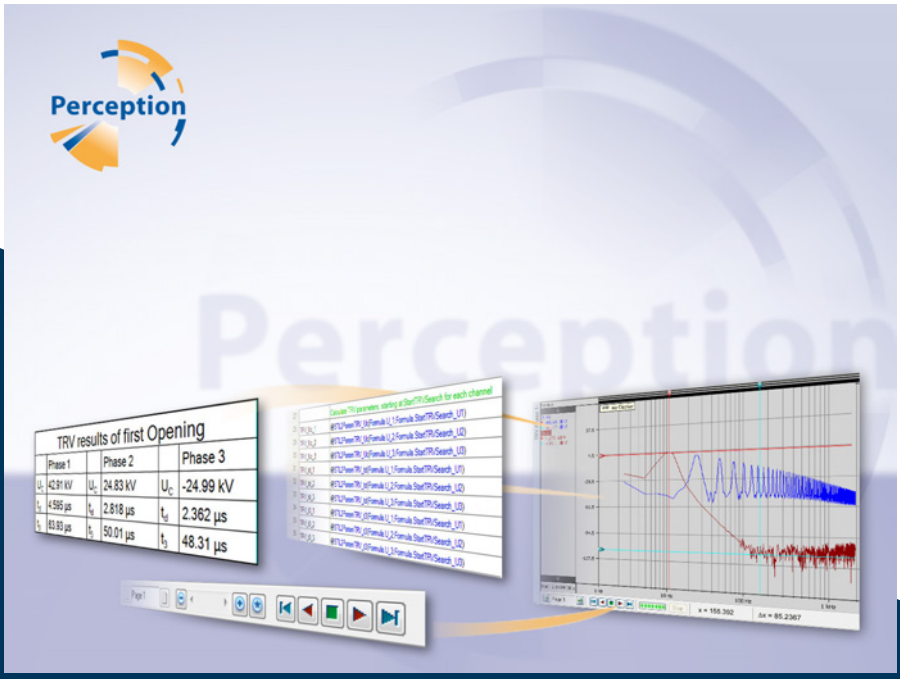
HBK
HOTTINGER BRÜEL & KJÆR

English

User Manual



Perception

STL Analysis Option

HBM

Brüel & Kjær

IMPRINT

Document version 3.0 - February 2025

For HBK's Terms and Conditions visit www.hbm.com/terms

Hottinger Brüel & Kjaer GmbH
Im Tiefen See 45
64293 Darmstadt
Germany
Tel: +49 6151 80 30
Fax: +49 6151 8039100
Email: info@hbm.com
www.hbm.com/highspeed

Copyright © 2025

All rights reserved. No part of the contents of this document may be reproduced or transmitted in any form or by any means without the written permission of the publisher.

LICENSE AGREEMENT AND WARRANTY

LICENSE AGREEMENT and WARRANTY

For information about LICENSE AGREEMENT AND WARRANTY refer to:
www.hbm.com/terms.

TABLE OF CONTENTS

TABLE OF CONTENTS

1 STL - Analysis Option

1.1 Introduction

Testing of switchgear devices and fuses requires advanced analysis capabilities to generate reproducible and accurate test results. Real world signals may be distorted, carry noise or spikes, but they still have to be evaluated appropriately. The STLA (Short- Circuit Testing Liaison Agreement) defined methods to unify the evaluation process of signals for HV electrical power equipment.

The calculations presented in this document are designed and implemented according to the STL technical report “Harmonisation of data processing methods for High Power Laboratories, September 2004”. Each implemented function in Perception is referenced to the relevant paragraph in the STL technical report.

The Perception STL Analysis option provides a set of calculations. For this it requires the Perception Analysis Option (a.k.a. formula database) also be installed. The calculation functions use advanced algorithms and methods like iterative loops, curve fitting, spike/noise suppression etc. to comply with the STL technical report. There are a number of generic calculations available for different purposes, a set of calculations for recovery voltage evaluation and a number of symmetric and asymmetric current calculations. In addition, various calculations for No-Load and travel recorder traces are available.

1.1.1 How to install the STL option

The Perception software requires a HASP key. HASP (Hardware Against Software Piracy) is a hardware-based (hardware key) software copy protection system that prevents unauthorized use of software applications.
Each HASP key contains a unique ID number used for personalization of the application according to the features and options purchased. The key is also used for storing licensing parameters, applications and customer-specific data. If you have purchased the STL option as a separate item, you will receive a personalized "key file". Use this file to unlock the additional features.

You can find the serial number of your key in **Help ► About Perception**

To update the key information:

1. Choose **Help ► Update Key...**
2. In the Open dialog locate the Key File (*.pKey) and click **Open**.
3. If everything is OK you will see the following message:

Fig. 1.1 Software copy protection dialog

4. Click **OK**.

After the installation you can go to **Help ► About Perception ► More...** to see all installed options.

You will need to restart the program before the changes take effect. The STL option is now available.

2 STL - Functions

2.1 @STLSignalStart

Function

This function will be used to recognize the start of a signal.

Syntax

@STLSignalStart*(Waveform; Frequency; StartPos; EndPos)*

Parameters

Waveform	Sinusoidal input waveform
Frequency	Frequency of the sinusoidal waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the start of the signal.

Description

This function uses the double threshold method to detect the start of a signal. The Y threshold is 3% of the Full Scale level and the X threshold is 0.5% of the signal period.

For a 50 Hz sinus (f(t) = sin (100 π t)) the threshold levels are:

```
Y-threshold: = 2 * 3 / 100 = 0.06 = 60 mVolt
X-threshold: = 0.05 / 50 = 0.001 = 1 msec
```

The function starts searching at “StartPos” and searches forward in time until it finds the signal start. If the signal start is not found before “EndPos” a double.NaN value will be returned.

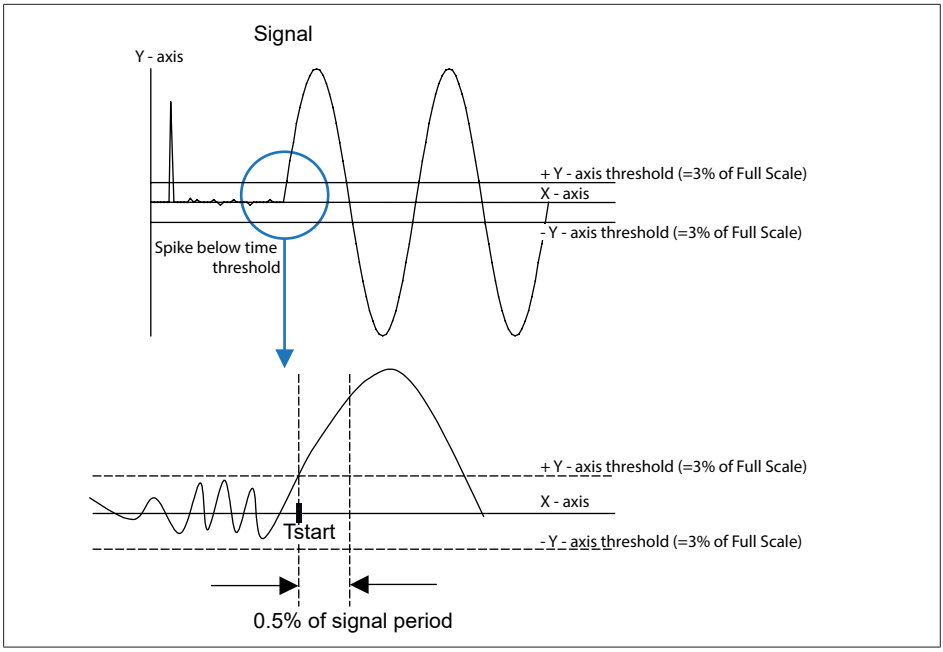


Fig. 2.1 STL SignalStart

This function uses the same method as the *STLSignalEnd*, but in opposite direction. See also the description of chapter "@STLSignalEnd“ on page 10 to get more information on the behavior of both signal start and end functions.

STL documentation reference

§ 6.2.2. Recognition of signal

2.2 @STLSignalEnd

Function

This function will be used to recognize the end of a signal.

Syntax

@STLSignalEnd*(Waveform; Frequency; StartPos; EndPos)*

Parameters

Waveform	Sinusoidal input waveform
Frequency	Frequency of the sinusoidal waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Description

This function uses the double threshold method to detect the end of a signal. The Y threshold is 3% of the Full Scale level and the X threshold is 0.5% of the signal period.

For a 50 Hz sinus (f(t) = sin (100 π t)) the threshold levels are:

```
Y-threshold: = 2 * 3 / 100 = 0.06 = 60 mVolt
X-threshold: = 0.05 / 50 = 0.001 = 1 msec
```

The function starts searching at “StartPos”, the search direction depends on the value of “EndPos”:

- Searches backwards if *“StartPos”* greater than *“EndPos”*
- Searches forwards if *“EndPos”* greater than *“StartPos”*

When *“StartPos”* and *“EndPos”* are not entered in the formula, the function starts searching at the beginning of the signal in forward direction until it finds an end condition or it stops when there are no more data points.

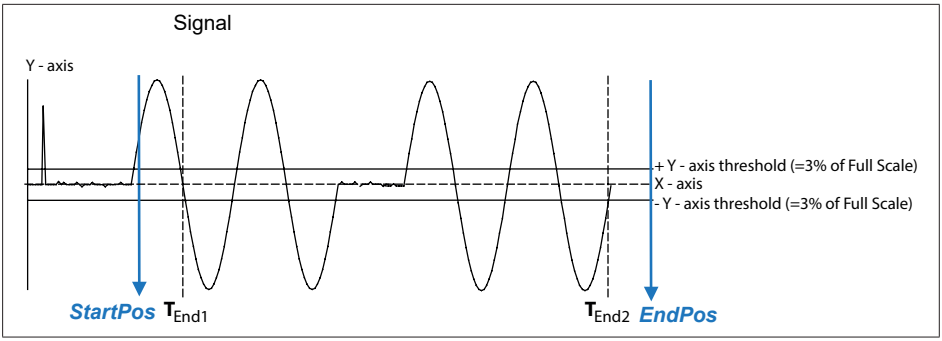


Fig. 2.2 STL SignalEnd

If the STLSignalEnd formula for the Signal is used as shown in the picture Figure 2.2 you will get the following End position:

- T_{End1} @STLSignalEnd (Formula.Signal; 50; StartPos; EndPos)

If the StartPos and EndPos in the formula is swapped you will find a different End position:

- T_{End2} @STLSignalEnd (Formula.Signal; 50; EndPos; StartPos

This function uses the same method as the *STLSignalStart*, but in opposite direction. See also the description of chapter "@STLSignalStart“ on page 8 to get more information on the behavior of both signal start and end functions.

Output

A Numerical value indicating the end of the signal.

STL documentation reference

§ 6.2.2. Recognition of signal

2.3 @STLNextZeroCrossing

Function

This function will be used to obtain the correct zero-crossing moment for a sinusoidal signal.

Syntax

@STLNextZeroCrossing*(Waveform; Frequency; StartPos; EndPos; SkipCount)*

Parameters

Waveform	Sinusoidal input waveform
Frequency	Optional; Frequency of the sinusoidal waveform, default 50Hz.
StartPos	Optional; beginning of search
EndPos	Optional; end of search
SkipCount	Optional; Number of zero crossings to be skipped, default 0.

Output

A Numerical value indicating the first found zero crossing time after “StartPos””, when however SkipCount is not zero then the returned value will be the found zero crossing time after skipping SkipCount zero crossings.

Description

This function searches for the next zero level crossing location beginning from “StartPos”. Once this location is found it will be used to define a time window. All data points in this time window will be used for a linear curve fitting. The zero crossing of the resulting line will be used to get a new time window. The linear curve fitting will be repeated. This sequence will be done several times. At least 10 points should be available in the time window.

If no zero crossing is found before *“EndPos”* a double.NaN value will be returned.

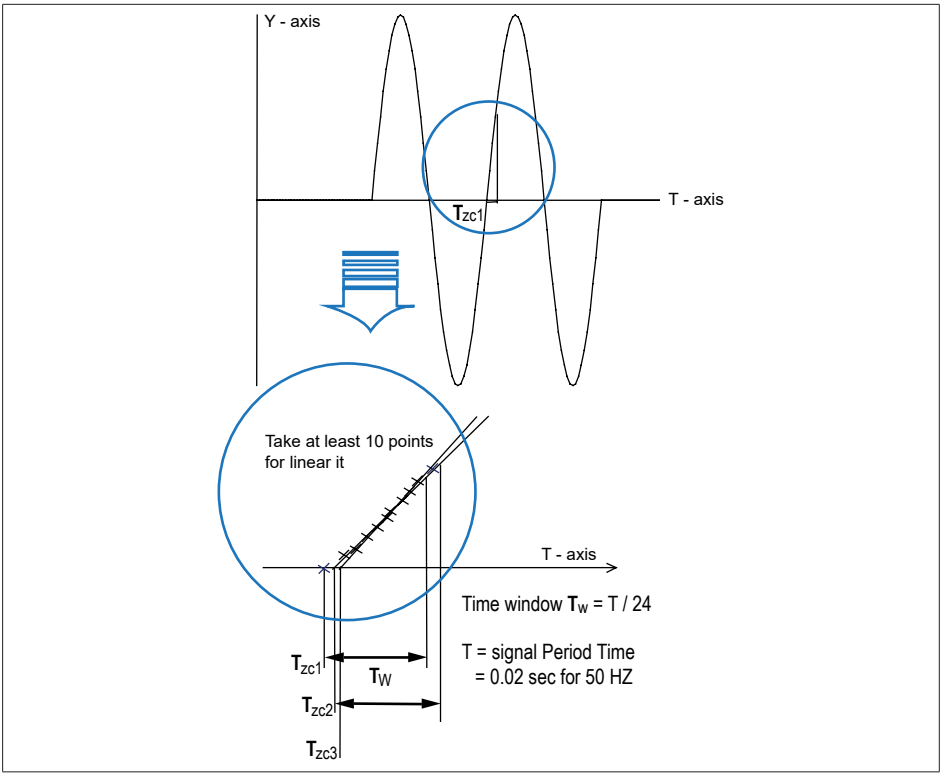


Fig. 2.3 STLNextZeroCrossing

STL documentation reference

§ 6.3.3. Recognition of signal

2.4 @STLPrevZeroCrossing

Function

This function will be used to obtain the correct zero-crossing location for a sinusoidal signal.

Syntax

@STLPrevZeroCrossing*(Waveform; Frequency; StartPos; EndPos; SkipCount)*

Parameters

Waveform	Sinusoidal input waveform
Frequency	Optional; Frequency of the sinusoidal waveform, default 50Hz.
StartPos	Optional; beginning of search
EndPos	Optional; end of search
SkipCount	Optional; Number of zero crossings to be skipped, default 0.

Output

A Numerical value indicating the first found zero crossing time before *"EndPos"*, when however *SkipCount* is not zero then the returned value will be the found zero crossing time before skipping *SkipCount* zero crossings.

Description

The function starts searching from *"StartPos"* and searches backward in time till it finds a zero crossing location. If no zero crossing is found before *"EndPos"* a double.NaN value will be returned. Once this location is found it will be used to define a time window. All data points in this time window will be used for a linear curve fitting. The zero crossing of the resulting line will be used to get a new time window. The linear curve fitting will be repeated. This sequence will be done several times. At least 10 points should be available in the time window.

This function uses the same method as the *STLNextZeroCrossing*, but in opposite direction.

STL documentation reference

§ 6.3.3. Calculation of the zero crossing of a signal

2.5 @STLNextCrestTime

Function

This function will be used to determine the time corresponding to the next peak value of a sinusoidal-like signal.

Syntax

@STLNextCrestTime*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Sinusoidal input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the time of the first found peak value after *"StartPos"*.

Description

This function uses the threshold method in combination with a parabolic curve fitting to detect the peak value of a sinusoidal signal. The function starts searching at *"StartPos"* and searches forward in time till it finds a peak. If no peak is found before *"EndPos"* a double.NaN value will be returned.

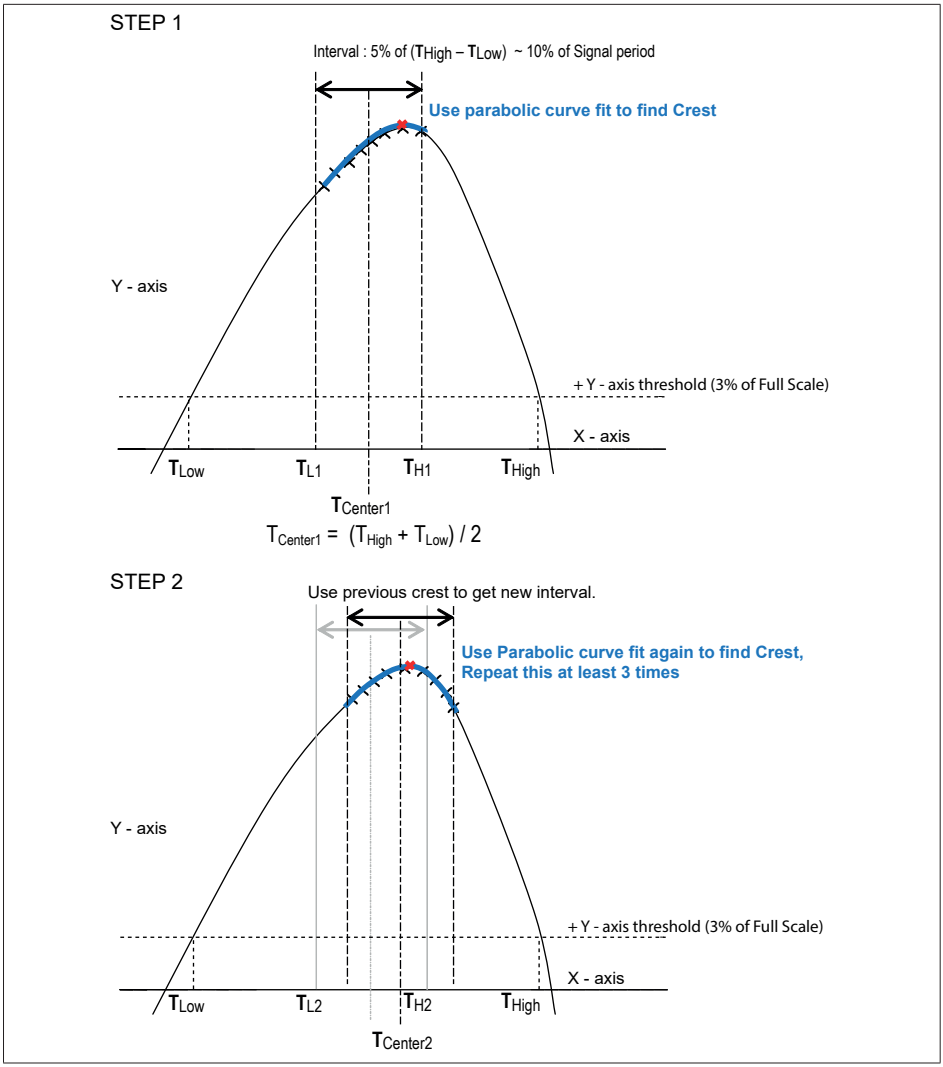


Fig. 2.4 STLNextCrestTime

STL documentation reference

§ 6.3.2. Calculation of the peak value of a signal

2.6 @STLPrevCrestTime

Function

This function will be used to determine the time corresponding to the previous peak value of a sinusoidal-like signal.

Syntax

@STLPrevCrestTime*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Sinusoidal input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the time of the first found peak before *"EndPos"*.

Description

This function uses the threshold method in combination with a parabolic curve fitting to detect the peak value of a sinusoidal signal. The function starts searching at *"EndPos"* and searches backward till it finds a peak.

If no peak is found before *"StartPos"* a double.NaN value will be returned. This function uses the same method as the *STLNextCrestTime*, but in opposite direction.

If the *"StartPos"* location is after the *"EndPos"* then the time parameters will be swapped.

STL documentation reference

§ 6.3.2. Calculation of the peak value of a signal

2.7 @STLNextCrestVal

Function

This function will be used to determine the next peak value of a sinusoidal-like signal.

Syntax

@STLNextCrestVal*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Sinusoidal input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the first found peak value after *"StartPos"*.

Description

This function uses the threshold method in combination with a linear and parabolic curve fitting to detect the peak value of a sinusoidal signal.
The function starts searching at *"StartPos"* and searches forward in time till it finds a peak. If no peak is found before *"EndPos"* a double.Nan value will be returned.
This function uses the same method as the *STLNextCrestTime*.

STL documentation reference

§ 6.3.2. Calculation of the peak value of a signal

2.8 @STLPrevCrestVal

Function

This function will be used to determine the previous peak value of a sinusoidal-like signal.

Syntax

@STLPrevCrestVal*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Sinusoidal input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the first found peak value found before *"EndPos"*.

Description

This function uses the threshold method in combination with a linear and parabolic curve fitting to detect the peak value of a sinusoidal signal. The function starts searching at *"EndPos"* and searches backward in time till it finds a peak. If no peak is found before *"StartPos"* a double.NaN value will be returned.
This function uses the same method as the *STLNextCrestVal.*

If the *"StartPos"* location is after the *"EndPos"* then the time parameters will be swapped.

STL documentation reference

§ 6.3.2. Calculation of the peak value of a signal

2.9 @STLFirstMaxCrestVal

Function

This function will be used to determine the max peak value of one of the first two peaks of a sinusoidal-like signal.

Syntax

@STLFirstMaxCrestVal*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Sinusoidal input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the max peak value out of the first two crests after *"StartPos"*.

Description

If doing short circuit test then the measured current is an asymmetric sinusoidal signal. Most of the time the first crest will be the maximum crest, but there are situations where the second crest will be larger then the first crest. This function can be used to get the first maximum crest out of those two crests.
The function starts searching at *"StartPos"* and searches forward in time until it finds two peaks. If not two peak are found before *"EndPos"* a double.NaN value will be returned.

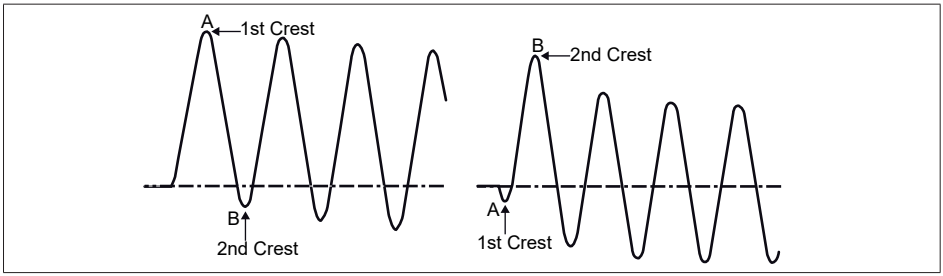


Fig. 2.5 STLFirstmaxCrestVal - Overview

Signal 1

```
@STLFirstMaxCresTime
(Formula.Signal1) = At
@STLFirstMaxCrestVal
(Formula.Signal1) = Au
```

Signal 2

```
@STLFirstMaxCrestTime
(Formula.Signal2) = Bt
@STLFirstMaxCrestVal
(Formula.Signal2) = Bu
```

STL documentation reference

None

2.10 @STLFirstMaxCrestTime

Function

This function will be used to determine the time corresponding to the max peak value of one of the first two peaks of a sinusoidal-like signal.

Syntax

@STLFirstMaxCrestTime*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Sinusoidal input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the time of the max peak value out of the first two crests after *"StartPos"*.

Description

This function works the same as the *STLFirstMaxCrestVal,* except it returns the time position of the max peak value out of the first two crests.

STL documentation reference

None

2.11 @STLValueFunction

Function

This function returns the instantaneous value of a signal at a specified time.

Syntax

@STLValue*(Waveform; XPosition)*

Parameters

Waveform	Input waveform
XPosition	X position at which the value of Waveform is to be determined

Output

A Numerical value indicating the value of a waveform at a specified x-position.

Description

This function returns the value of a waveform at a specified x-position.

The function uses the three succeeding data samples around the specified x position. The instantaneous value is the average of those three corresponding sampling values.

STL documentation reference

§ 6.3.1. Calculation of the instantaneous value of a signal

2.12 @STLNextSlopeAtZeroCrossing

Function

This function will be used to determine the slope at the next zero crossing of a sinusoidal-like signal.

Syntax

@STLNextSlopeAtZeroCrossing*(Waveform; Frequency; StartPos; EndPos)*

Parameters

Waveform	Sinusoidal input waveform
Frequency	Optional; Frequency of the sinusoidal waveform, default 50Hz.
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the slope at the next zero crossing found after *"StartPos"*.

Description

This function uses the crest function to find the next peak after *"StartPos"*. The slope at the first zero crossing point after the peak will be determined.

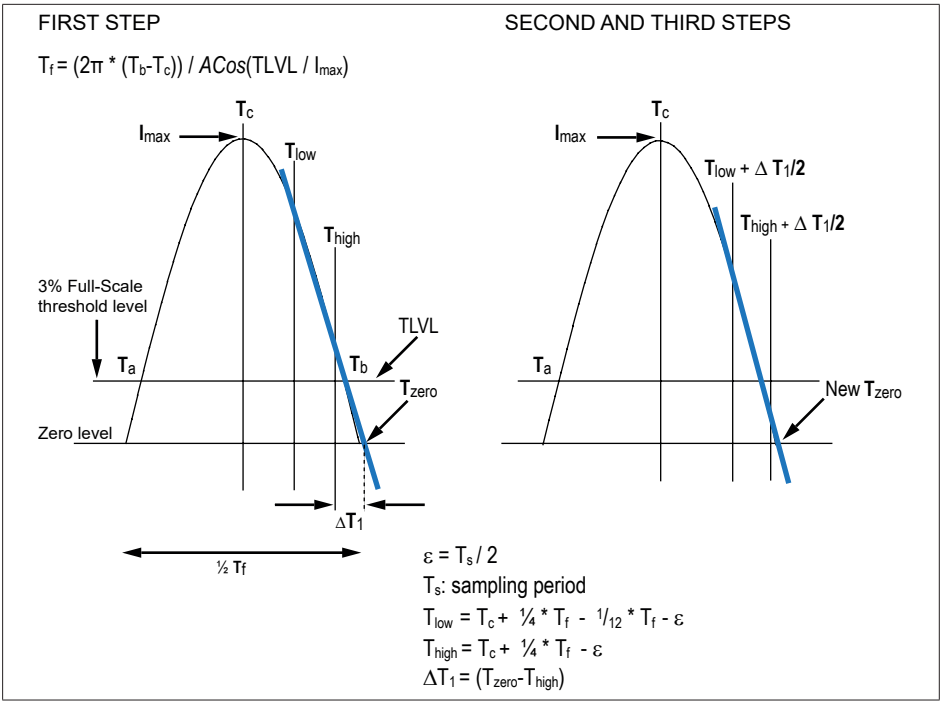


Fig. 2.6 STLNextSlopeAtZeroCrossing

STL documentation reference

§ 6.3.4. Determination of the slope (e.g. di/dt) at zero crossing of a signal

2.13 @STLPrevSlopeAtZeroCrossing

Function

This function will be used to determine the slope at the previous zero crossing of a sinusoidal-like signal.

Syntax

@STLPrevSlopeAtZeroCrossing*(Waveform; Frequency; StartPos; EndPos)*

Parameters

Waveform	Sinusoidal input waveform
Frequency	Optional; Frequency of the sinusoidal waveform, default 50Hz.
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the slope at the previous zero crossing found before *"EndPos"*.

Description

This function uses the crest function to find the previous peak before *"EndPos"*.

This function uses the same method as the *STLNextSlopeAtZeroCrossing*, but in opposite direction.

STL documentation reference

§ 6.3.4. Determination of the slope (e.g. di/dt) at zero crossing of a signal

2.14 @STLNext3CrestRMS

Function

This function will be used to determine the equivalent RMS value of the AC component of a signal by the 3-crest method.

Syntax

@STLNext3CrestRMS*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Sinusoidal input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the equivalent RMS value of the AC component of the “waveform” signal.

Description

This function uses the 3-crest method to calculate the r.m.s value of a sinusoidal signal. The function starts searching at *“StartPos”* and searches forward in time till it finds a peak. If no peak is found before *“EndPos”* a double.NaN value will be returned.
After it finds the first peak it searches for the next two peaks using the crest functions.

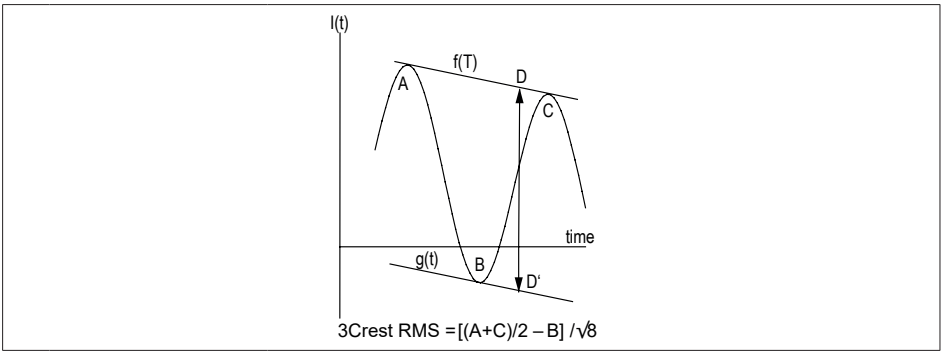


Fig. 2.7 STLNext3CrestRMS

STL documentation reference

§ 6.4.1. Evaluation of the equivalent RMS value of the AC component of a signal by the 3-crest method

2.15 @STLPrev3CrestRMS

Function

This function will be used to determine the equivalent RMS value of the AC component of a signal by the 3-crest method.

Syntax

@STLPrev3CrestRMS*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Sinusoidal input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the equivalent RMS value of the AC component of the *"waveform"* signal.

Description

This function uses the 3-crest method to calculate the r.m.s value of a sinusoidal signal.
The function starts searching from "StartPos" and searches backward in time till it finds a peak. If no peak is found before "EndPos" a double.NaN value will be returned.
After it finds the first peak it searches for the previous two peaks using the crest functions. This function uses the same method as the STLNext3CrestRMS, but in opposite direction.

STL documentation reference

§ 6.4.1. Evaluation of the equivalent RMS value of the AC component of a signal by the 3-crest method

2.16 @STLNextTrueRMS

Function

This function will be used to obtain the true RMS value of a sinusoidal signal.

Syntax

@STLNextTrueRMS*(Waveform; Frequency; StartPos; EndPos)*

Parameters

Waveform	Sinusoidal input waveform
Frequency	Frequency of the sinusoidal waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the true RMS value of *"Waveform"* between *"StartPos"* and *"EndPos"*.

Description

This function uses the zero crossing function to get the correct start and end time for calculating the true RMS value of a sinusoidal signal.
The function starts searching at "StartPos" and searches forward in time till it finds a zero crossing. From here it will search for the last zero crossing before "EndPos". This last zero crossing should have an identical polarity as the first zero crossing slope.

STL documentation reference

§ 6.4.2. Evaluation of the true RMS value of a signal

2.17 @STLPrevTrueRMS

Function

This function will be used to obtain the true RMS value of a sinusoidal signal.

Syntax

@STLPrevTrueRMS*(Waveform; Frequency; StartPos; EndPos)*

Parameters

Waveform	Sinusoidal input waveform
Frequency	Frequency of the sinusoidal waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the true RMS value of *"Waveform"* between *"StartPos"* and *"EndPos"*.

Description

This function uses the zero crossing function to get the correct start and end time for calculating the true RMS value of a sinusoidal signal.
The function starts searching at *"EndPos"* and searches backward in time till it finds a zero crossing. From here it will search for the last zero crossing before *"StartPos"*. This last zero crossing should have an identical polarity as the first zero crossing slope.
This function uses the same method as the *STLNextTrueRMS*, but in opposite direction.

STL documentation reference

§ 6.4.2. Evaluation of the true RMS value of a signal

2.18 @STL2ParamTRV_Uc

Function

This function will be used to obtain the **Uc** (Crest value) of a Transient Recovery Voltage (TRV) using a 2 parameter calculation.

Syntax

@STL2ParamTRV_Uc*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the Uc value of the TRV signal *"Waveform"* between *"StartPos"* and *"EndPos"*.

Description

This function uses the 2 parameter TRV method to determine Uc.
The function starts searching at *"StartPos"* for a signal start, it uses the double threshold method but with a lower level (1%). If no TRV is found before *"EndPos"* a double.NaN value will be returned.

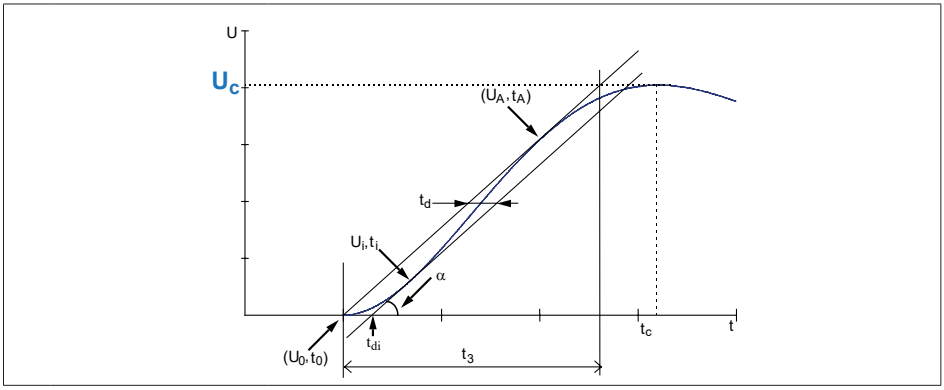


Fig. 2.8 STL2ParamTRV_Uc

STL documentation reference

§ 7.3.2. Two parameters TRV

2.19 @STL2ParamTRV_t3

Function

This function will be used to obtain the **t3** (rise time) of a Transient Recovery Voltage (TRV) using a 2 parameter calculation.

Syntax

@STL2ParamTRV_t3*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the t3 value of the TRV signal *"Waveform"* between *"StartPos"* and *"EndPos"*.

Description

This function uses the 2 parameter TRV method to determine t3.
The function starts searching at *"StartPos"* for a signal start, it uses the double threshold method but with a lower level (1%). If no TRV is found before *"EndPos"* a double.NaN value will be returned.

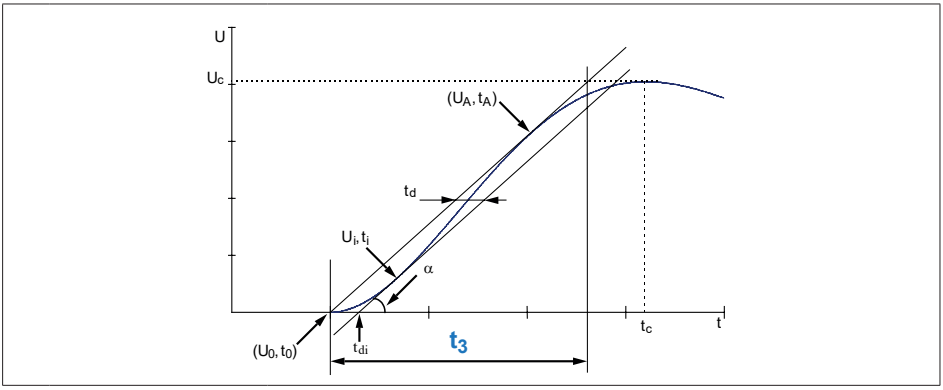


Fig. 2.9 STL2ParamTRV_t3

STL documentation reference

§ 7.3.2. Two parameters TRV

2.20 @STL2ParamTRV_td

Function

This function will be used to obtain the **td** (time delay) of a Transient Recovery Voltage (TRV) using a 2 parameter calculation.

Syntax

@STL2ParamTRV_td*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the td value of the TRV signal *"Waveform"* between *"StartPos"* and *"EndPos"*.

Description

This function uses the 2 parameter TRV method to determine td.
The function starts searching at *"StartPos"* for a signal start, it uses the double threshold method but with a lower level (1%). If no TRV is found before *"EndPos"* a double.NaN value will be returned.

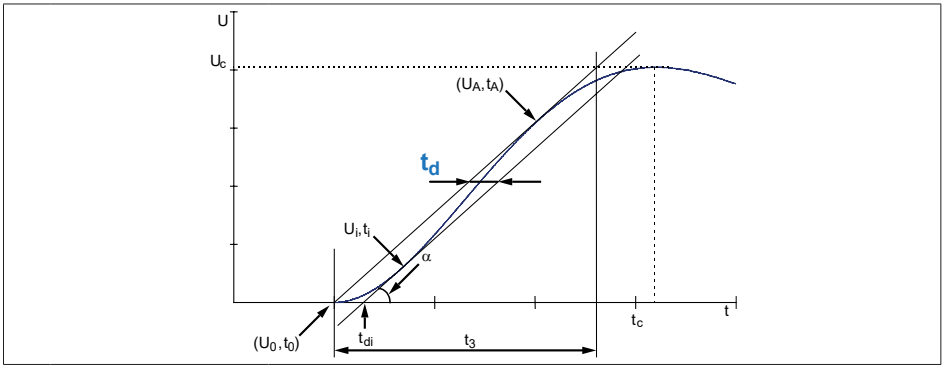


Fig. 2.10 STL2ParamTRV_tf

STL documentation reference

§ 7.3.2. Two parameters TRV

2.21 @STL4ParamTRV_Uc

Function

This function will be used to obtain the Uc of a Transient Recovery Voltage (TRV) using a 4 parameter calculation.

Syntax

@STL4ParamTRV_Uc*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the Uc value of the TRV signal *"Waveform"* between *"StartPos"* and *"EndPos"*.

Description

This function uses the 4 parameter TRV method to determine Uc.
The function starts searching at *"StartPos"* for a signal start, it uses the double threshold method but with a lower level (1%). If no TRV is found before *"EndPos"* a double.NaN value will be returned.

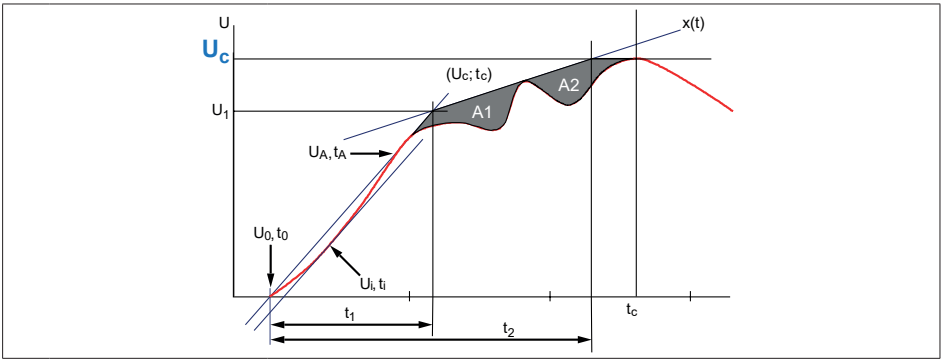


Fig. 2.11 STL4ParamTRV_Uc

STL documentation reference

§ 7.3.3. Four parameters TRV

2.22 @STL4ParamTRV_U1

Function

This function will be used to obtain the **U1** of a Transient Recovery Voltage (TRV) using a 4 parameter calculation.

Syntax

@STL4ParamTRV_Uc*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the U1 value of the TRV signal *"Waveform"* between *"StartPos"* and *"EndPos"*.

Description

This function uses the 4 parameter TRV method to determine U1.
The function starts searching at *"StartPos"* for a signal start, it uses the double threshold method but with a lower level (1%). If no TRV is found before *"EndPos"* a double.NaN value will be returned.

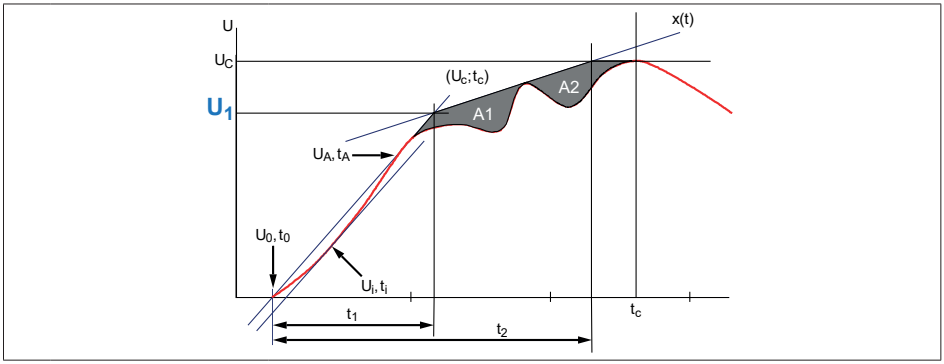


Fig. 2.12 STL4ParamTRV_U1

STL documentation reference

§ 7.3.3. Four parameters TRV

2.23 @STL4ParamTRV_td

Function

This function will be used to obtain the **td** of a Transient Recovery Voltage (TRV) using a 4 parameter calculation.

Syntax

@STL4ParamTRV_td*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the td value of the TRV signal *"Waveform"* between *"StartPos"* and *"EndPos"*.

Description

This function uses the 4 parameter TRV method to determine td.
The function starts searching at *"StartPos"* for a signal start, it uses the double threshold method but with a lower level (1%). If no TRV is found before *"EndPos"* a double.NaN value will be returned.

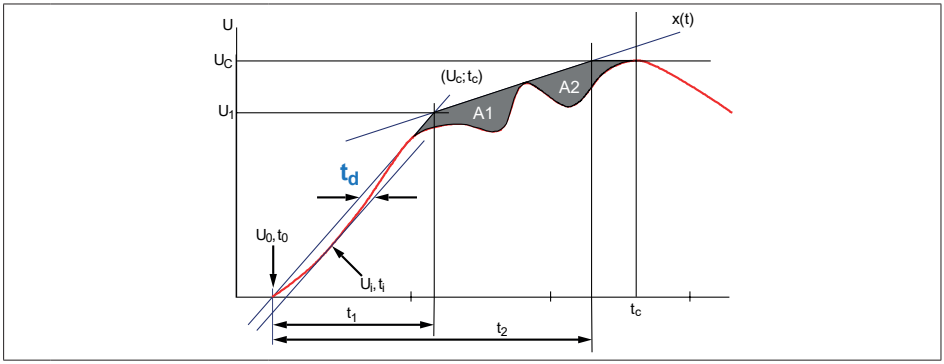


Fig. 2.13 STL4ParamTRV_td

STL documentation reference

§ 7.3.3. Four parameters TRV

2.24 @STL4ParamTRV_t1

Function

This function will be used to obtain the **t1** of a Transient Recovery Voltage (TRV) using a 4 parameter calculation.

Syntax

@STL4ParamTRV_t1*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the t1 value of the TRV signal *"Waveform"* between *"StartPos"* and *"EndPos"*.

Description

This function uses the 4 parameter TRV method to determine t1.
The function starts searching at *"StartPos"* for a signal start, it uses the double threshold method but with a lower level (1%). If no TRV is found before *"EndPos"* a double.NaN value will be returned.

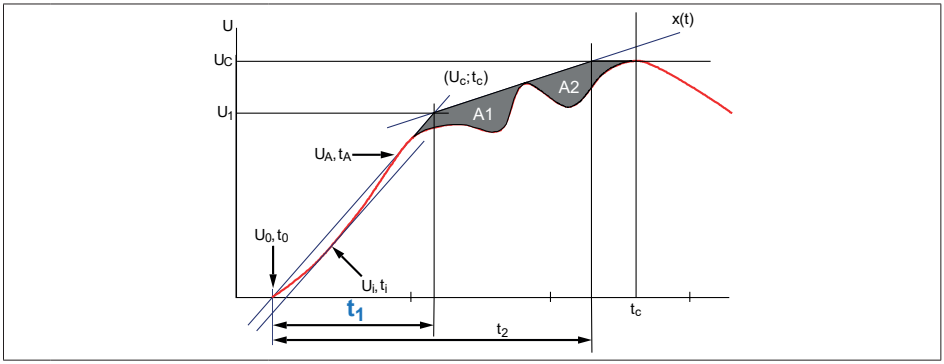


Fig. 2.14 STL4ParamTRV_t1

STL documentation reference

§ 7.3.3. Four parameters TRV

2.25 @STL4ParamTRV_t2

Function

This function will be used to obtain the **t2** of a Transient Recovery Voltage (TRV) using a 4 parameter calculation.

Syntax

@STL4ParamTRV_t2*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the t2 value of the TRV signal *"Waveform"* between *"StartPos"* and *"EndPos"*.

Description

This function uses the 4 parameter TRV method to determine t2.
The function starts searching at *"StartPos"* for a signal start, it uses the double threshold method but with a lower level (1%). If no TRV is found before *"EndPos"* a double.NaN value will be returned.

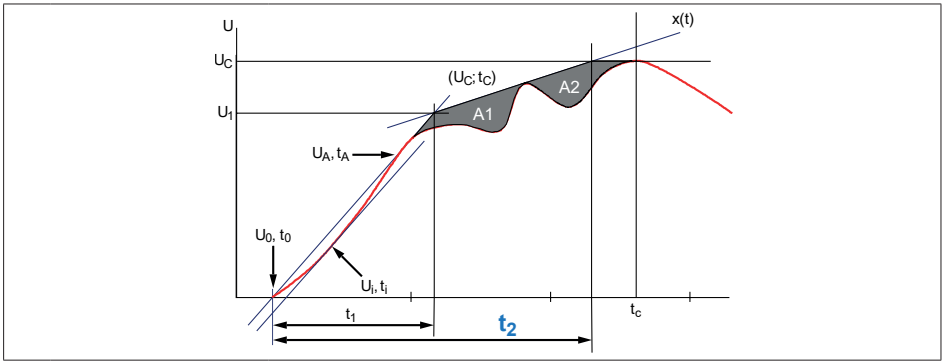


Fig. 2.15 STL4ParamTRV_t2

STL documentation reference

§ 7.3.3. Four parameters TRV

2.26 @STLOverVoltageVal

Function

This function will be used to obtain the overvoltage value of an input signal.

Syntax

@STLOverVoltageVal*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the value of the overvoltage of signal *"Waveform"* between *"StartPos"* and *"EndPos"*.

Description

Returns overvoltage value between *"StartPos"* and *"EndPos"*.
This function uses a sliding average of 3 consecutive points to calculate the mean value. The overvoltage value is the greatest mean value found around an expected maximum. If the signal is negative then the minimum value of this sliding average will be returned.

STL documentation reference

§ 7.4. Evaluation of overvoltages

2.27 @STLOverVoltageTime

Function

This function will be used to obtain the overvoltage time position of an input signal.

Syntax

@STLOverVoltageTime*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the time position of the overvoltage of signal *"Waveform"* between *"StartPos"* and *"EndPos"*.

Description

Returns overvoltage value between *"StartPos"* and *"EndPos"*.
It uses a sliding average of 3 consecutive points. If the signal is negative then the position of the minimum value of this sliding average will be returned.

STL documentation reference

§ 7.4. Evaluation of overvoltages

2.28 @STL3CrestDC

Function

This function will be used to return the percentage value of the d.c. component of an asymmetrical current.

Syntax

@STL3CrestDC*(Waveform; DCTime; StartPos; EndPos)*

Parameters

Waveform	Input waveform
DCTime	Time where DC percentage will be calculated
StartPos	Optional; start time of search for crests
EndPos	Optional; end time of search for crests

Output

A Numerical value indicating the percentage of the d.c. component compared to the AC signal.

Description

This function searches tree crests around the *"DCTime"*, these crests are used to construct two lines, f(t) and g(t). At time *"DCTime"* the points N and M are calculated, those points are located at the lines f(t) and g(t).

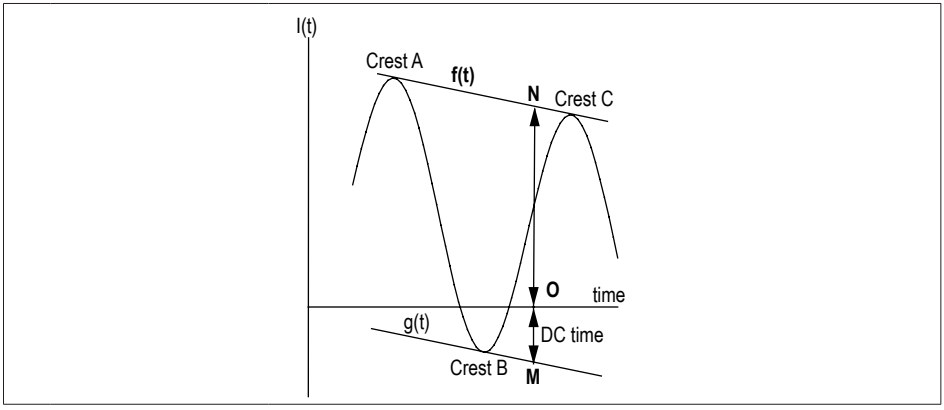


Fig. 2.16 STL3CrestDC

The next formula is used to get the percentage of the d.c. component:

```
D.C.-component = (ON-OM)/MN 100 %
```

The Crests have to be positioned between the *"StartPos"* and *"EndPos"*.

The DC component in the current signal is only available after the switching moment, the moment that the current has started. It is therefore only useful to have a DCTime which is between the **Start of current** and **End of current**, see (Fig. 2.17).

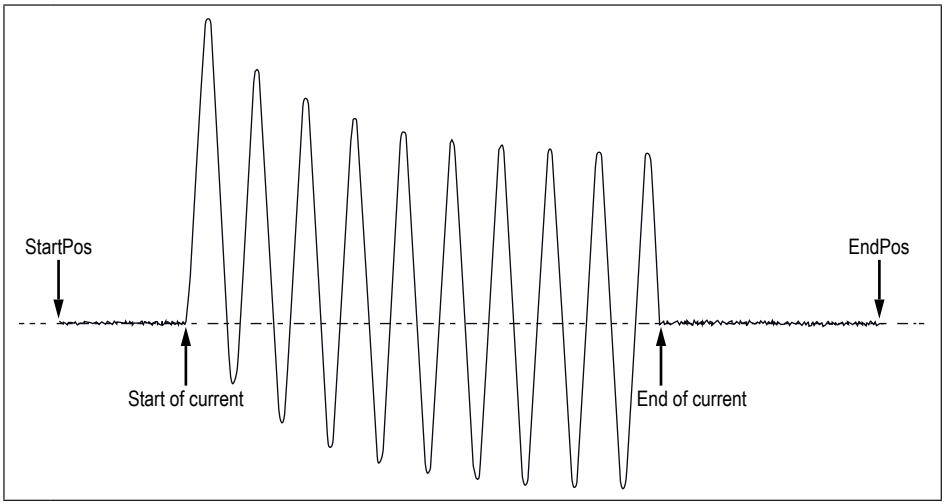


Fig. 2.17 DC component

STL documentation reference

§ 8.4.2 Percentage value of d.c. component

2.29 @STLExpCrestDC

Function

This function will return the time constant τ (Tau) of the exponential d.c. component of an asymmetrical current.

Syntax

@STLExpCrestDC*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the time constant of the exponential d.c. component of signal *"Waveform"* between *"StartPos"* and *"EndPos"*.

Description

This function uses all crests between *"StartPos"* and *"EndPos"* to calculate the exponential curve of the d.c. component. The input waveform contains an asymmetrical current.

Crest A and B are used to get point a,
Crest B and C are used to get point b,
Crest C and D are used to get point c,
Etc.

An exponential curve fitting will be done at the points a, b, c, etc.

The resulting curve will have the following formula:

$$DC(t) = ae^{-\frac{(t-t_0)}{\tau}} + C$$

τ = Exponential time constant

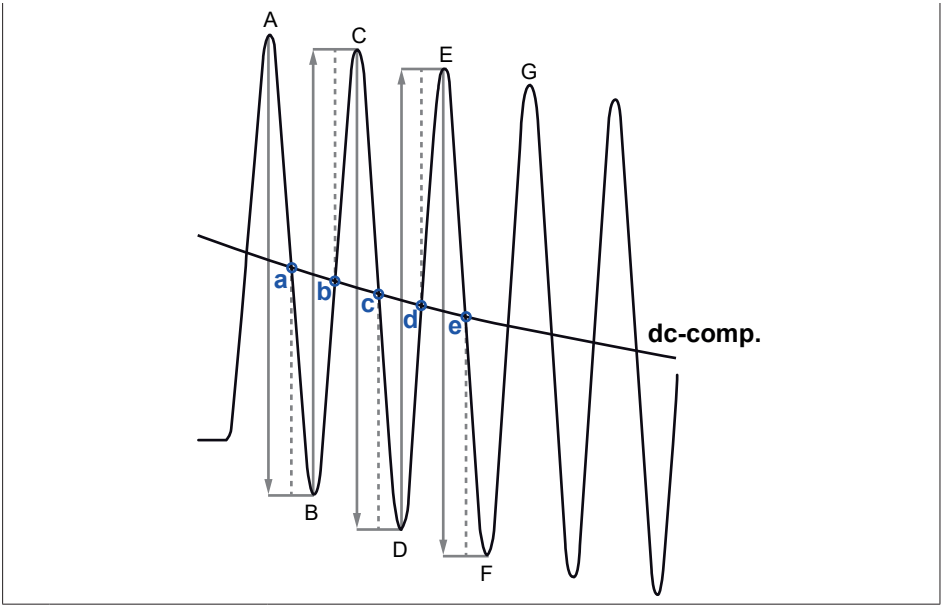


Fig. 2.18 STLExpCrestDC

STL documentation reference

§ 8.4.1 Evaluation of the percentage value of d.c. component

2.30 @STLExpDelayCrestDC

Function

This function will return the time delay constant t_0 of the exponential d.c. component of an asymmetrical current.

Syntax

@STLExpDelayCrestDC*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the time delay constant of the exponential d.c. component of signal *"Waveform"* between *"StartPos"* and *"EndPos"*.

Description

This function uses all crests between *"StartPos"* and *"EndPos"* to calculate the exponential curve of the d.c. component, see description of the function **STLExpCrestDC**.
The input waveform contains an asymmetrical current signal.

The DC component has the following formula:

$$DC(t) = ae^{-\frac{(t-t_0)}{\tau}} + C$$

$\mathbf{t_0}$ = Exponential time delay

STL documentation reference

§ 8.4.1 Evaluation of the percentage value of d.c. component

2.31 @STLExpFactorCrestDC

Function

This function will return the multiplication factor α of the exponential d.c. component of an asymmetrical current.

Syntax

@STLExpFactorCrestDC*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the time delay constant of the exponential d.c. component of signal *"Waveform"* between *"StartPos"* and *"EndPos"*.

Description

This function uses all crests between *"StartPos"* and *"EndPos"* to calculate the exponential curve of the d.c. component, see description of the function **STLExpCrestDC**.
The input waveform contains an asymmetrical current signal.

The DC component has the following formula:

$$DC(t) = ae^{-\frac{(t-t_0)}{\tau}} + C$$

Alpha = Multiplication Factor

STL documentation reference

§ 8.4.1 Evaluation of the percentage value of d.c. component

2.32 @STLExpOffsetCrestDC

Function

This function will return the offset of the exponential d.c. component of an asymmetrical current.

Syntax

@STLExpOffsetCrestDC*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the offset of the exponential d.c. component of signal *"Waveform"* between *"StartPos"* and *"EndPos"*.

Description

This function uses all crests between *"StartPos"* and *"EndPos"* to calculate the exponential curve of the d.c. component, see description of the function **STLExpCrestDC**.
The input waveform contains an asymmetrical current signal.

The DC component has the following formula:

$$DC(t) = ae^{-\frac{(t-t_0)}{\tau}} + C$$

C = Offset

STL documentation reference

§ 8.4.1 Evaluation of the percentage value of d.c. component

2.33 @STL_STCValue

Function

This function calculates the RMS value of a STC (Short Time Current) signal.

Syntax

@STL_STCValue*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating RMS value of a STC signal.

Description

Returns the RMS value of a STC signal. Data between *"StartPos"* and *"EndPos"* will be used. The function uses the 3 crest method to obtain 11 RMS values (Z_0,Z_1..Z_{10}). The returned rms value is the a weighted average of those 11 values.

The following formula is used:

$$I_t = \sqrt{\frac{1}{30}\left[z_0^2 + 4\left(z_1^2 + z_3^2 + z_5^2 + z_7^2 + z_9^2\right) + 2\left(z_2^2 + z_4^2 + z_6^2 + z_8^2\right) + z_{10}^2\right]}$$

Where:

For Z_0	The first 3 crests are used.
For Z_{10}	The last crest will be omitted and the 3 previous crests are used.
For Z_1 to Z_9	The used crests are equally spaced per 3 between the first and last used crests.

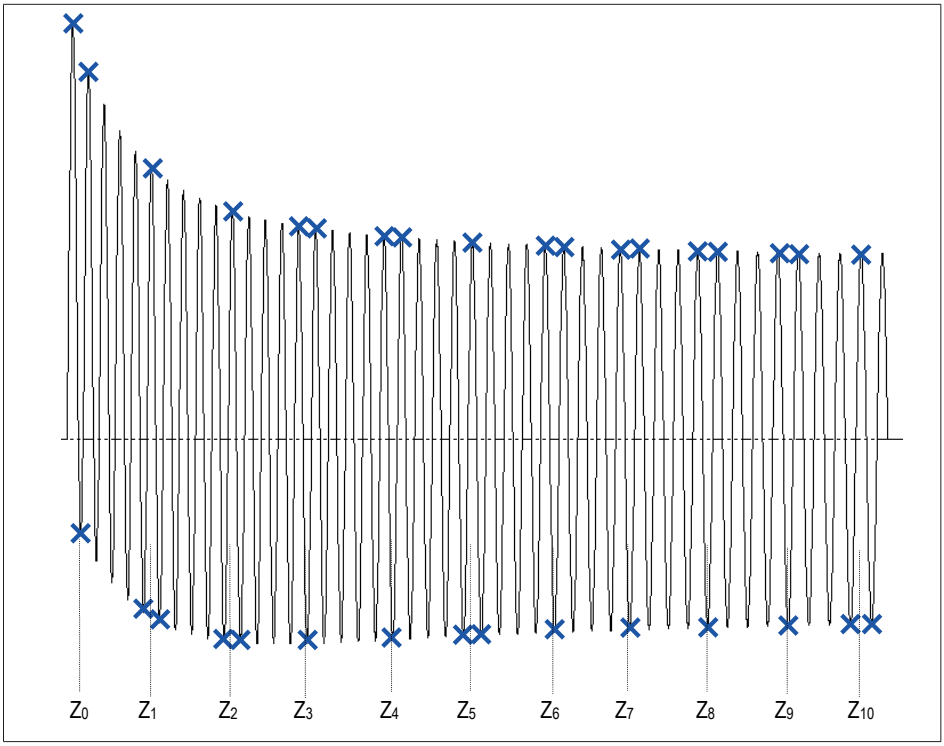


Fig. 2.19 STL_STCValue

STL documentation reference

§ 8.1.1 Short-time current tests

2.34 @STL_ShorterSTCValue

Function

This function calculates the RMS value of a shorter STC (Short Time Current) signal.

Syntax

@STL_ShorterSTCValue*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating RMS value of a shorter STC signal.

Description

Returns the RMS value of a shorter STC signal. Data between *"StartPos"* and *"EndPos"* will be used. The first and last crest will be omitted.

The function uses the 3 crest method to obtain RMS values (Z_0,Z_1..Z_n). The returned RMS value is the average of those n values.
The number n depends on the number of available crests.

For Z_0	Crests 2,3 and 4 are used
For Z_1 to Z_n	Each RMS value is obtained by a sliding of one crest to the previous one, the last crest will be omitted

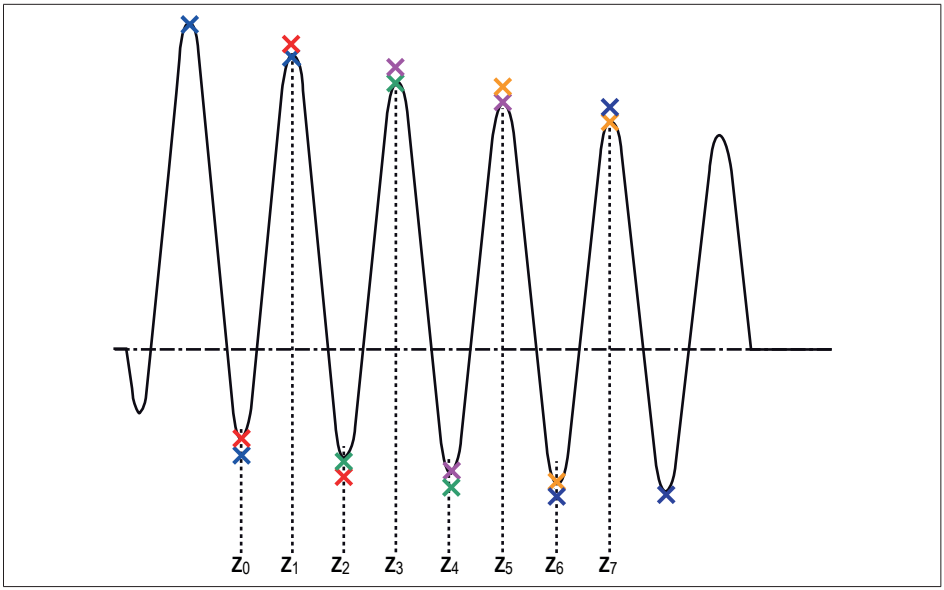


Fig. 2.20 STL_ShorterSTCValue

STL documentation reference

§ 8.1.2 Shorter short-time current tests

2.35 @STL_STCDuration

Function

This function will be used to obtain the duration of a STC (Short Time Current) signal.

Syntax

@STL_STCDuration*(Waveform; Frequency; StartPos; EndPos)*

Parameters

Waveform	STC input waveform
Frequency	*Frequency of the sinusoidal waveform*
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the duration in seconds of the STC signal between *"StartPos"* and *"EndPos"*.

Description

This function uses the STLSignalStart and STLSignalEnd to determine the start and end of the STC signal. The duration is the difference between those two values.

STL documentation reference

§ 8.1.1. Short-time current tests

2.36 @STL_ShorterSTCDuration

Function

This function will be used to obtain the duration of a STC (Short Time Current) signal.

Syntax

@STL_ShorterSTCDuration*(Waveform; Frequency; StartPos; EndPos)*

Parameters

Waveform	STC input waveform
Frequency	*Frequency of the sinusoidal waveform*
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the duration in seconds of the STC signal between *"StartPos"* and *"EndPos"*.

Description

This function is identical to STL_STCDuration.

STL documentation reference

§ 8.1.2. Shorter short-time current tests

2.37 @STLReadTestData

Function

This function is used to read test data generated by the TDG software program.

Syntax

@STLReadTestData*(Filename)*

Parameters

Filename	The filename of the ASCII file containing the TDG generated test data e.g. “C:\temp\Curve1.txt”

Output

A waveform of the input data. This waveform can be dropped into a Display.

Description

This function is used to import the signals generated by the Test Data Generator (TDG).
The imported data can now be shown in a display and it can also be used in the formulae database as input waveform for other STL functions.

STL documentation reference

§ 11 Software validation

2.38 @STLNoLoadClose

Function

This function calculates the moment of contact touch of a no-load signal.

Syntax

@STLNoLoadClose*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A numerical value indicating the time of contact touch (closing) of the input signal.

Description

Returns the time position of contact touch. The data between *"StartPos"* and *"EndPos"* will be used.

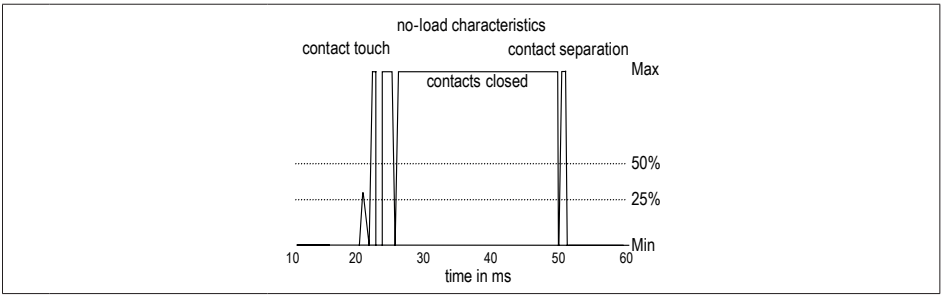


Fig. 2.21 STLNoLoadClose

The function starts detecting the maximum and minimum value of the input signal. From these two extremes two levels are derived, a 50% level and a 25% level. For a closing operation the signal level should be below the 25% level for at least 100 µs if this point is found then the function searches for the signal level >= 50% for at least 100 µs. The time of this last point will be returned as the contact touch time.

STL documentation reference

§ 9.1 Determination of no-load characteristics

2.39 @STLNoLoadOpen

Function

This function calculates the moment of contact separation of a no-load signal.

Syntax

@STLNoLoadOpen*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A numerical value indicating the time of contact separation (opening) of the input signal.

Description

Returns the time position of contact separation. The data between *"StartPos"* and *"EndPos"* will be used.

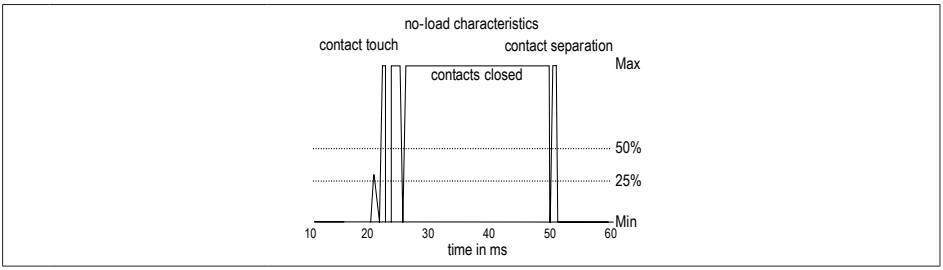


Fig. 2.22 STLNoLoadClose

The function starts detecting the maximum and minimum value of the input signal. From these two extremes two levels are derived, a 50% level and a 75% level. For an opening operation the signal level should be above the 75% level for at least 100 µs if this point is found then the function searches for the signal level <= 50% for at least 100 µs. The time of this last point will be returned as the separation time.

STL documentation reference

§ 9.1 Determination of no-load characteristics

2.40 @STLContactSpeed

Function

This function calculates the contact speed at a given time.

Syntax

@STLContactSpeed*(Waveform; Time)*

Parameters

Waveform	Input waveform
Time	Time where contact speed has to be calculated

Output

A numerical value indicating the contact speed at a given time, the dimension depends on the units of the Y-axis of the input signal and will be most likely “mm/s”.

Description

Returns the contact speed in mm/s at a given time.

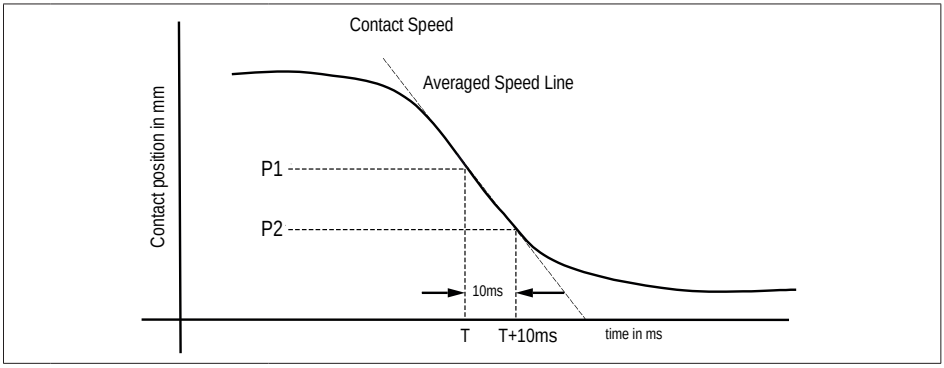


Fig. 2.23 STLContactSpeed

A linear curve fit algorithm using the points between T and T + 10 ms is used to get the average speed line. The slope of this line is the contact speed and will be returned by this function.

STL documentation reference

§ 9.1 Determination of no-load characteristics

2.41 @STLXRescale

Function

This function rescales a waveform, it modifies the display scaling which is used for calculating threshold levels.

Syntax

@STLXRescale*(Waveform; Yupper, YLower)*

Parameters

Waveform	Input waveform
YUpper	*New upper scale value*
YLower	New lower scale value

Output

The out put is a waveform with a modified display scaling (or Full Scale level).

Description

A waveform has two display scaling properties (display from and display to), these properties are used by the display to show initial vertical scaling values, or will be used after restoring the vertical scaling (key combination CTRL +NUM/). During the recording phase the display scaling properties of a waveform are set to the Full Scale level of the amplifier.
The *STLRescale* function can be used to modify these display scaling properties. By doing this, the threshold levels used for finding start and end of a signal (see "@STLSignalStart“ on page 8) are automatically modified as well, because the STL functions use the display scaling as being the Full Scale level.

The following pictures (Fig. 2.24 and Fig. 2.25) show the influence on the threshold level of the *STLRescale* function. The first picture shows a rescale to 10 kA and the next picture one to 4 kA. We see that the threshold level of the first picture is 600 A while the second one has a threshold level of 240 A. This rescale function can be used when a signal has been recorded with a way too high amplifier gain setting.

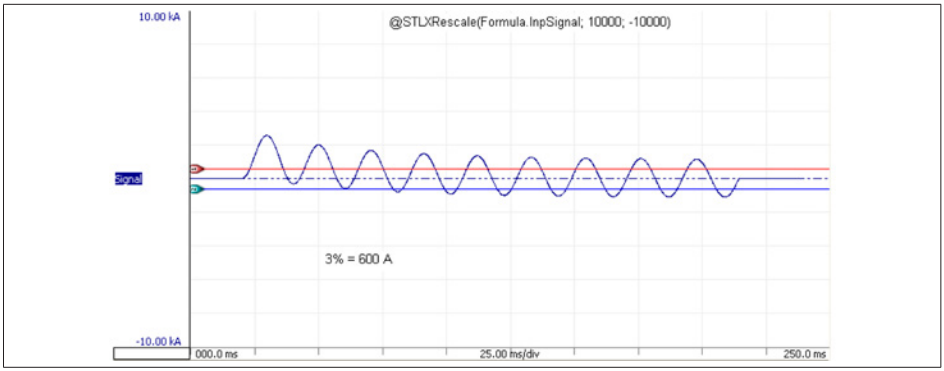


Fig. 2.24 STLXRescale (Part 1)

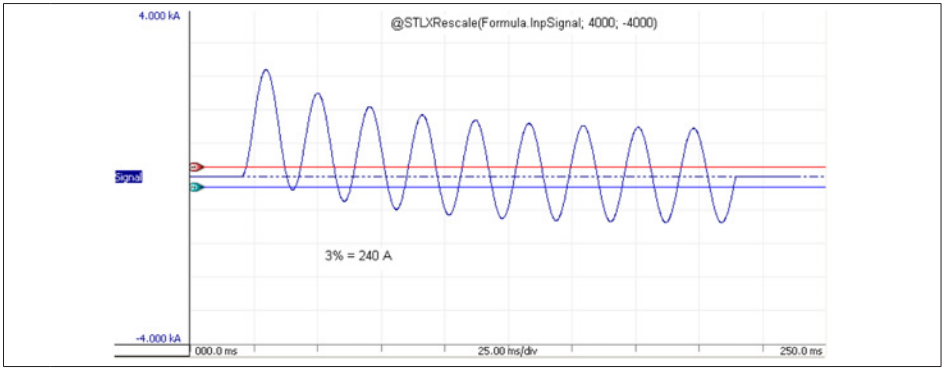


Fig. 2.25 STLXRescale (Part 2)

STL documentation reference

With this function the 3% threshold levels can be modified. The STL routines are using the waveforms “Display scaling” which is equal to the Full Scale level as mentioned in "@STLNextZeroCrossing“ on page 12 of the STL document.

2.42 @STLX_FirstValidCrestSignalStart

Function

This function will be used to find the first valid start location after doing a check on asymmetry and/or crest location. Depending on the parameters and the outcome of the checks crests can be skipped. The return value is the time location before the first found correct crest; this can either be a zero crossing or the beginning of the signal.

Syntax

@STLX_FirstValidCrestSignalStart*(Waveform Current; Number Frequency; Number Asymmetry Check; Number Crest Location Check; Number StartPos; Number EndPos)*

Parameters

Waveform	Sinusoidal input (current) waveform
Frequency	Optional; Frequency of the sinusoidal waveform, default 50 Hz.
Asymmetry Check	Optional; Enable 7% asymmetry check. The check is disabled when the Asymmetry Check is set to 0. If set to 1 the check is enabled. Default the check is enabled.
Crest Location Check	Optional; Enable crest location check; crest location should be larger than 75% of ¼ cycle. The check is disabled when the Crest Location Check is set to 0. If set to 1 the check is enabled. Default the check is enabled.
StartPos	Optional; Start time from where the first crest in the input signal will be searched. This parameter is optional if not entered the lowest possible time value is used.
EndPos	Optional; all search activities are limited to this end time. This parameter is optional if not entered the highest possible time value is used.

Output

The output is a new signal start position.

Description

This function determines a new start position. Most likely this new position will be used as the start position parameter for the power factor function **(@STLX_SymmetricalPowerFactor())**.

There is a check for asymmetry and a check for minimal crest location. Depending on these checks a new start time will be returned. This new start time will be the first zero crossing before a crest which succeeded against one of these checks, or the beginning of the input (current) signal.

- **Asymmetry Check:**

The asymmetry is the difference in% between the deflection below and above the first full cycle after the entered start position, related to the smaller deflection. If the difference is larger than 7% then the crest will be skipped and the first zero crossing after this crest will be used as return value.

But before doing this a new asymmetry check will be done on the new next crest, this check will be repeated as long as the asymmetry is above 7%. For more information about the Asymmetry calculations, please refer to the **@STL_PF_Asymmetry()** function (on page 71).

- **Crest Location Check:**

This check checks if the crest location is not too small compared to the expected frequency. The crest location should be at least 75% of ¼ cycle.

When looking at Figure 2.26 below this means that:

$$T_{Crest} > 0.75 * T_{Cylce}/4$$

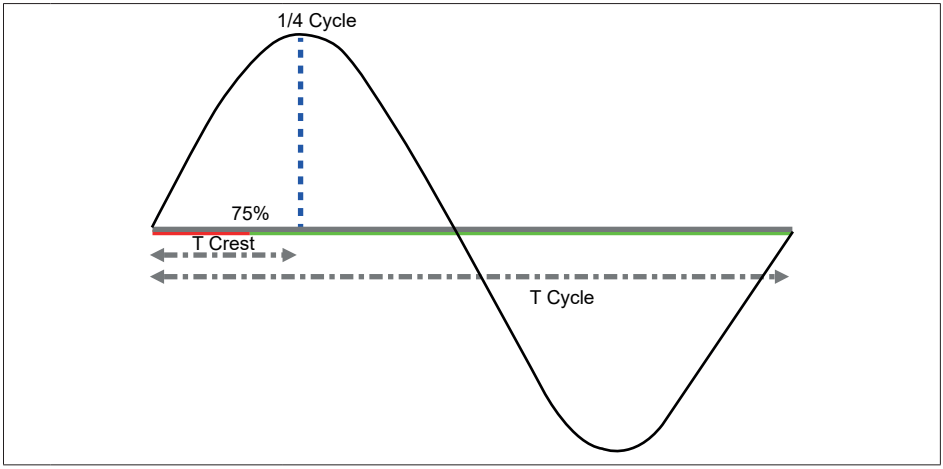
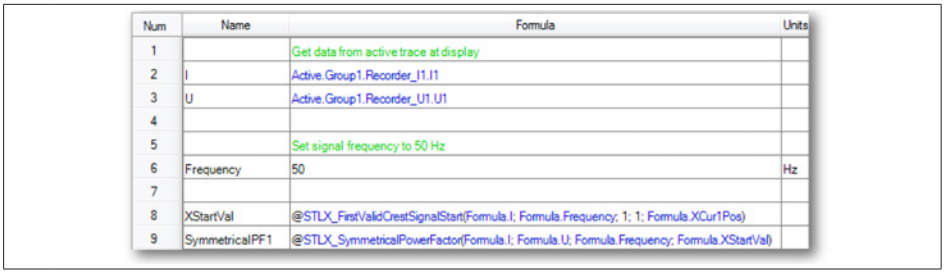


Fig. 2.26 Crest location check

Example
The following shows how you can use the **@STLX_FirstValidCrestSignalStart ()** function in combination with the **@STLX_SymmetricalPowerFactor()** function.



Num	Name	Formula	Units
1		Get data from active trace at display	
2	I	Active.Group1.Recorder_I1.I1	
3	U	Active.Group1.Recorder_U1.U1	
4			
5		Set signal frequency to 50 Hz	
6	Frequency	50	Hz
7			
8	XStartVal	@STLX_FirstValidCrestSignalStart(Formula.I; Formula.Frequency; 1; 1; Formula.XCur1Pos)	
9	SymmetricalPF1	@STLX_SymmetricalPowerFactor(Formula.I; Formula.U; Formula.Frequency; Formula.XStartVal)	

Fig. 2.27 @STLX_FirstValidCrestSignalStart() combined with @STLX_SymmetricalPowerFactor() function

2.43 @STLX_SymmetricalPowerFactor

Function

This function calculates the power factor for symmetric currents less or equal to 10 kA.

Syntax

@STLX_SymmetricalPowerFactor*(Waveform Current; Waveform Voltage; Frequency; StartPos; EndPos; Waveform Timing)*

Parameters

Waveform Current	Input current waveform
Waveform Voltage	Input voltage waveform
Frequency	Optional; Frequency of the sinusoidal waveform. This parameter is optional if it has not been entered 50 Hz will be used.
StartPos	Optional; Start time from where the first crest in the current signal will be searched. This parameter is optional if not entered the lowest possible time value is used.
EndPos	Optional; all search activities are limited to this end time. This parameter is optional if not entered the highest possible time value is used.
Waveform Timing	Optional; Voltage signal used for timing purposes. This parameter is optional if it is not available then the last complete cycle of the voltage signal before the start of current is used.

Output

The output is numerical value indicating the power factor expressed in percentage.

Description

The Power Factor is defined as the cos(φ) where φ is the phase shift between the current and the voltage signals.

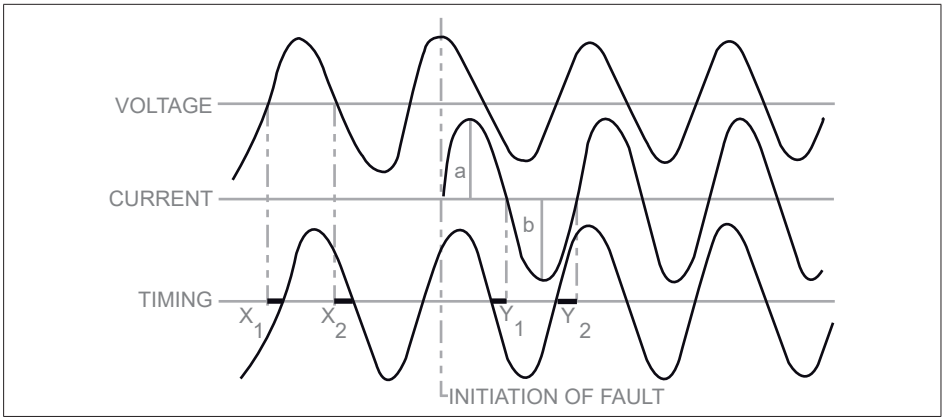


Fig. 2.28 STLX_SymmetricalPowerFactor (Part 1)

$$Power\ factor = \frac{\cos[(Y_1 + X_1 \times 180°]}{2} + \frac{\cos[(Y_2 + X_2 \times 180°]}{2}$$

In which:

- X and Y values are fractions of ½ cycle distance in which they occur.

When there is no timing signal available then a sinusoidal signal will be constructed using the last and second last zero crossings (in the same direction) of the voltage signal before the start of current. This constructed signal will be used to calculate the phase shifts Y_1 and Y_2, in this case X_1 and X_2 will become zero.

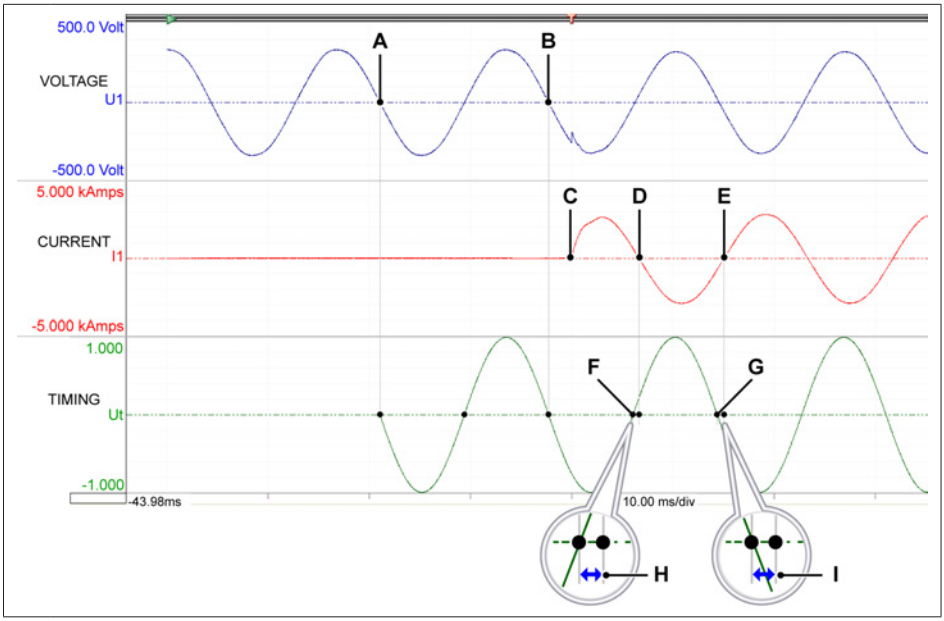


Fig. 2.29 STLX_SymetricalPowerFactor (Part 2)

A Second last zero crossing U1
B Last zero crossing U1
C Start of current
D First zero I1
E Second zero I1
F First zero UT
G Second zero UT
H Y1
I Y2

The power factor can only be calculated if the asymmetry of the current signal is no greater than 7%. The asymmetry is the difference in % between the deflection below and above the zero trace in the first full cycle related to the smaller deflection. There is a separate function to calculate this asymmetry this function is called: **STLX_PF_Asymmetry.**

When however the asymmetry is larger than 7% the optional *Startpos* should be used. The function searches for the first crest after the *Startpos* time, than it searches for the first zero crossing after this crest. Than it searches for the second crest after this zero crossing. This latest crest will be used to find the second zero crossing of the current signal.

Documentation reference

The implementation is according to the document UL 489 (ISBN 0-7629-0677-4) Appendix C3.2.

2.44 @STLX_SymmetricalPowerFactorNoAsymmetry

Function

This function calculates the power factor for symmetric currents less or equal to 10 kA without doing a 7% asymmetry check.

Syntax

@STLX_SymmetricalPowerFactorNoAsymmetry*(Waveform Current; Waveform Voltage; Frequency; StartPos; EndPos; Waveform Timing)*

Parameters

Waveform Current	Input current waveform
Waveform Voltage	Input voltage waveform
Frequency	Optional; Frequency of the sinusoidal waveform. This parameter is optional if it has not been entered 50 Hz will be used.
StartPos	Optional; Start time from where the first crest in the current signal will be searched. This parameter is optional if not entered the lowest possible time value is used.
EndPos	Optional; all search activities are limited to this end time. This parameter is optional if not entered the highest possible time value is used.
Waveform Timing	Optional; Voltage signal used for timing purposes. This parameter is optional if it is not available then the last complete cycle of the voltage signal before the start of current is used.

Output

The output is numerical value indicating the power factor expressed in percentage.

Description

This function is similar to the function @STLX_SymmetricalPowerFactor, however this function skips the 7% asymmetry check.

Documentation reference

The implementation is according to the document UL 489 (ISBN 0-7629-0677-4) Appendix C3.2.

2.45 @STLX_SymmetricalPowerFactorNoAsymmetryCheck

Function

This function calculates the power factor for symmetric currents less or equal to 10 kA without doing a 7% asymmetry test.

Syntax

@STLX_SymmetricalPowerFactor*(Waveform Current; Waveform Voltage; Frequency; StartPos; EndPos; Waveform Timing)*

Parameters

Waveform Current	Input current waveform
Waveform Voltage	Input voltage waveform
Frequency	Optional; Frequency of the sinusoidal waveform. This parameter is optional if it has not been entered 50 Hz will be used.
StartPos	Optional; Start time from where the first crest in the current signal will be searched. This parameter is optional if not entered the lowest possible time value is used.
EndPos	Optional; all search activities are limited to this end time. This parameter is optional if not entered the highest possible time value is used.
Waveform Timing	Optional; Voltage signal used for timing purposes. This parameter is optional if it is not available then the last complete cycle of the voltage signal before the start of current is used.

Output

The output is numerical value indicating the power factor expressed in percentage.

Description

The Power Factor is defined as the cos(φ) where φ is the phase shift between the current and the voltage signals. More detailed information can be found at the description of the function **@STLX_SymmetricalPowerFactor()** (see "@STLX_SymmetricalPowerFactor“ on page 64).

The calculations of this function are exactly the same as the STLX_SymmetricalPowerFactor() function; the only difference is the fact that the 7% asymmetrical check is not done.

Documentation reference

The implementation is according to the document UL 489 (ISBN 0-7629-0677-4) Appendix C3.2.

2.46 @STLX_PF_Asymmetry

Function

This function calculates the asymmetry in percentage of the sinusoidal input waveform.

Syntax

@STLX_PF_Asymmetry*(Waveform; Frequency; StartPos; EndPos)*

Parameters

Waveform	Input waveform
Frequency	Optional; Frequency of the sinusoidal waveform. This parameter is optional if it has not been entered 50 Hz will be used.
StartPos	Optional; Start time from where the first crest in the input signal will be searched. This parameter is optional if not entered the lowest possible time value is used.
EndPos	Optional; all search activities are limited to this end time. This parameter is optional if not entered the highest possible time value is used.

Output

The output is numerical value indicating the asymmetry of the input signal.

Description

This function returns a numerical value indicating the asymmetry in percentage for a sinusoidal signal. The asymmetry is the difference in % between the deflection below and above the first full cycle after the entered start position, related to the smaller deflection.
This function can be used to check if the asymmetry of a low voltage current is lower or equal to 7%. When this is true, it is possible to calculate the power factor of this current signal, see function *STLX_SymmetricalPowerFactor*.

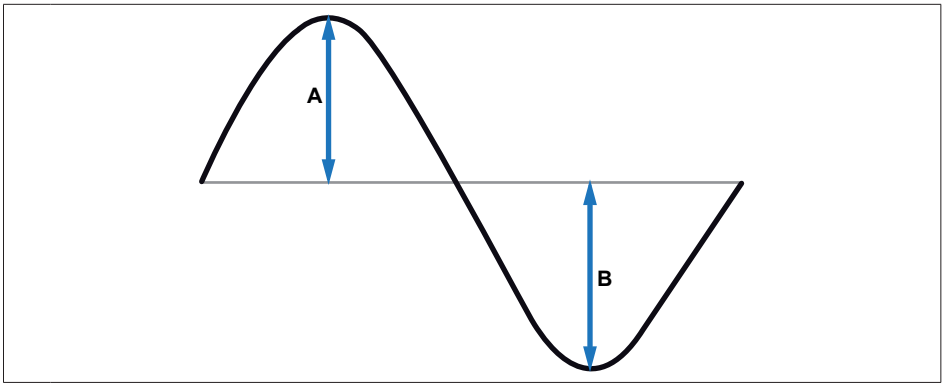


Fig. 2.30 STLX_PF_Asymmetry

If A > B than:

$$Asymmetry = 100 \times \frac{A-B}{B}$$

else:

$$Asymmetry = 100 \times \frac{B-A}{B}$$

Documentation reference

The implementation is according to the document UL 489 (ISBN 0-7629-0677-4) Appendix C3.2.

2.47 @STLX_PF_Crests

Function

This function returns the time position of either the first or second crest of the symmetrical current as used by the symmetrical powerfactor calculations.

Syntax

@STLX_PF_Crests*(Waveform;Crest; Frequency; StartPos; EndPos)*

Parameters

Waveform	Input waveform
Crest	Optional; Indicates selected crest: • 1 Return the time of first crest (default). • 2 Return the time of the second crest.
Frequency	Optional; Frequency of the sinusoidal waveform. This parameter is optional if it has not been entered 50 Hz will be used.
StartPos	Optional; Start time from where the first crest in the input signal will be searched. This parameter is optional if not entered the lowest possible time value is used.
EndPos	Optional; all search activities are limited to this end time. This parameter is optional if not entered the highest possible time value is used.

Output

The output is numerical value either indicating the time position of the first or second crest of a symmetrical current as used by the symmetrical powerfactor calculations.

Description

This function returns the time position of the first **(a)** or second **(b)** crest after the entered start time of a symmetrical current.

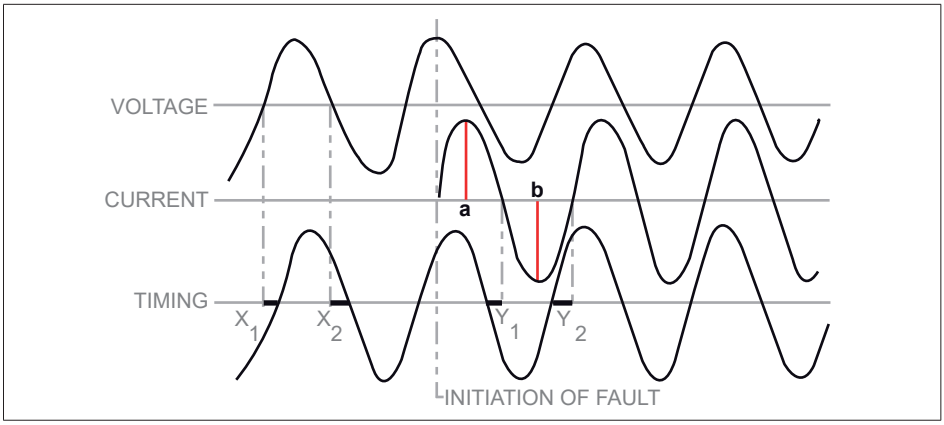


Fig. 2.31 STL_PF_Crests

Documentation reference

The implementation is according to the document UL 489 (ISBN 0-7629-0677-4) Appendix C3.2.

2.48 @STLX_PF_ZeroCrossings

Function

This function returns the time position of either the first, second or third zero crossing location of the symmetrical current as used by the symmetrical power-factor calculations.

Syntax

@STLX_PF_ZeroCrossings*(Waveform Current; Zero crossing; Frequency; StartPos; EndPos)*

Parameters

Waveform	Sinusoidal input (current) waveform
Zero crossing	Optional; Indicates selected zero crossing 1. Return time of the first zero crossing (default) 2. Return time of the second zero crossing 3. Return time of the third zero crossing
Frequency	Optional; Frequency of the sinusoidal waveform, default 50 Hz.
StartPos	Optional; Start time from where the first crest in the input signal will be searched. This parameter is optional if not entered the lowest possible time value is used.
EndPos	Optional; all search activities are limited to this end time. This parameter is optional if not entered the highest possible time value is used.

Output

The output is numerical value either indicating the time position of the first, second or third zero crossing location of a symmetrical current as used by the symmetrical power-factor calculations.

Description

This function returns the time position of the first (Z_1), second (Z_2) or third (Z_3) zero crossing after the entered start time of a symmetrical current

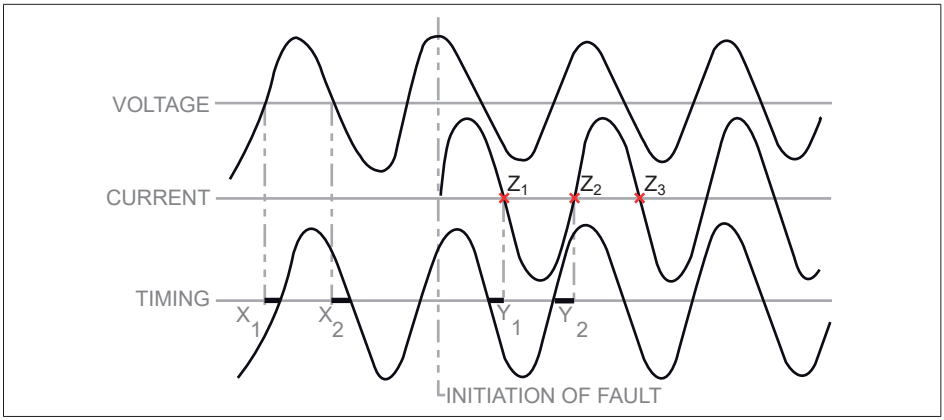


Fig. 2.32 STL_PF_Crests

Documentation reference

None

2.49 @STLX_PF_Frequency

Function

This function returns a value of the waveform indicating the calculated frequency.

Syntax

@STLX_PF_Frequency*(Waveform; Start; InitialCrest; UsedCrests)*

Parameters

Waveform	Sinusoidal current input signal.
Start	Start position is in x-units where searching should start.
InitialCrest	Optional; the first crest from the start to be used, the default value is 1.
UsedCrests	Optional; the number of crests used to calculate the frequency, the default value is 3.

Output

A numerical value of the frequency calculated over specific parameters.

Description

This function returns a number which is the calculated frequency of a sinusoidal signal. The function uses the interval between two specified crests to calculate the frequency. The interval is defined by two parameters, the first parameter defines the first crest and the second parameter defines how many crests should be used. The example below explains how this works.

Example

@STLX_PF_Frequency(Formula.I1; Formula.T0; 5; 3)
This function uses the interval between the crest 5 and 7 to calculate the frequency.

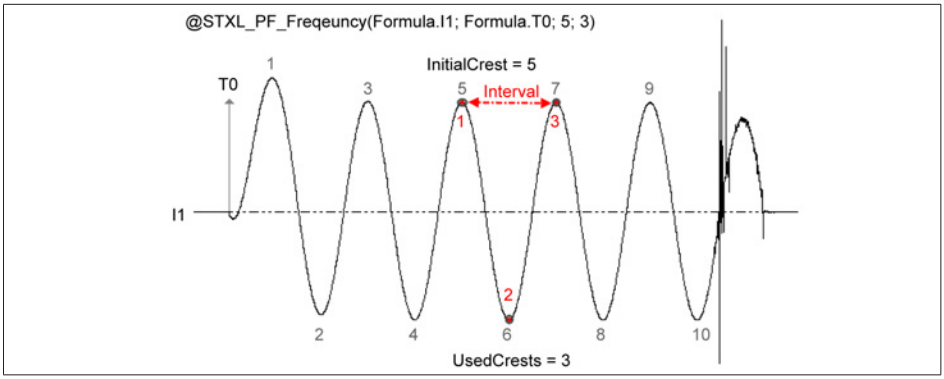


Fig. 2.33 STLX_PF_Frequency

Documentation reference

None

2.50 @STLX_DC_ExpEnvelope

Function

This function returns a waveform being the exponential envelopes on asymmetrical currents signals.

Syntax

@STLX_DC_ExpEnvelope*(Waveform; Method; StartIntv; EndIntv; Start; End)*

Parameters

Waveform	Input asymmetrical current waveform.
Method	Optional; defines the crests used for the curve fit. • 1 Only positive (or top) crests are used (default). • -1 Only negative (or bottom) crests are used. This parameter is optional if not available the default value will be used.
StartIntv	Optional; the beginning of the interval for searching of the crests, this parameter is optional and if it is not available then the beginning of the signal will be used.
EndIntv	Optional; the end of the interval for searching of the crests, this parameter is optional and if it is not available the end of the signal will be used.
Start	Optional; the start time of the DC output curve. By default the start of current is used.
End	Optional; the end time of the DC output curve. By default the position of the 8th crest is used.

Output

The output is a waveform containing the exponential envelope of the asymmetrical current input signal.

Description

Depending on the *Method* either positive or negative crests are used to create an exponential envelope. An exponential curve fitting method is used to do this. The algorithm uses a maximum of 4 successive crests, the first crest is the first crest found after the *StartIntv* time. If however this crest is a minor crest and the other successive crests are monotonically in- or decreasing then the first crest will be skipped automatically. By default the *StartIntv* time is the start of the current signal.

When the *Start* and *End* parameters are not entered then the envelope starts at the current start and ends after the 8th crest.
Fig. 2.34 shows an example signal and its top and bottom exponential envelope.

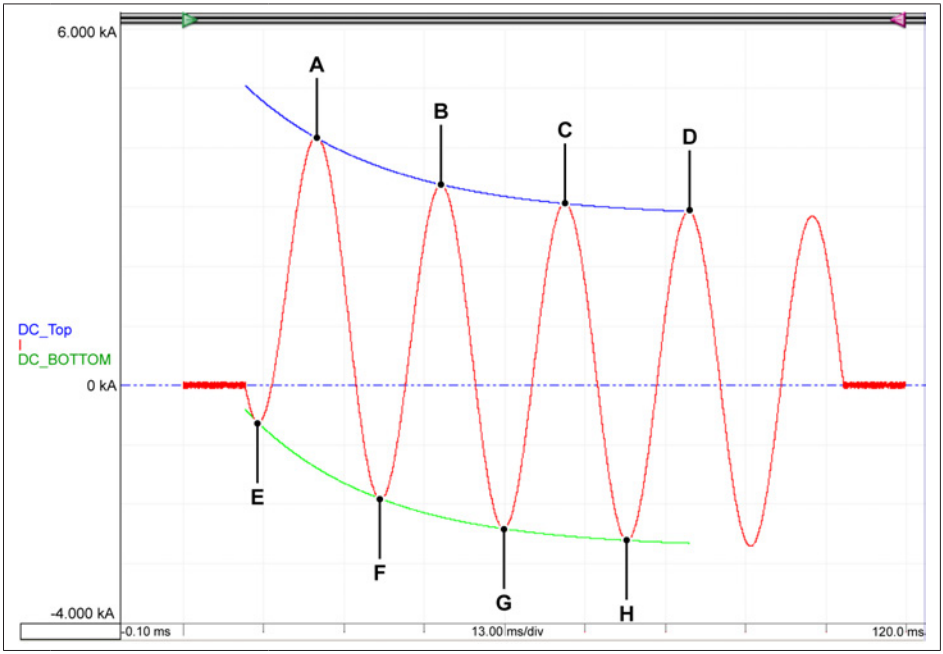


Fig. 2.34 STLX_DC_ExpEnvelope

A Top crest 1
B Top crest 2
C Top crest 3
D Top crest 3
E Bottom crest 1
F Bottom crest 2
G Bottom crest 3
H Bottom crest 4

Documentation reference

None

2.51 @STLX_DCEnd

Function

Returns the end location of a DC signal.

Syntax

@STLX_DCEnd*(Signal; Start; End; Threshold)*

Parameters

Signal	Input waveform (DC signal)
Start	(optional) The start position in x-units where the searching should start.
End	(optional) The end position in x-units where the searching should stop.
Threshold	(optional) The threshold level % related to the Maximum value of the DC signal. The default value is 0.1%.

Output

A number being the end location of a DC signal found between start and end.

Description

This function returns the end of a DC signal. It is internally using the threshold and the maximum value (or minimum value when DC signal is below zero) to set a level (threshold * max /100) for roughly locating the start and end of the DC signal. From here special logic and a linear interpolation is used to find a better start and end locations of the DC signal.

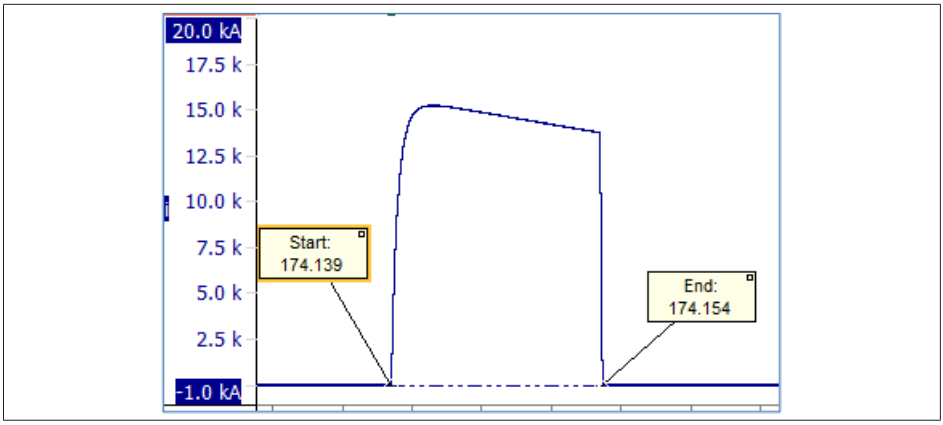


Fig. 2.35 STLX_DCEnd

Example

@STLX_DCEnd(Active.Group1.Recorder1.Ch1) returns the end of the DC signal Active.Group1.Recorder1.Ch1

See Also:

"@STLX_DCStart“ on page 83

2.52 @STLX_DCStart

Function

Returns the start location of a DC signal.

Syntax

@STLX_DCStart*(Signal; Start; End; Threshold)*

Parameters

Signal	Input waveform (DC signal)
Start	(optional) The start position in x-units where the searching should start.
End	(optional) The end position in x-units where the searching should stop.
Threshold	(optional) The threshold level % related to the Maximum value of the DC signal. The default value is 0.1%.

Output

A number being the start location of a DC signal found between start and end.

Description

This function returns the start of a DC signal. It is internally using the threshold and the maximum value (or minimum value when DC signal is below zero) to set a level (threshold * max /100) for roughly locating the start and end of the DC signal. From here special logic and a linear interpolation is used to find a better start and end locations of the DC signal.

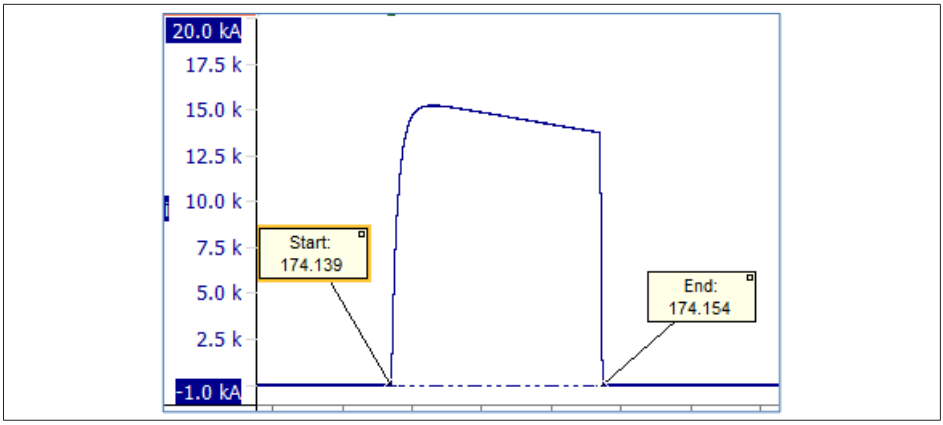


Fig. 2.36 STLX_DCStart

Example

@STLX_DCStart(Active.Group1.Recorder1.Ch1) returns the start location of the DC signal Active.Group1.Recorder1.Ch1

See Also:

"@STLX_DCEnd“ on page 81

2.53 @ STLX_AsymmetricalPowerfactor

Function

This function returns the power factor percentage of the input ratio for asymmetrical current signals.

Syntax

@STLX_AsymmetricalPowerFactor*(Ratio; Phase)*

Parameters

Ratio	The input ratio (Asymmetrical RMS / Symmetrical RMS)
Phase	Optional; used number of phases: • 1 = Single Phase • 3 = Three phase This parameter is optional if it has not been defined than single phase is used.

Output

The output is a numerical value indicating the short circuit power factor percentage of the input ratio for asymmetrical current signals.

Description

This function uses a lookup table to determine the short-circuit power factor percentage. The input *Ratio* parameter has to be calculated as follows:

For single phase:

$$M_M = \frac{\text{Asymmetrical rms Amperes}}{\text{Symmetrical rms Amperes}}$$

For 3 phases:

$$M_A = \frac{\text{Total 3 phases Asymmetrical rms Amperes}}{\text{Total 3 phases Symmetrical rms Amperes}}$$

The second parameter defines whether one or three phases are used.
The lookup table below is used to determine the power factor related to M_M or M_A.

Determination of the Power Factor related to (M_M) / (M_A)					
Short-circuit Power factor, percent	**Ratio M_M**	**Ratio M_A**	**Short-circuit Power factor, percent**	**Ratio M_M**	**Ratio M_A**
0	1.732	1.394	**30**	1.130	1.064
1	1.697	1.374	**31**	1.122	1.062
2	1.662	1.354	**32**	1.113	1.057
3	1.630	1.336	**33**	1.106	1.053
4	1.599	1.318	**34**	1.098	1.050
5	1.569	1.302	**35**	1.091	1.046
6	1.540	1.286	**36**	1.085	1.043
7	1.512	1.271	**37**	1.079	1.040
8	1.486	1.256	**38**	1.073	1.037
9	1.461	1.242	**39**	1.068	1.034
10	1.437	1.229	**40**	1.062	1.031
11	1.413	1.216	**41**	1.058	1.029
12	1.391	1.204	**42**	1.053	1.027
13	1.370	1.193	**43**	1.049	1.025
14	1.350	1.182	**44**	1.045	1.023
15	1.331	1.172	**45**	1.041	1.021
16	1.312	1.162	**46**	1.038	1.019
17	1.295	1.152	**47**	1.035	1.017
18	1.278	1.144	**48**	1.032	1.016
19	1.262	1.135	**49**	1.029	1.014
20	1.247	1.127	**50**	1.026	1.013
21	1.232	1.119	**55**	1.016	1.008
22	1.219	1.112	**60**	1.009	1.004
23	1.205	1.105	**65**	1.005	1.002
24	1.193	1.099	**70**	1.002	1.001
25	1.181	1.092	**75**	1.0008	1.0004
26	1.170	1.087	**80**	1.0002	1.00001
27	1.159	1.081	**85**	1.00004	1.00002
28	1.149	1.076	**100**	1.00000	1.00000
29	1.139	1.071			

Tip

Below 50 ***Short-circuit power factor %****, the closest* ***Short-circuit power factor %*** *integer from the table is returned.*
Above 50 ***Short-circuit power factor %****, linear interpolation is used to return the closest* ***Short-circuit power factor %*** *integer from the table.*

Documentation reference

The implementation is according to the document UL 489 (ISBN 0-7629-0677-4) Appendix C4, table C4.1.

2.54 @STLX_AsymmetricalPowerFactorDecimals

Function

This function returns the power factor percentage of the input ratio for asymmetrical current signals.

Syntax

@STLX_AsymmetricalPowerFactorDecimals*(Ratio; Phase)*

Parameters

Ratio	The input ratio (Asymmetrical RMS / Symmetrical RMS)
Phase	Optional; used number of phases: • 1 = Single Phase • 3 = Three phase This parameter is optional if it has not been defined than single phase is used.

Output

The output is a numerical value indicating the short circuit power factor percentage of the input ratio for asymmetrical current signals.

Description

This function is similar to the function **@STLX_AsymmetricalPowerfactor**, however when using the lookup table to determine the power factor related to M_M or M_A a linear interpolation is used between the two nearest points in this table. Therefore the power factor now has a decimal part.

Documentation reference

The implementation is according to the document UL 489 (ISBN 0-7629-0677-4) Appendix C6 Direct Current Circuits.

2.55 @STLX_ShortCircuitTimeConstant

Function

This function calculates the short circuit time constant of direct current circuits.

Syntax

@STLX_ShortCircuitTimeConstant*(Waveform Current; StartPos; EndPos; Current Start)*

Parameters

Waveform Current	Input current waveform
StartPos	Optional; Start time from where the evaluation should start. This parameter is optional if not entered the lowest possible time value is used.
EndPos	Optional; all evaluation activities are limited to this end time. This parameter is optional if not entered the highest possible time value is used.
Current Start	Optional; The timing position where the current starts, this is the moment of closing the circuit. This parameter is optional if it is not available then this function will detect itself where the current is starting.

Output

The output is numerical value indicating the short circuit time constant.

Description

The short circuit time constant is defined as the time needed for the current to go from zero to 0.632 of the maximum current. See diagram below (see Fig. 2.37).

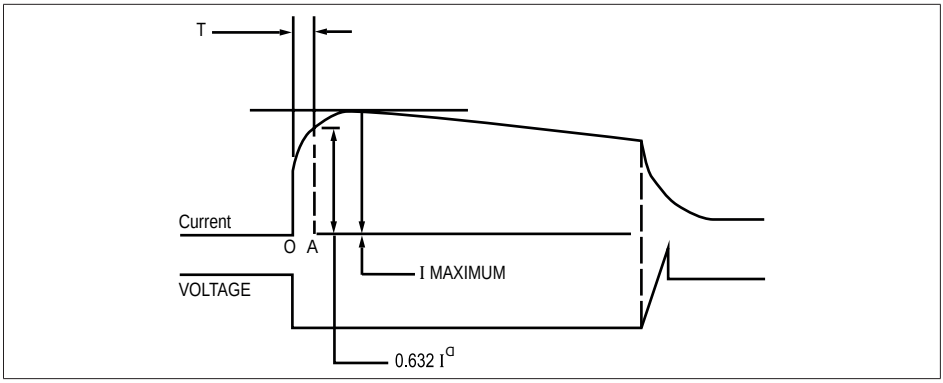


Fig. 2.37 Short circuit time constant

Documentation reference

The implementation is according to the document UL 489 (ISBN 0-7629-0677-4) Appendix C6 Direct Current Circuits.

2.56 @STLX_SignalStart

Function

This function is used to recognize the start of a signal.

Syntax

@STLX_SignalStart*(Waveform; Frequency; StartPos; EndPos)*

Parameters

Waveform	Sinusoidal input waveform
Frequency	*Frequency of the sinusoidal waveform*
StartPos	Optional; beginning of search. This parameter is optional if not entered the lowest possible time value is used.
EndPos	Optional; end of search. This parameter is optional if not entered the highest possible time value is used.

Output

A Numerical value indicating the start of the signal.

Description

This function is using the existing @STLSignalStart() function, however, to be more accurate it is doing some extra steps:

- The function automatically rescales the input signal (similar to the @STLXRescale() function).
- The @STLSignalStart() is using the rescaled signal to find the start **(Tstart1)**.
- Via a linear interpolation around the found start point this function tries to find a better start position **(TStart2)**. This location is the zero crossing location of the line found via the linear interpolation.

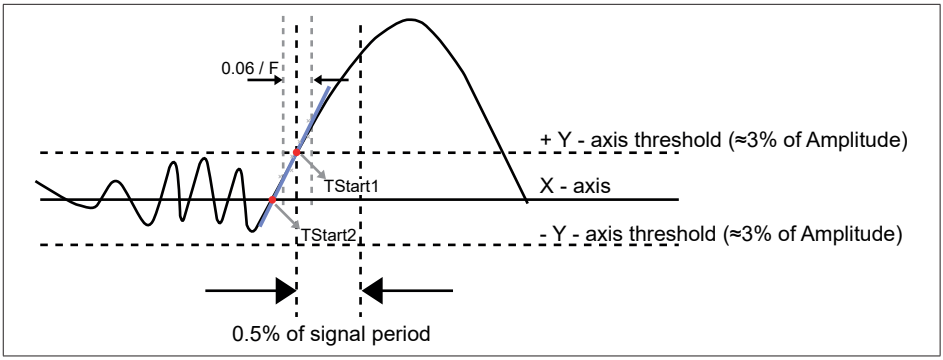


Fig. 2.38 STLXSignal - TStart1 and TStart2 positions

For more information, see "@STLSignalStart“ on page 8.

2.57 @STLX_SignalEnd

Function

This function is used to recognize the end of a signal.

Syntax

@STLX_SignalEnd*(Waveform; Frequency; StartPos; EndPos)*

Parameters

Waveform	Sinusoidal input waveform
Frequency	*Frequency of the sinusoidal waveform*
StartPos	Optional; beginning of search. This parameter is optional if not entered the lowest possible time value is used.
EndPos	Optional; end of search. This parameter is optional if not entered the highest possible time value is used.

Output

A Numerical value indicating the start of the signal.

Description

This function is using the existing @STLSignalEnd() function.
The function starts searching at “StartPos”, the search direction depends on the value of “EndPos”:

- Searches backwards if “StartPos” is greater than “EndPos”.
- Searches forwards if “EndPos” is greater than “StartPos”.

When “StartPos” and “EndPos” are not entered in the formula, the function starts searching at the beginning of the signal in forward direction until it finds an end condition or it stops when there are no more data points.

The following steps are done to get a more accurate end position:

- The function automatically rescales the input signal (similar to the @STLXRescale() function).
- The @STLSignalEnd() is using the rescaled signal to find the end location.
- Via a linear interpolation around the found end point this function tries to find a better end position. This location is the zero crossing location of the line found via the linear interpolation.

For more information, see "@STLSignalEnd“ on page 10 and "@STLX_SignalStart“ on page 91.

2.58 @STLX_GetActionTime

Function

This function returns the action time of a tripping signal.

Syntax

@STLX_GetActionTime*(Waveform; StartPos; EndPos)*

Parameters

Waveform	Sinusoidal input waveform
StartPos	Optional; beginning of search
EndPos	Optional; end of search

Output

A Numerical value indicating the start of the first action time after the StartPos time.

Description

For switchgear application tripping signals are often used to mark the start of an opening and closing operation. This time is typically called the **action** time.

- This function will determine the first action time after the StartPos time.
- The function can handle analog and digital tripping signals.
- The polarity of the tripping signals can be either positive or negative, the STLX_GetActionTime() will find the correct action time in both cases.

Fig. 2.39 Analog tripping signals

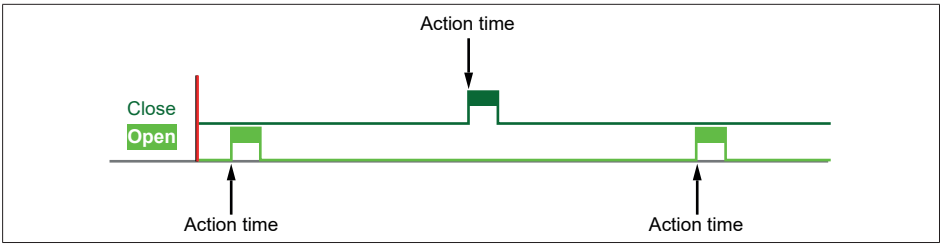


Fig. 2.40 Digital tripping signals

Documentation reference

None

2.59 @STLX_OffsetCorrection

Function

Removes the offset at the beginning of a signal.

Syntax

@STLX_OffsetCorrection*(Signal; StartOffset Interval; EndOffset Interval)*

Parameters

Signal	The signal for which the offset will be corrected.
StartOffset Interval	(optional) Defines the start of the interval used to find the offset; by default, this is the start of the signal.
EndOffset Interval	(optional) Defines the end of the interval used to find the offset; by default, this is 20 ms after the entered start location (StartOffset interval).

Output

A number being the start location of a DC signal found between start and end.

Description

This function removes the offset mostly at the beginning of a signal. It calculates the average over the offset interval period and subtracts this from the signal. It works for both positive or negative offsets. By default, the offset interval starts at the beginning of the signal and has a length of 20 ms. However, this can be changed via the two optional parameters.

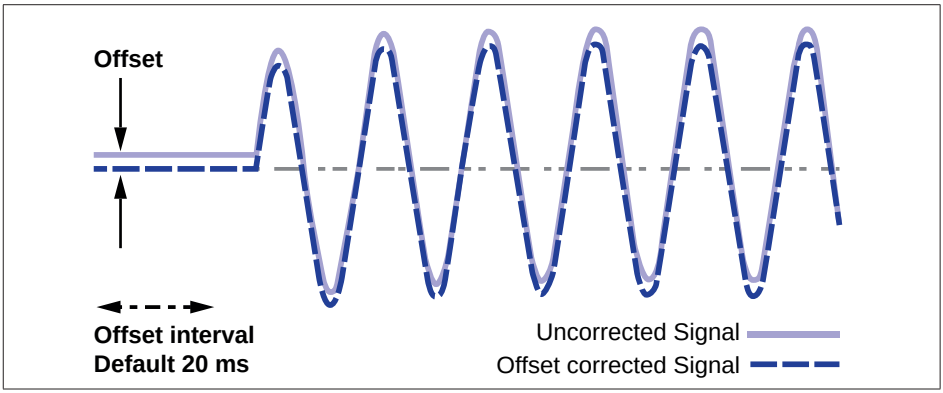


Fig. 2.41 STLX_OffsetCorrection

Example

@STLX_OffsetCorrection(Active.Group1.Recorder1.Ch1) returns a new trace where the offset has been removed

Documentation reference

None

A STL Analysis User Keys Actions

A.1 Introduction

The STL comes with predefined **User Key Actions**; these actions make it possible to do basic STL actions like (for example) *go to the next crest* by a User Key click.

Tip

For more information about User Keys (general), please refer to the A05025_01_E00_00_ Perception User Keys and Macros User manual.

The STL User Key actions not only position the cursor in the active display but also publish information as data sources. All these data sources will start with Temp.STL.

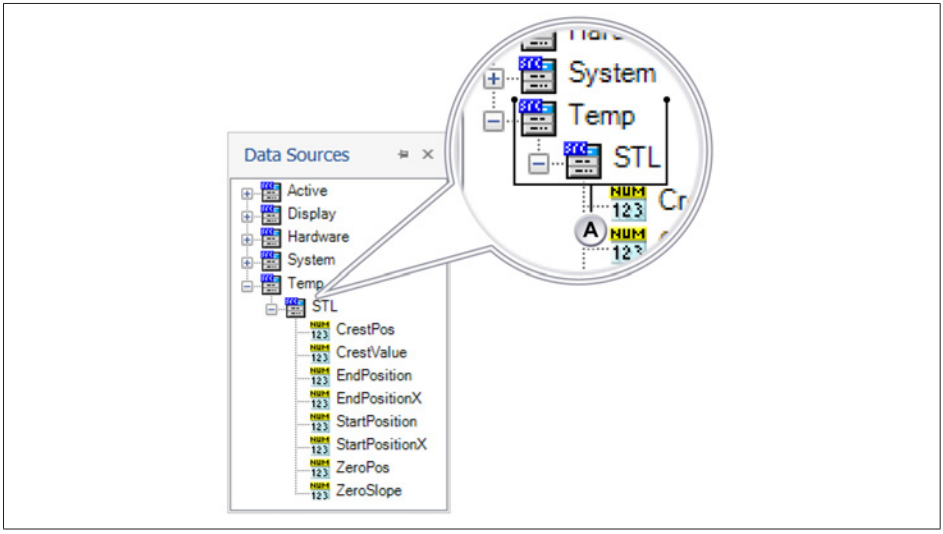


Fig. A.1 STL data sources

A Temp/STL nodes in STL data source

These data sources are located in the Temp data source tree because they are not persistent but temporally, this means they are not saved into the workbench or experiment.

In this chapter the various **STL** User Keys are described.

A STL ANALYSIS USER KEYS ACTIONS

STL User Key actions

A.1.1 Clear all STL User Key data sources

Clear all STL User Key data sources.

A.1.2 Delete all STL User Key data sources

Delete all STL User Key data sources.

A.1.3 Find first STL crest

Find the first STL crest from the **beginning of the recording (A)** or from the **position of the active cursor (B)** in the **Find first Crest Configuration** dialog.

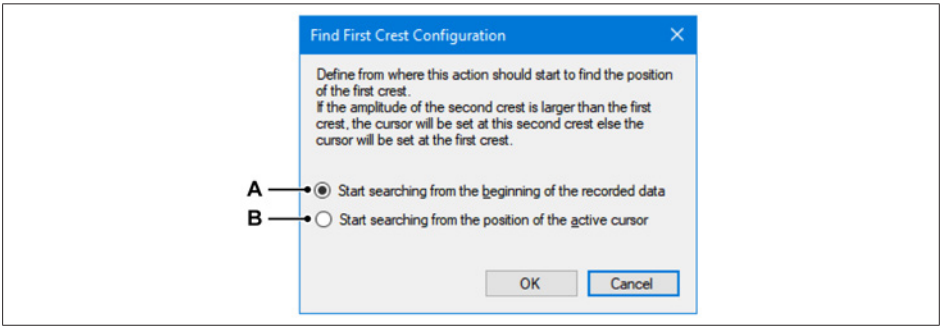


Fig. A.2 Find First Crest Configuration

The cursor will be placed at the location of the first found STL crest and the data sources Temp.STL.CrestPos and Temp.STL.CrestValue will be updated with the time location and the value of the found first crest.

A.1.4 Find the next STL crest

Find the next STL crest from the position of the active cursor. The cursor will be placed at the location of the found STL crest and the data sources **Temp.STL.CrestPos** and **Temp.STL.CrestValue** will be updated with the time location and the value of the found first crest.

A.1.5 Find the previous STL crest

Find the previous STL crest from the position of the active cursor. The cursor will be placed at the location of the found STL crest and the data sources **Temp.STL.CrestPos** and **Temp.STL.CrestValue** will be updated with the time location and the value of the found first crest.

A.1.6 Find next STL zero crossing

Find the next STL zero crossing from the position of the active cursor in the **Frequency Entry** dialog.

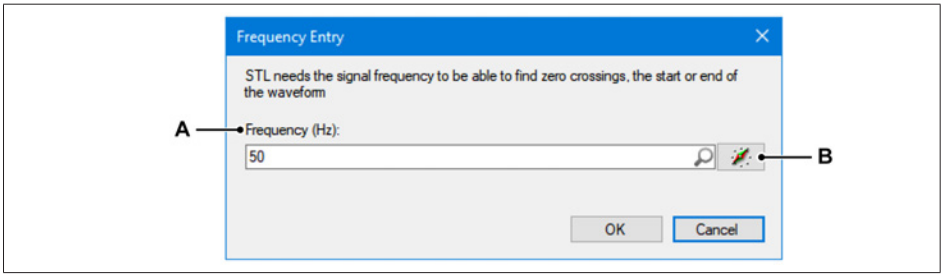


Fig. A.3 Frequency Entry

A Used frequency

B The data sources explorer

The cursor will be placed at the location of the found zero crossing and the data sources **Temp.STL.ZeroPos** and **Temp.STL.ZeroSlope** will be updated with the time location of the zero crossing and the value of slope used to find this zero crossing. For more information, please refer to "@STLNextZeroCrossing“ on page 12.

A.1.7 Find previous STL zero crossing

Find the previous STL zero crossing, works like „Find next STL zero crossing“ function, but searches backwards.

A.1.8 Find STL Start and End

Find the STL start and end position. The time interval between cursor 1 and 2 is used to find the start and end positions. When the action finds the start and end positions it moves cursor 1 to the start and cursor 2 to the end location. The data sources **Temp.STL.StartPosition** and **Temp.STL.EndPosition** will be updated with the found time locations. The used frequency can be entered with the Frequency Entry dialog (see Fig. A.3).

A.1.9 Find STLX Start and End

Find the STL start and end position using the extended versions, see the STL functions: "@STLX_SignalStart“ on page 91 and "@STLX_SignalEnd“ on page 93.
The time interval between cursor 1 and 2 is used to find the start and end positions. When the action finds the start and end positions it moves cursor 1 to the start and cursor 2 to the end location. The data sources T**emp.STL.StartPositionX** and **Temp.STL.EndPositionX** will be updated with the found time locations.

Index

Symbole

A

C

D

E

F

G

I

L

N

O

P

R

S

V

Hottinger Brüel & Kjaer GmbH
www.hbkworld.com
info@hbkworld.com

A02696_03_E00_00 10/02/2025