NAME

vnacal_make_scalar_parameter, vnacal_make_vector_parameter, vnacal_make_unknown_parameter, vnacal_make_correlated_parameter, get_parameter_value, vnacal_delete_parameter − vnacal s-parameter functions

SYNOPSIS

#include <vnacal.h>

VNACAL_MATCH (also **VNACAL_ZERO**)
VNACAL_OPEN (also **VNACAL_ONE**)
VNACAL_SHORT

int vnacal_make_scalar_parameter(vnacal_t ****vcp*, double complex** *gamma***);**

int vnacal_make_vector_parameter(vnacal_t ****vcp*,**
const double * *frequency_vector***, int** *frequencies***, const double complex *** *gamma_vector***);**

int vnacal_make_unknown_parameter(vnacal_t ****vcp*, int** *initial_guess***);**

int vnacal_make_correlated_parameter(vnacal_t ****vcp*, int** *other***,**
const double * *sigma_frequency_vector***, int** *sigma_frequencies***, const double *** *sigma_vector***);**

double complex vnacal_get_parameter_value(vnacal_t ****vcp*, int** *parameter***, double** *frequency***);**

int vnacal_delete_parameter(vnacal_t ****vcp*, int** *parameter***);**

Link with *-lvna -lyaml -lm*.

DESCRIPTION

These functions are used to specify the S-parameters for calibration standards used by the **vnacal_new_add_***() functions.

Instead of taking a matrix of complex numbers that specify the S-parameters directly, the **vnacal_new_add_***() functions instead take a matrix of type **int**, filled with either the predefined constants: **VNACAL_MATCH**, **VNACAL_OPEN**, **VNACAL_SHORT**, **VNACAL_ZERO**, **VNACAL_ONE**, or the integer handles returned from the four **vnacal_make_***_**parameter**() functions described here. There are two reasons for this extra level of abstraction. First, it unifies the interface between scalar parameters − parameters that are constant across all frequencies (e.g. -1.0 for short) − and vector parameters that are given at a list of frequency points. Second, it allows for parameters to be specified as unknown: parameters that the library has to solve for.

All of the functions take a pointer to a **vnacal_t** obtained from **vnacal_create**() or **vnacal_load**(). See **vnacal**(3).

vnacal_make_scalar_parameter() creates a frequency-independent parameter with a reflection coefficient of *gamma*.

vnacal_make_vector_parameter() creates a frequency-dependent parameter, where *frequency_vector* and *gamma_vector* are vectors of length *frequencies* giving the frequency and value at each point. The library uses rational function interpolation to interpolate between frequencies if the given frequency points don't align with the calibration frequencies.

vnacal_make_unknown_parameter() creates a parameter with unknown value, where *initial_guess* is either one of the pre-defined constants or the value returned from **vnacal_make_scalar_parameter**() or **vnacal_make_vector_parameter**(), giving a starting point for the solution.

vnacal_make_correlated_parameter() creates a parameter with unknown value that is correlated with another (possibly unknown) parameter, *other*, with per-frequency standard deviations of the differences given in *sigma_vector*. The *sigma_frequency_vector* argument gives the list of frequencies where sigma values are specified, and *sigma_frequencies* gives the length of both *sigma_vector* and *sigma_frequency_vector*. This type of parameter is useful for stochastic modeling of connection non-repeatability.

If *sigma_frequencies* is 1, then *sigma_frequency_vector* is ignored, and *sigma_vector*[0] is taken as a

frequency-independent value. Otherwise, the frequencies given in *sigma_frequencies* must be positive increasing values that overlap with, but don't necessarily have to be the same as, those given in either **vnacal_new_set_frequency_vector**() or those given in a vector parameter referred to by *other*. The library uses natural cubic-spline interpolation to interpolate sigma values between frequency points as needed.

vnacal_get_parameter_value() returns the value of the parameter at a given frequency. If given a scalar parameter, **vnacal_get_parameter_value**() ignores *frequency*, and simply returns the fixed gamma value; if given a vector parameter, it computes the value at the given frequency, interpolating as necessary. The *frequency* parameter must lie within the frequency range of the parameter. If given an unknown or correlated parameter, this function returns the most recent value computed by **vnacal_new_solve**(), or fails if the parameter has not been solved.

vnacal_delete_parameter() removes a parameter from the **vnacal_t** structure. It's not an error to delete a parameter that has been added to a **vnacal_new_t** structure – a copy of the parameter will continue to exist internally until the last reference has been released.

RETURN VALUE

The **vnacal_make_*_parameter**() functions return an integer handle to the newly formed parameter on success or -1 on error. The **vnacal_get_parameter_value**() function returns a complex number on success or HUGE_VAL on error. **vnacal_delete_parameter**() returns 0 on success or -1 on error.

ERRORS

On error, these functions invoke the *error_fn*, given to **vnacal_create**() or **vnacal_load**() if provided, set **errno** to one of the following values and return failure.

EINVAL **vnacal_make_vector_parameter**() was given an invalid frequency vector. **vnacal_make_unknown_parameter**(), **vnacal_make_correlated_parameter**() or **vnacal_delete_parameter**() was called with an invalid parameter handle.

ENOMEM
The library was unable to allocate memory.

SEE ALSO

vnacal(3), **vnacal_new**(3), **vnaconv**(3), **vnadata**(3), **vnaerr**(3)