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

EPTF CLL Variable, Function Description

Contents

[1 Introduction 2](#_Toc453592890)

[1.1 Revision history 2](#_Toc453592891)

[1.2 How to read this Document 2](#_Toc453592892)

[1.3 Scope 2](#_Toc453592893)

[1.4 Recommended way of reading 2](#_Toc453592894)

[1.5 Typographical conventions 2](#_Toc453592895)

[1.6 Abbreviations 3](#_Toc453592896)

[1.7 Terminology 3](#_Toc453592897)

[2 General Description 4](#_Toc453592898)

[2.1 Variable creation 4](#_Toc453592899)

[2.2 Type of providers 5](#_Toc453592900)

[2.3 Supported value types 5](#_Toc453592901)

[3 Functional Interface 6](#_Toc453592902)

[3.1 Naming Conventions 6](#_Toc453592903)

[3.2 Public Functions 6](#_Toc453592904)

[3.2.1 Initialization 6](#_Toc453592905)

[3.2.2 Variable creation 6](#_Toc453592906)

[3.2.3 Get/Set functions 7](#_Toc453592907)

[3.2.4 Subscribing 9](#_Toc453592908)

[3.2.5 Unsubscribing 12](#_Toc453592909)

[3.2.6 The calc function 13](#_Toc453592910)

[3.2.7 The guard functions 13](#_Toc453592911)

[3.2.8 The postProc functions 14](#_Toc453592912)

[3.2.9 Refresh Content 14](#_Toc453592913)

[3.2.10 Adjust Content 14](#_Toc453592914)

[3.2.11 Disabling/Enabling adjustContent on subscribers 15](#_Toc453592915)

[3.2.12 Saving/Loading from configuration data 16](#_Toc453592916)

[3.2.13 String conversion 17](#_Toc453592917)

[3.2.14 Cleanup 18](#_Toc453592918)

[3.3 The Refresh Mechanism 18](#_Toc453592919)

[3.4 Summary Table of all public functions for EPTF Variables 18](#_Toc453592920)

[4 References 22](#_Toc453592921)

# Introduction

## Revision history

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Rev | Characteristics | Prepared |
| 2007-11-26 | PA1 | First draft version | ETHJGI |
| 2007-12-05 | PA2 | Updated after review | ETHJGI |
| 2008-02-14 | PA3 | FloatList, newVar added | ETHJGI |
| 2008-11-07 | PB1 | T\_Data-3\_Variable | ETHJGI |
| 2008-11-11 | PB2 | Updated after review | ETHJGI |
| 2010-06-28 | PC1 | Custom refresh rate | ETHJGI |
| 2011-02-07 | PD1 | CharstringList variable type added | ETHJGI |
| 2011-05-24 | PE1 | The adjustable flag is introduced | ETHJGI |
| 2012-01-18 | PF1 | Settable MaxWaitTime for ByeAck | ETHJGI |
| 2012-07-02 | PG1 | String conversion functions added | ETHJGI |
| 2016-05-09 | PH1 | 3.2.4.2 Remote subscription chapter expanded | ESZILSZ |
| 2016-06-09 | H | Updated for release | ESZILSZ |

## How to read this Document

This is the Function Description for the Variable feature of the Ericsson Performance Test Framework (TitanSim), EPTF Core Library (CLL). EPTF CLL is developed for the TTCN-3 [1] Toolset with TITAN [2]. For more information on the EPTF CLL please consult the Product Revision Information.

## Scope

This document is to specify the content and functionality of the Variable feature of the EPTF CLL.

## Recommended way of reading

The readers are supposed to get familiar with the concept and functionalities of EPTF CLL [3]. 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.

## Abbreviations

CLL Core Library

EPTF Ericsson Performance Test Framework

TitanSim New synonym for the EPTF Framework

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

## Terminology

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

*Calculated variables* Variables that are automatically calculated from other EPTF Variables

*Calculating function (calcFn)* This function is used in EPTF Variables with local provider to calculate its value automatically.

*Content* The actual value of the EPTF Variable

*Direct content* Value type of the EPTF Variables

*Parameters* Variables that can have no provider

*Provider* The Variable that affects the value of the given EPTF Variable

*Provider, local* An EPTF Variable ah a local provider if it is calculated from other local EPTF Variable

*Provider, remote* An EPTF Variable has a remote provider if its value is the mirror of the value of the remote EPTF Variable

*Remote subscribers* Variables that are links to EPTF Variables in remote components. They can be used to access the value of Variables in remote components.

*Referenced content* Reference to a value. Contains the memory address of the component variable

*Subscription, local* If an EPTF Variable is a local provider of some other EPTF Variable, that Variable is the local subscriber

*Subscription, remote* If an EPTF Variable has a remote provider, it is the remote subscriber of that Variable

*Variable* An enhanced component variable that makes it possible to access the values of other Variables in remote compoents

*Variable, direct* An EPTF Variable that creates the storage place for the value also

*Variable, reference* An EPTF Variable without the storage place. It contains only reference to the component variable that stores the value

# General Description

This document specifies the Variable feature of the EPTF CLL.

The EPTF Variable feature makes it possible to

* access component variables in remote components
* create automatically calculated variables triggered by the refresh of other parameters

To be able to use EPTF Variable, the user component should extend the EPTF\_Var\_CT component.

## Variable creation

EPTF Variables can be assigned to existing component variables or can be created on their own. The two different classes based on this are:

* Direct content
* Referenced content

The EPTF Variables with direct content store the value of the Variable in their content without using any external component variables. The place of storage is created automatically in the background. However, the Variables with referenced content need a component variable where their values are actually stored. These kinds of EPTF Variables contain references to the component variables in their content. This reference is basically the memory address of the referred component variable. Changing the value of these variables will change the value of the referenced component variable and vice versa. But the EPTF Variables that depend on this Variable are only notified if the change of the value is registered into the system.

## Type of providers

The value of EPTF Variables can depend on the values of other EPTF Variables. We call those Variables which determine the value of the EPTF Variable providers. The Variable is notified automatically about the change of the value of any of its providers by the built-in refresh mechanism.

The EPTF Variables can have one of the following types of providers:

* no provider (the Variable is parameter)
* local (the value of the Variable is calculated from other Variables)
* remote (the Variable is a link to another EPTF Variable in a remote component)

With using EPTF Variables, it is possible to view the value of another EPTF Variable from a different (remote) component. We call the EPTF Variable which shows the value of another EPTF Variable in a remote component the remote subscriber, and the one which provides the value, the provider. The EPTF Variables can only have one remote provider.

In case of changing the value of an EPTF Variable, the value of any calculated Variables that depend on this Variable (subscribers) is updated automatically. Also the value of all remote subscribers to this EPTF Variable is updated by the refresh mechanism built into the EPTF Variable feature. The rate of the refresh can be specified during subscription. The period of the refresh is calculated as the minimal refresh interval multiplied by the refresh rate.

The values of the EPTF Variables are returned and can be accessed via the type ETPF\_Var\_DirectContent, which is simply the union of types supported by the EPTF Variable feature.

## Supported value types

The EPTF Variables can be created with the following types:

* integer
* float
* boolean
* charstring
* octetstring
* hexstring
* bitstring
* EPTF\_IntegerList
* EPTF\_FloatList
* EPTF\_CharstringList
* EPTF\_StatusLED

# Functional Interface

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

## Naming Conventions

All functions have the prefix f\_EPTF\_Var\_.

## Public Functions

### Initialization

Before using the EPTF Variable functions the

f\_EPTF\_Var\_init\_CT(…)

function should be called. This initializes the EPTF Variable feature.

### Variable creation

Only explained for the integer type. It is the same for the other types respectively.

#### Creating a direct-content Variable

To create an integer type Variable, call the function

f\_EPTF\_Var\_newInt(name, initial value, index)

The given initial value will be used to initialize the Variable. Its index is returned in the *index* argument. This index can be used later to access this Variable. Any EPTF Variables are uniquely defined by the component reference where they are created and this index.

All EPTF Variables should have a unique name on the component they are created. The name cannot be an empty string. The uniqueness is checked before creation. If the name is not unique an error is generated and the execution is stopped.

To create an EPTF Variable for all value types using the same function

f\_EPTF\_Var\_newVar(name, initial value) return index

The type of the initial value in this case contains the type information of the variable. The index of the variable id returned in the return value.

These functions create EPTF Variables with direct content.

#### Creating a referenced-content Variable

To create an EPTF Variable as a reference to an existing integer type component variable call the function:

f\_EPTF\_Var\_newIntRef(name,intCompVar,index)

The intCompVar argument is the component variable to which the new EPTF Variable is assigned, and the index returns the index of the Variable which can be used to access this Variable later.

### Get/Set functions

#### Get/Set value

To get the value of an EPTF Variable the functions

f\_EPTF\_Var\_getIntValue(index) return integer

f\_EPTF\_Var\_getContent(index, currentContent)

can be used. The first returns the integer value, the second returns it as a direct-content. These functions work for direct- and referenced EPTF Variables.

To modify the content of the EPTF Variable with the given index call:

f\_EPTF\_Var\_setContent(index, directContent)

The directContent argument specifies the new value as a direct content. For referenced EPTF Variable the value of the referred component Variable is modified.

This function is equivalent with the assignment operator of component variables:

v\_pi := 3.14;

v\_2pi := 2.0\*v\_pi;

or just the value of the component variables (=get).

The above example can be rewritten with the get/set functions as:

f\_EPTF\_Var\_setContent(v\_pi\_index,{floatVal:=3.14});

f\_EPTF\_Var\_setContent(v\_2pi\_index,{ floatVal:=2.0\*f\_EPTF\_Var\_getFloatValue(v\_pi\_index)});

The set function checks the type of the argument and the type of the EPTF Variable. In case of type mismatch it will stop with an error.

Note, that the set function modifies the content of the EPTF Variable without invoking the refresh mechanism. After the last set-function is called it is the user’s responsibility to activate the refresh mechanism by another function call to *f\_EPTF\_Var\_refreshContent*, see Section 3.2.9.

For example instead of invoking the update mechanism for all argument change of the sum function, it is possible to update all arguments and then invoke the refresh at the end.

To change the value of an EPTF Variable and activating the refresh mechanism automatically the adjust functions should be used (see later).

#### Remote get value

To get the value of an EPTF Variable in a remote component without subscription:

f\_EPTF\_Var\_getRemoteContent(remoteCompRef, remoteVarName, directContent)

This function returns true if the content of the given Variable in the remote component is available, false otherwise. The function blocks the execution until the response arrives.

#### Get the reference of a component variable

In some cases the reference to a component variable might be needed. For example to be able to subscribe to a remote Variable when the local EPTF Variable should be a referenced Variable.

To get the reference to an integer component variable call the function:

f\_EPTF\_Var\_getIntRef (intCompVar) return octetstring

This function will return a reference as octetstring for the given component variable.

To dereference a reference as an integer:

f\_EPTF\_Var\_derefInt(compVarRef) return integer

This function returns the value as an integer that is referred by the given reference.

To modify the integer value at a reference call:

f\_EPTF\_Var\_modifyIntRefValue(compVarRef, newIntValue)

#### Get the timeline of a variable

The following function can be used the get the timeline of values of a variable:

f\_EPTF\_Var\_getTimeLine(varIdx,timeline)

The timeline of the EPTF Variable with varIdx is returned in the timeline argument of the function.

#### Set the maximum waiting time for ByeAck messages

The following function sets the waiting time for ByeAck messages during the cleanup process:

f\_EPTF\_Var\_setMaxWaitTimeForByeAck(waitTime)

This function has to be called before the cleanup process started.

### Subscribing

There are two types of subscription: local and remote.

#### Local subscription

Local subscription is like defining a Variable that is calculated from other EPTF Variables. To define this type of ETPF Variable, the calculating function (calcFn) together with its arguments (list of indexes of other EPTF variables) has to be specified.

The calcFn function has the signature:

EPTF\_Var\_fcb\_CalcFn(in integer index, in EPTF\_IntegerList args, inout EPTF\_Var\_DirectContent returnValue)

When the calcFn function is activated, the returnValue is set to the actual value of the EPTF Variable. The calcFn function should modify that value.

To define the provider for an EPTF variable, call the function:

f\_EPTF\_Var\_subscribeLocal(index, provider)

where provider has the type EPTF\_Var\_LocalProvider, and specifies the: {funcRef, arglist} pair. Here funcref is a function pointer to the function type above, and its arglist is given by the integer list at arglist.

Now the EPTF Variable with index is calculated by calling the calcFn given by the provider argument.

#### Remote subscription

Subscribing for an EPTF Variable in a remote component makes it possible to view or adjust the value of the subscribed variable from the current component. Subscribed variables can be understood as a link to the original variable. It is not possible to subscribe more than once to the same remote EPTF Variable.

There are four different ways to subscribe to a remote EPTF Variable:

* **realtime**: the subscriber variable is notified about the refresh of the provider value instantly

For the other subscription modes the subscriber is notified about the refresh periodically. The length of the period can be set by the module parameter tsp\_EPTF\_Var\_SyncInterval, which is set to 10 secs as default.

The periodic (buffered) subscription modes are:

* **timeLine**: all refreshed values are buffered during the synchronization interval. At the end of the interval they are sent in one message to the remote components containing the timelines of the values
* **sampled**: only the last refreshed value (the value that was set by the refresh mechanism) is sent to the subscriber
* **sampledAtSync**: the value of the Variable is sampled at the synchronization event, and this value is sent to the subscriber. In this case, the values set by the refresh mechanism are not used, refreshing the provider variable is not necessary.
* **pull**: the value of the variable is only sent to the subscriber if it is requested. When the value of the pull mode subscriber variable is needed, a request is sent to the provider and the value is returned to the subscriber. The value of the subscriber variable is updated with the received value. When the value of the subscriber is adjusted, a request is sent to the provider.

It is possible to set the timeline of the EPTF Variable manually using the function

f\_EPTF\_Var\_updateTimeLine(index, contentNew, timeStamp)

The timeline of the variable can be determined by the function

f\_EPTF\_Var\_getTimeLine(index, timeLine, refreshRateId)

The refreshRateId determines the refresh rate for which the timeline should be returned. If -1 is given (or not specified), the timeline for the refresh rate of the provider variable is returned. The 0 value corresponds to the default refresh interval.

Also the timeline of the Variable can be send to all of its buffered remote subscribers independently of the synchronization interval:

f\_EPTF\_Var\_sendTimeLine(index)

This function resets the timeline of the given EPTF Variable before exit.

To subscribe for a remote EPTF Variable, call the function:

f\_EPTF\_Var\_subscribeRemote(compRef,remoteProviderName, subscriptionMode,index,localName, refreshRate)

The remoteProviderName argument specifies the name of the EPTF Variable to subscribe for on the component compRef. This function call will create a local EPTF Variable with direct-content automatically. The index of the new Variable is returned in the index argument. The localName argument can be used to specify the name of the new Variable. If not specified the name is auto-generated as:

“remoteCompName.remoteProviderName”

The refreshRate parameter specifies the rate of refresh for this subscription. The refresh period is calculated as the refreshRate multiplied by the minSyncInterval. If this argument is not specified (or <=0 value is given) the default refresh rate is used with period defined by the syncInterval parameter. The minSyncInterval and syncInterval parameters can be set by the functions:

f\_EPTF\_Var\_setSyncInterval and f\_EPTF\_Var\_setMinSyncInterval

To subscribe for a remote EPTF Variable, but create a referenced EPTF Variable call:

f\_EPTF\_Var\_subscribeRemoteRef(varReference,remoteCompRef, remoteProviderName, subscriptionMode,index,localName, refreshRate)

This function works as the previous, but it needs an additional argument for the reference to the local component variable. To get this kind of reference call the function f\_EPTF\_Var\_getIntRef (see above) for an integer component variable and use the intType to select the type of the reference. In this way the value of the remote EPTF Variable will be written into the local component variable.

The alternative way to subscribe as a reference is to use the component variable itself. For integer type component variable:

f\_EPTF\_Var\_subscribeRemoteIntRef(intCompVar, remoteCompRef, remoteProviderName, subscriptionMode,index,localName, refreshRate)

To re-subscribe an EPTF Variable with a given index after it was unsubscribed from its remote provider call the function:

f\_EPTF\_Var\_resubscribeRemote(compRef,providerName,subscrMode, index,localName, refreshRate)

If no localName is specified the previous name is used. To generate new name automatically for the resubscriber variable set its name to empty string before resubscribing and do not specify the localName.

### Unsubscribing

To unsubscribe from the provider(s) of the EPTF Variable call:

f\_EPTF\_Var\_unsubscribe (index)

The index argument is the index of the local EPTF Variable that will be unsubscribed from its provider(s). The local variable becomes an EPTF Variable with empty provider (i.e. a parameter).

This function removes the given EPTF Variable from the subscriber list of its providers and removes its provider (i.e. the calcFn function in case of local provider).

The function does nothing for an EPTF Variable with empty provider.

### The calc function

As stated above, an EPTF Variable that has a local provider has a user given calc function. This calc function is used by the library to calculate its new value, when needed from the values of other EPTF Variables. This calc function can also be called by the user anytime by calling the function:

f\_EPTF\_Var\_callCalcFn(index,retVal)

The index identifies the EPTF Variable of which calc function is called. The return value of the calc function is set into the direct-content type argument retVal. Calling this function has not got any effect on the EPTF Variables themselves (i.e. no refresh mechanism activated, value is not written into the content).

This function can be applied to any EPTF Variables, but if they not have a calc function it will return the current content as direct-content.

### The guard functions

The user defined guard functions can be used to check whether a new value which is to be set into the EPTF Variable (with empty provider) is acceptable or not. If acceptable by all the guard functions the new value is written into the content.

The guard function are called automatically by the library, but can be called by the user anytime with the function:

f\_EPTF\_Var\_callGuardFns(index,newContent) return true

The index argument determines the EPTF Variable, newContent is the new value of the EPTF Variable in direct-content format which is to be tested by the guards. If all guard functions accept the new value this function returns true, otherwise false.

The function before calling the guards, performs a type-checking. If the type of the new value does not match with the previous type of the EPTF Variable the function returns false without calling the guards.

This function can be applied to any EPTF Variables, but if they not have a guard function it will return true.

New guard function can be added to an EPTF Variable by:

f\_EPTF\_Var\_addGuardFn(index, guardFn)

An existing guard function can be removed by the function:

f\_EPTF\_Var\_removeGuardFn(index, guardFn)

### The postProc functions

The user defined postProc functions can be used to perform some task after the content of an EPTF Variable is updated. These functions are called automatically by the library, but can be called by the user anytime with the function:

f\_EPTF\_Var\_callPostProcFns(index)

The index argument selects the EPTF Variable.

### Refresh Content

When the value of an EPTF Variable was modified with a set function or an assignment, the EPTF library should be notified about this change in order to the subscribers of that variable could be updated by the library.

To initiate the refresh mechanism of the library for a given EPTF Variable call the function:

f\_EPTF\_Var\_refreshContent(index)

This function will call the calc function if present (i.e. for EPTF Variables with local providers) and initiate the refresh mechanism, which in turn will notify all subscribers of the EPTF Variable.

### Adjust Content

To set a new value into the EPTF Variable and initiate the refresh mechanism, call the function:

f\_EPTF\_Var\_adjustContent(index, newValue, handlerFn)

The index argument specifies the EPTF Variable and the newValue argument its new value to be set, as a direct-content. This function is non-blocking. When the adjust response arrives the call-back function handlerFn is called with the results of the adjustment.

This function will stop with an error for EPTF Variables that have a local provider. For those variables the f\_EPTF\_Var\_refreshContent function should be used.

For EPTF Variables that have an empty provider it calls the guard functions. If all guards accept the new value it initiates the refresh mechanism.

If the Variable is a subscriber of a remote EPTF Variable (it has a remote provider) it sends an adjust request message to it. After its guards were called, a response is sent to the sender of the adjust request. If the adjust is accepted all subscribers of the owner will be refreshed.

The blocking version of the above function is:

f\_EPTF\_Var\_adjustContent\_Blocking(index, newValue)

This blocks the execution until adjust response is available. It returns true upon successful adjust.

To adjust the value of an EPTF Variable on a remote component without subscription, use the function:

f\_EPTF\_Var\_adjustRemoteContent(remoteCompRef, remoteVarName, newValue, handlerFn)

The blocking version of this function is:

f\_EPTF\_Var\_adjustRemoteContent\_Blocking(remoteCompRef, remoteVarName, newContent)

Returns true if adjust is successful.

### Disabling/Enabling adjustContent on subscribers

Adjusting the content of a Variable can be disabled or enabled on subscribers with the following function:

f\_EPTF\_Var\_setSubsCanAdjust(in integer pl\_idx, in boolean pl\_subsCanAdjust)

If the pl\_subsCanAdjust parameter is set to false, the content of the Variable cannot be changed on all subscribers (and their subscribers) of the original variable with index pl\_idx. This function can be called for Variables that have no providers. If f\_EPTF\_Var\_adjustContent is called for a subscriber, it will call the handlerFn function (if specified) with a false result.

To enable the adjust in all subscibers the pl\_subsCanAdjust parameter should be set to true.

The current state of the ‘adjustable’ flag can be retrieved by the function:

f\_EPTF\_Var\_getSubsCanAdjust(in integer pl\_idx)

This can be called for every Variable.

If the state of the adjustable’ flag of a Variable changes, functions registered by

f\_EPTF\_Var\_addSubsCanAdjustNotifyFn(in integer pl\_idx, in EPTF\_Var\_GenericFn pl\_subsCanAdjustNotifyFn)

for that Variable will be called. Any call-back function can be removed by

f\_EPTF\_Var\_removeSubsCanAdjustNotifyFn(in integer pl\_idx, in EPTF\_Var\_GenericFn pl\_subsCanAdjustNotifyFn)

### Saving/Loading from configuration data

An EPTF Variable can be registered to be saved with the function:

f\_EPTF\_Var\_registerVarToSave(in charstring pl\_name)

The variable with the name specified in the pl\_name parameter will be saved by the save functions.

The opposite of this function (i.e. to remove a variable from the list of saved variables) is the function:

f\_EPTF\_Var\_deregisterVarFromSave(in charstring pl\_name)

To save all registered variables to a charstring in a module parameter format:

f\_EPTF\_Var\_saveVars2tsp(

in charstring pl\_moduleParName := "tsp\_EPTF\_Var\_cfg",

in boolean pl\_removeSelfNameFromParamName := true

) runs on EPTF\_Var\_CT return charstring

The parameter pl\_moduleParName specifies the name of the module parameter that will be assigned to the saved list of EPTF Variable values. The removeSelfNameFromParamName parameter can be used to set the name of the variables in the configuration file. The value false means that the name in the output will be the same as the name of the variables. Whereas the value true means that they will be prefixed by the selfName of the component.

The same string without the module parameter header is generated by the function

f\_EPTF\_Var\_saveVars(in boolean pl\_removeSelfNameFromParamName := true)

To save the module parameter to a file, the following function can be used:

function f\_EPTF\_Var\_save(

in charstring pl\_fileName := tsp\_EPTF\_Var\_snapshotFilename,

in charstring pl\_timeStampFormat := tsp\_EPTF\_Var\_snapshotTimestamp,

in charstring pl\_moduleparName := "",

in boolean pl\_removeSelfNameFromParamName := false

) runs on EPTF\_Var\_CT return charstring

It saves the string generated by the function f\_EPTF\_Var\_saveVars2tsp to the file named pl\_fileName+pl\_timeStampFormat. The extension of the file shall be set in the pl\_timeStampFormat parameter. If the pl\_moduleparName is not specified the module parameter name will be set to “tsp\_EPTF\_Var\_<selfName>\_cfg”.

To load an EPTF Variable from configuration data call the function

f\_EPTF\_Var\_loadVarFromCfg(name, cfg data, remove selfName)

This function searches the given Variable name in the config data and adjusts the value of the existing variable to that in the config.

The name of the Variable in the config data should be the same as:

* The name of the existing Variable – if the *remove selfName* parameter false
* selfName + “.” + name of the existing Variable – if *remove selfName* is true (default), i.e. the selfName (the name of the component) is removed from the name in the cfg.

The index of the loaded Variable is returned in its return value.

To automatically create the Variable with initial value read from the cfg:

f\_EPTF\_Var\_createVarFromCfg(name, cfg data, remove selfName)

This function creates a new EPTF Variable with the given name if it is found in cfg, and also sets its initial value. The index of the new Variable is returned in the return value.

### String conversion

The content of any EPTF\_Variable can be converted to charstring, and their content can be set using a value given as a charstring.

The following function converts the current value of an EPTF\_Variable to charstring and returns it in its return value:

f\_EPTF\_Var\_content2str(in integer pl\_idx) return charstring

The inverse of the function above sets (adjusts) the value of the EPTF\_Variable from a value in charstring format. If the conversion is successful, it returns 0, otherwise a nonzero error code is returned:

f\_EPTF\_Var\_str2content(in integer pl\_idx, in charstring pl\_valueStr) return integer

### Cleanup

Before exiting the component that extends EPTF\_Var\_CT the function

f\_EPTF\_Var\_cleanup\_CT()

should be called to properly exit the EPTF\_Var\_CT component. This function unsubscribes all EPTF\_Variables that are subscribers of EPTF Variables in the exiting component and closes all connections related to the EPTF Variable feature.

This function should never be called explicitly, because it is registered as a clean-up function in the function f\_EPTF\_Var\_init\_CT() and will be called automatically at exit.

## The Refresh Mechanism

The refresh mechanism implemented for the EPTF Variables makes it possible to update the value of all subscribers of an EPTF Variable if its value was changed.

The refresh mechanism is initiated automatically if the value of the EPTF Variable was changed via the function f\_EPTF\_Var\_adjustContent, or can be initiated explicitly by calling the function f\_EPTF\_Var\_refreshContent. If the f\_EPTF\_Var\_adjustContent function is called, the new value is first tested with the guardFns. If all of them accept the new value only then initiated the refresh mechanism.

The following steps are executed during the refresh mechanism for the EPTF Variables:

1. The new value is written into the content of the EPTF Variable
2. The timeline is updated (if needed)
3. The postProc functions are called
4. All subscribers are notified

## Summary Table of all public functions for EPTF Variables

Table 1. Summary of EPTF Variable functions

| Function name | Description for different provider types | | |
| --- | --- | --- | --- |
| Empty | local | remote |
| f\_EPTF\_Var\_newInt, f\_EPTF\_Var\_newIntRef  f\_EPTF\_Var\_newVar | Creates this type of Variable | - | - |
| f\_EPTF\_Var\_subscribeLocal | Sets the provider to local provider | Results in an error | Results in an error |
| f\_EPTF\_Var\_subscribeRemote,  f\_EPTF\_Var\_subscribeRemoteRef | - | - | Creates this type of variable, sets the provider to remote |
| f\_EPTF\_Var\_unsubscribe | Does nothing | Removes the local provider an sets the provider to empty | Removes the remote provider and sets the provider to empty |
| f\_EPTF\_Var\_getContent | Returns the content | | |
| f\_EPTF\_Var\_setContent | Overwrites the previous content with the new value | | |
| calcFn | - | Called when content needs to be refreshed | - |
| guardFns | Called when adjust is needed | Never called | Never called |
| postProcFns | Called after new value was set (by refresh or adjust) | Called after new value was set (by refresh or adjust) | Never called |
| f\_EPTF\_Var\_refreshContent | Initiates the refresh mechanism: notifies all subscribers | Calls the calcFn and initiates the refresh mechanism: notifies all subscribers | Initiates the refresh mechanism: notifies all subscribers. It is possible to call the function, but unnecessary because all subscribers are notified automatically when the value changes |
| f\_EPTF\_Var\_adjustContent | Sets new value to the content, but before that the guards are called. The refresh mechanism is initiated: PostProcFns are called at the end | Not allowed. The function call results in an error. | The new value of the variable is forwarded to the provider in an adjust message. If fails adjustNack will be received. |
| f\_EPTF\_Var\_setSubsCanAdjust | Sets if adjust is allowed in subscribers of this Variable | Not allowed | Not allowed |
| f\_EPTF\_Var\_getSubsCanAdjust | Returns the value of the ‘adjustable’ flag | Returns the value of the ‘adjustable’ flag | Returns the value of the ‘adjustable’ flag |
| f\_EPTF\_Var\_addSubsCanAdjustNotifyFn  f\_EPTF\_Var\_removeSubsCanAdjustNotifyFn  f\_EPTF\_Var\_callSubsCanAdjustNotifyFns | Add, remove and call functions that are called when the value of the ‘adjustable’ flag changes. | Not allowed. | Add, remove and call functions that are called when the value of the ‘adjustable’ flag changes. |
| f\_EPTF\_Var\_loadVarFromCfg | Sets new value from cfg to the content, but before that the guards are called. The refresh mechanism is initiated: PostProcFns are called at the end | Not allowed. The function call results in an error. | The new value of the variable is forwarded to the provider in an adjust message. If fails adjustNack will be received. |
| f\_EPTF\_Var\_createVarFromCfg | Creates a new Variable and sets its value from cfg to the content | - | - |
| f\_EPTF\_Var\_registerVarToSave, f\_EPTF\_Var\_deregisterVarFromSave f\_EPTF\_Var\_saveVars2tsp, f\_EPTF\_Var\_saveVars, f\_EPTF\_Var\_save | The variables specified are registered to save,  and saved | | |
| f\_EPTF\_Var\_content2str | Returns the value of the variable as a charsting | | |
| f\_EPTF\_Var\_str2content | Sets the value of the variable from a string value and notifies all subscibers | Not allowed | Sets the value of the provider variable from a string value and notifies all subscibers. |
| f\_EPTF\_Var\_getTimeLine | Returns the timeline of a variable | | |
| f\_EPTF\_Var\_updateTimeLine | Update the timeline of a variable | - | - |
| f\_EPTF\_Var\_sendTimeLine | Send the timeline of a variable to all of its subscribers | | |
| f\_EPTF\_Var\_setSyncInterval | Sets the period of the default synchronization method | | |
| f\_EPTF\_Var\_setMinSyncInterval | Sets the minimal period (elementary time step) for the custom synchronization refresh rates. Its default is the Scheduler’s tsp\_EPTF\_ELEMENTARY\_TIMESTEP\_PARAM value | | |
| f\_EPTF\_Var\_getRefreshPeriod | Returns the refresh period for a given refresh rate id | | |
| f\_EPTF\_Var\_getRefreshRate | Returns the refresh rate for a given refresh rate id | | |
| f\_EPTF\_Var\_getRefreshRateId | Returns the refresh rate id for a given refresh rate | | |
| f\_EPTF\_Var\_setMaxWaitTimeForByeAck | To set the maximal waiting time for ByeAck messages during cleanup | | |

# References

1. ETSI ES 201 873-1 v3.2.1 (2007-02)  
   The Testing and Test Control Notation version 3. [Part 1: Core Language](http://www.etsi.org/deliver/etsi_es/201800_201899/20187301/03.02.01_60/es_20187301v030201p.pdf)
2. 1/198 17-CRL 113 200/5 Uen  
   User Guide for the TITAN TTCN-3 Test Executor
3. 155 17-CNL 113 512 Uen   
   EPTF CLL for TTCN-3 toolset with TITAN, Function Specification
4. EPTF CLL for TTCN-3 toolset with TITAN, [Reference Guide](http://ttcn.ericsson.se/TCC_Releases/Libraries/EPTF_Core_Library_CNL113512/doc/apidoc/html/index.html)