradeType	Grade	GradeRatio Applies	Selected Plates	Selected Bricks
gt1x2Grade			Edge	Edge
gt1x2TriGrade			Edge	Edge
gt1x3Grade			Edge	Edge
gt2x3Grade			Edge	Edge
gt2x3TriGrade			Edge	Edge
gtQuadTriGrade1			Edge	Edge
gtQuadCutOut		True	Plate	Face
gtTriGrade1			Edge	Edge
gtTriGrade2			Edge	Edge
gtTriGrade3			Plate	Face
gt2x4Grade			Edge	Edge
gtQuarterQuadGrade			Edge	Edge
gtFullQuarterCircleGrade		True	Edge	Edge
gtQuarterCircleCut		True	Edge	Edge
gtFullQuarterCircleCut			Edge	Edge

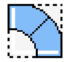
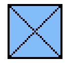
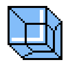
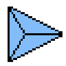
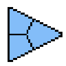
`gtQuarterAnnulusCut`		True	Edge	Edge
`gtQuadGradeTri`			Plate	Face
`gtBrickCornerGrade`				Edge
`gtTriGrade4`			Plate	Face
`gtTriGrade5`			Plate	Face

Table Types

Strand7 defines a number of different table types, for use in different contexts. In the GUI these are defined under **LAYOUTS/Tables**, and they are identified in the API by the following constants.

TableType	Table Description
`ttAccVsTime`	Acceleration vs time
`ttDispVsTime`	Displacement vs time
`ttForceDisplacement`	Force vs displacement
`ttForceVelocity`	Force vs velocity
`ttMomentCurvature`	Moment vs curvature
`ttMomentRotation`	Moment vs rotation
`ttStrainTime`	Strain vs time
`ttStressStrain`	Stress vs strain
`ttTemperatureVsTime`	Temperature vs time
`ttVelVsTime`	Velocity vs time
`ttVsFrequency`	Factor vs frequency/period
`ttVsPosition`	Factor vs position
`ttVsTemperature`	Factor vs temperature
`ttVsTime`	Factor vs time
`ttVsVelocity`	Factor vs velocity

The functions

St7NewTableType
St7DeleteTableType
St7SetTableTypeName
St7GetTableTypeName
St7GetNumTableTypeRows
St7SetTableTypeData
St7GetTableTypeData

all uniquely identify tables by means of their TableType parameter (listed above) and TableID.

Functions that pass a TableID parameter always assume a given TableType is being passed. Usually a given function parameter will only set a given table type. In the case of *St7SetPropertyTable* and *St7GetPropertyTable*, the assumed type depends on the parameter PropTableType. The constants to define this parameter are listed below, along with the TableType they are setting.

PropTableType	Table Type	Information
`ptBeamStiffModVsTemp`	Factor vs temperature	Beam **Modulus or Stiffness vs Temperature**

`ptBeamAlphaVsTemp`	Factor vs temperature	Beam **Thermal Expansion vs Temperature**
`ptBeamConductVsTemp`	Factor vs temperature	Beam **Conductivity vs Temperature**
`ptBeamCpVsTemp`	Factor vs temperature	Beam **Specific Heat vs Temperature**
`ptBeamStiffModVsTime`	Factor vs time	Beam **Modulus or Stiffness vs Time**
`ptBeamConductVsTime`	Factor vs time	Beam **Conductivity vs Time**
`ptSpringAxialVsDisp`	Force vs Displacement	Spring **Axial Force vs Displacement**
`ptSpringTorqueVsTwist`	Moment vs Rotation	Spring **Torque vs Angle of Twist**
`ptSpringAxialVsVelocity`	Force vs Velocity	Spring **Axial Force vs Velocity**
`ptBeamStressVsStrain`	Stress vs Strain	Beam **Stress vs Strain**
`ptBeamMomentK1`	Moment vs Curvature	Beam **Moment vs Curvature Plane 1**
`ptBeamMomentK2`	Moment vs Curvature	Beam **Moment vs Curvature Plane 2**
`ptConnectionShear1`	Force vs Displacement	Connection **Translation Stiffness 1**
`ptConnectionShear2`	Force vs Displacement	Connection **Translation Stiffness 2**
`ptConnectionAxial`	Force vs Displacement	Connection **Translation Stiffness 3**
`ptConnectionBend1`	Moment vs Rotation	Connection **Rotation Stiffness 1**
`ptConnectionBend2`	Moment vs Rotation	Connection **Rotation Stiffness 2**
`ptConnectionTorque`	Moment vs Rotation	Connection **Rotation Stiffness 3**
`ptBeamYieldVsTemp`	Factor vs Temperature	Beam **Yield vs Temperature**
`ptPlateModVsTemp`	Factor vs Temperature	Plate **Modulus vs Temperature**
`ptPlateAlphaVsTemp`	Factor vs Temperature	Plate **Thermal Expansion vs Temperature**
`ptPlateConductVsTemp`	Factor vs Temperature	Plate **Conductivity vs Temperature**
`ptPlateCpVsTemp`	Factor vs Temperature	Plate **Specific Heat vs Temperature**
`ptPlateModVsTime`	Factor vs time	Plate **Modulus vs Time**
`ptPlateConductVsTime`	Factor vs time	Plate **Conductivity vs Time**
`ptPlateStressVsStrain`	Stress vs Strain	Plate **Stress vs Strain**
`ptPlateYieldVsTemp`	Factor vs Temperature	Plate **Yield vs Temperature**
`ptBrickModVsTemp`	Factor vs Temperature	Brick **Modulus vs Temperature**
`ptBrickAlphaVsTemp`	Factor vs Temperature	Brick **Thermal Expansion vs Temperature**
`ptBrickConductVsTemp`	Factor vs Temperature	Brick **Conductivity vs Temperature**
`ptBrickCpVsTemp`	Factor vs Temperature	Brick **Specific Heat vs Temperature**
`ptBrickModVsTime`	Factor vs time	Brick **Modulus vs Time**
`ptBrickConductVsTime`	Factor vs time	Brick **Conductivity vs Time**
`ptBrickStressVsStrain`	Stress vs Strain	Brick **Stress vs Strain**
`ptBrickYieldVsTemp`	Factor vs Temperature	Brick **Yield vs Temperature**

Solver Options

Solver Types

Strand7 offers the following solvers:

`stLinearStatic`	Linear static analysis for displacement and stress.
`stLinearBuckling`	Linear elastic buckling of structures subjected to internal forces calculated using one of stLinearStatic, stNonlinearStatic or stQuasiStatic.
`stLoadInfluence`	Linear influence surface calculation for displacement, force, moment and stress variables.
`stNonlinearStatic`	Nonlinear static analysis for displacement and stress. Nonlinearity can be material, geometric and contact.
`stQuasiStatic`	Nonlinear quasi-static response of structures subject to constant or time-varying loads. Nonlinearity can be material, geometric and contact.
`stNaturalFrequency`	Undamped natural frequencies of structures, with or without internal forces; internal forces calculated using one of stLinearStatic, stNonlinearStatic or stQuasiStatic.
`stHarmonicResponse`	Dynamic steady state response of structures subjected to sinusoidal excitations (external forces, base displacement, base velocity or base acceleration) using modal superposition of modes calculated using stNaturalFrequency.
`stSpectralResponse`	Spectral response of structures due to loading defined by way of a frequency response spectrum, using modal superposition of modes calculated using stNaturalFrequency.
`stLinearTransientDynamic`	Linear dynamic response of structures subject to constant or time-varying loads, using either full system integration or modal superposition of modes calculated using stNaturalFrequency.
`stNonlinearTransientDynamic`	Nonlinear dynamic response of structures subject to constant or time-varying loads, using full system integration. Nonlinearity can be material, geometric and contact.
`stSteadyHeat`	Steady state linear or nonlinear heat transfer analysis of structures for conduction, convection and radiation.
`stTransientHeat`	Transient linear or nonlinear heat transfer analysis of structures for conduction, convection and radiation with time dependent material and boundary conditions.

Solver Modes

Solvers may be launched by the Strand7 API in one of several modes, relating to the creation and display of a solver window. These inputs, used by *St7InsituStress*, *St7RunSolver* and *St7RunSolverProcess*, are described below.

`smBackgroundRun`	No solver dialog is created, process terminates on completion.
`smNormalCloseRun`	Full solver dialog is displayed, process terminates on completion.
`smNormalRun`	Full solver dialog is displayed, process waits for manual termination.
`smProgressRun`	Solver progress bar is displayed, process terminates on completion.

Result Types

It is possible to control which of the calculated results are written to the result file, using the functions *St7SetEntityResult* and *St7GetEntityResult*.

Mechanical solvers

`srNodeAcceleration`	Node acceleration.
`srNodeInertia`	Node inertia force; applicable to harmonic response and spectral response.
`srNodeReaction`	Node reaction.
`srNodeVelocity`	Node velocity.
`srBeamForce`	Beam force.
`srBeamMNLStress`	Beam MNL stresses.
`srBeamStrain`	Beam strain.
`srPlateStress`	Plate stress.
`srPlateStrain`	Plate strain.
`srBrickStress`	Brick stress.
`srBrickStrain`	Brick strain.
`srElementNodeForce`	Element node force.
`srLinkForce`	Link node force.

Heat solvers

`hrNodeFlux`	Node heat flux.
`hrBeamFlux`	Beam heat flux.

`hrPlateFlux`	Plate heat flux.
`hrBrickFlux`	Brick heat flux.
`hrLinkFlux`	Link heat flux.

Modal solvers

`frBeamForcePattern`	Beam force pattern.
`frBeamStrainPattern`	Beam strain pattern.
`frPlateStressPattern`	Plate stress pattern.
`frPlateStrainPattern`	Plate strain pattern.
`frBrickStressPattern`	Brick stress pattern.
`frBrickStrainPattern`	Brick strain pattern.

Solver Parameters

Solver defaults in Strand7 are set under the **SOLVERS** tab. These defaults fall into three categories of information; logical flags (checkboxes), integer parameters (fields setting an integer) or double parameters (fields accepting a real number).

Logical Parameters

The functions *St7SetSolverDefaultsLogical* and *St7GetSolverDefaultsLogical* may be used to manipulate the logical solver parameters indicated by the constants below.

`spAddKg`	Applies to geometric nonlinear analysis. If True, the geometric or stress stiffness matrix, KG, is added to the global stiffness matrix to form a tangent stiffness matrix.
`spAllowExtraIterations`	Applies to the nonlinear solvers. If True, extra iterations beyond spMaxIterationNonlin are allowed when the solution is convergent but not yet converged.
`spAppendRemainingTime`	Applies to the time-based solvers when continuing a previous analysis. If True, the remaining time in the time table, depending on the restart time, is used to continue the analysis. If False, the entire time table is appended to the continuing analysis.
`spAutoNewmarkAlpha`	Applies to transient dynamic analysis with the Newmark method. If True, the Newmark Alpha parameter is calculated automatically from the Beta parameter. If False, a user-specifed value of Alpha is used.

`spAutoPCGIterations`	If True, the maximum number of PCG iterations when using the PCG solver scheme is automatically set.
`spAutoScaleKg`	Applies to geometric nonlinear analysis. If True, the geometric stiffness matrix, KG, is automatically scaled to help improve convergence, where necessary.
`spAutoShift`	Applies to natural frequency analysis. If True, a frequency shift is automatically applied when rigid body motion is detected.
`spAutoWorkingSet`	Applies to natural frequency and linear buckling analysis. If True, the working set of eigenvectors available for subspace iteration is automatically expanded.
`spCablesAsMultiCase`	Applies to cables in linear static analysis. If True, a global stiffness matrix is assembled for each load case, solving one load case at a time. Each assembly calculates the stiffness of the cable based on the loads in that load case. If False, a single global stiffness matrix is assembled for all load cases, with the cable stiffness calculated from a single specified load case; this cable stiffness is used for all the considered load cases.
`spCalcDampingRatios`	Applies to natural frequency analysis. If True, damping ratios are calculated for each mode and reported in the solver log file.
`spCheckEigenvector`	Applies to natural frequency and linear buckling analysis. If True, iteration continues until both the eigenvalue and the eigenvector are converged. If False, iteration continues only until the eigenvalue is converged.
`spDampingForce`	Applies to transient dynamic analysis. If True, the internal damping forces due to viscous and Rayleigh damping are added to the nodal reactions.
`spDoInstantNTA`	Applies to nonlinear transient dyanamic analysis with creep. If True, the quasi-static initial stress state is established before commencing the time-based analysis.
`spDoResidualsCheck`	Applies to linear staic analysis. If True, a residuals check is performed to assess the equilibrium of the solution.
`spDoSturm`	Applies to natural frequency and linear buckling analysis. If True, a Sturm check is performed at the end of the eigenvalue extraction procedure to detect whether any modes are missing from the calculated set.
`spExcludeMassX`	Applies to dynamic analysis. If True, mass components in the global X direction of the mass matrix are set to zero.

`spExcludeMassY`	Applies to dynamic analysis. If True, mass components in the global Y direction of the mass matrix are set to zero.
`spExcludeMassZ`	Applies to dynamic analysis. If True, mass components in the global Z direction of the mass matrix are set to zero.
`spForceSingularityCheck`	Applies to the structural solvers. If True, the presence of singularities in the global stiffness matrix is detected before factorising the matrix.
`spFullSystemTransient`	Applies to linear transient dynamic analysis. If True, full system integration is used. If False, modal superposition is used.
`spIgnoreCompressiveBeamKg`	Applies to geometric nonlinear analysis. If True, the geometric stiffness matrix, KG, is not added for beam elements in compression.
`spIncludeFollowerLoadKG`	Applies to linear buckling analysis. If True, the geometric stiffness due to follower loads is added to the global geometric stress stiffness matrix, KG.
`spIncludeLinkReactions`	For the structural solvers, if True, nodal forces due to links are added to the nodal reactions. For the heat solvers, if True, nodal flux due to links is added to the nodal reactions.
`spIncludeRotationalMass`	Applies to dynamic analysis. If True, rotational mass components in the global mass matrix are included. If False, they are set to zero.
`spInertiaForce`	Applies to transient dynamic analysis. If True, the internal forces due to inertia are added to the nodal reactions.
`spLegacyMaxStress`	Use legacy material model for Max Stress materials (not recommended).
`spLimitDisplacementNLA`	Applies to the nonlinear solvers. If True, an upper limit of displacement, set by spMaxDisplacementNLA, is imposed in the analysis. If this displacement is exceeded, the solver will terminate either immediately or at the end of the current step or sub-step, depending on the state of spSaveFinalSubStep.
`spLimitRotationNLA`	Applies to the nonlinear solvers. If True, an upper limit of rotation, set by spMaxRotationNLA, is imposed in the analysis. If this rotation is exceeded, the solver will terminate, either immediately or at the end of the current step or sub-step, depending on the state of `spSaveFinalSubStep`.
`spLumpedLoadBeam`	Applies to the structural solvers. If True, loads applied to beam elements will be represented by a simplified lumped nodal force vector. If False, a consistent nodal vector containing forces and moments will be used.

`spLumpedLoadPlate`	Applies to the structural solvers. If True, loads applied to plate elements will be represented by a simplified lumped nodal force vector. If False, a consistent nodal vector containing forces and moments will be used.
`spLumpedMassBeam`	If True, the mass matrix for beam elements will be represented by a lumped diagonal matrix. If False, a consistent, non-diagonal mass matrix will be used.
`spLumpedMassBrick`	If True, the mass matrix for brick elements will be represented by a lumped diagonal matrix. If False, a consistent, non-diagonal mass matrix will be used.
`spLumpedMassPlate`	If True, the mass matrix for plate elements will be represented by a lumped diagonal matrix. If False, a consistent, non-diagonal mass matrix will be used.
`spNonlinearGeometry`	Applies to the nonlinear structural solvers. If True, geometric nonlinearity is considered in the analysis.
`spNonlinearHeat`	Applies to the heat solvers. If True, nonlinearity is considered in the analysis.
`spNonlinearMaterial`	Applies to the nonlinear structural solvers. If True, material nonlinearity is considered in the analysis.
`spPredictImpact`	Applies to nonlinear transient dynamic analysis. If True, additional time steps are inserted near contact activation events to help improve the capture of the contact event.
`spReducedLogFile`	If True, the solver log file will limit the number of times the same warning or note is generated. The limit is set by spMaxNumRepeatedMessages. If False, all warnings and notes are generated.
`spSaveCQCSpectral`	Applies to spectral response analysis. If True, the CQC spectral results are generated and stored in the result file.
`spSaveFinalSubStep`	When spLimitDisplacementNLA or spLimitRotationNLA are set, by also setting spSaveFinalSubStep to True the final sub-step in the analysis will be saved to the result file. If False, that sub-step will not be saved as the solver will terminate immediately.
`spSaveIntermediate`	Applies to the nonlinear solvers. If True, the results of sub-steps generated by any of the automatic sub-stepping options will be stored in the result file. If False, the sub-steps will not be stored.

`spSaveLastRestartStep`	Applies to the nonlinear solvers. If True, the restart file will contain only the information required to restart the analysis from the last saved step. If False, the restart file will contain the information required to restart the analysis from any saved step.
`spSaveRestartFile`	Applies to the nonlinear solvers. If True, a restart file will be saved enabling the analysis to be subsequently restarted.
`spSaveSRSSSpectral`	Applies to spectral response analysis. If True, the SRSS spectral results are generated and stored in the result file.
`spSaveTableInsertedSteps`	Applies to the time-based solvers. If True, additional time steps are added to the analysis to coincide with points contained in the time tables associated with considered load cases.
`spScaleSupports`	Applies to nonlinear analysis. If True, the stiffness of element face support attributes is scaled, if required to help achieve better convergence.
`spShowConvergenceGraph`	If True, the solver window will initially show the convergence graph.
`spShowMessages`	If True, the solver window will initially show the solver messages.
`spShowProgress`	If True, the progress bar will initially be shown in the solver window.
`spSolverGeneratesCombinations`	Applies to linear static analysis. If True, combinations are generated by the solver as part of the linear static analysis.
`spSpectralBaseExcitation`	Applies to spectral response analysis. If True, base excitation cases are considered in the analysis.
`spSpectralLoadExcitation`	Applies to spectral response analysis. If True, load excitation cases are considered in the analysis.
`spSuppressAllSingularities`	If True, all singularities detected in the global stiffness matrix are artificially suppressed by the addition of a small fictitious stiffness. If False, the singularity will cause the solver to terminate with an error.

Integer Parameters

The functions *St7SetSolverDefaultsInteger* and *St7GetSolverDefaultsInteger* may be used to manipulate the integer solver parameters indicated by the constants below.

`spBeamKgType`	Applies to geometric nonlinear analysis to set how the beam geometric stress stiffness matrix, KG, is calculated; either scSimplifiedBeamKg or scCompleteBeamKg.

`spBeamLength`	Applies to geometric nonlinear analysis to set whether the beam length is updated or not during the analysis; either scInitialBeamLength or scUpdatedBeamLength.
`spCurveFitTimeUnit`	Applies to quasi-static and nonlinear transient dynamic analysis when creep nonlinearity is included: creep curve fit time unit; one of tuMilliSec, tuSec, tuMin, tuHour or tuDay.
`spDynamicAutoStepping`	Applies to quasi-static and nonlinear transient dynamic analysis: dynamic sub-stepping option; one of scDynamicAutoStepNone, scDynamicAutoStepTime, scDynamicAutoStepDispTime or scDynamicAutoStepDispDisp.
`spExpandWorkingSet`	Applies to linear buckling and natural frequency analysis when spAutoWorkingSet is False; additional modes to be included in the subspace iteration, which can help with convergence.
`spFiniteStrainDefinition`	Finite strain definition for geometric nonlinear analysis; one of scFiniteStrainNominal, scFiniteStrainEng, scFiniteStrainGreen.
`spFormNonlinHeatStiffMatrix`	Applies to nonlinear transient heat transfer analysis: matrix update option; one of scHeatMatrixEveryRow, scHeatMatrixSavedStep or scHeatMatrixEveryStep.
`spFormStiffMatrix`	Applies to the nonlinear structural solvers: stiffness matrix update option; one of scStiffnessMatrixEveryIteration, scStiffnessMatrixTwoIterations, scStiffnessMatrixOneIteration, scStiffnessMatrixAutomatic.
`spMaxConjugateGradientIter`	Maximum number of PCG iterations when using the PCG solver scheme.
`spMaxIterationEig`	Applies to linear buckling and natural frequency analysis; maximum number of iterations for the eigenvalue solver.
`spMaxIterationHeat`	Maximum number of iterations for nonlinear steady heat transfer analysis.
`spMaxIterationNonlin`	Applies to nonlinear static, quasi-static and nonlinear transient dynamic analysis; maximum number of iterations for a load step.
`spMaxNumViscoUnits`	Applies to the nonlinear solvers with creep; maximum number of visco-elastic creep units to be used in any creep material. The value is a hint for the solver rather than an absolute setting.
`spMaxNumRepeatedMessages`	Maximum number of log file notes and warnings of the same type.

`spMaxUpdateInterval`	Maximum number of iterations between stiffness matrix updates when spFormStiffMatrix=scStiffnessMatrixAutomatic.
`spMinNumViscoUnits`	Applies to the nonlinear solvers with creep; minimum number of visco-elastic creep units to be used in any creep material. The value is a hint for the solver rather than an absolute setting.
`spNumBeamSlicesModal`	Applies to natural frequency, harmonic response and spectral response; number of beam slices to be generated for force and moment results along a beam element.
`spNumBucklingModes`	Number of modes to be calculated in linear buckling analysis.
`spNumFrequency`	Number of modes to be calculated in natural frequency analysis.
`spStaticAutoStepping`	Applies to nonlinear static analysis: static sub-stepping option; one of scStaticAutoStepNone, scStaticAutoStepLoad, scStaticAutoStepDispLoad, scStaticAutoStepDispDisp or scStaticAutoStepDispArc.
`spTreeStartNumber`	Refers to a node number, which applies when the Matrix Node Ordering option is set to Tree.

Double Parameters

The functions *St7SetSolverDefaultsDouble* and *St7GetSolverDefaultsDouble* may be used to manipulate the double solver parameters (real numbers) indicated by the constants below.

`spBucklingShift`	Buckling shift for linear buckling analysis; modes nearest to spBucklingShift are calculated.
`spCavityVolumeAveraging`	Cavity volume averaging factor for convergence control of nonlinear analysis of ideal gas cavities.
`spClusterZeroDiagonal`	Constraint matrix zero diagonal for the factorisation of link matrices.
`spConjugateGradientTol`	Solver tolerance to define convergence when using the PCG solver scheme.
`spCurveFitTime`	Applies to the nonlinear solvers with creep; creep curve fit time parameter used to interpolate creep strain as a function of time to the calculation of creep coefficients.
`spDrillStiffFactorQ4`	Drilling stiffness multiplier for Quad4 shell elements.

`spDrillStiffFactorQ8`	Drilling stiffness multiplier for Tri6, Quad8 and Quad9 shell elements.
`spDrillStiffFactorT3`	Drilling stiffness multiplier for Tri3 shell elements.
`spEigenTolerance`	Applies to linear buckling and natural frequency analysis; eigenvalue tolerance for defining convergence of eigenvalues, and convergence of eigenvectors when spCheckEigenvector=True.
`spFrequencyShift`	Frequency shift for natural frequency analysis; modes nearest to spFrequencyShift are calculated.
`spFrictionModulusRatio`	Applies to point contact elements in the nonlinear structural solvers and sets the ratio between tangent and secant modulus for point contact elements with friction to help improve convergence.
`spGlobalZeroDiagonal`	Matrix zero diagonal for the global stiffness matrix; a diagonal less than spGlobalZeroDiagonal signifies a singular matrix.
`spMaxDispChange`	Applies to the nonlinear structural solvers when spFormStiffMatrix=scStiffnessMatrixAutomatic; maximum residual displacement ratio change before the stiffness matrix is updated.
`spMaxDisplacementNLA`	Applies to the nonlinear structural solvers when spLimitDisplacementNLA is True. The solver will terminate when the total displacements exceed this value.
`spMaxDynamicPointContactFactor`	Applies to the nonlinear structural solvers; maximum scaling factor applicable to dynamic stiffness point contact elements.
`spMaxEigenRatio`	Ratio between maximum and minimum Eigenvalues during Subspace iteration; Eigenvalues that exceed this ratio are removed from the Subspace.
`spMaxFibreStrainInc`	Applies to the nonlinear material structural solvers; maximum MNL beam fibre strain increment before sub-stepping is invoked.
`spMaxRotation`	Applies to the nonlinear structural solvers; maximum incremental rotation change before sub-stepping is invoked.
`spMaxDispRatio`	Applies to the nonlinear structural solvers; maximum residual displacement change, as a ratio of the bounding box length of the model, before sub-stepping is invoked.
`spMaxNormalsAngle`	Maximum angle between normals of adjacent plate elements within which an average normal direction will be used for the addition of drilling stiffness.

`spMaxResidualChange`	Applies to the nonlinear structural solvers when spFormStiffMatrix=scStiffnessMatrixAutomatic; maximum residual force norm change before the stiffness matrix is updated.
`spMaxRotationNLA`	Applies to the nonlinear structural solvers when spLimitRotationNLA is True. The solver will terminate when the total rotations exceed this value.
`spMinArcLengthFactor`	Applies to the nonlinear static solver when the Displacement control (Arc length) sub-stepping option is used; minimum arc length reduction factor for non-convergent load steps. The arc length in non-convergent load steps will be reduced until the step is converged or this factor is reached.
`spMinDimension`	Minimum element dimension, such as beam length or plate edge length, below which it is considered to be zero.
`spMinInternalAngle`	Minimum element internal angle in plate elements, below which a warning is issued. The setting does not affect the element matrix.
`spMinLoadReductionFactor`	Minimum load reduction factor for nonlinear analysis with sub-stepping enabled. The load in non-convergent load steps will be reduced until the step is converged or this factor is reached.
`spMinTimeStep`	Applies to the quasi-static and nonlinear transient dynamic solvers; the time step in non-convergent steps will be reduced until the step is converged or this time step is reached.
`spMNLTangentRatio`	Applies to elements that consider nonlinear material behaviour in the nonlinear structural solvers; sets the ratio between tangent and secant modulus to help improve convergence.
`spNewmarkAlpha`	Applies to linear and nonlinear transient dynamic analysis with the Newmark time integration scheme when spAutoNewmarkAlpha=False; alpha parameter.
`spNewmarkBeta`	Applies to linear and nonlinear transient dynamic analysis with the Newmark time integration scheme; beta parameter.
`spNonlinDispTolerance`	Applies to the nonlinear structural solvers; displacements in a load step are considered converged when the ratio of the iterative displacement norm to the total displacement norm is below this value.
`spNonlinHeatTolerance`	Applies to the steady heat solver; temperatures are considered converged when the ratio of the temperature change norm to the total temperature norm is below this value.

`spNonlinResidualTolerance`	Applies to the nonlinear structural solvers; residual forces in a load step are considered acceptable, and the load step therefore converged, when the ratio of the current residual force norm to the total applied force norm is below this value.
`spRelaxationFactor`	Applies to transient heat transfer analysis; relaxation parameter to control the time integration between an explicit scheme (spRelaxationFactor = 0.0) and an implicit scheme (spRelaxationFactor = 1.0).
`spSpacingBias`	Applies to the nonlinear solvers with creep; creep curve fit spacing parameter used to bias the insertion of time sampling points either closer to time = 0.0 or closer to time = spCurveFitTime.
`spTimeStepParam`	Applies to the nonlinear solvers with creep; relaxation parameter to control the creep strain time integration between an explicit scheme (spTimeStepParam = 0.0) and an implicit scheme (spTimeStepParam = 1.0).
`spTransientReferenceTemperature`	Applies to quasi-static and nonlinear transient dynamic analysis when the temperature time history in the analysis comes from a transient heat transfer analysis; reference temperature to be used for thermal strain.
`spUpdateDirContactCheckPoint`	Applies to the nonlinear structural solvers; the axis of update-direction point contacts is kept constant when the compressed element length is less than its initial length times this factor.
`spWilsonTheta`	Applies to linear transient dynamic analysis with the Wilson-Theta time integration scheme; theta parameter.
`spZeroBucklingEigen`	Applies to linear buckling anlysis; eigenvalues (buckling factors) less than this value are considered as zero.
`spZeroContactFactor`	Applies to nonlinear structural analysis; inactive point contact elements are scaled by this factor when adding their contribution to the global stiffness matrix.
`spZeroDiagonal`	For the purpose of issuing a warning, diagonals in the local element stiffness matrix below this value are considered as zero.
`spZeroDisplacement`	Applies to the nonlinear structural solvers; an iterative displacement norm less than this value considers the load step to be converged as far as displacements are concerned, irrespective of the displacement norm ratio.
`spZeroForce`	Applies to the nonlinear structural solvers; a residual force norm less than this value considers the load step to be converged as far

	as residual forces are concerned, irrespective of the force norm ratio.
`spZeroFrequency`	Applies to natural frequency analysis; frequencies less than this value are considered as zero.
`spZeroModalDisp`	Translational displacement components in the Eigenvector of a natural frequency analysis are not considered for the calculation of modal mass and modal stiffness unless their magnitude is greater than this value.
`spZeroRotation`	Used by the structural solvers for the detection of zero rotational stiffness and for the suppression of drilling degrees of freedom on plate elements. At a node, a rotational stiffness that is less than this value times the maximum rotational stiffness will be considered a singularity; it will be suppressed if it corresponds to a plate drilling degree of freeedom or reported in the log file, if spForceSingularityCheck=True.
`spZeroTranslation`	Used by the structural solvers for the detection of zero translational stiffness when spForceSingularityCheck=True. At a node, a translational stiffness that is less than this value times the maximum translational stiffness will be considered a singularity and will be reported in the log file.

Node Results

Node results include displacement, velocity, acceleration, phase, reaction, temperature, heat flux and influence. The functions *St7GetNodeResult*, *St7GetNodeResultEx*, *St7GetNodeResultUCS* and *St7GetNodeResultExUCS* can be used to access these result quantities. Result types can be selected by the following input:

`ResultType`

> One of rtNodeDisp, rtNodeVel, rtNodeAcc, rtNodePhase, rtNodeReact, rtNodeTemp, rtNodeFlux, rtNodeInfluence or rtNodeInertia.

Results obtained using *St7GetNodeResult* and *St7GetNodeResultUCS* are returned in a 6-element array of data. Results obtained using *St7GetNodeResultEx* and *St7GetNodeResultExUCS* are returned in a 14-element array of data.

ResultType: [rtNodeDisp, rtNodeVel, rtNodeAcc, rtNodePhase, rtNodeReact, rtNodeInertia]

These refer to vector results of nodal displacement, velocity, acceleration, phase, reaction and inertia, respectively.

All four functions will return the first six result components in the 123456 format via the following array indices:

[0..2] – correspond to the translational degrees of freedom, and

[3..5] – correspond to the rotational degrees of freedom.

The function *St7GetNodeResultEx* and *St7GetNodeResultExUCS* will return an additional eight results combining the component results via the following array indices:

[6] – corresponds to the translational magnitude given by $\sqrt{[0]^2 + [1]^2}$,

[7] – corresponds to the translational magnitude given by $\sqrt{[1]^2 + [2]^2}$,

[8] – corresponds to the translational magnitude given by $\sqrt{[2]^2 + [0]^2}$,

[9] – corresponds to the translational magnitude given by $\sqrt{[0]^2 + [1]^2 + [2]^2}$,

[10] – corresponds to the rotational magnitude given by $\sqrt{[3]^2 + [4]^2}$,

[11] – corresponds to the rotational magnitude given by $\sqrt{[4]^2 + [5]^2}$,

[12] – corresponds to the rotational magnitude given by $\sqrt{[5]^2 + [3]^2}$, and

[13] – corresponds to the rotational magnitude given by $\sqrt{[3]^2 + [4]^2 + [5]^2}$.

ResultType: [rtNodeInfluence]

Result components are returned in the 123456 format via the following array indices:

[0..2] – correspond to translational degrees of freedom, and

[3..5] – correspond to rotational degrees of freedom.

ResultType: [rtNodeTemp, rtNodeFlux]

Results are the scalar quantities of temperature and total heat flux into the node, respectively.

[0] – scalar quantity of interest.

Beam Results

Beam results include force, stress, strain, release, cable position, flux, creep strain, strain energy, beam displacement and beam reactions. The functions *St7GetBeamResultArray*, *St7GetBeamResultArrayPos*, *St7GetBeamResultEndPos* and *St7GetBeamResultSinglePos* can be used to access these result quantities.

Result quantities can be selected via a combination of the following inputs:

`ResultType`

> One of rtBeamForce, rtBeamAllStrain, rtBeamAllTotalStrain, rtBeamAllStress, rtBeamAxialStress, rtBeamBendingStress, rtBeamFibreStress, rtBeamAvShearStress, rtBeamShearStress, rtBeamCombinedStress, rtPipeHoopStress, rtBeamYieldAreaRatio, rtBeamCableXYZ, rtBeamFlux, rtBeamNodeFlux, rtBeamGradient, rtBeamCreepStrain, rtBeamEnergy, rtBeamNodeReact, rtBeamDisp, rtBeamBirthDisp or rtBeamUser.

`ResultSubType`

> One of stBeamLocal, stBeamPrincipal, stBeamGlobal or a UCS ID.

Results are returned in a one-dimensional array BeamResult, which consists of contiguous blocks of data. Each block corresponds to a location along the beam. The length NumColumns of each block is also returned by the function as it depends on the requested ResultType and ResultSubType.

The number of these blocks is dependent on the function that is called – for example, *St7GetBeamResultSinglePos* will return one such block, whereas *St7GetBeamResultArray* will return an array containing NumStations such blocks.

Constants are provided that index specific result quantities within each block of data. For example, the axial force at the i^{th} beam station is stored in the location:

```
BeamResult[(i-1)*NumColumns+ipBeamAxialF]
```

These constants are specific to the result type requested – appropriate constants for each result type are listed in the rest of this section.

ResultType: rtBeamForce with
ResultSubType: stBeamPrincipal

> ipBeamSF1 – Shear force in the 1 axis direction.
>
> ipBeamBM1 – Bending moment in the 1 axis direction.
>
> ipBeamSF2 – Shear force in the 2 axis direction.
>
> ipBeamBM2 – Bending moment in the 2 axis direction.
>
> ipBeamAxialF – Axial force.
>
> ipBeamTorque – Torque.

ResultType: rtBeamForce with
ResultSubType: stBeamLocal

> ipBeamSFx – Shear force in the local x axis direction.
>
> ipBeamBMx – Bending moment in the local x axis direction.

ipBeamSFy – Shear force in the local y axis direction.

ipBeamBMy – Bending moment in the local y axis direction.

ipBeamAxialF – Axial force.

ipBeamTorque – Torque.

ResultType: rtBeamForce with ResultSubType: stBeamGlobal

At any cut section, the forces/moments are those required to keep End 2 of the beam in equilibrium.

ipBeamFX – Internal force in the global X direction.

ipBeamMX – Internal moment in the global X direction.

ipBeamFY – Internal force in the global Y direction.

ipBeamMY – Internal moment in the global Y direction.

ipBeamFZ – Internal force in the global Z direction.

ipBeamMZ – Internal moment in the global Z direction.

ResultType: rtBeamForce with ResultSubType: UCS ID

At any cut section, the forces/moments are those required to keep End 2 of the beam in equilibrium.

ipBeamFX – Internal force in the global 1 direction of the UCS.

ipBeamMX – Internal moment in the global 1 direction of the UCS.

ipBeamFY – Internal force in the global 2 direction of the UCS.

ipBeamMY – Internal moment in the global 2 direction of the UCS.

ipBeamFZ – Internal force in the global 3 direction of the UCS.

ipBeamMZ – Internal moment in the global 3 direction of the UCS.

ResultType: rtBeamAllStress

ipMinFibreStress – Minimum fibre stress.

ipMaxFibreStress – Maximum fibre stress.

ipMaxShearStress1 – Maximum shear stress in the 1 axis direction.

ipMaxShearStress2 – Maximum shear stress in the 2 axis direction.

ipShearF1MeanShearStress – Average stress due to shear force in the 1 axis direction.

ipShearF2MeanShearStress – Average stress due to shear force in the 2 axis direction.

ipShearStressMag – Shear stress magnitude.

ipMaxPrincipalStress – Maximum principal stress.

ipMinPrincipalStress – Minimum principal stress.

ipMinPipeHoopStress – Minimum hoop stress.

ipMaxPipeHoopStress – Maximum hoop stress.

ipMinAxialStress – Minimum axial stress.

ipMaxAxialStress – Maximum axial stress.

ipMinBendingStress1 – Minimum bending stress in the 1 axis direction.

ipMaxBendingStress1 – Maximum bending stress in the 1 axis direction.

ipMinBendingStress2 – Minimum bending stress in the 2 axis direction.

ipMaxBendingStress2 – Maximum bending stress in the 2 axis direction.

ipYieldAreaRatio – Portion of beam section that has yielded.

ipVonMisesStress – von Mises stress.

ipTrescaStress – Tresca stress.

ipTorqueShearStress – Shear stress due to torque.

ipShearF1ShearStress – Largest magnitude shear stress due to shear force in the 1 axis direction.

ipShearF2ShearStress – Largest magnitude shear stress due to shear force in the 2 axis direction.

ResultType: rtBeamAxialStress, rtBeamBendingStress, rtBeamFibreStress, rtBeamAvShearStress, rtBeamShearStress, rtBeamCombinedStress, rtPipeHoopStress, rtBeamYieldAreaRatio

Since beam stress results can require significant computation to determine, requesting only a subset of the components is a way of speeding up result extraction in applications where not all stress components are relevant. With subset stress result types the index positions are the same as for rtBeamAllStress, but inactive components will not be calculated. Active components for each of the subset stress result types are tabulated below.

	ipMinAxialStress	ipMaxAxialStress	ipMinBendingStress1	ipMaxBendingStress1	ipMinBendingStress2	ipMaxBendingStress2	ipMinFibreStress	ipMaxFibreStress	ipShearF1MeanShearStress	ipShearF2MeanShearStress	ipMaxShearStress1	ipMaxShearStress2	ipShearStressMag	ipTorqueShearStress	ipShearF1ShearStress	ipShearF2ShearStress	ipMinPrincipalStress	ipMaxPrincipalStress	ipVonMisesStress	ipTrescaStress	ipMinPipeHoopStress	ipMaxPipeHoopStress	ipYieldAreaRatio
rtBeamAllStress	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
rtBeamAvShearStress									■	■													
rtBeamAxialStress	■	■																					
rtBeamBendingStress			■	■	■	■																	
rtBeamCombinedStress																	■	■	■	■			
rtBeamFibreStress							■	■															
rtBeamShearStress											■	■	■	■	■	■							
rtBeamYieldAreaRatio																							■
rtPipeHoopStress																					■	■	

ResultType: rtBeamAllStrain, rtBeamAllTotalStrain

ipAxialStrain – Axial strain.

ipCurvature1 – Curvature in the 1 axis direction.

ipCurvature2 – Curvature in the 2 axis direction.

ipTwist – Twist.

ipMinFibreStrain – Minimum fibre strain.

ipMaxFibreStrain – Maximum fibre strain.

ResultType: rtBeamCreepStrain

ipMinFibreCreepStrain – Minimum fibre creep strain.

ipMaxFibreCreepStrain – Maximum fibre creep strain.

ipMinFibreCreepStrainRate – Minimum fibre creep strain rate.

ipMaxFibreCreepStrainRate – Maximum fibre creep strain rate.

ipShrinkageStrain – Shrinkage strain.

ResultType: rtBeamCableXYZ

Global displacement components indexed according to the 123 convention.

ResultType: rtBeamEnergy

ipBeamEnergyStored – Stored energy.

ipBeamEnergySpent – Spent energy.

ResultType: rtBeamNodeReact with ResultSubType: stBeamGlobal

Global reaction components indexed according to the 123456 convention.

ResultType: rtBeamNodeReact with ResultSubType: UCS ID

UCS reaction components indexed according to the 123456 convention.

ResultType: [rtBeamDisp/rtBeamBirthDisp] with ResultSubType: [stBeamLocal/stBeamGlobal/stBeamPrincipal]

Displacement defined in 123456 format for the coordinate system indicated by ResultSubType.

ResultType: rtBeamFlux

Heat flux within element.

ResultType: rtBeamGradient

Temperature gradient.

ResultType: rtBeamNodeFlux

Heat flux at nodes.

Plate Results

Plate results include stress, strain, strain energy, force moment, effective force, curvature, ply stress, ply strain, ply reserve, heat flux, temperature gradient, reinforcement design, creep strain, soil characteristics, nodal reactions and user defined quantities. The *St7GetPlateResultArray* function can be used to access these result quantities.

Result quantities can be selected via a combination of the following inputs:

`ResultType`

> One of rtPlateStress, rtPlateEffectiveStress, rtPlateStrain, rtPlateTotalStrain, rtPlateEnergyDensity, rtPlateEnergyIntegral, rtPlateForce, rtPlateEffectiveForce, rtPlateMoment, rtPlateCurvature, rtPlateTotalCurvature, rtPlatePlyStress, rtPlatePlyStrain, rtPlatePlyReserve, rtPlateFlux, rtPlateNodeFlux, rtPlateGradient, rtPlateRCDesign, rtPlateCreepStrain, rtPlateSoil, rtPlateUser, rtPlateNodeReact, rtPlateNodeDisp or rtPlateNodeBirthDisp.

`ResultSubType`

> One of stPlateLocal, stPlateGlobal, stPlateCombined, stPlateSupport, stPlateDevLocal, stPlateDevGlobal, stPlateDevCombined, stPlateCavity or the ID of a UCS into which result components are to be transformed. Note that the Global Cartesian coordinate system is defined as UCS 1.

The rtPlateEffectiveForce quantity is not available for axisymmetric plates.

The Layers input has a different interpretation depending on ResultType. When ResultType is rtPlateRCDesign, use the reinforcement layer number (1 to 4). When ResultType is one of rtPlatePlyStress, rtPlatePlyStrain or rtPlatePlyReserve, refer to the table below. For other result types the input is not relevant.

Layer input	**Result output**
Ply number	One specified ply.
rsPlyMinValue	Minimum of all plies.
rsPlyMaxValue	Maximum of all plies.
rsPlyMaxMag	Maximum magnitude of all plies.
rsPlyMinValueActivePlies	Minimum of plies enabled by *True* to perform the angle check; False to skip the angle check. St7EnablePlyPropertyResults.
rsPlyMaxValueActivePlies	Maximum of plies enabled by *True* to perform the angle check; False to skip the angle check. St7EnablePlyPropertyResults.
rsPlyMaxMagActivePlies	Maximum magnitude of plies enabled by *True* to perform the angle check; False to skip the angle check. St7EnablePlyPropertyResults.

Results are returned in a one-dimensional array `PlateResult`, which consists of contiguous blocks of data. Each block corresponds to a location on the plate. The length `NumColumns` of each block depends on the requested result quantity and is returned by *St7GetPlateResultArray*. The total number of these blocks `NumPoints` depends on the input `SampleLocation` and is also returned. The total number of quantities returned in `PlateResult` is therefore `NumPoints*NumColumns`.

Constants are provided that index specific result quantities within each block of data. For example, the plate local xy force at the ith Gauss point is stored in the location:

```
PlateResult[(i-1)*NumColumns+ipPlateLocalxy]
```

These constants are specific to the result type requested – appropriate constants for each result type are listed in the rest of this section.

ResultType: [rtPlateStress/rtPlateEffectiveStress/rtPlateStrain/rtPlateTotalStrain/rtPlateCurvature/rtPlateTotalCurvature/rtPlateCreepStrain/rtPlateMoment/rtPlateForce/rtPlateEffectiveForce] with
ResultSubType: stPlateLocal

ipPlateLocalxx – Local xx component.

ipPlateLocalyy – Local yy component.

ipPlateLocalzz – Local zz component.

ipPlateLocalxy – Local xy component.

ipPlateLocalyz – Local yz component.

ipPlateLocalxz – Local zx component.

ResultType: [rtPlateStress/rtPlateEffectiveStress/rtPlateStrain/rtPlateTotalStrain/rtPlateCurvature/rtPlateTotalCurvature/rtPlateForce/rtPlateEffectiveForce/rtPlateMoment/rtPlateCreepStrain] with
ResultSubType: stPlateGlobal

ipPlateGlobalXX – Global XX component.

ipPlateGlobalYY – Global YY component.

ipPlateGlobalZZ – Global ZZ component.

ipPlateGlobalXY – Global XY component.

ipPlateGlobalYZ – Global YZ component.

ipPlateGlobalZX – Global ZX component.

Exceptionally, for axisymmetric plates use the following constants:

ipPlateAxiGlobalRR – Axisymmetric RR component.

ipPlateAxiGlobalZZ – Axisymmetric ZZ component.

ipPlateAxiGlobalTT – Axisymmetric TT component.

ipPlateAxiGlobalRZ – Axisymmetric RZ component.

ResultType: [rtPlateStress/rtPlateEffectiveStress/rtPlateStrain/rtPlateTotalStrain/rtPlateCurvature/rtPlateTotalCurvature/rtPlateForce/rtPlateEffectiveForce/

rtPlateMoment/rtPlateCreepStrain] with ResultSubType: UCS ID

ipPlateUCSXX – UCS 11 component.

ipPlateUCSYY – UCS 22 component.

ipPlateUCSZZ – UCS 33 component.

ipPlateUCSXY – UCS 12 component.

ipPlateUCSYZ – UCS 23 component.

ipPlateUCSZX – UCS 31 component.

ResultType: [rtPlateStress/rtPlateEffectiveStress] with ResultSubType: stPlateCombined

ipPlateCombPrincipal11 – Principal 11 component.

ipPlateCombPrincipal22 – Principal 22 component.

ipPlateCombPrincipalAngle – Principal axis angle.

ipPlateCombVonMises – von Mises quantity.

ipPlateCombTresca – Tresca quantity.

ipPlateCombMohrCoulomb – Mohr Coulomb quantity.

ipPlateCombDruckerPrager – Drucker-Prager quantity.

ipPlateCombYieldIndex – Yield index.

ipPlateCombMagnitude – Maximum absolute principal component.

Exceptionally, for axisymmetric plates use the following constants:

ipPlateAxiCombPrincipal11 – Axisymmetric principal 11 component.

ipPlateAxiCombPrincipal22 – Axisymmetric principal 22 component.

ipPlateAxiCombPrincipal33 – Axisymmetric principal 33 component.

ipPlateAxiCombVonMises – Axisymmetric von Mises quantity.

ipPlateAxiCombTresca – Axisymmetric Tresca quantity.

ipPlateAxiCombMohrCoulomb – Axisymmetric Mohr-Coulomb quantity.

ipPlateAxiCombDruckerPrager – Axisymmetric Drucker-Prager quantity.

ipPlateAxiCombYieldIndex – Axisymmetric yield index.

ipPlateAxiCombMagnitude – Maximum absolute principal component.

ResultType: rtPlateStrain/rtPlateTotalStrain with ResultSubType: stPlateCombined

ipPlateCombPrincipal11 – Principal 11 component.

ipPlateCombPrincipal22 – Principal 22 component.

ipPlateCombPrincipalAngle – Principal axis angle.

ipPlateCombVonMises – von Mises quantity.

ipPlateCombTresca – Tresca quantity.

ipPlateCombPlasticStrain – Plastic strain.

ipPlateCombMagnitude – Maximum absolute principal component.

Exceptionally, for axisymmetric plates use the following constants:

ipPlateAxiCombPrincipal11 – Axisymmetric principal 11 component.

ipPlateAxiCombPrincipal22 – Axisymmetric principal 22 component.

ipPlateAxiCombPrincipal33 – Axisymmetric principal 33 component.

ipPlateAxiCombVonMises – Axisymmetric von Mises quantity.

ipPlateAxiCombTresca – Axisymmetric Tresca quantity.

ipPlateAxiCombPlasticStrain – Axisymmetric plastic strain quantity.

ResultType: rtPlateCreepStrain with ResultSubType: stPlateCombined

ipPlateCombPrincipal11 – Principal 11 component.

ipPlateCombPrincipal22 – Principal 22 component.

ipPlateCombPrincipalAngle – Principal axis angle.

ipPlateCombVonMises – von Mises quantity.

ipPlateCombTresca – Tresca quantity.

ipPlateCombMagnitude – Maximum absolute principal component.

ipPlateCombCreepEffRate – Effective creep strain rate.

ipPlateCombCreepShrinkage – Concrete shrinkage strain.

Exceptionally, for axisymmetric plates use the following constants:

ipPlateAxiCombPrincipal11 – Axisymmetric principal 11 component.

ipPlateAxiCombPrincipal22 – Axisymmetric principal 22 component.

ipPlateAxiCombPrincipal33 – Axisymmetric principal 33 component.

ipPlateAxiCombVonMises – Axisymmetric von Mises quantity.

ipPlateAxiCombTresca – Axisymmetric Tresca quantity.

ipPlateAxiCombCreepEffRate – Axisymmetric effective creep strain rate.

ipPlateAxiCombCreepShrinkage – Axisymmetric concrete shrinkage strain.

ResultType: [rtPlateForce/rtPlateEffectiveForce/rtPlateMoment/rtPlateCurvature/rtPlateTotalCurvature] with ResultSubType: stPlateCombined

ipPlateCombPrincipal11 – Principal 11 component.

ipPlateCombPrincipal22 – Principal 22 component.

ipPlateCombPrincipalAngle – Principal axis angle.

ipPlateCombVonMises – von Mises quantity.

ipPlateCombTresca – Tresca quantity.

ipPlateCombMagnitude – Maximum absolute principal component.

ResultType: rtPlateStress with ResultSubType: stPlateSupport

ipPlateEdgeSupport – Edge support component.

ipPlateFaceSupport – Face support component.

ResultType: [rtPlateStress/rtPlateEffectiveStress] with ResultSubType: stPlateDevLocal

ipPlateLocalMean – Mean.

ipPlateLocalDevxx – Deviatoric xx component.

ipPlateLocalDevyy – Deviatoric yy component.

ResultType: [rtPlateStress/rtPlateEffectiveStress] with ResultSubType: stPlateDevGlobal

ipPlateGlobalMean – Mean.

ipPlateGlobalDevXX – Deviatoric XX component.

ipPlateGlobalDevYY – Deviatoric YY component.

ipPlateGlobalDevZZ – Deviatoric ZZ component.

Exceptionally, for axisymmetric plates use the following constants:

ipPlateAxiGlobalMean – Mean.

ipPlateAxiGlobalDevRR – Axisymmetric deviatoric RR component.

ipPlateAxiGlobalDevZZ – Axisymmetric deviatoric ZZ component.

ipPlateAxiGlobalDevTT – Axisymmetric deviatoric TT component.

ResultType: [rtPlateStress/rtPlateEffectiveStress] with ResultSubType: stPlateDevCombined

ipPlateCombMean – Mean.

ipPlateCombDev11 – Deviatoric principal 11 component.

ipPlateCombDev22 – Deviatoric principal 22 component.

Exceptionally, for axisymmetric plates use the following constants:

ipPlateAxiCombMean – Mean.

ipPlateAxiCombDev11 – Axisymmetric deviatoric principal 11 component.

ipPlateAxiCombDev22 – Axisymmetric deviatoric principal 22 component.

ipPlateAxiCombDev33 – Axisymmetric deviatoric principal 33 component.

ResultType: [rtPlateEnergyDensity/rtPlateEnergyIntegral]

ipPlateEnergyStored – Stored energy.

ipPlateEnergySpent – Spent energy.

ResultType: rtPlatePlyStress

ipPlyStress11 – Ply 11 component.

ipPlyStress22 – Ply 22 component.

ipPlyStress12 – Ply 12 component.

ipPlyILSx – Interlamina Sx component.

ipPlyILSy – Interlamina Sy component.

ipPlyStressPrincipal11 – Principal 11 component.

ipPlyStressPrincipal22 – Principal 22 component.

ipPlyStressVonMises – von Mises quantity.

ipPlyStressTresca – Tresca quantity.

ResultType: rtPlatePlyStrain

ipPlyStrain11 – Ply 11 component.

ipPlyStrain22 – Ply 22 component.

ipPlyStrain12 – Ply 12 component.

ipPlyStrainPrincipal11 – Principal 11 component.

ipPlyStrainPrincipal22 – Principal 22 component.

ResultType: rtPlatePlyReserve

ipPlyMaxStress – Maximum stress.

ipPlyMaxStrain – Maximum strain.

ipPlyTsaiHill – Tsai-Hill measure.

ipPlyModTsaiWu – Modified Tsai-Wu measure.

ipPlyHoffman – Hoffman measure.

ipPlyInterlam – Interlamina stress.

ResultType: rtPlateRCDesign

ipPlateRCWoodArmerMoment – Wood-Armer moment.

ipPlateRCWoodArmerForce – Wood-Armer force.

ipPlateRCSteelArea – Steel area.

ipPlateRCConcreteStrain – Concrete strain ratio.

ipPlateRCSteelAreaLessBase – Steel area less base area.

ipPlateRCUserSteelStress – User steel stress.

ipPlateRCUserConcreteStrain – User concrete strain.

ipPlateRCBlockRatio – Block ratio.

ResultType: rtPlateSoil

ipPlateSoilTotalPorePressure – Total pore pressure.

ipPlateSoilExcessPorePressure – Excess pore pressure.

ipPlateSoilOCRIndex – OCR index.

ipPlateSoilStateIndex – Failure index.

ipPlateSoilVoidRatio – Void ratio.

ResultType: rtPlateNodeReact with
ResultSubType: stPlateGlobal

Global reaction components indexed according to the 123456 convention.

ResultType: rtPlateNodeReact with
ResultSubType: UCS ID

UCS reaction components indexed according to the 123456 convention.

ResultType: [rtPlateNodeDisp/rtPlateNodeBirthDisp] with
ResultSubType: stPlateGlobal

Global displacement components indexed according to the 123456 convention.

ResultType: [rtPlateNodeDisp/rtPlateNodeBirthDisp] with
ResultSubType: UCS ID

UCS displacement components indexed according to the 123456 convention.

ResultType: rtPlateUser

A scalar result calculated by the user defined equation defined by *St7GetResultUserEquation*. See *User Defined Results* for details.

ResultType: [rtPlateFlux/rtPlateGradient] with
ResultSubType: stPlateLocal

ipPlateFluxLocalx – Local x component.

ipPlateFluxLocaly – Local y component.

ipPlateFluxLocalMagxy – Flux magnitude.

ResultType: [rtPlateFlux/rtPlateGradient] with ResultSubType: stPlateGlobal

ipPlateFluxGlobalX – Global X component.

ipPlateFluxGlobalY – Global Y component.

ipPlateFluxGlobalZ – Global Z component.

ipPlateFluxGlobalMagXY – Magnitude of global projection on the XY plane.

ipPlateFluxGlobalMagYZ – Magnitude of global projection on the XY plane.

ipPlateFluxGlobalMagZX – Magnitude of global projection on the XY plane.

ipPlateFluxGlobalMagXYZ – Flux Magnitude

ResultType: [rtPlateFlux/rtPlateGradient] with ResultSubType: UCS ID

ipPlateFluxUCSX – UCS 1 component.

ipPlateFluxUCSY – UCS 2 component.

ipPlateFluxUCSZ – UCS 3 component.

ipPlateFluxUCSMagXY – Magnitude of projection on the UCS 12 plane.

ipPlateFluxUCSMagYZ – Magnitude of projection on the UCS 23 plane.

ipPlateFluxUCSMagZX – Magnitude of projection on the UCS 31 plane.

ipPlateFluxUCSMagXYZ – Flux magnitude.

ResultType: [rtPlateNodeFlux]

Heat flux at nodes.

Brick Results

Brick results include stress, strain, strain energy, heat flux, temperature gradient, creep strain, soil characteristics, nodal reactions and user defined quantities. The *St7GetBrickResultArray* function can be used to access these result quantities.

Result quantities can be selected via a combination of the following inputs:

`ResultType`

> One of rtBrickStress, rtBrickEffectiveStress, rtBrickStrain, rtBrickTotalStrain, rtBrickEnergyDensity, rtBrickEnergyIntegral, rtBrickFlux, rtBrickNodeFlux, rtBrickGradient, rtBrickCreepStrain, rtBrickSoil, rtBrickUser, rtBrickNodeReact, rtBrickNodeDisp or rtBrickNodeBirthDisp.

`ResultSubType`

> One of stBrickLocal, stBrickGlobal, stBrickCombined, stBrickSupport, stBrickDevLocal, stBrickDevGlobal, stBrickDevCombined, stBrickCavity or the ID of a UCS into which result components are to be transformed. Note that the Global Cartesian coordinate system is defined as UCS 1.

Results are returned in a one-dimensional array `BrickResult`, which consists of contiguous blocks of data. Each block corresponds to a location on the brick. The length `NumColumns` of each block depends on the requested result quantity and is returned by *St7GetBrickResultArray*. The total number of these blocks `NumPoints` depends on the input `SampleLocation` and is also returned. The total number of quantities returned in `BrickResult` is therefore `NumPoints*NumColumns`.

Constants are provided that index specific result quantities within each block of data. For example, the brick local xy stress component at the i^{th} Gauss point is stored in the location:

```
BrickResult[(i-1)*NumColumns+ipBrickLocalxy]
```

These constants are specific to the result type requested – appropriate constants for each result type are listed in the rest of this section.

ResultType: [rtBrickStress/rtBrickEffectiveStress/rtBrickStrain/rtBrickTotalStrain/rtBrickCreepStrain] with ResultSubType: stBrickLocal

ipBrickLocalxx – Local xx component.

ipBrickLocalyy – Local yy component.

ipBrickLocalzz – Local zz component.

ipBrickLocalxy – Local xy component.

ipBrickLocalyz – Local yz component.

ipBrickLocalzx – Local zx component.

ResultType: [rtBrickStress/rtBrickEffectiveStress/rtBrickStrain/rtBrickTotalStrain/rtBrickCreepStrain] with ResultSubType: stBrickGlobal

ipBrickGlobalXX – Global XX component.

ipBrickGlobalYY – Global YY component.

ipBrickGlobalZZ – Global ZZ component.

ipBrickGlobalXY – Global XY component.

ipBrickGlobalYZ – Global YZ component.

ipBrickGlobalZX – Global ZX component.

ResultType: [rtBrickStress/rtBrickEffectiveStress/rtBrickStrain/rtBrickTotalStrain/rtBrickCreepStrain] with ResultSubType: UCS ID

ipBrickUCSXX – UCS 11 component.

ipBrickUCSYY – UCS 22 component.

ipBrickUCSZZ – UCS 33 component.

ipBrickUCSXY – UCS 12 component.

ipBrickUCSYZ – UCS 23 component.

ipBrickUCSZX – UCS 31 component.

ResultType: [rtBrickStress/rtBrickEffectiveStress] with ResultSubType: stBrickCombined

ipBrickCombPrincipal11 – Principal 11 component.

ipBrickCombPrincipal22 – Principal 22 component.

ipBrickCombPrincipal33 – Principal 33 component.

ipBrickCombVonMises – von Mises quantity.

ipBrickCombTresca – Tresca quantity.

ipBrickCombMohrCoulomb – Mohr-Coulomb quantity.

ipBrickCombDruckerPrager – Drucker-Prager quantity.

ipBrickCombMean – Mean.

ipBrickCombYieldIndex – Yield index.

ipBrickCombMagnitude – Maximum absolute principal component.

ResultType: rtBrickStrain/rtBrickTotalStrain with ResultSubType: stBrickCombined

ipBrickCombPrincipal11 – Principal 11 component.

ipBrickCombPrincipal22 – Principal 22 component.

ipBrickCombPrincipal33 – Principal 33 component.

ipBrickCombVonMises – von Mises quantity.

ipBrickCombTresca – Tresca quantity.

ipBrickCombPlasticStrain – Plastic strain.

ipBrickCombMean – Mean.

ipBrickCombMagnitude – Maximum absolute principal component.

ResultType: rtBrickCreepStrain with
ResultSubType: stBrickCombined

ipBrickCombPrincipal11 – Principal 11 component.

ipBrickCombPrincipal22 – Principal 22 component.

ipBrickCombPrincipal33 – Principal 33 component.

ipBrickCombVonMises – von Mises quantity.

ipBrickCombTresca – Tresca quantity.

ipBrickCombCreepEffRate – Effective creep rate.

ipBrickCombCreepShrinkage – Concrete shrinkage strain.

ipBrickCombMagnitude – Maximum absolute principal component.

ResultType: rtBrickStress with
ResultSubType: stPlateSupport

ipBrickFaceSupport – Face support.

ResultType: [rtBrickStress/rtBrickEffectiveStress] with
ResultSubType: stBrickDevLocal

ipBrickLocalMean – Mean.

ipBrickLocalDevxx – Local deviatoric xx component.

ipBrickLocalDevyy – Local deviatoric yy component.

ipBrickLocalDevzz – Local deviatoric zz component.

ResultType: [rtBrickStress/rtBrickEffectiveStress] with
ResultSubType: stBrickDevGlobal

ipBrickGlobalMean – Mean.

ipBrickGlobalDevXX – Global deviatoric XX component.

ipBrickGlobalDevYY – Global deviatoric YY component.

ipBrickGlobalDevZZ – Global deviatoric ZZ component.

ResultType: [rtBrickStress/rtBrickEffectiveStress] with
ResultSubType: stBrickDevCombined

ipBrickCombDevMean – Mean.

ipBrickCombDev11 – Principal 11 deviatoric component.

ipBrickCombDev22 – Principal 22 deviatoric component.

ipBrickCombDev33 – Principal 33 deviatoric component.

ResultType: rtBrickNodeReact with
ResultSubType: stBrickGlobal

Global reaction components indexed according to the 123456 convention.

ResultType: rtBrickNodeReact with
ResultSubType: UCS ID

UCS reaction components indexed according to the 123456 convention.

ResultType: [rtBrickNodeDisp/rtBrickNodeBirthDisp] with
ResultSubType: stBrickGlobal

Global displacement components indexed according to the 123456 convention.

ResultType: [rtBrickNodeDisp/rtBrickNodeBirthDisp] with
ResultSubType: UCS ID

UCS displacement components indexed according to the 123456 convention.

ResultType: [rtBrickEnergyDensity/rtBrickEnergyIntegral]

ipBrickEnergyStored – Stored energy.

ipBrickEnergySpent – Spent energy.

ResultType: rtBrickSoil

ipBrickSoilTotalPorePressure – Total pore pressure.

ipBrickSoilExcessPorePressure – Excess pore pressure.

ipBrickSoilOCRIndex – OCR index.

ipBrickSoilStateIndex – Failure index.

ipBrickSoilVoidRatio – Void ratio.

ResultType: rtBrickUser

A scalar result calculated by the user defined equation defined by *St7GetResultUserEquation.* See *User Defined Results* for details.

ResultType: [rtBrickFlux/rtBrickGradient] with
ResultSubType: stBrickLocal

ipBrickFluxLocalx – Local X component.

ipBrickFluxLocaly – Local Y component.

ipBrickFluxLocalz – Local Z component.

ipBrickFluxLocalMagxy – Magnitude of local projection on the XY plane.

ipBrickFluxLocalMagyz – Magnitude of local projection on the YZ plane..

ipBrickFluxLocalMagzx – Magnitude of local projection on the ZX plane..

ipBrickFluxLocalMagxyz – Flux magnitude.

ResultType: [rtBrickFlux/rtBrickGradient] with
ResultSubType: stBrickGlobal

ipBrickFluxGlobalX – Global X component.

ipBrickFluxGlobalY – Global Y component.

ipBrickFluxGlobalZ – Global Z component.

ipBrickFluxGlobalMagXY – Magnitude of global projection on the XY plane.

ipBrickFluxGlobalMagYZ – Magnitude of global projection on the YZ plane.

ipBrickFluxGlobalMagZX – Magnitude of global projection on the ZX plane.

ipBrickFluxGlobalMagXYZ – Flux magnitude.

ResultType: [rtBrickFlux/rtBrickGradient] with
ResultSubType: UCS ID

ipBrickFluxUCSX – UCS 1 component.

ipBrickFluxUCSY – UCS 2 component.

ipBrickFluxUCSZ – UCS 3 component.

ipBrickFluxUCSMagXY – Magnitude of projection on the UCS 12 plane.

ipBrickFluxUCSMagYZ – Magnitude of projection on the UCS 23 plane.

ipBrickFluxUCSMagZX – Magnitude of projection on the UCS 31 plane.

ipBrickFluxUCSMagXYZ – Flux magnitude.

ResultType: [rtBrickNodeFlux]

Heat flux at nodes.

User Defined Results

The calculation of user defined results based on primary result quantities is made possible by the definition of a text equation (string) that uses the primary result quantities as variables (primary variables). In the Strand7 GUI, this is a contour option available for beams, plates and bricks via the **Results Settings** dialog and the **Equations** menu under **VISUAL/Results**. The following functions are used to manage these equations in the Strand7 API:

- *St7SetResultUserEquation*
- *St7GetResultUserEquation*
- *St7StoreResultUserEquation*
- *St7DeleteStoredResultUserEquation*
- *St7ReplaceStoredResultUserEquation*
- *St7RetrieveStoredResultUserEquation*
- *St7GetNumStoredResultUserEquations*
- *St7SetStoredResultUserEquation*

Syntax

Equation strings are case-insensitive and whitespace is ignored. They are formed as a parseable string containing primary variables in square brackets […], numeric constants, operators and functions.

The following is an example of a valid equation for plate elements; it defines the von Mises stress as a function of the principal stresses in 2D:

SQRT(0.5*(([S11]-[S22])^2 + [S22]^2 + [S11]^2))

Here **[S11]** is a primary variable, **0.5** is a constant, **SQRT()** is a function, and **^** is an operator, amongst other components.

Beam Primary Variables

Beam primary variables include element forces and moments along the beam, in either the principal or local directions, as well as a range of section properties such as cross section area, moments of area, and so on (note that in a tapered beam, the section properties may vary along the beam length). The following table lists all the available beam primary variables.

Beam Primary Variables	Description
[SF1]	Shear force in principal 1 direction
[SF2]	Shear force in principal 2 direction
[SFx]	Shear force in local x direction
[SFy]	Shear force in local y direction
[AxF]	Axial force
[BM1]	Bending moment in principal 1 plane
[BM2]	Bending moment in principal 2 plane
[BMx]	Bending moment in local x plane
[BMy]	Bending moment in local y plane

[Trq]	Torque
[CvD1]	Deformational curvature in principal 1 plane
[CvD2]	Deformational curvature in principal 2 plane
[AxDS]	Deformational axial strain
[CvT1]	Total curvature in principal 1 plane
[CvT2]	Total curvature in principal 2 plane
[AxTS]	Total axial strain
[Tws]	Twist
[I11]	Maximum second moment of area
[I22]	Minimum second moment of area
[Ixx]	Moment of area about local x axis
[Iyy]	Moment of area about local y axis
[Ixy]	Cross moment of area about local x-y axes
[Area]	Cross section area
[J]	Torsion constant
[SA1]	Shear area in principal 1 direction
[SA2]	Shear area in principal 2 direction
[S11]	Plastic modulus in principal 1 direction
[S22]	Plastic modulus in principal 2 direction
[Sxx]	Plastic modulus in local x direction
[Syy]	Plastic modulus in local y direction
[Z11p]	Section modulus in the principal 1 direction for stress on the positive side
[Z11n]	Section modulus in the principal 1 direction for stress on the negative side
[Z22p]	Section modulus in the principal 2 direction for stress on the positive side
[Z22n]	Section modulus in the principal 2 direction for stress on the negative side
[Zxxp]	Section modulus in the local x direction for stress on the positive side
[Zxxn]	Section modulus in the local x direction for stress on the negative side
[Zyyp]	Section modulus in the local y direction for stress on the positive side
[Zyyn]	Section modulus in the local y direction for stress on the negative side

Plate Primary Variables

Plate primary variables are generally defined with the convention [ABCD], where:

A – Result type; one of stress [S], deformation strain [E], total strain [T], force [F], moment [M], deformation curvature [K] or total curvature [L].

B – Component axis directions defined by the coordinate system (e.g. xx, yy, 11, etc).

C – Coordinate system; one of local [L], global [G], UCS [U] or combined.

D – Optional plate surface definition, from which the primary variable result is to be extracted; one of mid-plane [-MP], -z [-NZ] or +z [-PZ]. This applies only to the contour shown on the rendered element in the model window; if the option is omitted, the contour displayed depends on how the plate element is rendered (i.e. whether it is rendered as solid or as surface). For the extraction of user defined results on plate elements via functions such as St7GetPlateResultArray, the surface is specifed in the function call, therefore this option is not relevant.

In addition, user defined results for plates may also use the membrane thickness [TM] and the bending thickness [TB] of the plate.

Not all plate primary variables require all of A, B, C and D. The following table illustrates some of the variations of plate primary variable definitions for stress, strain, force and curvature.

Plate Primary Variables	Description
[SXXL]	Direct stress in the local x direction on the surface that is displayed
[SXXL-NZ]	Direct stress in the local x direction on the -z surface
[SYYG]	Direct stress in the global Y direction on the surface that is displayed
[SYYG-PZ]	Direct stress in the global Y direction on the +z surface
[SXXU]	Direct stress in the first axis direction of the current UCS on the surface that is displayed
[SXYU-NZ]	Shear stress in the first two axis directions of the current UCS on the -z surface
[EXXL]	Direct deformation strain in the local x direction on the surface that is displayed
[EXXL-NZ]	Direct deformation strain in the local x direction on the -z surface
[EYYG]	Direct deformation strain in the global Y direction on the surface that is displayed
[EYYG-PZ]	Direct deformation strain in the global Y direction on the +z surface
[EXXU]	Direct deformation strain in the first axis direction of the current UCS on the surface that is displayed
[EXYU-NZ]	Shear deformation strain in the first two axis directions of the current UCS on the -z surface
[FXXL]	Direct force in the local x direction (surface is not relevant for force results)
[FYYG]	Direct force in the global Y direction (surface is not relevant for force results)
[FXXU]	Direct force in the first axis direction of the current UCS (surface is not relevant for force results)
[FXYU]	Shear force in the first two axis directions of the current UCS (surface is not relevant for force results)
[MXXL]	Direct moment in the local x direction (surface is not relevant for moment results)
[MYYG]	Direct moment in the global Y direction (surface is not relevant for moment results)

[MXXU]	Direct moment in the first axis direction of the current UCS (surface is not relevant for moment results)
[MXYU]	Twisting moment in the first two axis directions of the current UCS (surface is not relevant for moment results)
[KXXL]	Direct deformation curvature in the local x direction (surface is not relevant for curvature results)
[KYYG]	Direct deformation curvature in the global Y direction (surface is not relevant for curvature results)
[KXXU]	Direct deformation curvature in the first axis direction of the current UCS (surface is not relevant for curvature results)
[KXYU]	Twist in the first two axis directions of the current UCS (surface is not relevant for curvature results)
[S11-PZ]	Maximum principal stress on the +z surface
[SMEAN-NZ]	Mean stress on the -z surface
[SVM-NZ]	von Mises stress on the -z surface
[STR-NZ]	Tresca stress on the -z surface
[TM]	Plate membrane thickness
[TB]	Plate bending thickness

Brick Primary Variables

Brick primary variables are generally defined with the convention [ABC], where:

A – Result type; one of stress [S], deformation strain [E] or total strain [T].

B – Component axis directions defined by the coordinate system (e.g. xx, yy, 11, etc).

C – Coordinate system; one of local [L], global [G], UCS [U] or combined.

Not all brick primary variables require all of A, B and C. The following table illustrates some of the variations of brick primary variables for stress and strain.

Brick Primary Variables	**Description**
[SXXL]	Direct stress in the local x direction
[SYYG]	Direct stress in the global Y direction
[SXXU]	Direct stress in the first axis direction of the current UCS
[SXYU]	Shear stress in the first two axis directions of the current UCS
[EXXL]	Direct deformation strain in the local x direction
[EYYG]	Direct deformation strain in the global Y direction
[EXXU]	Direct deformation strain in the first axis direction of the current UCS
[EXYU]	Shear deformation strain in the first two axis directions of the current UCS

[S11]	Maximum principal stress
[S33]	Minimum principal stress
[SMEAN]	Mean stress
[SVM]	von Mises stress
[STR]	Tresca stress

Numeric Constants

Numeric constants may be entered in integer format (e.g. 100), in simple floating point format (e.g. 3.14), or in scientific notation (e.g. 1.67E-2, where **E** separates the mantissa from the exponent). If used, the decimal point must be entered as the decimal separator defined in Windows. This is typically the period (.) or the comma (,).

Operators

The equation string supports the standard operators: addition +, subtraction -, multiplication *, division / and exponentiation ^, as well as brackets () to control the order of operations (otherwise the standard BODMAS convention applies).

Functions

The following functions may be used in the equation string.

Function	**Description**
ABS	absolute value
ACOS	arccosine
ASIN	arcsine
ATAN	arctangent
COS	cosine
EXP	natural exponent (base e)
FACT	factorial
HACOS	hyperbolic arccosine
HASIN	hyperbolic arcsine
HATAN	hyperbolic arctangent
HCOS	hyperbolic cosine
HSIN	hyperbolic sine
HTAN	hyperbolic tangent
IF	If (logical test; value if true; value if false) where “;” is the list separator character
IFNEG	returns argument when argument is negative, 0.0 otherwise
IFNEGB	returns 1.0 when argument is negative, 0.0 otherwise
IFPOS	returns argument when argument is positive, 0.0 otherwise

IFPOSB	returns 1.0 when argument is positive, 0.0 otherwise
LN	natural log (base e)
LOG	log base 10
MAX	returns the maximum of a series of values separated by the list separator character
MIN	returns the minimum of a series of values separated by the list separator character
SIN	sine
SQR	square
SQRT	square root
TAN	tangent

Creep Definitions

A number of the creep laws available in Strand7 require the specification of coefficients. These coefficients are defined by the Doubles array passed by *St7SetCreepBasicData* and *St7GetCreepBasicData*. The positions of these coefficients in this array are listed below, along with their relevant creep law.

Primary Power Law – clPrimaryPower

Doubles[0..3] – Coefficients C1, C2, C3 and CT.

Secondary Power Law – clSecondaryPower

Doubles[0..2] – Coefficients C1, C2 and CT.

Primary + Secondary Power Law – clPrimarySecondaryPower

Doubles[0..6] – Coefficients C1, C2, C3, CT1, C4, C5 and CT2.

Secondary Hyperbolic Creep – clSecondaryHyperbolic

Doubles[0..3] – Coefficients C1, C2, C3 and CT.

Secondary Exponential Creep – clSecondaryExponential

Doubles[0..2] – Coefficients C1, C2 and CT.

Theta Projection Creep – clThetaProjection

Doubles[0..3] – Coefficients A1, A2, A3 and A4.

Doubles[4..7] – Coefficients B1, B2, B3 and B4.

Doubles[8..11] – Coefficients C1, C2, C3 and C4.

Doubles[12..15] – Coefficients D1, D2, D3 and D4.

Generalised Graham Creep – clGenGraham

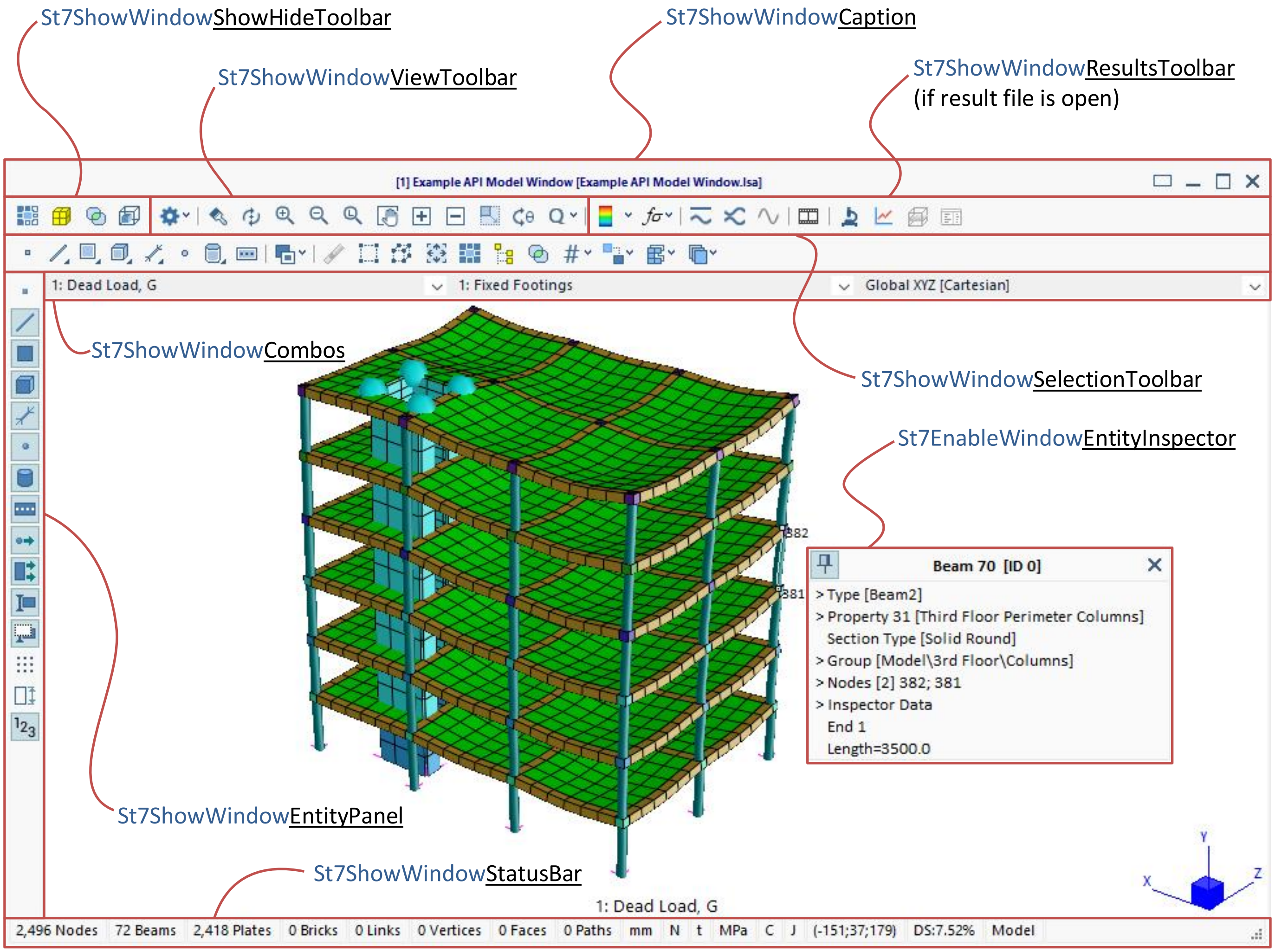
Doubles[0..7] – Coefficients C1, C2, C3, C4, C5, C6, C7 and CT.

Generalised Blackburn Creep – clGenBlackburn

Doubles[0..6] – Coefficients C1, C2, C3, C4, C5, C6 and C7.

Model Window Components

The model window has a number of optional components that may be activated or deactivated using the functions in *Model Window*. The components are shown below, along with the functions used to show or enable them.



RGB Colours

When colours are passed to or returned from a Strand7 API function, the 32-bit RGB format integer is used. The format is composed of four bytes, the least significant three of which carry the colour components.

Most Significant			Least Significant
Null	Blue (0-255)	Green (0-255)	Red (0-255)

For example, consider the colour defined by the components Red = 33, Green = 165 and Blue = 239. The colour would be passed to Strand7 as

$$
\begin{aligned}
RGB &= (256^2 \times Blue) + (256 \times Green) + Red \\
&= 15705377
\end{aligned}
$$

When passing in a colour value, the most significant bit should always be 0. That is, the RGB value should always be less than 16777216.

The functions *St7RGBToColour* and *St7ColourToRGB* are provided to facilitate the conversion of colour representations.

Entity Contours

The following constants are passed by *St7SetEntityContourIndex* and *St7GetEntityContourIndex* as the parameter Index. They identify the pre-processor contour types listed in the **Entity Display** combo boxes.

Beam Contour Types

Constant	Description
ctBeamNone	No contour
ctBeamLength	Beam length
ctBeamAxis1	Coordinate system axis 1 component
ctBeamAxis2	Coordinate system axis 2 component
ctBeamAxis3	Coordinate system axis 3 component
ctBeamEA	EA axial stiffness
ctBeamEI11	EI1 bending stiffness
ctBeamEI22	EI2 bending stiffness
ctBeamGJ	GJ torsional stiffness
ctBeamEAFactor	Scaled EA axial stiffness
ctBeamEI11Factor	Scaled EI1 bending stiffness
ctBeamEI22Factor	Scaled EI2 bending stiffness
ctBeamGJFactor	Scaled GJ torsional stiffness
ctBeamOffset1	Offset in the local beam 1 direction
ctBeamOffset2	Offset in the local beam 2 direction
ctBeamStiffnessFactor1	Stiffness factor for shear stiffness 1
ctBeamStiffnessFactor2	Stiffness factor for shear stiffness 2
ctBeamStiffnessFactor3	Stiffness factor for axial stiffness
ctBeamStiffnessFactor4	Stiffness factor for bending stiffness in plane 1 (about 2 axis)
ctBeamStiffnessFactor5	Stiffness factor for bending stiffness in plane 2 (about 1 axis)

ctBeamStiffnessFactor6	Stiffness factor for torque stiffness
ctBeamMassFactor	Mass factor
ctBeamSupportM1	Support in the -1 principal beam direction
ctBeamSupportP1	Support in the +1 principal beam direction
ctBeamSupportM2	Support in the -2 principal beam direction
ctBeamSupportP2	Support in the +2 principal beam direction
ctBeamSupportGapM1	Compression-only support activation gap in the -1 beam direction
ctBeamSupportGapP1	Compression-only support activation gap in the +1 beam direction
ctBeamSupportGapM2	Compression-only support activation gap in the -2 beam direction
ctBeamSupportGapP2	Compression-only support activation gap in the +2 beam direction
ctBeamTemperature	Applied node temperature
ctBeamPreTension	Pre-tension
ctBeamPreStrain	Pre-strain
ctBeamPreCurvature1	Pre-curvature in the local beam 1 direction
ctBeamPreCurvature2	Pre-curvature in the local beam 2 direction
ctBeamTempGradient1	Temperature gradient in the local beam 1 direction
ctBeamTempGradient2	Temperature gradient in the local beam 2 direction
ctBeamPipePressureIn	Internal pipe pressure
ctBeamPipePressureOut	External pipe pressure
ctBeamPipeTempIn	Internal pipe temperature
ctBeamPipeTempOut	External pipe temperature
ctBeamConvectionCoeff	Convection coefficient
ctBeamConvectionAmbient	Convection ambient temperature
ctBeamRadiationCoeff	Radiation coefficient

ctBeamRadiationAmbient	Radiation ambient temperature
ctBeamHeatFlux	Heat flux
ctBeamHeatSource	Heat source
ctBeamAgeAtFirstLoading	Age at first loading
ctBeamPropertyName	Property name
ctBeamMaterialName	Material name
ctBeamSectionName	Cross section name

Plate Contour Types

Constant	**Description**
ctPlateNone	No Contour
ctPlateAspectRatioMin	Minimum aspect ratio
ctPlateAspectRatioMax	Maximum aspect ratio
ctPlateWarping	Warping
ctPlateInternalAngle	Internal angle
ctPlateInternalAngleRatio	Internal angle ratio
ctPlateDiscreteThicknessM	Discrete membrane thickness
ctPlateContinuousThicknessM	Continuous membrane thickness
ctPlateDiscreteThicknessB	Discrete bending thickness
ctPlateContinuousThicknessB	Continuous bending thickness
ctPlateOffset	Normal offset
ctPlateStiffnessFactor1	Cxx membrane stiffness factor
ctPlateStiffnessFactor2	Cyy membrane stiffness factor
ctPlateStiffnessFactor3	Cgg membrane stiffness factor
ctPlateStiffnessFactor4	Dxx bending stiffness factor

ctPlateStiffnessFactor5	Dyy bending stiffness factor
ctPlateStiffnessFactor6	Dgg bending stiffness factor
ctPlateStiffnessFactor7	Gxx transverse stiffness factor
ctPlateStiffnessFactor8	Gyy transverse stiffness factor
ctPlateMassFactor	Mass factor
ctPlateArea	Area
ctPlateAxis1	Coordinate system axis 1 component
ctPlateAxis2	Coordinate system axis 2 component
ctPlateAxis3	Coordinate system axis 3 component
ctPlateTemperature	Applied temperature
ctPlateEdgeNormalSupport	Edge normal support
ctPlateEdgeLateralSupport	Edge lateral support
ctPlateEdgeSupportGap	Edge support gap
ctPlateFaceNormalSupportMinusZ	Face normal support on the -z plate surface
ctPlateFaceNormalSupportPlusZ	Face normal support on the +z plate surface
ctPlateFaceLateralSupportMinusZ	Face lateral support on the -z plate surface
ctPlateFaceLateralSupportPlusZ	Face lateral support on the +z plate surface
ctPlateFaceSupportGapMinusZ	Compression only face support activation gap on the -z plate surface
ctPlateFaceSupportGapPlusZ	Compression only face support activation gap on the +z plate surface
ctPlatePreStressX	Pre-stress in the local plate x direction
ctPlatePreStressY	Pre-stress in the local plate y direction
ctPlatePreStressZ	Pre-stress in the local plate z direction
ctPlatePreStressMagnitude	Pre-stress magnitude
ctPlatePreStrainX	Pre-strain in the local plate x direction

ctPlatePreStrainY	Pre-strain in the local plate y direction
ctPlatePreStrainZ	Pre-strain in the local plate z direction
ctPlatePreStrainMagnitude	Pre-strain magnitude
ctPlatePreCurvatureX	Pre-curvature in the local plate x direction
ctPlatePreCurvatureY	Pre-curvature in the local plate y direction
ctPlatePreCurvatureMagnitude	Pre-curvature magnitude
ctPlateTempGradient	Temperature gradient
ctPlateEdgeNormalPressure	Edge normal pressure
ctPlateEdgeShear	Edge shear stress
ctPlateEdgeTransverseShear	Edge transverse shear stress
ctPlateEdgeGlobalPressure	Edge global pressure magnitude
ctPlateEdgeGlobalPressureX	Edge global pressure in the global X direction
ctPlateEdgeGlobalPressureY	Edge global pressure in the global Y direction
ctPlateEdgeGlobalPressureZ	Edge global pressure in the global Z direction
ctPlateNormalPressureMinusZ	Normal pressure (-z)
ctPlateNormalPressurePlusZ	Normal pressure (+z)
ctPlateGlobalPressureMinusZ	Global pressure magnitude (-z)
ctPlateGlobalPressureXMinusZ	Pressure in the global X direction (-z)
ctPlateGlobalPressureYMinusZ	Pressure in the global Y direction (-z)
ctPlateGlobalPressureZMinusZ	Pressure in the global Z direction (-z)
ctPlateGlobalPressurePlusZ	Global pressure magnitude (+z)
ctPlateGlobalPressureXPlusZ	Pressure in the global X direction (+z)
ctPlateGlobalPressureYPlusZ	Pressure in the global Y direction (+z)
ctPlateGlobalPressureZPlusZ	Pressure in the global Z direction (+z)

ctPlateFaceShearX	Face shear stress in the local plate x direction
ctPlateFaceShearY	Face shear stress in the local plate y direction
ctPlateFaceShearMagnitude	Face shear stress magnitude
ctPlateNSMass	Non-structural mass
ctPlateDynamicFactor	Non-structural mass dynamic factor
ctPlateConvectionCoeff	Edge convection coefficient
ctPlateConvectionAmbient	Edge convection ambient temperature
ctPlateRadiationCoeff	Edge radiation coefficient
ctPlateRadiationAmbient	Edge radiation ambient temperature
ctPlateHeatFlux	Edge heat flux
ctPlateConvectionCoeffPlusZ	Face convection coefficient (+z)
ctPlateConvectionCoeffMinusZ	Face convection coefficient (-z)
ctPlateConvectionAmbientPlusZ	Face convection ambient temperature (+z)
ctPlateConvectionAmbientMinusZ	Face convection ambient temperature (-z)
ctPlateRadiationCoeffPlusZ	Face radiation coefficient (+z)
ctPlateRadiationCoeffMinusZ	Face radiation coefficient (-z)
ctPlateRadiationAmbientPlusZ	Face radiation ambient temperature (+z)
ctPlateRadiationAmbientMinusZ	Face radiation ambient temperature (-z)
ctPlateHeatSource	Heat source
ctPlateSoilStressSV	Soil stress Sv
ctPlateSoilStressK0	Soil stress K0
ctPlateSoilStressSH	Soil stress Sh
ctPlateSoilRatioOCR	Soil OCR (over-consolidation ratio)
ctPlateSoilRatioE0	Soil ratio e0

ctPlateSoilFluidLevel	Soil fluid level
ctPlateAgeAtFirstLoading	Age at first loading
ctPlatePropertyName	Property name
ctPlateMaterialName	Material name

Brick Contour Types

Constant	Description
ctBrickNone	No contour
ctBrickAspectRatioMin	Minimum aspect ratio
ctBrickAspectRatioMax	Maximum aspect ratio
ctBrickVolume	Brick volume
ctBrickDeterminant	Determinant of brick Jacobian matrix
ctBrickInternalAngle	Internal angle
ctBrickMixedProduct	Mixed product
ctBrickDihedral	Dihedral angle ratio
ctBrickAxis1	Coordinate system axis 1 component
ctBrickAxis2	Coordinate system axis 2 component
ctBrickAxis3	Coordinate system axis 3 component
ctBrickTemperature	Applied temperature
ctBrickNormalSupport	Face normal support
ctBrickLateralSupport	Face lateral support
ctBrickSupportGap	Compression only face support activation gap
ctBrickPreStressX	Pre-stress in the local brick x direction
ctBrickPreStressY	Pre-stress in the local brick y direction
ctBrickPreStressZ	Pre-stress in the local brick z direction

ctBrickPreStressMagnitude	Pre-stress magnitude
ctBrickPreStrainX	Pre-strain in the local brick x direction
ctBrickPreStrainY	Pre-strain in the local brick y direction
ctBrickPreStrainZ	Pre-strain in the local brick z direction
ctBrickPreStrainMagnitude	Pre-strain magnitude
ctBrickNormalPressure	Normal pressure
ctBrickGlobalPressureMagnitude	Global pressure magnitude
ctBrickGlobalPressureX	Pressure in the global X direction
ctBrickGlobalPressureY	Pressure in the global Y direction
ctBrickGlobalPressureZ	Pressure in the global Z direction
ctBrickShearX	Face shear in the local brick face x direction
ctBrickShearY	Face shear in the local brick face y direction
ctBrickShearMagnitude	Face shear magnitude
ctBrickNSMass	Non-structural mass
ctBrickDynamicFactor	Non-structural mass dynamic factor
ctBrickConvectionCoeff	Convection coefficient
ctBrickConvectionAmbient	Convection ambient temperature
ctBrickRadiationCoeff	Radiation coefficient
ctBrickRadiationAmbient	Radiation ambient temperature
ctBrickHeatFlux	Heat flux
ctBrickHeatSource	Heat source
ctBrickSoilStressSV	Soil stress Sv
ctBrickSoilStressK0	Soil stress K0
ctBrickSoilStressSH	Soil stress Sh

ctBrickSoilRatioOCR	Soil OCR (over-consolidation ratio)
ctBrickSoilRatioE0	Soil ratio e0
ctBrickSoilFluidLevel	Soil fluid level
ctBrickAgeAtFirstLoading	Age at first loading
ctBrickPropertyName	Property name
ctBrickMaterialName	Material name

Result Display Options

It is possible to set the result displays found in **Results Settings** for a Strand7 API model window when a result file is open. The display options found in Results Settings can be configured with the functions *St7SetBeamResultDisplay*, *St7SetPlateResultDisplay*, *St7SetBrickResultDisplay* and *St7SetLinkResultDisplay*. These functions all pass an Integers array, whose contents are detailed below – note that the bold type indicates the **Results Settings** combo box name seen in the GUI.

Result type – Integers[ipResultType]

SHOW AS:

rtAsNone – No result display.

rtAsContour – Display result as element contour.

rtAsDiagram – Display result as element diagram.

rtAsVector – Display result as element vector.

Result quantity – Integers[ipResultQuantity]

QUANTITY:

For beams, plates, bricks and links:

rqDispC – Displacement results.

rqInfluenceC – Influence results.

rqVelC – Velocity results.

rqAccC – Acceleration results.

rqPhaseC – Phase results.

rqReactC – Reaction results.

rqNodeForceC – Element node force results.

rqNodeInertiaC – Node inertia results.

rqTempC – Temperature results.

rqNodeFluxC – Flux results.

For beams:

rqBeamForceC – Beam force results.

rqBeamStrainC – Beam deformational strain results.

rqBeamTotalStrainC – Beam total strain results.

rqBeamStressC – Beam stress results.

rqBeamCreepStrainC – Beam creep strain results.

rqBeamEnergyC – Beam energy results.

rqBeamFluxC – Beam heat flux results.

rqBeamTGradC – Beam temperature gradient results.

rqBeamUserC – Beam user defined equation results.

For plates:

rqPlateForceC – Plate force results.

rqPlateMomentC – Plate moment results.

rqPlateStressC – Plate stress results.

rqPlateStrainC – Plate deformational strain results.

rqPlateTotalStrainC – Plate total strain results.

rqPlateCurvatureC – Plate deformational curvature results.

rqPlateTotalCurvatureC – Plate total curvature results.

rqPlateCreepStrainC – Plate creep strain results.

rqPlateEnergyC – Plate energy results.

rqPlateFluxC – Plate heat flux results.

rqPlateTGradC – Plate temperature gradient results.

rqPlateRCDesignC – Plate concrete reinforcement results.

rqPlatePlyStressC – Plate ply stress results.

rqPlatePlyStrainC – Plate ply strain results.

rqPlatePlyReserveC – Plate ply reserve factor results.

rqPlateSoilC – Plate soil results.

rqPlateUserC – Plate user defined equation results.

For bricks:

rqBrickStressC – Brick stress results.

rqBrickStrainC – Brick deformational strain results.

rqBrickTotalStrainC – Brick total strain results.

rqBrickCreepStrainC – Brick creep strain results.

rqBrickEnergyC – Brick energy results.

rqBrickFluxC – Brick heat flux results.

rqBrickTGradC – Brick temperature gradient results.

rqBrickSoilC – Brick soil results.

rqBrickUserC – Brick user defined equation results.

For links:

rqLinkForceC – Link force results.

rqLinkFluxC – Link flux results.

rqLinkMPLReactionC – Reaction MPL summation results.

Result axis – Integers[ipResultSystem]

SYSTEM:

stBeamLocal – use beam local axes.

stBeamPrincipal – use beam principal axes.

stBeamGlobal – use global axes.

stPlateLocal – use plate local axes.

stPlateGlobal – use global axes.

stPlateCombined – use plate combined results.

stBrickLocal – use brick local axes.

stBrickGlobal – use global axes.

stBrickCombined – use brick combined results.

stLinkGlobal – use global axes.

Alternatively, you can use the UCS ID in which frame the result is to be defined.
Similar to the GUI behaviour, UCS ID must agree with that set by *St7SetWindowUCSCase*, for expected results. Note that the UCS IDs in a model begin at an index of 2.

In the special case of presenting the steel requirement of reinforced concrete plate results, the following constants are used:

rsAreaPerLength.

rsBarSpacing.

rsBarDiameter.

rsAreaPerAreaSlab.

rsAreaPerAreaBase.

In the special case of presenting the laminate plate results; one of the following constants can be used:

Layer input	**Result output**
Ply number	One specified ply.
rsPlyMinValue	Minimum of all plies.
rsPlyMaxValue	Maximum of all plies.
rsPlyMaxMag	Maximum magnitude of all plies.

rsPlyMinValueActivePlies	Minimum of plies enabled by *True* to perform the angle check; False to skip the angle check. St7EnablePlyPropertyResults.
rsPlyMaxValueActivePlies	Maximum of plies enabled by *True* to perform the angle check; False to skip the angle check. St7EnablePlyPropertyResults.
rsPlyMaxMagActivePlies	Maximum magnitude of plies enabled by *True* to perform the angle check; False to skip the angle check. St7EnablePlyPropertyResults.

Result surface – Integers[ipResultSurface]

For plates, the **SURFACE** for general results:

psPlateMidPlane – contour the mid-plane.

psPlateMinusZ – contour the value occurring on the local -z side of the plate.

psPlatePlusZ – contour the value occurring on the local +z side of the plate.

For reinforced concrete plate results, the **SURFACE** is the reinforcement layer (1, 2, 3 or 4).

Result components – Integers[ipResultComponent]

COMPONENT:

The position index of the desired result component in the combo box labelled **COMPONENT:** in the **Results Settings**. Indices start at 1 at the top of the combo list, and increment downwards.

Note that this setting is in reference to the GUI, and its behaviour depends on Integers[ipResultType], Integers[ipResultQuantity] and Integers[ipResultSystem]. For example, it would be ignored when Integers[ipResultType] = rtAsDiagram or Integers[ipResultType] = rtAsVector.

When reinforced concrete plate results are presented, the following constants may be used: rcWoodArmerMoment, rcWoodArmerForce, rcSteelRequirementMin, rcConcreteStrain, rcSteelRequirementLessBase, rcUserSteelStress, rcUserConcreteStrain or rcBlockRatio.

Result vector components – Integers[ipVectorStyle]

Where Integers[ipResultType] = rtAsVector with appropriate Integers[ipResultQuantity] and Integers[ipResultSystem] settings, the vector style is set by **COMPONENT:**

vtVectorTranslationMag – the translation vector is drawn as a single arrow in the appropriate direction.

vtVectorTranslationComponents – the translation vector is drawn as three component arrows.

vtVectorRotationMag – the rotation vector is drawn as a single arrow in the appropriate direction.

vtVectorRotationComponents – the rotation vector is drawn as three component arrows.

For vtVectorTranslationComponents and vtVectorRotationComponents above, the flags labelled by **AXES:** are manipulated by the following indices into Integers.

Integers[ipVector1] – display the first vector translational component, btTrue or btFalse.

Integers[ipVector2] – display the second vector translational component, btTrue or btFalse.

Integers[ipVector3] – display the third vector translational component, btTrue or btFalse.

Integers[ipVector4] – display the first vector rotational component, btTrue or btFalse.

Integers[ipVector5] – display the second vector rotational component, btTrue or btFalse.

Integers[ipVector6] – display the third vector rotational component, btTrue or btFalse.

Diagram Flags – Integers[ipDiagram1..ipDiagram6]

Where Integers[ipResultType] = rtAsDiagram for beam elements, the flags labelled **COMPONENT:** are manipulated by the following indices into Integers.

Integers[ipDiagram1] – shear force in **Plane 1**/**Plane x**; either btTrue or btFalse.

Integers[ipDiagram2] – bending moment in **Plane 1**/**Plane x**; either btTrue or btFalse.

Integers[ipDiagram3] – shear force in **Plane 2**/**Plane y**; either btTrue or btFalse.

Integers[ipDiagram4] – bending moment in **Plane 2**/**Plane y**; either btTrue or btFalse.

Integers[ipDiagram5] – axial force; either btTrue or btFalse.

Integers[ipDiagram6] – torque, btTrue or btFalse.

Custom Results

User defined results files (custom result files) can be created directly using the Strand7 API. Once created, these results can be opened and manipulated in the same way as normal Strand7 result files.

The basic workflow for creating a custom result file involves the following steps.

St7NewResFile – to create a new, empty custom result file.

St7SetResFileNumCases – to set up a number of result cases to store results in. Optionally they can also be given names using *St7SetResFileCaseName*.

St7SetResFileQuantity – to set result file quantity types to be stored in the custom result file.

St7SetResFileBeamStations – to set the number of beam stations, should the model include beam results. This function should be called once per result case in the custom result file, not once per beam element. The set number of stations applies to all beam elements in the model for the specified result case.

St7SetResFileNodeResult/*St7SetResFileBeamResult*/*St7SetResFileBeamReleaseResult*/*St7SetResFilePlateResult*/*St7SetResFilePlatePressureResult*/*St7SetResFileBrickResult* – to write elemental results.

St7CloseResFile – to close the custom result file.

St7NewResFile will initialise the units of the result file with a set of units based on the length unit of the model file with which the result file is associated. The result file units are set as follows:

Model Length Unit	**Units automatically assigned to custom result file**		
	Force	**Mass**	**Stress**
m	N	kg	Pa
cm	kgf	kg	kgf/cm^2
mm	N	t	N/mm^2 (MPa)
ft	lbf	sl	lb/ft^2 (psf)
in	lbf	lb	lbf/in^2 (psi)

The other two base units, temperature and energy, are independent of all other units; the custom result file will be assigned the same temperature and energy units as defined in the associated model file. *St7GetResFileUnits* can be used to retrieve the units in the result file. Irrespective of the units assigned to the custom result file, result quantities will always be available in any units system (i.e., in the units of the model file that opens the result file).

The following basic quantities may be stored in custom result files. They are manipulated using the functions *St7SetResFileQuantity*, *St7ClearResFileQuantity*, *St7GetResFileQuantityState*.

Node Results

rtNodeDisp, rtNodeVel, rtNodeAcc, rtNodeReact, rtNodeTemp or rtNodeFlux.

Beam Results

rtBeamForce, rtBeamAllStrain, rtBeamNodeReact, rtBeamExtraResults or rtBeamFlux.

Plate Results

rtPlateStress, rtPlateStrain, rtPlateNodeReact or rtPlateFlux.

Brick Results

rtBrickStress, rtBrickStrain, rtBrickNodeReact or rtBrickFlux.

Node Results

The functions *St7SetResFileNodeResult*/*St7GetResFileNodeResult* pass an array Doubles of nodal results relevant to the nominated result Quantity.

Quantity: [rtNodeDisp/rtNodeVel/rtNodeAcc/rtNodeReact]

These quantities refer to vector results of nodal displacement, velocity, acceleration, and reaction, respectively. The following indices into the array Doubles specify translational and rotational results:

ipNodeResFileDX – in the global X direction.

ipNodeResFileDY – in the global Y direction.

ipNodeResFileDZ – in the global Z direction.

ipNodeResFileRX – about the global X axis.

ipNodeResFileRY – about the global Y axis.

ipNodeResFileRZ – about the global Z axis.

The units of the result data must be consistent with the length unit of the result file based on the following table.

Quantity	**Type**	**Result File Length Unit**				
		m	**cm**	**mm**	**ft**	**in**
rtNodeDisp	Translation	m	cm	mm	ft	in
rtNodeDisp	Rotation	rad	rad	rad	rad	rad
rtNodeVel	Translation	m/s	cm/s	mm/s	ft/s	in/s
rtNodeVel	Rotation	rad/s	rad/s	rad/s	rad/s	rad/s
rtNodeAcc	Translation	m/s^2	cm/s^2	mm/s^2	ft/s^2	in/s^2
rtNodeAcc	Rotation	rad/s^2	rad/s^2	rad/s^2	rad/s^2	rad/s^2
rtNodeReact	Translation	N	kgf	N	lbf	lbf
rtNodeReact	Rotation	N.m	kgf.cm	N.mm	lbf.ft	lbf.in

Quantity: [rtNodeTemp/rtNodeFlux]

These quantities refer to scalar quantities of nodal temperature and heat flux, respectively. The following index into the array Doubles specifies both results:

ipNodeResTempFlux – Temperature/flux result.

The units of the result data must be consistent with the temperature and energy units of the result file based on the following table.

Quantity	Result File Units
	Temperature, Energy
rtNodeTemp	temperature
rtNodeFlux	energy/s

Beam Results

The functions *St7SetResFileBeamResult*/*St7GetResFileBeamResult* pass an array Doubles of nodal results relevant to the nominated result Quantity. For station-dependent results such as rtBeamForce and rtBeamAllStrain, the number of beam stations should be set for the result file using *St7SetResFileBeamStations*. End release results are handled by the dedicated functions *St7SetResFileBeamReleaseResult*/*St7GetResFileBeamReleaseResult*.

Quantity: rtBeamForce

For each beam station, a block of results kBeamResFileForceSize long is stored in Doubles. The following constants index the results in each block:

ipBeamResFileSF1 – Shear force in the principal 1 axis direction.

ipBeamResFileSF2 – Shear force in the principal 2 axis direction.

ipBeamResFileAxial – Axial force.

ipBeamResFileBM1 – Bending moment in the principal 1 axis direction.

ipBeamResFileBM2 – Bending moment in the principal 2 axis direction.

ipBeamResFileTorque – Torque.

For example, the axial force at the i^{th} beam station is stored at:

```
Doubles[(i-1)*kBeamResFileForceSize + ipBeamResFileAxial].
```

The units of the result data must be consistent with the length units of the result file based on the following table.

rtBeamForce	Result File Length Unit				
	m	cm	mm	ft	in
Force	N	kgf	N	lbf	lbf
Moment	N.m	kgf.cm	N.mm	lbf.ft	lbf.in

Quantity: rtBeamAllStrain

For each beam station, a block of results kBeamResFileStrainSize long is stored in Doubles. The following constants index the results in each block:

ipBeamResFileAxialStrain – Axial strain.

ipBeamResFileCurvature1 – Curvature in the principal 1 axis direction.

ipBeamResFileCurvature2 – Curvature in the principal 2 axis direction.

ipBeamResFileTwist – Twisting strain.

The units of the result data must be consistent with the length units of the result file based on the following table.

rtBeamAllStrain	**Result File Length Unit**				
	m	**cm**	**mm**	**ft**	**in**
Axial Strain	none	none	none	none	none
Curvature	m^{-1}	cm^{-1}	mm^{-1}	ft^{-1}	in^{-1}
Twist	m^{-1}	cm^{-1}	mm^{-1}	ft^{-1}	in^{-1}

Quantity: rtBeamNodeReact

For each beam end, a block of results kBeamResFileReactSize long is stored in Doubles. The following constants index the results in each block:

ipBeamResFileFX – Force reaction in the X axis direction.

ipBeamResFileFY – Force reaction in the Y axis direction.

ipBeamResFileFZ – Force reaction in the Z axis direction.

ipBeamResFileMX – Moment reaction in the X axis direction.

ipBeamResFileMY – Moment reaction in the Y axis direction.

ipBeamResFileMZ – Moment reaction in the Z axis direction.

The units of the result data must be consistent with the length units of the result file based on the following table.

rtBeamNodeReact	**Result File Length Unit**				
	m	**cm**	**mm**	**ft**	**in**
Force	N	kgf	N	lbf	lbf
Moment	N.m	kgf.cm	N.mm	lbf.ft	lbf.in

Quantity: Beam End Release

Beam end release results are assigned using *St7SetResFileBeamReleaseResult*. The following constants index the results in the array:

`ipRelEnd1Dir1` – End 1 translational release in the 1 axis direction.

`ipRelEnd1Dir2` – End 1 translational release in the 2 axis direction.

`ipRelEnd1Dir3` – End 1 translational release in the 3 axis direction.

`ipRelEnd1Dir4` – End 1 rotational release about the 1 axis direction.

`ipRelEnd1Dir5` – End 1 rotational release about the 2 axis direction.

`ipRelEnd1Dir6` – End 1 rotational release about the 3 axis direction.

`ipRelEnd2Dir1` – End 2 translational release in the 1 axis direction.

`ipRelEnd2Dir2` – End 2 translational release in the 2 axis direction.

`ipRelEnd2Dir3` – End 2 translational release in the 3 axis direction.

`ipRelEnd2Dir4` – End 2 rotational release about the 1 axis direction.

`ipRelEnd2Dir5` – End 2 rotational release about the 2 axis direction.

`ipRelEnd2Dir6` – End 2 rotational release about the 3 axis direction.

The units of the result data must be consistent with the length unit of the result file based on the following table.

Release Direction	Result File Length Unit				
	m	**cm**	**mm**	**ft**	**in**
Dir1, Dir2, Dir3	m	cm	mm	ft	in
Dir4, Dir5, Dir6	rad	rad	rad	rad	rad

Quantity: rtBeamFlux

For each beam end, a block of results kBeamResFileFluxSize long is stored in Doubles. The following constants index the results in each block:

ipBeamResFileF – Beam heat flux at the end.

ipBeamResFileG – Beam temperature gradient at the end.

The units of the result data must be consistent with the energy, length and temperature units of the result file based on the following table.

rtBeamFlux	Result File Units
	Energy, Length, Temperature
Flux	energy/s/length^2
Temperature Gradient	temperature/length

Quantity: rtBeamNodeFlux

For each beam end, one flux result is stored in Doubles.

The units of the result data must be consistent with the energy units of the result file based on the following table.

rtBeamNodeFlux	Result File Units
	Energy
Flux	energy/s

Plate Results

The functions *St7SetResFilePlateResult*/*St7GetResFilePlateResult* pass an array Doubles containing Gauss point or node results depending on the nominated result Quantity. When the results are intended to correspond to a material nonlinear analysis, a flag NonlinearMaterial is set to btTrue to indicate that extra indices in Doubles are used to define the nonlinear stress state, for plate/shell type elements.

Result blocks

The number of Gauss point or node result blocks expected by *St7SetResFilePlateResult* depends on the number of nodes on the element, according to the following table. This is independent of solution, material or element type and may differ compared with the raw results produced directly by the solvers.

Plate nodes	Plate type	Number of Gauss point related result blocks required	Number of node related result blocks required
3	Tri3	3	3
6	Tri6	3	6
4	Quad4	4	4
8	Quad8	4	8
9	Quad9	4	9

Quantity: rtPlateStress resultants for plate/shell type elements

For each Gauss point, a block of results kPlateShellResFileStressSize long is stored in Doubles. The following constants index the results in each block.

ipPlateShellResFileNxx – Membrane force per unit width in the local x axis direction.

ipPlateShellResFileNyy – Membrane force per unit width in the local y axis direction.

ipPlateShellResFileNxy – Membrane shear force per unit width in the local xy axis direction.

ipPlateShellResFileMxx – Moment per unit width in the local x axis direction.

ipPlateShellResFileMyy – Moment per unit width in the local y axis direction.

ipPlateShellResFileMxy – Twisting moment per unit width in the local xy axis direction.

ipPlateShellResFileQxz – Transverse shear force per unit width in the local xz axis direction.

ipPlateShellResFileQyz – Transverse shear force per unit width in the local yz axis direction.

Linear plate stress results are then derived from the plate internal force results and the plate thickness. The plate thickness is taken from the model file associated with the custom result file.

The units of the result data must be consistent with the length unit of the result file based on the following table.

rtPlateStress	**Result File Length Unit**				
	m	**cm**	**mm**	**ft**	**in**
Force	N/m	kgf/cm	N/mm	lbf/ft	lbf/in
Moment	N.m/m	kgf.cm/cm	N.mm/mm	lbf.ft./ft	lbf.in/in
Transverse Shear Force	N/m	kgf/cm	N/mm	lbf/ft	lbf/in

Quantity: rtPlateStress for plate/shell type elements with NonlinearMaterial = btTrue

Nonlinear plate stress results additionally set data at the following indices in each block.

ipPlateShellResFileSxxMinusZ – Direct stress in the local x axis direction, at the -z surface.

ipPlateShellResFileSyyMinusZ – Direct stress in the local y axis direction, at the -z surface.

ipPlateShellResFileSxyMinusZ – Shear stress in the local xy axis direction, at the -z surface.

ipPlateShellResFileSxxMidPlane – Direct stress in the local x axis direction, at the mid-plane surface.

ipPlateShellResFileSyyMidPlane – Direct stress in the local y axis direction, at the mid-plane surface.

ipPlateShellResFileSxyMidPlane – Shear stress in the local xy axis direction, at the mid-plane surface.

ipPlateShellResFileSxxPlusZ – Direct stress in the local x axis direction, at the +z surface.

ipPlateShellResFileSyyPlusZ – Direct stress in the local y axis direction, at the +z surface.

ipPlateShellResFileSxyPlusZ – Shear stress in the local xy axis direction, at the +z surface.

The units of the result data must be consistent with the length unit of the result file based on the following table.

rtPlateStress	Result File Length Unit				
	m	cm	mm	ft	in
Stress	N/m^2	kgf/cm^2	N/mm^2	lbf/ft^2	lbf/in^2

Quantity: rtPlateStrain for plate/shell type elements

For each Gauss point, a block of results kPlateShellResFileStrainSize long is stored in Doubles. The following constants index the results in each block.

ipPlateShellResFileExx – In-plane strain in the local x axis direction.

ipPlateShellResFileEyy – In-plane strain in the local y axis direction.

ipPlateShellResFileExy – In-plane shear strain in the local xy axis direction.

ipPlateShellResFileEzz – Strain in the local z axis direction.

ipPlateShellResFileKxx – Curvature in the local x axis direction.

ipPlateShellResFileKyy – Curvature in the local y axis direction.

ipPlateShellResFileKxy – Twist in the local xy axis direction.

ipPlateShellResFileTxz – Transverse strain in the local zx axis direction.

ipPlateShellResFileTyz – Transverse strain in the local yz axis direction.

ipPlateShellResFileStoredE – Stored elastic strain energy density.

ipPlateShellResFileSpentE – Irreversible work performed, as an energy density.

Strain results on the surfaces of the plate are derived from the strain and curvature results together with the plate thickness. The plate thickness is taken from the model file associated with the custom result file.

The units of the result data must be consistent with the length unit of the result file based on the following table.

rtPlateStrain	Result File Length Unit				
	m	cm	mm	ft	in
Strain	none	none	none	none	none
Curvature	m^{-1}	cm^{-1}	mm^{-1}	ft^{-1}	in^{-1}
Transverse Shear Strain	none	none	none	none	none
Energy Density	$N.m/m^2$	$kgf.cm/cm^2$	$N.mm/mm^2$	$lbf.ft/ft^2$	$lbf.in/in^2$

Quantity: rtPlateStress for 2D plane stress/strain type elements

For each Gauss point, a block of results kPlate2DResFileStressSize long is stored in Doubles. The following constants index the results in each block.

ipPlate2DResFileSXX – Direct stress in the global X axis direction.

ipPlate2DResFileSYY – Direct stress in the global Y axis direction.

ipPlate2DResFileSXY – Direct shear stress in the global XY axis direction.

ipPlate2DResFileSZZ – Direct stress in the global Z axis direction.

The units of the result data must be consistent with the length unit of the result file based on the following table.

rtPlateStress	**Result File Length Unit**				
	m	**cm**	**mm**	**ft**	**in**
Stress	N/m^2	kgf/cm^2	N/mm^2	lbf/ft^2	lbf/in^2

Quantity: rtPlateStrain for 2D plane stress/strain type elements

For each Gauss point, a block of results kPlate2DResFileStrainSize long is stored in Doubles. The following constants index the results in each block.

ipPlate2DResFileEXX – Strain in the global X axis direction.

ipPlate2DResFileEYY – Strain in the global Y axis direction.

ipPlate2DResFileEXY – Shear strain in the global XY axis direction.

ipPlate2DResFileEZZ – Strain in the global Z axis direction.

ipPlate2DResFileStoredE – Stored elastic strain energy density.

ipPlate2DResFileSpentE – Irreversible work performed, as an energy density.

The units of the result data must be consistent with the length unit of the result file based on the following table.

rtPlateStrain	**Result File Length Unit**				
	m	**cm**	**mm**	**ft**	**in**
Strain	none	none	none	none	none
Energy Density	$N.m/m^2$	$kgf.cm/cm^2$	$N.mm/mm^2$	$lbf.ft/ft^2$	$lbf.in/in^2$

Quantity: rtPlateStress for 2D axisymmetric type elements

For each Gauss point, a block of results kPlateAxiResFileStressSize long is stored in Doubles. The following constants index the results in each block.

ipPlateAxiResFileSRR – Direct stress in the axisymmetric R axis direction.

ipPlateAxiResFileSZZ – Direct stress in the axisymmetric Z axis direction.

ipPlateAxiResFileSTT – Direct stress in the axisymmetric T axis direction.

ipPlateAxiResFileSRZ – Shear stress in the axisymmetric RZ axis direction.

The units of the result data must be consistent with the length unit of the result file based on the following table.

rtPlateStress	Result File Length Unit				
	m	cm	mm	ft	in
Stress	N/m^2	kgf/cm^2	N/mm^2	lbf/ft^2	lbf/in^2

Quantity: rtPlateStrain for 2D axisymmetric type elements

For each Gauss point, a block of results kPlateAxiResFileStrainSize long is stored in Doubles. The following constants index the results in each block.

ipPlateAxiResFileERR – Strain in the axisymmetric R axis direction.

ipPlateAxiResFileEZZ – Strain in the axisymmetric Z axis direction.

ipPlateAxiResFileETT – Strain in the axisymmetric T axis direction.

ipPlateAxiResFileERZ – Shear strain in the axisymmetric RZ axis direction.

ipPlateAxiResFileStoredE – Stored elastic strain energy density.

ipPlateAxiResFileSpentE – Irreversible work performed, as an energy density.

The units of the result data must be consistent with the length unit of the result file based on the following table.

rtPlateStrain	Result File Length Unit				
	m	cm	mm	ft	in
Strain	none	none	none	none	none
Energy Density	$N.m/m^3$	$kgf.cm/cm^3$	$N.mm/mm^3$	$lbf.ft/ft^3$	$lbf.in/in^3$

Quantity: rtPlateStress for 3D membrane type elements

For each Gauss point, a block of results kPlateMembraneResFileStressSize long is stored in Doubles. The following constants index the results in each block.

ipPlateMembraneResFileSXX – Direct stress in the local x direction.

ipPlateMembraneResFileSYY – Direct stress in the local y direction.

ipPlateMembraneResFileSXY – Shear stress in the local xy direction.

The units of the result data must be consistent with the length unit of the result file based on the following table.

rtPlateStress	Result File Length Unit				
	m	cm	mm	ft	in
Stress	N/m^2	kgf/cm^2	N/mm^2	lbf/ft^2	lbf/in^2

Quantity: rtPlateStrain for 3D membrane type elements

For each Gauss point, a block of results kPlateMembraneResFileStrainSize long is stored in Doubles. The following constants index the results in each block.

ipPlateMembraneResFileExx – Strain in the local x direction.

ipPlateMembraneResFileEyy – Strain in the local y axis direction.

ipPlateMembraneResFileExy – Shear strain in the local xy axis direction.

ipPlateMembraneResFileEzz – Strain in the local z axis direction.

ipPlateMembraneResFileStoredE – Stored elastic strain energy density.

ipPlateMembraneResFileSpentE – Irreversible work performed, as an energy density.

The units of the result data must be consistent with the length unit of the result file based on the following table.

rtPlateStrain	**Result File Length Unit**				
	m	**cm**	**mm**	**ft**	**in**
Strain	none	none	none	none	none
Energy Density	$N.m/m^2$	$kgf.cm/cm^2$	$N.mm/mm^2$	$lbf.ft/ft^2$	$lbf.in/in^2$

Quantity: rtPlateStress resultants for shear panel type elements

For each Gauss point, a block of results kPlateShearPanelResFileStressSize long is stored in Doubles. The following constants index the result in each block.

ipPlateShearPanelResFileNxy – Membrane shear force per unit width in the local xy direction.

The units of the result data must be consistent with the length unit of the result file based on the following table.

rtPlateStress	**Result File Length Unit**				
	m	**cm**	**mm**	**ft**	**in**
Shear Force	N/m	kgf/cm	N/mm	lbf/ft	lbf/in

Quantity: rtPlateStrain for shear panel type elements

For each Gauss point, a block of results kPlateShearPanelResFileStrainSize long is stored in Doubles. The following constants index the results in each block.

ipPlateShearPanelResFileExy – Shear strain in the local xy direction.

ipPlateShearPanelResFileStoredE – Stored elastic strain energy density.

ipPlateShearPanelResFileSpentE – Irreversible work performed, as an energy density.

The units of the result data must be consistent with the length unit of the result file based on the following table.

rtPlateStrain	Result File Length Unit				
	m	cm	mm	ft	in
Shear Strain	none	none	none	none	none
Energy Density	N.m/m^2	kgf.cm/cm^2	N.mm/mm^2	lbf.ft/ft^2	lbf.in/in^2

Quantity: rtPlateNodeReact

For each node, a block of results kPlateResFileReactSize long is stored in Doubles. The following constants index the results in each block.

ipPlateResFileFX – Reaction force in the global X axis direction.

ipPlateResFileFY – Reaction force in the global Y axis direction.

ipPlateResFileFZ – Reaction force in the global Z axis direction.

ipPlateResFileMX – Reaction moment about the global X axis direction.

ipPlateResFileMY – Reaction moment about the global Y axis direction.

ipPlateResFileMZ – Reaction moment about the global Z axis direction.

The units of the result data must be consistent with the length units of the result file based on the following table.

rtPlateNodeReact	Result File Length Unit				
	m	cm	mm	ft	in
Force	N	kgf	N	lbf	lbf
Moment	N.m	kgf.cm	N.mm	lbf.ft	lbf.in

Quantity: Plate Normal Pressure

Plate normal pressure applied in the analysis should be assigned to surfaces of a plate/shell element by using *St7SetResFilePlatePressureResult*. This information is required to calculate normal stress on the plate surface. The units of the pressure must be consistent with the length units of the result file based on the following table.

Plate Pressure	Result File Length Unit				
	m	cm	mm	ft	in
Pressure	N/m^2	kgf/cm^2	N/mm^2	lbf/ft^2	lbf/in^2

Quantity: rtPlateFlux

For each Gauss point, a block of results kPlateResFileFluxSize long is stored in Doubles. The following constants index the results in each block.

ipPlateResFileFxx – Heat flux in the local x axis direction.

ipPlateResFileFyy – Heat flux in the local y axis direction.

ipPlateResFileGxx – Temperature gradient in the local x axis direction.

ipPlateResFileGyy – Temperature gradient in the local y axis direction.

The units of the result data must be consistent with the energy, length and temperature units of the result file based on the following table.

rtPlateFlux	**Result File Units**
	Energy, Length, Temperature
Flux	energy/s/length^2
Temperature Gradient	temperature/length

Quantity: rtPlateNodeFlux

For each node, one flux result is stored in Doubles.

The units of the result data must be consistent with the energy units of the result file based on the following table.

rtPlateNodeFlux	**Result File Units**
	Energy
Flux	energy/s

Brick Results

The functions *St7SetResFileBrickResult*/*St7GetResFileBrickResult* pass an array Doubles of nodal results relevant to the nominated result Quantity.

Result blocks

The number of Gauss point or node result blocks expected by *St7SetResFileBrickResult* depends on the number of nodes on the element, according to the following table. This is independent of solution, material or element type and may differ compared with the raw results produced directly by the solvers.

Brick nodes	**Brick type**	**Number of Gauss point related result blocks required**	**Number of node related result blocks required**
4	Tetra4	1	4
10	Tetra10	4	10
5	Pyra5	8	5
13	Pyra13	8	13

6	Wedge6	6	6
15	Wedge15	6	15
8	Hexa8	8	8
16	Hexa16	8	16
20	Hexa20	8	20

Quantity: rtBrickStress

For each Gauss point, a block of results kBrickResFileStressSize long is stored in Doubles. The following constants index the results in each block.

ipBrickResFileSXX – Direct stress in the global x axis direction.

ipBrickResFileSYY – Direct stress in the global y axis direction.

ipBrickResFileSZZ – Direct stress in the global z axis direction.

ipBrickResFileSXY – Shear stress in the global xy axis direction.

ipBrickResFileSYZ – Shear stress in the global yz axis direction.

ipBrickResFileSZX – Shear stress in the global zx axis direction.

The units of the result data must be consistent with the length unit of the result file based on the following table.

rtBrickStress	**Result File Length Unit**				
	m	**cm**	**mm**	**ft**	**in**
Stress	N/m^2	kgf/cm^2	N/mm^2	lbf/ft^2	lbf/in^2

Quantity: rtBrickStrain

For each Gauss point, a block of results kBrickResFileStrainSize long is stored in Doubles. The following constants index the results in each block.

ipBrickResFileExx – Strain in the global x axis direction.

ipBrickResFileEyy – Strain in the global y axis direction.

ipBrickResFileEzz – Strain in the global z axis direction.

ipBrickResFileExy – Shear strain in the global xy axis direction.

ipBrickResFileEyz – Shear strain in the global yz axis direction.

ipBrickResFileEzx – Shear strain in the global zx axis direction.

ipBrickResFileStoredE – Stored elastic strain energy density.

ipBrickResFileSpentE – Irreversible work performed, as an energy density.

The units of the result data must be consistent with the length unit of the result file based on the following table.

rtBrickStrain	Result File Length Unit				
	m	cm	mm	ft	in
Strain	none	none	none	none	none
Energy Density	N.m/m^3	kgf.cm/cm^3	N.mm/mm^3	lbf.ft/ft^3	lbf.in/in^3

Quantity: rtBrickNodeReact

For each node, a block of results kBrickResFileReactSize long is stored in Doubles. The following constants index the results in each block.

ipBrickResFileFX – Reaction force in the global X axis direction.

ipBrickResFileFY – Reaction force in the global Y axis direction.

ipBrickResFileFZ – Reaction force in the global Z axis direction.

The units of the result data must be consistent with the length units of the result file based on the following table.

Quantity	Result File Length Unit				
	m	cm	mm	ft	in
rtBrickNodeReact	N	kgf	N	lbf	lbf

Quantity: rtBrickFlux

For each Gauss point, a block of results kBrickResFileFluxSize long is stored in Doubles. The following constants index the results in each block.

ipBrickResFileFXX – Heat flux in the global x axis direction.

ipBrickResFileFYY – Heat flux in the global y axis direction.

ipBrickResFileFZZ – Heat flux in the global z axis direction.

ipBrickResFileGXX – Temperature gradient in the global x axis direction.

ipBrickResFileGYY – Temperature gradient in the global y axis direction.

ipBrickResFileGZZ – Temperature gradient in the global z axis direction.

The units of the result data must be consistent with the energy, length and temperature units of the result file based on the following table.

rtBrickFlux	Result File Units
	Energy, Length, Temperature
Flux	energy/s/length^2
Temperature Gradient	temperature/length

Quantity: rtBrickNodeFlux

For each node, one flux result is stored in Doubles.

The units of the result data must be consistent with the energy units of the result file based on the following table.

rtBrickNodeFlux	Result File Units
	Energy
Flux	energy/s

Strand7 Function Index

Finite element analysis.

Strand7 Pty Limited

Suite 1, Level 5, 65 York Street
Sydney NSW 2000 Australia
Tel +61 2 9264 2977
info@strand7.com
www.strand7.com

Strand7®
FINITE ELEMENT ANALYSIS