libnecpp.h File Reference

nec++ Library Functions. More...

Go to the source code of this file.

Typedefs

typedef struct **nec_context** **nec_context**

Functions

Initialization and Cleanup

Functions dealing with antenna simulation contexts. The contexts should be created before a simulation begins and deleted after the simluation is complete to recover any memory allocated.

nec_context * **nec_create** (void)
Create an **nec_context** and initialize it. More...

long **nec_delete** (**nec_context** *in_context)
Delete an **nec_context** object. More...

Antenna Geometry

Functions for creating wires and surface patches, as well as geometry transformations

long **nec_wire** (**nec_context** *in_context, int tag_id, int segment_count, double xw1, double yw1, double zw1, double xw2, double yw2, double zw2, double rad, double rdel, double rrad)
Create a straigt wire. More...

long **nec_sp_card** (**nec_context** *in_context, int ns, double x1, double y1, double z1, double x2, double y2, double z2)
Surface Patch (SP Card) More...

long **nec_sc_card** (**nec_context** *in_context, int i2, double x3, double y3, double z3, double x4, double y4, double z4)
Surface Patch Continuation (SC Card) More...

long **nec_gm_card** (**nec_context** *in_context, int itsi, int nrpt, double rox, double roy, double roz, double xs, double ys, double zs, int its)
Coordinate Transformation. More...

long **nec_gx_card** (**nec_context** *in_context, int i1, int i2)
Reflection in Coordinate Planes. More...

long **nec_geometry_complete** (**nec_context** *in_context, int gpflag)
Indicate that the geometry is complete (GE card) More...

Error Handling

Functions for error handling and utility

long	**nec_benchmark** (void) Benchmark the libnecpp engine. A score of 1 is roughly an Athlon XP 1800. More...
const char *	**nec_error_message** (void) Get the last error message All functions return a long. If this is != 0. Then an error has occurred. The error message can be retrieved with this function.

Antenna Environment

Functions for specifying the ground and antenna excitation, frequency and loading.

long	**nec_medium_parameters** (**nec_context** *in_context, double permittivity, double permeability) Set the prameters of the medium (permittivity and permeability) More...
long	**nec_gn_card** (**nec_context** *in_context, int iperf, int nradl, double epse, double sig, double tmp3, double tmp4, double tmp5, double tmp6) Ground Card Examples: More...
long	**nec_fr_card** (**nec_context** *in_context, int in_ifrq, int in_nfrq, double in_freq_mhz, double in_del_freq) FR card. More...
long	**nec_ek_card** (**nec_context** *in_context, int itmp1) To control use of the extended thin-wire kernel approximation. More...
long	**nec_ld_card** (**nec_context** *in_context, int ldtyp, int ldtag, int ldtagf, int ldtagt, double tmp1, double tmp2, double tmp3) LD card (Loading) More...
long	**nec_ex_card** (**nec_context** *in_context, int extype, int i2, int i3, int i4, double tmp1, double tmp2, double tmp3, double tmp4, double tmp5, double tmp6) EX card (Excitation) More...
long	**nec_excitation_voltage** (**nec_context** *in_context, int tag, int **segment**, double v_real, double v_imag) Voltage Source Excitation. More...
long	**nec_excitation_current** (**nec_context** *in_context, double x, double y, double z, double a, double beta, double moment) Current Source Excitation. More...
long	**nec_excitation_planewave** (**nec_context** *in_context, int n_theta, int n_phi, double theta, double phi, double eta, double dtheta, double dphi, double pol_ratio) Planewave Excitation (Linear Polarization) More...
long	**nec_tl_card** (**nec_context** *in_context, int itmp1, int itmp2, int itmp3, int itmp4, double tmp1, double tmp2, double tmp3, double tmp4, double tmp5, double tmp6)

long **nec_nt_card** (**nec_context** *in_context, int itmp1, int itmp2, int itmp3, int itmp4, double tmp1, double tmp2, double tmp3, double tmp4, double tmp5, double tmp6)

long **nec_xq_card** (**nec_context** *in_context, int itmp1)
XQ Card (Execute) More...

long **nec_gd_card** (**nec_context** *in_context, double tmp1, double tmp2, double tmp3, double tmp4)

Simulation Output

Functions for calculating radiation patterns, and requesting printed output of simulation results.

long **nec_rp_card** (**nec_context** *in_context, int calc_mode, int n_theta, int n_phi, int output_format, int normalization, int D, int A, double theta0, double phi0, double delta_theta, double delta_phi, double radial_distance, double gain_norm)
Standard radiation pattern parameters. More...

long **nec_pt_card** (**nec_context** *in_context, int itmp1, int itmp2, int itmp3, int itmp4)
Print Flag (Printing of Currents. More...

long **nec_pq_card** (**nec_context** *in_context, int itmp1, int itmp2, int itmp3, int itmp4)

long **nec_kh_card** (**nec_context** *in_context, double tmp1)

long **nec_ne_card** (**nec_context** *in_context, int itmp1, int itmp2, int itmp3, int itmp4, double tmp1, double tmp2, double tmp3, double tmp4, double tmp5, double tmp6)

long **nec_nh_card** (**nec_context** *in_context, int itmp1, int itmp2, int itmp3, int itmp4, double tmp1, double tmp2, double tmp3, double tmp4, double tmp5, double tmp6)

long **nec_cp_card** (**nec_context** *in_context, int itmp1, int itmp2, int itmp3, int itmp4)

long **nec_pl_card** (**nec_context** *in_context, char *ploutput_filename, int itmp1, int itmp2, int itmp3, int itmp4)

Analysis of Output

Functions for calculating statistics from simulation outputs. These are useful for automatic optimization.

double **nec_gain** (**nec_context** *in_context, int freq_index, int theta_index, int phi_index)
Get the gain from a radiation pattern. More...

double **nec_gain_max** (**nec_context** *in_context, int freq_index)
Get the maximum gain from a radiation pattern. More...

double **nec_gain_min** (**nec_context** *in_context, int freq_index)
Get the minimum gain from a radiation pattern. More...

double **nec_gain_mean** (**nec_context** *in_context, int freq_index)
Get the mean gain from a radiation pattern. More...

double **nec_gain_sd** (**nec_context** *in_context, int freq_index)
Get the standard deviation of the gain from a radiation pattern. More...

double **nec_gain_rhcp_max** (**nec_context** *in_context, int freq_index)

double **nec_gain_rhcp_min** (**nec_context** *in_context, int freq_index)

double	**nec_gain_rhcp_mean** (**nec_context** *in_context, int freq_index)
double	**nec_gain_rhcp_sd** (**nec_context** *in_context, int freq_index)
double	**nec_gain_lhcp_max** (**nec_context** *in_context, int freq_index)
double	**nec_gain_lhcp_min** (**nec_context** *in_context, int freq_index)
double	**nec_gain_lhcp_mean** (**nec_context** *in_context, int freq_index)
double	**nec_gain_lhcp_sd** (**nec_context** *in_context, int freq_index)
double	**nec_impedance_real** (**nec_context** *in_context, int freq_index) Impedance: Real Part.
double	**nec_impedance_imag** (**nec_context** *in_context, int freq_index) Impedance: Imaginary Part.

Detailed Description

nec++ Library Functions.

How to use libNEC.

Have a look at **test_nec.c**

Function Documentation

long nec_benchmark (void)

Benchmark the libnecpp engine. A score of 1 is roughly an Athlon XP 1800.

Example:

Return values

err `0` indicates that the result is successful and `1` indicates that an error occurred. Call **nec_error_message()** for a detailed message.

References **nec_context::benchmark()**.

nec_context* nec_create (void)

Create an **nec_context** and initialize it.

Return values

context* An **nec_context** pointer.

Note: Do NOT delete or free the nec_context yourself, rather call nec_delete() to free memory associated with the nec simulation.

Examples:

test_nec.c.

References **nec_context::initialize()**.

long nec_delete (nec_context * in_context)

Delete an **nec_context** object.

Example:

Return values

err `0` indicates that the result is successful and `1` indicates that an error occurred. Call **nec_error_message()** for a detailed message.

Examples:

test_nec.c.

long nec_ek_card (nec_context * in_context,
int itmp1
)

To control use of the extended thin-wire kernel approximation.

Parameters

itmp1
- `-1` Return to normal kernel
- `0` Use Extended thin wire kernel

Example:

Return values

err `0` indicates that the result is successful and `1` indicates that an error occurred. Call **nec_error_message()** for a detailed message.

Examples:
test_nec.c.

References **nec_context::set_extended_thin_wire_kernel()**.

```
long nec_ex_card ( nec_context * in_context,
                   int           extype,
                   int           i2,
                   int           i3,
                   int           i4,
                   double        tmp1,
                   double        tmp2,
                   double        tmp3,
                   double        tmp4,
                   double        tmp5,
                   double        tmp6
                 )
```

EX card (Excitation)

Parameters

in_context The **nec_context** created with **nec_create()**

extype Type of excitation

- 0 - voltage source (applied-E-field source).
- 1 - incident plane wave, linear polarization.
- 2 - incident plane wave, right-hand (thumb along the incident k vector) elliptic polarization.
- 3 - incident plane wave, left-hand elliptic polarization.
- 4 - elementary current source.
- 5 - voltage source (current-slope-discontinuity).

i2 Tag number the source segment. This tag number along with the number to be given in (i3), which identifies the position of the segment in a set of equal tag numbers, uniquely definer the source segment.

- 0 - Blank or zero in field (i2) implies that the Source segment will be identified by using the absolute segment number in the next field (i3).

i3 Equal to m, specifies the mth segment of the set of segments whose tag numbers are equal to the number set by the previous parameter. If the previous parameter is zero, the number in (i3) must be the absolute segment number of the source.

i4 Meaning Depends on the extype parameter. See http://www.nec2.org/part_3/cards/ex.html

Example:

Return values

err `0` indicates that the result is successful and `1` indicates that an error occurred. Call **nec_error_message()** for a detailed message.

Remarks

Simpler versions of the function are provided for common uses. These are nec_voltage_excitation, nec_current_excitation and nec_planewave_excitation.

The meaning of the floating point parameter depends on the excitation type. See http://www.nec2.org/part_3/cards/ex.html for more details.

Examples:

test_nec.c.

References **nec_context::ex_card()**.

long nec_excitation_current (nec_context * in_context,
double x,
double y,
double z,
double a,
double beta,
double moment
)

Current Source Excitation.

Parameters

in_context	The **nec_context** created with **nec_create()**
x	- X position in meters.
y	- Y position in meters.
z	- Z position in meters.
a	- a in degrees. a is the angle the current source makes with the XY plane as illustrated on figure 15.
beta	- beta in degrees. beta is the angle the projection of the current source on the XY plane makes with the X axis.
moment	- "Current moment" of the source. This parameter is equal to the product Il in amp meters.

Example:

Return values

err `0` indicates that the result is successful and `1` indicates that an error occurred. Call **nec_error_message()** for a detailed message.

Remarks

Only one incident plane wave or one elementary current source is al- lowed at a time. Also plane-wave or current-source excitation is not allowed with voltage sources. If the excitation types are mixed, the program will use the last excitation type encountered.

References **nec_context::ex_card()**.

```
long nec_excitation_planewave ( nec_context * in_context,
                                int           n_theta,
                                int           n_phi,
                                double        theta,
                                double        phi,
                                double        eta,
                                double        dtheta,
                                double        dphi,
                                double        pol_ratio
                              )
```

Planewave Excitation (Linear Polarization)

Parameters

- **in_context** The **nec_context** created with **nec_create()**
- **n_theta** - Number of theta angles desired for the incident plane wave .
- **n_phi** - Number of phi angles desired for the incident plane wave.
- **theta** - Theta in degrees. Theta 19 defined in standard spherical coordinates as illustrated
- **phi** - Phi in degrees. Phi is the standard spherical angle defined lned in the XY plane.
- **eta** - Eta in degrees. Eta is the polarization angle defined as the angle between the theta unit vector and the direction of the electric field for linear polarization or the major ellipse axis for elliptical polarization.
- **dtheta** - Theta angle stepping increment in degrees.
- **dphi** - Phi angle stepping increment in degrees.
- **pol_ratio** - Ratio of minor axis to major axis for elliptic polarization (major axis field strength - 1 V/m).

Example:

Return values

- **err** `0` indicates that the result is successful and `1` indicates that an error occurred. Call **nec_error_message()** for a detailed message.

Remarks

Only one incident plane wave or one elementary current source is al- lowed at a time. Also plane-wave or current-source excitation is not allowed with voltage sources. If the excitation types are mixed, the program will use the last excitation type encountered.

References **nec_context::ex_card()**.

long nec_excitation_voltage (nec_context * in_context, int tag, int segment, double v_real, double v_imag)

Voltage Source Excitation.

Parameters

in_context	The **nec_context** created with **nec_create()**
tag	Tag number of the source segment. This tag number along with the number to be given in (segment), which identifies the position of the segment in a set of equal tag numbers, uniquely definer the source segment. • 0 - Blank or zero in field (tag) implies that the Source segment will be identified by using the absolute segment number in the next field (segment).
segment	Equal to m, specifies the mth segment of the set of segments whose tag numbers are equal to the number set by the previous parameter. If the previous parameter is zero, the number in (segment) must be the absolute segment number of the source.
v_real	real part of the voltage excitation (Volts)
v_imag	imaginary part of the voltage excitation (Volts)

Example:

Return values

err `0` indicates that the result is successful and `1` indicates that an error occurred. Call **nec_error_message()** for a detailed message.

Remarks

Only one incident plane wave or one elementary current source is al- lowed at a time. Also plane-wave or current-source excitation is not allowed with voltage sources. If the excitation types are mixed, the program will use the last excitation type encountered.

References **nec_context::ex_card()**.

```
long nec_fr_card ( nec_context *  in_context,
                   int            in_ifrq,
                   int            in_nfrq,
                   double         in_freq_mhz,
                   double         in_del_freq
                 )
```

FR card.

Parameters

in_context	The **nec_context** created with **nec_create()**
in_ifrq	0 is a linear range of frequencies, 1 is a log range.
in_nfrq	The number of frequencies
in_freq_mhz	The starting frequency in MHz.
in_del_freq	The frequency step (in MHz for ifrq = 0)

Example:

Return values

err `0` indicates that the result is successful and `1` indicates that an error occurred. Call **nec_error_message()** for a detailed message.

Examples:

test_nec.c.

References **nec_context::fr_card()**.

```
double nec_gain ( nec_context * in_context,
                  int           freq_index,
                  int           theta_index,
                  int           phi_index
                )
```

Get the gain from a radiation pattern.

Parameters

- **freq_index** The rp_card frequency index. If this parameter is 0, then the first simulation results are used. Subsequent simulations will store their results at higher indices.
- **theta_index** The theta index (starting at zero) of the radiation pattern
- **phi_index** The phi index (starting at zero) of the radiation pattern

Returns

The gain in dB or -999.0 if no radiation pattern had been previously requested.

Remarks

This function requires a previous **nec_rp_card()** method to have been called (with the gain normalization set to 5)

References **nec_context::get_gain()**.

```
double nec_gain_max ( nec_context * in_context,
                      int           freq_index
                    )
```

Get the maximum gain from a radiation pattern.

Parameters

- **freq_index** The rp_card frequency index. If this parameter is 0, then the first simulation results are used. Subsequent simulations will store their results at higher indices.

Returns

The maximum gain in dB or -999.0 if no radiation pattern had been previously requested.

Remarks

This function requires a previous **nec_rp_card()** method to have been called (with the gain normalization set to 5)

Examples:

test_nec.c.

double nec_gain_mean (nec_context * in_context,
int freq_index
)

Get the mean gain from a radiation pattern.

Parameters

freq_index The rp_card frequency index. If this parameter is 0, then the first simulation results are used. Subsequent simulations will store their results at higher indices.

Returns

The mean gain in dB or -999.0 if no radiation pattern had been previously requested.

Remarks

This function returns the mean over the sphere.

This function requires a previous **nec_rp_card()** method to have been called (with the gain normalization set to 5)

Examples:

test_nec.c.

double nec_gain_min (nec_context * in_context,
int freq_index
)

Get the minimum gain from a radiation pattern.

Parameters

freq_index The rp_card frequency index. If this parameter is 0, then the first simulation results are used. Subsequent simulations will store their results at higher indices.

Returns

The minimum gain in dB or -999.0 if no radiation pattern had been previously requested.

Remarks

This function requires a previous **nec_rp_card()** method to have been called (with the gain normalization set to 5)

double nec_gain_sd (nec_context * in_context,
int freq_index
)

Get the standard deviation of the gain from a radiation pattern.

Parameters

freq_index The rp_card frequency index. If this parameter is 0, then the first simulation results are used. Subsequent simulations will store their results at higher indices.

Returns

The standard deviation in dB or -999.0 if no radiation pattern had been previously requested.

Remarks

This function returns the standard deviation over the sphere.

This function requires a previous **nec_rp_card()** method to have been called (with the gain normalization set to 5)

Examples:

test_nec.c.

long nec_geometry_complete (nec_context * in_context,
int gpflag
)

Indicate that the geometry is complete (GE card)

Parameters

in_context The **nec_context** created with **nec_create()**

gpflag Geometry ground plain flag.

- `0` - no ground plane is present.
- `1` - Indicates a ground plane is present. Structure symmetry is modified as required, and the current expansion is modified so that the currents an segments touching the ground (x, Y plane) are interpolated to their images below the ground (charge at base is zero)
- `-1` - indicates a ground is present. Structure symmetry is modified as required. Current expansion, however, is not modified, Thus, currents on segments touching the ground will go to zero at the ground.

Example:

Return values

err `0` indicates that the result is successful and `1` indicates that an error occurred. Call **nec_error_message()** for a detailed message.

Examples:

test_nec.c.

References **nec_context::geometry_complete()**.

```
long nec_gm_card ( nec_context *  in_context,
                   int            itsi,
                   int            nrpt,
                   double         rox,
                   double         roy,
                   double         roz,
                   double         xs,
                   double         ys,
                   double         zs,
                   int            its
                 )
```

Coordinate Transformation.

Parameters

- **itsi** Tag number increment.
- **nprt** The number of new Structures to be generated
- **ROX** Angle in degrees through which the structure is rotated about the X-axis. A positive angle causes a right-hand rotation.
- **ROY** Angle of rotation about Y-axis.
- **ROZ** Angle of rotation about
- **XS** X, Y. Z components of vector by which
- **YS** structure is translated with respect to
- **ZS** the coordinate system.
- **ITS** This number is input as a decimal number but is rounded to an integer before use. Tag numbers are searched sequentially until a segment having a tag of this segment through the end of the sequence of segments is moved by the card. If ITS is zero the entire structure is moved.

Example:

Return values

err `0` indicates that the result is successful and `1` indicates that an error occurred. Call **nec_error_message()** for a detailed message.

```
long nec_gn_card ( nec_context * in_context,
                   int           iperf,
                   int           nradl,
                   double        epse,
                   double        sig,
                   double        tmp3,
                   double        tmp4,
                   double        tmp5,
                   double        tmp6
                 )
```

Ground Card Examples:

1) Infinite ground plane nec_gn_card(nec, 1, 0, 0, 0, 0, 0, 0, 0);

2) Radial Wire Ground Plane (4 wires, 2 meters long, 5mm in radius) nec_gn_card(nec, 4, 0, 0.0, 0.0, 2.0, 0.005, 0.0, 0.0)

Parameters

iperf Ground-type flag

- `-1` Nullifies ground parameters previously used and sets free-space condition. The remainder of the parameters are ignored in this case.
- `0` Finite ground, reflection coefficient approximation
- `1` Perfectly conducting ground.
- `2` Finite ground, Sommerfeld/Norton method.

nradl Number of radial wires in the ground screen approximation, O implies no ground screen.

epse Relative dielectric constant for ground in the vicinity of the antenna. Zero in the case of perfect ground.

sig Conductivity in mhos/meter of the ground in the vicinity of the antenna. Use zero in the case of a perfect ground. If SIG is input as a negative number, the complex dielectric constant Ec = Er -j sigma/omaga epslon is set to EPSR - |SIG|.

Example:

Return values

err `0` indicates that the result is successful and `1` indicates that an error occurred. Call **nec_error_message()** for a detailed message.

Examples:

test_nec.c.

References **nec_context::gn_card()**.

**long nec_gx_card (nec_context * in_context,
int i1,
int i2
)**

Reflection in Coordinate Planes.

Parameters

- **i1** - Tag number increment.
- **i2** - This integer is divided into three independent digits, in columns 8, 9, and 10 of the card, which control reflection in the three orthogonal coordinate planes. A one in column 8 causes reflection along the X-axis (reflection in Y, Z plane); a one in column 9 causes reflection along the Y-axis; and a one in column 10 causes reflection along the Z axis. A zero or blank in any of these columns causes the corres- ponding reflection to be skipped.

Remarks

Any combination of reflections along the X, Y and Z axes may be used. For example, 101 for (I2) will cause reflection along axes X and Z, and 111 will cause reflection along axes X, Y and Z. When combinations of reflections are requested, the reflections are done in reverse alphabetical order. That is, if a structure is generated in a single octant of space and a GX card is then read with I2 equal to 111, the structure is first reflected along the Z-axis; the structure and its image are then reflected along the Y-axis; and, finally, these four structures are reflected along the X-axis to fill all octants. This order determines the position of a segment in the sequence and, hence, the absolute segment numbers.

The tag increment I1 is used to avoid duplication of tag numbers in the image segments. All valid tags on the original structure are incremented by I1 on the image. When combinations of reflections are employed, the tag increment is doubled after each reflection. Thus, a tag increment greater than or equal to the largest tag an the original structure will ensure that no duplicate tags are generated. For example, if tags from 1 to 100 are used on the original structure with I2 equal to 011 and a tag increment of 100, the first reflection, along the Z-axis, will produce tags from 101 to 200; and the second reflection, along the Y-axis, will produce tags from 201 to 400, as a result of the increment being doubled to 200.

Example:

Return values

- **err** `0` indicates that the result is successful and `1` indicates that an error occurred. Call **nec_error_message()** for a detailed message.

```
long nec_ld_card ( nec_context * in_context,
                   int           ldtyp,
                   int           ldtag,
                   int           ldtagf,
                   int           ldtagt,
                   double        tmp1,
                   double        tmp2,
                   double        tmp3
                 )
```

LD card (Loading)

Parameters

in_context	The nec_context created with nec_create()
ldtyp	Type of loading (5 = segment conductivity)
ldtag	Tag (zero for absolute segment numbers, or in conjunction with 0 for next parameter, for all segments)
ldtagf	Equal to m specifies the mth segment of the set of segments whose tag numbers equal the tag number specified in the previous parameter. If the previous parameter (LDTAG) is zero, LDTAGF then specifies an absolute segment number. If both LDTAG and LDTAGF are zero, all segments will be loaded.
ldtagt	Equal to n specifies the nth segment of the set of segments whose tag numbers equal the tag number specified in the parameter LDTAG. This parameter must be greater than or equal to the previous parameter. The loading specified is applied to each of the mth through nth segments of the set of segments having tags equal to LDTAG. Again if LDTAG is zero, these parameters refer to absolute segment numbers. If LDTAGT is left blank, it is set equal to the previous parameter (LDTAGF).

Example:

Return values

err `0` indicates that the result is successful and `1` indicates that an error occurred. Call nec_error_message() for a detailed message.

Remarks

Floating Point Input for the Various Load Types:

References nec_context::ld_card().

long nec_medium_parameters (nec_context * in_context, double permittivity, double permeability)

Set the prameters of the medium (permittivity and permeability)

Parameters

- **permittivity** The electric permittivity of the medium (in farads per meter)
- **permeability** The magnetic permeability of the medium (in henries per meter)

Remarks

From these parameters a speed of light is chosen.

Example:

Return values

- **err** `0` indicates that the result is successful and `1` indicates that an error occurred. Call **nec_error_message()** for a detailed message.

References **nec_context::medium_parameters()**.

long nec_pt_card (nec_context * in_context,
int itmp1,
int itmp2,
int itmp3,
int itmp4
)

Print Flag (Printing of Currents.

Parameters

IPTFLG Print control flag, specifies the type of format used in printing segment currents. The options are:

- `-2` - all currents printed. This it a default value for the program if the card is Omitted.
- `-1` - suppress printing of all wire segment currents.
- `0` - current printing will be limited to the segments specified by the next three parameters.
- `1` - currents are printed by using a format designed for a receiving pattern (refer to output section in this manual Only currents for the segments specified by the next three parameters are printed.
- `2` - same as for 1 above; in addition, however, the current for one Segment will Cue normalized to its maximum, ant the normalized values along with the relative strength in tB will be printed in a table. If the currents for more than one segment are being printed, only currents from the last segment in the group appear in the normalized table.
- `3` - only normalized currents from one segment are printed for the receiving pattern case.

IPTAG - Tag number of the segments for which currents will be printed.

IPTAGF - Equal to m, specifies the mth segment of the set of segments having the tag numbers of IPTAG, at which printing of currents starts. If IPTAG is zero or blank, then IPTAGF refers to an absolute segment number. If IPTAGF is blank, the current is printed for all segments.

IPTAGT - Equal to n specifies the nth segment of the set of segments having tag numbers of IPTAG. Currents are printed for segments having tag number IPTAG starting at the m th segment in the set and ending at the nth segment. If IPTAG is zero or blank, then IPTAGF and IPTAGT refer to absoulte segment numbers. In IPTAGT is left blank, it is set to IPTAGF.

Example:

Return values

err `0` indicates that the result is successful and `1` indicates that an error occurred. Call **nec_error_message()** for a detailed message.

References **nec_context::pt_card()**.

```
long nec_rp_card ( nec_context * in_context,
                   int           calc_mode,
                   int           n_theta,
                   int           n_phi,
                   int           output_format,
                   int           normalization,
                   int           D,
                   int           A,
                   double        theta0,
                   double        phi0,
                   double        delta_theta,
                   double        delta_phi,
                   double        radial_distance,
                   double        gain_norm
                 )
```

Standard radiation pattern parameters.

Parameters

calc_mode This integer selects the mode of calculation for the radiated field. Some values of (calc_mode) will affect the meaning of the remaining parameters on the card. Options available for calc_mode are:

- 0 - normal mode. Space-wave fields are computed. An infinite ground plane is included if it has been specified previously on a GN card; otherwise, the antenna is in free space.
- 1 - surface wave propagating along ground is added to the normal space wave. This option changes the meaning of some of the other parameters on the RP card as explained below, and the results appear in a special output format. Ground parameters must have been input on a GN card. The following options cause calculation of only the space wave but with special ground conditions. Ground conditions include a two-medium ground (cliff where the media join in a circle or a line), and a radial wire ground screen. Ground parameters and dimensions must be input on a GN or GD card before the RP card is read. The RP card only selects the option for inclusion in the field calculation. (Refer to the GN and GD cards for further explanation.)
- 2 - linear cliff with antenna above upper level. Lower medium parameters are as specified for the second medium on the GN card or on the GD card.
- 3 - circular cliff centered at origin of coordinate system: with antenna above upper level. Lower medium parameters are as specified for the second medium on the GN card or on the GD card.

- 4 - radial wire ground screen centered at origin.
- 5 - both radial wire ground screen and linear cliff.
- 6 - both radial wire ground screen ant circular cliff.

n_theta The number of theta angles.

n_phi The number of phi angles.

output_format The output format:

- 0 major axis, minor axis and total gain printed.
- 1 vertical, horizontal ant total gain printed.

normalization Controls the type of normalization of the radiation pattern

- 0 no normalized gain.
- 1 major axis gain normalized.
- 2 minor axis gain normalized.
- 3 vertical axis gain normalized.
- 4 horizontal axis gain normalized.
- 5 total gain normalized.

D Selects either power gain or directive gain for both standard printing and normalization. If the structure excitation is an incident plane wave, the quantities printed under the heading "gain" will actually be the scattering cross section (a/lambda 2) and will not be affected by the value of d. The column heading for the output will still read "power" or "directive gain," however.

- 0 power gain.
- 1 directive gain.

A - Requests calculation of average power gain over the region covered by field points.

- 0 no averaging.
- 1 average gain computed.
- 2 average gain computed, printing of gain at the field points used for averaging is suppressed. If n_theta or NPH is equal to one, average gain will not be computed for any value of A since the area of the region covered by field points vanishes.

theta0 - Initial theta angle in degrees (initial z coordinate in meters if calc_mode = 1).

phi0 - Initial phi angle in degrees.

delta_theta - Increment for theta in degrees (increment for z in meters if calc_mode = 1).

delta_phi - Increment for phi in degrees.

radial_distance - Radial distance (R) of field point from the origin in meters. radial_distance is optional. If it is zero, the radiated electric field will have the factor exp(-jkR)/R omitted. If a value of R is specified, it should represent a point in the far-field region since near components of the field cannot be obtained with an RP card. (If calc_mode = 1, then radial_distance represents the cylindrical coordinate phi in meters and is not optional. It must be greater than about one wavelength.)

gain_norm - Determines the gain normalization factor if normalization has been requested in the normalization parameter. If gain_norm is zero, the gain will be normalized to its maximum value. If gain_norm is not zero, the gain wi11 be normalized to the value of gain_norm.

Example:

Return values

err `0` indicates that the result is successful and `1` indicates that an error occurred. Call **nec_error_message()** for a detailed message.

Remarks

The field point is specified in spherical coordinates (R, sigma, theta), except when the surface wave is computed. For computing the surface wave field (calc_mode = l), cylindrical coordinates (phi, theta, z) are used to accurately define points near the ground plane at large radial distances.

The rp_card() function allows automatic stepping of the field point to compute the field over a region about the antenna at uniformly spaced points.

The integers n_theta and n_phi and floating point numbers theta0, phi0, delta_theta, delta_phi, radial_distance, and gain_norm control the field-point stepping.

- The **nec_rp_card()** function will cause the interaction matrix to be computed and factored and the structure currents to be computed if these operations have not already been performed. Hence, all required input parameters must be set before the **nec_rp_card()** function is called.
- At a single frequency, any number of **nec_rp_card()** calls may occur in sequence so that different field-point spacings may be used over different regions of space. If automatic frequency stepping is being used (i.e., in_nfrq on the **nec_fr_card()** function is greater than one), only one **nec_rp_card()** function will act as data inside the loop. Subsequent calls to **nec_rp_card()** will calculate patterns at the final frequency.
- When both n_theta and n_phi are greater than one, the angle theta (or Z) will be stepped faster than phi.
- When a ground plane has been specified, field points should not be requested below the ground (theta greater than 90 degrees or Z less than zero.)

Examples:

test_nec.c.

References **nec_context::rp_card()**.

```
long nec_sc_card ( nec_context * in_context,
                   int           i2,
                   double        x3,
                   double        y3,
                   double        z3,
                   double        x4,
                   double        y4,
                   double        z4
                 )
```

Surface Patch Continuation (SC Card)

Parameters

in_context	The **nec_context** created with **nec_create()**
i2	Weird integer parameter.
x3	The x coordinate of patch corner 3.
y3	The y coordinate of patch corner 3.
z3	The z coordinate of patch corner 3.
x4	The x coordinate of patch corner 4.
y4	The y coordinate of patch corner 4.
z4	The z coordinate of patch corner 4.

Example:

Return values

err `0` indicates that the result is successful and `1` indicates that an error occurred. Call **nec_error_message()** for a detailed message.

Remarks

All co-ordinates are in meters.

```
long nec_sp_card ( nec_context * in_context,
                   int           ns,
                   double        x1,
                   double        y1,
                   double        z1,
                   double        x2,
                   double        y2,
                   double        z2
                 )
```

Surface Patch (SP Card)

Parameters

in_context	The nec_context created with nec_create()
ns	The Patch Type. • `0` (default) arbitrary patch shape • `1` rectangular patch • `2` triangular patch • `3` quadrilateral patch
x1	The x coordinate of patch corner1.
y1	The y coordinate of patch corner1.
z1	The z coordinate of patch corner1.
x2	The x coordinate of patch corner2.
y2	The y coordinate of patch corner2.
z2	The z coordinate of patch corner2.

Example:

Return values

err `0` indicates that the result is successful and `1` indicates that an error occurred. Call nec_error_message() for a detailed message.

Remarks

All co-ordinates are in meters, except for arbitrary patches where the angles are in degrees

```
long nec_wire ( nec_context * in_context,
                int           tag_id,
                int           segment_count,
                double        xw1,
                double        yw1,
                double        zw1,
                double        xw2,
                double        yw2,
                double        zw2,
                double        rad,
                double        rdel,
                double        rrad
              )
```

Create a straigt wire.

Parameters

in_context	The nec_context created with nec_create()
tag_id	The tag ID.
segment_count	The number of segments.
xw1	The x coordinate of the wire starting point.
yw1	The y coordinate of the wire starting point.
zw1	The z coordinate of the wire starting point.
xw2	The x coordinate of the wire ending point.
yw2	The y coordinate of the wire ending point.
zw2	The z coordinate of the wire ending point.
rad	The wire radius (meters)
rdel	For tapered wires, the. Otherwise set to 1.0
rrad	For tapered wires, the. Otherwise set to 1.0

Example:

Return values

err `0` indicates that the result is successful and `1` indicates that an error occurred. Call nec_error_message() for a detailed message.

Remarks

All co-ordinates are in meters.

Examples:

test_nec.c.

References nec_context::wire().

Referenced by c_geometry::wire().

long nec_xq_card (nec_context * in_context,
int itmp1
)

XQ Card (Execute)

Purpose: To cause program execution at points in the data stream where execution is not automatic. Options on the card also allow for automatic generation of radiation patterns in either of two vertical cuts.

Parameters

- **in_context** The nec_context created with nec_create()
- **itmp1** Options controlled by (I1) are: 0 - no patterns requested (normal case). 1 - generates a pattern cut in the XZ plane, i.e., phi = 0 degrees and theta varies from 0 degrees to 90 degrees in 1 degree steps. 2 - generates a pattern cut in the YZ plane, i.e., phi = 90 degrees theta varies from 0 degrees to 90 degrees in 1 degree steps. 3 - generates both of the cuts described for the values 1 and 2.

Example:

Return values

- **err** `0` indicates that the result is successful and `1` indicates that an error occurred. Call nec_error_message() for a detailed message.

References nec_context::xq_card().