|  |  |
| --- | --- |
|  |  |

EPTF CLL Statistics Measure, Function Description

Contents

[1.1 Revision history 3](#_Toc327539280)

[1.2 How to Read this Document 3](#_Toc327539281)

[1.3 References 3](#_Toc327539282)

[1.4 Scope 3](#_Toc327539283)

[1.5 Recommended way of reading 4](#_Toc327539284)

[1.6 Typographical conventions 4](#_Toc327539285)

[1.7 Abbreviations 4](#_Toc327539286)

[1.8 Terminology 4](#_Toc327539287)

[2 General Description 5](#_Toc327539288)

[2.1 Detailed description of supported statistics 6](#_Toc327539289)

[2.1.1 Content 7](#_Toc327539290)

[2.1.2 Delta 7](#_Toc327539291)

[2.1.3 DeltaSum 7](#_Toc327539292)

[2.1.4 Min 7](#_Toc327539293)

[2.1.5 Max 7](#_Toc327539294)

[2.1.6 Mean 7](#_Toc327539295)

[2.1.7 StandardDev 7](#_Toc327539296)

[2.1.8 EPS 8](#_Toc327539297)

[2.1.9 Chrono 8](#_Toc327539298)

[2.1.10 Density 8](#_Toc327539299)

[2.1.11 Percentile95 9](#_Toc327539300)

[2.1.12 PercentileP 10](#_Toc327539301)

[2.1.13 Limits 10](#_Toc327539302)

[2.1.14 Custom 11](#_Toc327539303)

[3 Functional Interface 13](#_Toc327539304)

[3.1 Naming Conventions 13](#_Toc327539305)

[3.2 General Public Functions 13](#_Toc327539306)

[3.2.1 Initialization 13](#_Toc327539307)

[3.2.2 Creating Statistics 13](#_Toc327539308)

[3.2.3 Resetting Statistics 14](#_Toc327539309)

[3.2.4 Updating Statistics 14](#_Toc327539310)

[3.2.5 Enabling/Disabling Statistics 14](#_Toc327539311)

[3.2.6 Get/Set Functions 15](#_Toc327539312)

[3.2.7 Convenience Functions 15](#_Toc327539313)

[3.2.8 Creating an EPTF Variable from a Statistics 15](#_Toc327539314)

[3.3 Summary Table of General Public Functions for EPTF Statistics Measure 16](#_Toc327539315)

[3.4 Special functions 17](#_Toc327539316)

[3.4.1 Delta, DeltaSum, Min and Max 17](#_Toc327539317)

[3.4.2 Mean 17](#_Toc327539318)

[3.4.3 StandardDev 17](#_Toc327539319)

[3.4.4 EPS 18](#_Toc327539320)

[3.4.5 Chrono 18](#_Toc327539321)

[3.4.6 Density 19](#_Toc327539322)

[3.4.7 Percentile95 19](#_Toc327539323)

[3.4.8 PercentileP 20](#_Toc327539324)

[3.4.9 Limits 20](#_Toc327539325)

Introduction

## Revision history

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Rev | Characteristics | Prepared |
| 2007-11-29 | PA1 | First draft version | <EANDDAR> |
| 2008-01-09 | PA2 | Density stat added, EPS modified | ETHJGI |
| 2008-01-28 | PA3 | Updated after review | ETHJGI |
| 2008-11-28 | PB1 | Updated for new statistics | EAKOPER |
| 2009-03-24 | PB2 | Stop function added to chrono | ETHJGI |
| 2009-05-19 | PB3 | PercentileP statistics added | EBENMOL |
| 2009-09-02 | PC1 | EPS statistics updated | EBENMOL |
| 2012-05-17 | PD1 | Limits stat added | ETHJGI |

## How to Read this Document

This is the Function Description for the Statistics Measure Feature of the Ericsson Performance Test Framework (TitanSim), Core Load Library (CLL). TitanSim CLL is developed for the TTCN-3 ‎[1] Toolset with TITAN ‎[2]. For more information on the TitanSim CLL please consult the Product Revision Information ‎[3]. Additionally, to understand the functionality of this feature, read the documentation of the Variable ‎[6] feature.

## References

1. ETSI ES 201 873-1 v3.2.1 (2007-02)  
   The Testing and Test Control Notation version 3. Part 1: Core Language
2. 1/198 17-CRL 113 200 Uen  
   User Guide for the TITAN TTCN-3 Test Executor
3. 109 21-CNL 113 512-2 Uen   
   TitanSim CLL for TTCN-3 toolset with TITAN, Product Revision Information
4. 155 17-CNL 113 512 Uen   
   TitanSim CLL for TTCN-3 toolset with TITAN, Function Specification
5. TitanSim CLL for TTCN-3 toolset with TITAN, Reference Guide  
   <http://ttcn.ericsson.se/products/libraries.shtml>
6. 18/155 16-CNL 113 512 Uen  
   EPTF CLL Variable Function Description

## Scope

This document is to specify the content and functionality of the Statistics Measure feature of the TitanSim CLL.

## Recommended way of reading

The readers are supposed to get familiar with the concept and functionalities of TitanSim CLL ‎[4] in general, and the Variable ‎[6] feature. They should get familiar with the list of acronyms and the glossary in Section ‎1.7 and ‎1.8, respectively.

## Typographical conventions

Important concepts are denoted by *italic* font wherever they are first used in the given context. Moreover, whenever a concept is mentioned that has a special meaning as described in the Glossary (Section ‎1.8) of this document, then these occurrences are marked with an initial arrow, e.g., *🡪 TitanSim Statistics*.

## Abbreviations

CLL Core Load Library

EPTF Ericsson Load Test Framework, formerly TITAN Load Test Framework

TitanSim Ericsson Load Test Framework, formerly TITAN Load Test Framework

TTCN-3 Testing and Test Control Notation version 3 ‎[1]

## Terminology

*TitanSim Core (Load) Library(CLL)* is that part of the TitanSim software that is totally project independent. (I.e., which is not protocol-, or application-dependent). The TitanSim CLL is to be supplied and supported by the TCC organization. Any TitanSim CLL development is to be funded centrally by Ericsson

*TitanSim Variables* contain values, from which *🡪 TitanSim Statistics* can be calculated automatically on changes or periodically.

*TitanSim Statistics Measure* is a feature, which is responsible for creating *🡪 TitanSim Statistics* and updating their value when required.

*TitanSim Statistics* are Statistics of TTCN-3 Variables. Possible Statistics are among others minimum, maximum, mean, standard deviation, etc. and also the content of a Variable can be a Statistics itself. The values of such Statistics are automatically and periodically updated in the background.

# General Description

This document specifies the Statistics Measure feature of the TITANSim CLL.

The EPTF Statistics Measure feature makes it possible to

* Create stand-alone Statistics or Statistics belonging to a given Variable
* Measure statistical data and update the value of Statistics
* Reset Statistics
* Enable/Disable Statistics
* Create an EPTF Variable from a given Statistics

EPTF Statistics can be created in two different ways. The first case is when the Statistics is created belonging to an EPTF Variable. In this case, a certain type of Statistics is created for the given EPTF Variable, and this Statistics is always updated by Statistics Measure if the value of the EPTF Variable changes. Thus, the value of Statistics is available according to push model.

The second case is, if the user creates a stand-alone Statistics of a given type. In this case, no Variable is assigned, it is the user`s responsibility to update the value of this Statistics and add new data to it, when required.

Statistics can have different types:

* content: The value of a Variable (it cannot be a stand-alone Statistics)
* delta: The new value – the old value (last measured)
* deltaSum: The new value – the initial value (first measured)
* min: minimal value ever
* max: maximal value ever
* mean: mean value
* standardDev: contains both standard deviation and mean
* EPS: measures the number of events during a measurement period (events/sec) and the length of the measurement period (sec).  For Variables the events are the refresh events
* chrono: measures the time
* density: counts the number of values that fall between user defined boundaries. The scale of the boundaries can be set to linear or logarithmic between a minimal and maximal boundary value.
* percentile95: statistics values are ordered into increasing order and percentile95 statistics value is calculated as the element at the number of elements \* 0.95 position (position is rounded upwards if it is not an integer value). The user should set boundary intervals to quantify the measurement space. The final statistics value will be the upper boundary of the interval in which the percentile95 element falls or the maximum of elements if the percentile95 element is above the higher boundary.
* percentileP: this statistic is the general version of percentile95 statistics. The statistics values are ordered into increasing order and pth percentile statistics value is calculated as the element at the number of elements \* pth position. The user should set boundary intervals to quantify the measurement space. The final statistics value will be the upper boundary of the interval in which the p percentile element falls.
* limits: calculates a statusLED based on predefined threshold for certain colors. Also relative difference to a reference value can be used for the calculation.
* custom: provides the ability to the user to create own statistics. The user should register its on custom statistics functions consistently with existing structure of EPTF Statistics Measure feature. The registered custom statistic then can be applied like other statistic type.

The aim of the EPTF Statistics Measure feature is to calculate and handle all these type of Statistics in one component.

To be able to use EPTF Statistics Measure, the user component should extend the EPTF\_StatMeasure\_CT component.

## Detailed description of supported statistics

The statistics that are created for an EPTF Variable updated with a new measured data when the value of Variable is refreshed by the refresh mechanism.

New measured data can be added manually to stand-alone statistics and statistics that belong to an EPTF Variable by the addData functions.

### Content

The content statistics is defined to be able to interpret the value of an EPTF Variable as a statistics. This statistics does not contain any data, only the reference to the EPTF Variable.

This statistics cannot be defined for stand-alone statistics, and cannot be reset.

### Delta

The delta statistics measures the difference between the last measured value and the current value.

### DeltaSum

This statistics measures the difference between the current value and the first value after the statistics was reset.

### Min

This statistics measures the minimal value since reset.

### Max

This statistics measures the maximal value since reset.

### Mean

This statistics measures the mean, i.e. the arithmetic average of the data values added since reset.

### StandardDev

This statistics measures the mean and standard deviation, i.e. the arithmetic average and the variance of the data values added since reset.

### EPS

This statistics measures how many new values were added during the measurement period and also the length of the measurement period. Before the value of the statistics is read out the update function should be called. The length of the measurement period can be specified in seconds, or if not specified the time since addition of the first data is used. This statistics can be used to measure the number of events during a certain time (if 1 sec is used for the duration), the average events/sec during the measurement period, or the time during a period.

### Chrono

This statistics measures the time during a user given period, or the elapsed time since the start of the measurement.

### Density

This statistics can measure the number of data falling between user given boundaries. Also the number of data below the minimal and above the maximal boundary is measured.

It is possible the set automatically calculated boundaries between a minimal and maximal value on a linear or a logarithmic scale.

This statistics corresponds to the density function without normalization. If all measured counter is divided by the sum of all counters we will get the density function.

An example measured density statistics is shown on the figure below:

Value

Figure 1. Example for the density statistics

Counters

max

min

0 1 2 3 … N+1

A data value falls into a coloumn if:

* Lower boundary <= Value < Upper boundary
* The 0th column measures the number of values that falls below the minimal boundary: value < min boundary
* The last column measures the values that larger or equal the maximal boundary: value <= max boundary

If there is N boundary specified the number of intervals measured will be N+1.

### Percentile95

Percentile95 statistic provides the lowest element from a sequence of numbers that is bigger then the 95 percent of all the elements.

Percentile95 statistic is based on density (‎2.1.10) and max (‎2.1.5) statistics. Values are categorized by density statistic. The position of the required element and its boundary is computed incrementally. If the computed boundary is the highest boundary where no upper limit exists, the statistic value will be the maximum of the elements, otherwise statistic value will be the highest limit of the computed boundary.

The precision of the statistic can be improved by defining boundaries more frequently however it results in performance loss. It is recommended that the user creates the boundaries for the necessary precision only.

### PercentileP

PercentileP statistic provides the lowest element from a sequence of numbers that is bigger then the pth percent of all the elements. This is a general version of percentile95 statistics (‎2.1.11), where p value is set to 95.

PercentileP statistic is based on density (‎2.1.10) and max (‎2.1.5) statistics. Values are categorized by density statistic. The position of the required element and its boundary is computed incrementally. If the computed boundary is the highest boundary where no upper limit exists, the statistic value will be the maximum of the elements, otherwise statistic value will be the highest limit of the computed boundary.

The precision of the statistic can be improved by defining boundaries more frequently however it results in performance loss. It is recommended that the user creates the boundaries for the necessary precision only.

### Limits

The limits statistics provides a StatusLED with the color and the text determined from the last measured data. Each color have a lower limit value. The color of the limits statistics LED will be set to the color whose limit is the highest below the last measured data. If no such limit is found, the color is set to the default color of the limits statistics.

The limits statistics can be calculated from a source variable. The type of the variable can be integer or float. The value of the variable is compared to the limit values and the color of the LED will be set accordingly.

The limits statistics can also have a reference value variable. The type of the reference variable can be integer or float. If specified the value that is compared to the specified LED color limits in this case is the relative difference from the reference value in percentage:

100\*(data-reference)/reference

The limit values for different colors can be changed anytime. When they are changed, the limits statistics is reset. It is possible to set different limit for the same color more than once.

Default color

GreenLimit

BlueLimit

GreenLimit

green

blue

green

Figure 2: The color of the limits statistics

### Custom

Custom statistic enables the user to create her/his own statistics. The functions realizing the new statistic must be registered by the user by providing the statistic name, the previously mentioned custom statistic functions with their current argument list to the f\_EPTF\_StatMeasure\_newStat function.

Provided argument list is type of EPTF\_IntegerList which can be used as a number of pointers to TTCN databases.

Custom statistics and registered functions can be used trough the following interface:

#### New statistics function (f\_EPTF\_StatMeasure\_newStat\_custom)

Registers the new statistics and creates the statistics also based on provided statistic name, statistic functions and current argument list.

Like other statistics, custom statistic can be created from a variable or can be created stand alone. If statistics is created from a variable, the add data function is called automatically on value changes.

The function also resets and enables the statistic, than calls user defined new statistic function.

#### Enable statistics function (f\_EPTF\_StatMeasure\_enableData\_custom)

Change the state of statistics to enabled. Statistic can only be updated when its state is set to enabled.

If the statistic has been created from a variable, the value changes of its variable will trigger the statistic.

Also user defined enable statistic function is applied here.

#### Disable statistics function (f\_EPTF\_StatMeasure\_disableData\_custom)

Change the state of statistics to disabled. Statistic can’t be updated when its state is set to disabled.

If the statistic has been created from a variable, the value changes of its variable will not trigger the statistic.

Also user defined disable statistic function is applied here.

#### Reset statistics function (f\_EPTF\_StatMeasure\_resetStat\_custom)

Reset the statistic.

By default the function only set the statistic state to reset enabled, but doesn’t perform its resetting. It is recommended to act on reset from add data function.

Also user defined reset statistic function is applied here which can override previous behavior if necessary.

#### Create variable from statistics function (f\_EPTF\_StatMeasure\_ createVarFromStat\_custom)

The function creates a variable from the statistic by applying user defined create variable from statistics function.

The type of custom statistic is not fixed, therefore user must fix its type and create this function according to it.

The created variable is synchronized automatically to the statistic value of its custom statistic.

#### Add data function (f\_EPTF\_StatMeasure\_addData\_custom)

If statistic is enabled, it updates statistics (optionally using the provided new statistic value) and also updates the variable created from the given statistic (if one exists).

It is recommended to act on statistics reset within this function.

#### Get value function (f\_EPTF\_StatMeasure\_getStat\_custom)

The function acquires the current value of the custom statistic.

#### Stat value to string function (f\_EPTF\_StatMeasure\_custom2str)

The function converts the current statistic value to string.

# Functional Interface

Apart from this description a cross-linked reference guide for the TitanSim CLL Functions can be reached for on-line reading ‎[5].

## Naming Conventions

All functions have the prefix f\_EPTF\_StatMeasure\_

## General Public Functions

The following sections describe the functions that are available for all statistics.

### Initialization

Before using the EPTF Statistics Measure functions the

f\_EPTF\_StatMeasure\_init\_CT(componentName)

function should be called. This initializes the EPTF Statistics Measure feature.

### Creating Statistics

f\_EPTF\_StatMeasure\_newStat(varID, statType [, customStatName, customStatFunctions, customStatArguments])

This function creates a new Statistics of a given type (e.g. min, max, etc.) belonging to a Variable and returns the ID of the new Statistics. Note, that percentileP statistics will be created with p value 50.0 that can be changed later.

f\_EPTF\_StatMeasure\_newStat\_\*(varID [,customStatName, customStat Functions, customStatArguments])

To create stand-alone Statistics of a given type, this function should be called substituting \* with the Statistics type (e.g. delta). In place of the *varID*, in this case the -1 value should be written. Note, that for the content Statistics type, it is not possible to create stand-alone Statistics (i.e. without a Variable assigned to it).

f\_EPTF\_StatMeasure\_addData\_\*(statID, newData)

If no variable was given when the statistics was created, this function can be used to add new data.  If the statistics was created for an existing EPTF Variable data will be automatically added when the value of the variable is refreshed, and it is considered as a new measured value.

Note that this function only works with enabled Statistics.

Note that the Variable belonging to EPS statistics won’t be updated automatically, additional functions has to be called (See ‎3.2.4).

f\_EPTF\_StatMeasure\_createAllStat(varID)

This function creates all possible Statistics belonging to the given Variable ID.

### Resetting Statistics

f\_EPTF\_StatMeasure\_resetStat(statID)

f\_EPTF\_StatMeasure\_resetStats(statIDList)

These functions can be used to reset given Statistics. The Statistics will be initialized. Then, a new measurement can be started.

Note that the value of the given Statistics is reset only when the startMeasurement or the addData function is called afterwards.

To reset the value of the statistics, the function

f\_EPTF\_StatMeasure\_initStatValue(statID)

should be called. It resets the statistics and initializes its value. The created variable is updated also.

### Updating Statistics

f\_EPTF\_StatMeasure\_update\_EPS(statID, period)

In case of Statistics belonging to Variables, the Statistics are automatically updated in the background, so that the user does not have to call any update functions.

For EPS type of Statistics, however, the user generally wants to define the measurement period, for which the EPS Statistics is calculated. For this aim, the update\_EPS function should be called. If measurement period is not set or set to -1.0, the time is automatically measured using the elapsed time since the addition of the first data.

f\_EPTF\_StatMeasure\_registerDefaultUpdate\_EPS(statID)

This function should be used to set the Variable belonging to EPS statistics to be updated automatically. If this function is not called f\_EPTF\_StatMeasure\_update\_EPS() has to be called to update the Variable.

### Enabling/Disabling Statistics

f\_EPTF\_StatMeasure\_enableStat(statID)

f\_EPTF\_StatMeasure\_disableStats(statID)

Statistics can be enabled or disabled. Only enabled Statistics are updated with new values, disabled Statistics will not be updated and the f\_EPTF\_StatMeasure\_addData\_\* function does not change their data either. New Statistics are created as enabled.

### Get/Set Functions

f\_EPTF\_StatMeasure\_getVarIdx(statID)

This function returns the ID of the EPTF Variable the given statistics calculated from. If the Statistics does not belong to any Variable, the returned ID is -1.

f\_EPTF\_StatMeasure\_getStat\_\*(statID, statValue)

This function can be called to receive the value of a Statistics *statID* of a certain type (\* should be substituted by the type) in the *statValue* parameter. This function returns false if the value of the statistics is not available, e.g. when no data was added after reset.

f\_EPTF\_StatMeasure\_getStatType(statID)

This function returns the type (content, min, max, etc.) of the Statistics.

### Convenience Functions

f\_EPTF\_StatMeasure\_value2str(statID)

This function converts the value of the Statistics to charstring for easier printing.

### Creating an EPTF Variable from a Statistics

f\_EPTF\_StatMeasure\_createVarFromStat(statID, VarName)

This function can be useful if the user wants to print the Statistics onto the GUI, or reach the value of a Statistics from a distant component.

The function creates an EPTF Variable form the given Statistics. The name of the created Variable can be given by *VarName*; if left empty, this name is auto-generated.

The value of the created EPTF Variables is updated automatically for all Statistics (including stand-alone Statistics) when the value of the corresponding Statistics changes.

## Summary Table of General Public Functions for EPTF Statistics Measure

Table 1. Summary of Statistics Measure Functions

|  |  |
| --- | --- |
| Function name | Description |
| f\_EPTF\_StatMeasure\_init\_CT | Initializes the StatMeasure component. |
| f\_EPTF\_StatMeasure\_newStat | Creates new Statistics and returns its ID. |
| f\_EPTF\_StatMeasure\_addData\_\* | Adds new data to a Statistics of a given type (\*). |
| f\_EPTF\_StatMeasure\_createAllStat | Creates all available Statistics for a Variable. |
| f\_EPTF\_StatMeasure\_resetStat | Resets a Statistics. |
| f\_EPTF\_StatMeasure\_resetStats | Resets all Statistics defined in the input parameter. |
| f\_EPTF\_StatMeasure\_initStatValue | Resets the Statistics and initializes its value. |
| f\_EPTF\_StatMeasure\_update\_EPS | This function should be called to update the EPS type Statistics. |
| f\_EPTF\_StatMeasure\_enableStat | Enables a Statistics. |
| f\_EPTF\_StatMeasure\_disableStats | Disables a Statistics |
| f\_EPTF\_StatMeasure\_getVarIdx | Returns the Variable ID the Statistics belongs to, or -1 |
| f\_EPTF\_StatMeasure\_getStat\_\* | Returns the actual value of a Statistics of a given type (\*) |
| f\_EPTF\_StatMeasure\_getStatType | Returns the type of the Statistics. |
| f\_EPTF\_StatMeasure\_value2str | Converts the Statistics value to charstring. |
| f\_EPTF\_StatMeasure\_createVarFromStat | Creates a Variable from a Statistics with a given name. |
| f\_EPTF\_StatMeasure\_changeToFloat\_\* | Changes the value type to float of the statistics delta, deltaSum, min and max. |

## Special functions

This section describes additional functions that are available for a given statistics only.

### Delta, DeltaSum, Min and Max

The function f\_EPTF\_StatMeasure\_changeToFloat\_\* can be used to change the type of these statistics to float if they are not standalone stats. The type of these standalone statistics is set to integer by default. This function has to be called before a variable is created from the statistics by the f\_EPTF\_StatMeasure\_createVarFromStat function.

### Mean

The function f\_EPTF\_StatMeasure\_createVarFromStat\_N\_mean can be used to create EPTF Variable from the auxiliary data N (i.e. the number of data samples) of the mean statistics.

### StandardDev

The function f\_EPTF\_StatMeasure\_getStat\_standardDev\_mean can be used to get the mean of the data. This mean value is calculated automatically for the standardDev statistics. It is not necessary to create an additional mean statistics.

The function f\_EPTF\_StatMeasure\_getStatIdx\_standardDev\_mean can be used to get the statID of the mean statistics that is maintained automatically for the standardDev statistics. Do not reset, enable or disable this statistics! This ID can be used to create an EPTF Variable from this statistics.

The function f\_EPTF\_StatMeasure\_standardDev\_mean2str can be used to convert the mean statistics belonging to the standardDev statistics to string.

The function f\_EPTF\_StatMeasure\_createVarFromStat\_N\_standardDev can be used to create EPTF Variable from the auxiliary data N (i.e. the number of data samples) of the standardDev statistics.

The function f\_EPTF\_StatMeasure\_createVarFromStat\_S\_standardDev can be used to create EPTF Variable from the auxiliary data S (i.e. the sum of deviation squares of data samples) of the standardDev statistics.

The function f\_EPTF\_StatMeasure\_createVarFromStat\_Mean\_standardDev can be used to create EPTF Variable from the auxiliary data Mean (i.e. the mean of data samples) of the standardDev statistics.

### EPS

To calculate the EPS statistics and update its value and set the time of the measurement call f\_EPTF\_StatMeasure\_update\_EPS. If the period is not specified the time of the measurement is determined automatically. If 1 sec is specified for the period the value of the statistics will give the number of events since the start of the measurement.

The measurement is started when the f\_EPTF\_StatMeasure\_startMeasurement\_EPS is called or if the first data is added by the f\_EPTF\_StatMeasure\_addData\_EPS function after f\_EPTF\_StatMeasure\_resetStat\_EPS is called.

The current time since the start of the measurement can be read by f\_EPTF\_StatMeasure\_getTime\_EPS.

To read out the length of the measurement period (i.e. the time between start and stop) call the function f\_EPTF\_StatMeasure\_getMeasurementLength\_EPS.

Additional calls to f\_EPTF\_StatMeasure\_update\_EPS can be used to update the stop time and the measurement length. The variable belonging to EPS statistics will get a new value if this function is called.

If user wants the Variable belonging to the EPS statistics to be updated automatically, he has to call the function f\_EPTF\_StatMeasure\_registerDefaultUpdate\_EPS after he created this type of statistics.

### Chrono

The function f\_EPTF\_StatMeasure\_start\_chrono starts the chronometer. If this function is not called the measurement is started when the first data is added by the function f\_EPTF\_StatMeasure\_addData\_chrono. However, f\_EPTF\_StatMeasure\_addData\_chrono updates the value of the statistics also.

The function f\_EPTF\_StatMeasure\_getMeasurementLength\_chrono returns the length of the measurement. It is the time between start and update in seconds. Same as f\_EPTF\_StatMeasure\_getStat\_chrono but the value of the statistics is in the return value.

The f\_EPTF\_StatMeasure\_getTime\_chrono function returns the time elapsed since the measurement was started.

The function f\_EPTF\_StatMeasure\_update\_chrono updates the measurement length. The value of the statistics is set to the time since start.

The function f\_EPTF\_StatMeasure\_stop\_chrono stops the chronometer and updates the value of the statistics. After the chronometer is stopped new data cannot be added with the addData function. The statistics has to be reset to add further data. Also after stopping the chronometer, the f\_EPTF\_StatMeasure\_getTime\_chrono will return the time between starting and stopping the chronometer.

### Density

The function f\_EPTF\_StatMeasure\_setBoundaries\_density can be used to set the boundaries manually. When this function is called, the statistics is reset automatically.

The function f\_EPTF\_StatMeasure\_setScale\_density can be used to generate the boundaries automatically between the lower and the upper boundary. The scale can be set to linear or logarithmic. When this function is called, the statistics is reset automatically.

The f\_EPTF\_StatMeasure\_getStat\_normalized\_density function returns the density function. It is normalized so that the sum of all values is 1.

### Percentile95

The function f\_EPTF\_StatMeasure\_setBoundaries\_percentile95 can be used to set the boundaries manually. When this function is called, the statistics is reset automatically.

The function f\_EPTF\_StatMeasure\_setScale\_percentile95 can be used to generate the boundaries automatically between the lower and the upper boundary. The scale can be set to linear or logarithmic. When this function is called, the statistics is reset automatically.

The functions f\_EPTF\_StatMeasure\_getStatIdx\_density\_percentile95 and f\_EPTF\_StatMeasure\_getStatIdx\_max\_percentile95 return the index of the density and max statistics used by percentile95.

The functions

* f\_EPTF\_StatMeasure\_createVarFromStat\_max\_percentile95
* f\_EPTF\_StatMeasure\_createVarFromStat\_density\_percentile95
* f\_EPTF\_StatMeasure\_createVarFromStat\_boundaries\_density\_percentile95

can be used to create EPTF Variables from the max, density and the boundaries of the percentile95 statistics.

### PercentileP

The function f\_EPTF\_StatMeasure\_setBoundaries\_percentileP is the same as in percentile95 statistics. It can be used to set the boundaries manually. When this function is called, the statistics is reset automatically.

The function f\_EPTF\_StatMeasure\_setScale\_percentileP can be used to generate the boundaries automatically between the lower and the upper boundary. The scale can be set to linear or logarithmic. When this function is called, the statistics is reset automatically.

The functions f\_EPTF\_StatMeasure\_getStatIdx\_density\_percentileP and f\_EPTF\_StatMeasure\_getStatIdx\_max\_percentileP return the index of the density and max statistics used by percentileP.

The function f\_EPTF\_StatMeasure\_setValueP\_percentileP can be used to set the p value of the statistics where the pth percentile should be calculated. The p value can be between 0.0 and 1.0. If it is set to 0.95, percentileP statistics calculates the same values as percentile95 statistics. The value 0.5 corresponds to the Median of the data values. When this function is called, the statistics is reset automatically.

The functions

* f\_EPTF\_StatMeasure\_createVarFromStat\_max\_percentileP
* f\_EPTF\_StatMeasure\_createVarFromStat\_density\_percentileP
* f\_EPTF\_StatMeasure\_createVarFromStat\_boundaries\_density\_percentileP
* f\_EPTF\_StatMeasure\_createVarFromStat\_valueP\_percentileP

can be used to create EPTF Variables from the max, the density, the boundaries and the actually used p value of the percentile95 statistics.

### Limits

The limits statistics can be created by the f\_EPTF\_StatMeasure\_newStat\_limits function. It is possible to set the limit values, the default led color, the reference variable via its arguments. The text field of the LED can show the value of the last measured data used to calculate the statistics. This can be enabled by this function also. This statistics, like the others, can be calculated automatically from an EPTF Variable. This variable can also be given for the function. The type of the variable can be integer or float.

The function f\_EPTF\_StatMeasure\_setLimits\_limits can be used to set the limit values for different LED colors. When it is called the statistics is reset.

The function f\_EPTF\_StatMeasure\_getLimits\_limits returns the limits currently used to calculate the limits statistics.