Discover the Experience

EB tresos® AutoCore Generic 6 CAL documentation

Module release 1.2.2

EB tresos® AutoCore Generic 6 CAL documentation

Elektrobit Automotive GmbH
Am Wolfsmantel 46
91058 Erlangen, Germany
Phone: +49 9131 7701 0
Fax: +49 9131 7701 6333
E-mail: info.automotive@elektrobit.com

Technical support

Europe	Japan	USA
Phone: +49 9131 7701 6060	Phone: +81 3 5577 6110	Phone: +1 888 346 3813

Support URL

http://automotive.elektrobit.com/support

Legal notice

Table of Contents

1. Overview

Welcome to the Cal (Cal) release notes and documentation.

This document provides:

- Chapter 2, Cal module release notes
- Chapter 3, Cal user's guide: concept information and configuration instructions
- Chapter 4, Cal module references: configuration parameters and the application programming interface

2. Cal module release notes

- AUTOSAR R4.0 Rev 3
- AUTOSAR SWS document version: 1.2.0
- Module version: 1.2.2.B117106
- Supplier: Elektrobit Automotive GmbH

2.1. Change log

This chapter lists the changes between different versions.

Module version 1.2.2

2013-11-28

- ASCCAL-83 Fixed known issue: Redefinition of the element with the short name path /AUTOSAR_Cal in the Bswmd.

Module version 1.2.1

2013-10-11

- ASCCAL-72 Fixed known issue: Cal random service functions use incorrect calls to the Cpl API functions.

Module version 1.2.0

2013-06-14

- Added compression and decompression feature.

Module version 1.1.0

2013-02-14

Module version 1.0.3

2012-11-16

- File Cal_Version.h is added.
- MISRA violations fixed.

Module version 1.0.2

2012-08-30

- Added Cpl Stub.
- Fixed generated primitive names for KeyExchangeCalcPubVal and KeyExchangeCalcSecret services.

Module version 1.0.1

2012-05-08

- First production ready release.

Module version 1.0.0

2012-03-31

- Initial prototype release.

2.2. New features

- Compression and Decompression Services

  Description:

  The Cal module provides interfaces for Compression and Decompression services.

2.3. EB-specific enhancements

This chapter lists the enhancements provided by the module.

- This module provides no EB-specific enhancements.

2.4. Deviations

This chapter lists the deviations of the module from the AUTOSAR standard.

- The following Cal services are not supported:
  - SymEncrypt
  - AsymEncrypt
  - AsymDecrypt
  - SignatureGenerate
  - Checksum
  - SymKeyWrapSym
  - SymKeyWrapAsym
  - AsymPrivateKeyExtract
  - AsymPrivateKeyWrapSym
  - AsymPrivateKeyWrapAsym
- The AUTOSAR_SWS_CryptoAbstractionLibrary document specifies some requirements, which are not applicable to the Cal module but to the underlying Cpl module. Therefore these requirements are not implemented in the Cal. It is assumed that the used Cpl fullfills these requirements.
- The CryptoAbstractionLibrary does not perform Inter Module Checks, because only the Cpl library is imported. The Cal does not refer to a defined version of the Cpl. The CryptoAbstractionLibrary does not include any other modules.

2.5. Limitations

This chapter lists the limitations of the module.

- For this module no limitations are known.

3. Cal user's guide

3.1. Overview

This user's guide describes the `Crypto Abstraction Library` (`Cal`) module and explains the basic functionality of the `Cal`. It also decsribes the modules necessary to configure the `Cal` module. The `Cal` module reference provides further information on configuring the `Cal` itself.

Note that this user's guide is intended for readers who have good knowledge of AUTOSAR and about the purpose of the `Cal`. The information provided here should help you to integrate the `Cal` in your AUTOSAR project.

- Section Section 3.2, “Background information” provides an overview of the basic functionality of the `Cal`.
- Section Section 3.3, “Configuring Cal” provides information on related modules that are needed in order to configure the `Cal`.
- For details on configuring the `Cal` itself, refer to the parameter descriptions provided in the `Cal` module reference Chapter 4, Cal module references.

3.2. Background information

The `Crypto Abstraction Library` is a library, which makes cryptographic primitives available for applications. It defines an interface to use cryptographic routines.

The `Cal` is called from an application, a complex device driver or a software component, and calls the `Cryptographic Primitive Library` (`Cpl`) itself. The cryptographic algorithms are implemented in the `Cpl`.

Furthermore the `Cal` provides synchronous services to enable a standardized access to basic cryptographic functionalities for all software modules and software components.

As the `Cal` is a library, it is not related to a certain layer of the AUTOSAR Layered Software Architecture. The services of the `Cal` are always executed in the context of the calling function.

The `Cal` supports reentrant access to all services and allows parallel access to different services.

3.2.1. Functional overview

The `Cal` module provides access to the following cryptographic services:

- Asymmetrical key extraction
- Compression
- Decompression
- Hash
- Key derivation
- Key exchange
- Mac generation
- Mac verification
- Random
- Signature verification
- Symmetrical block encryption
- Symmetrical block decryption
- Symmetrical decryption
- Symmetrical key extraction

3.3. Configuring Cal

To be able to use the `Cal`, it has to be integrated into the software environment.

The `Cal` calls the primitive functions from the `Cpl`.

To use the `Cal` module, you must configure additional a `Cpl` module. For each `Cal` service, which contains a configuration, there must be a corresponding `Cpl` configuration.

Configure a service (e.g. symmetrical block decryption) of the `Cal` as follows:

- Open the `Cal` module configuration.
- Switch to the tab of the service to configure.
- Click the button to create a service configuration.

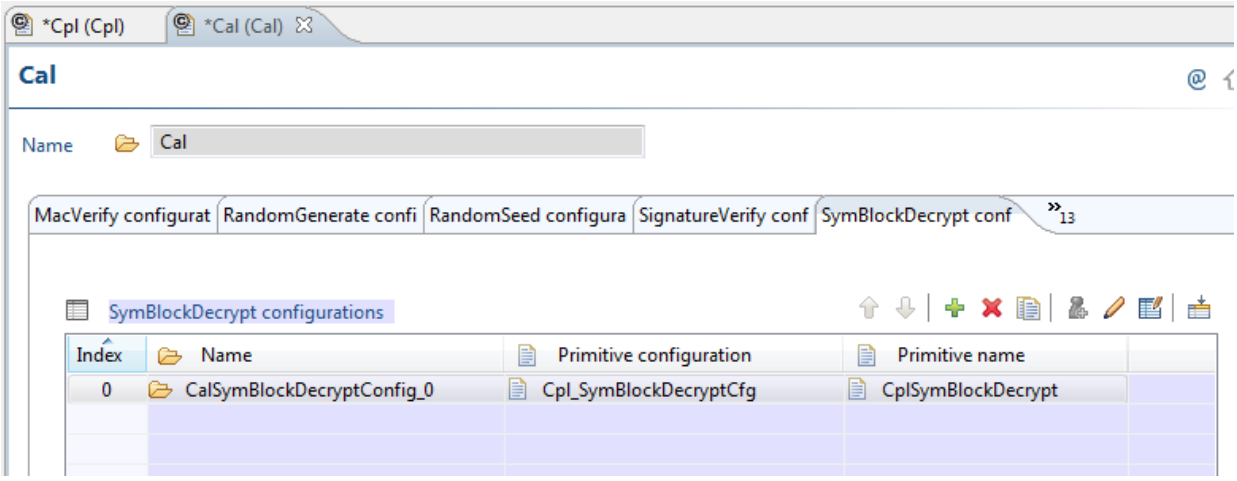

Figure 3.1. Cal service tab

- The *Name* of the configuration shall be used by the application, when the service APIs are called for this service configuration.
- The *Primitive configuration* must hold the name of the `Cpl` configuration, which shall be used by the service.
- The *Primitive name* must hold the name of the `Cpl` primitive, which shall be used.
- Switch to the tab *General*.

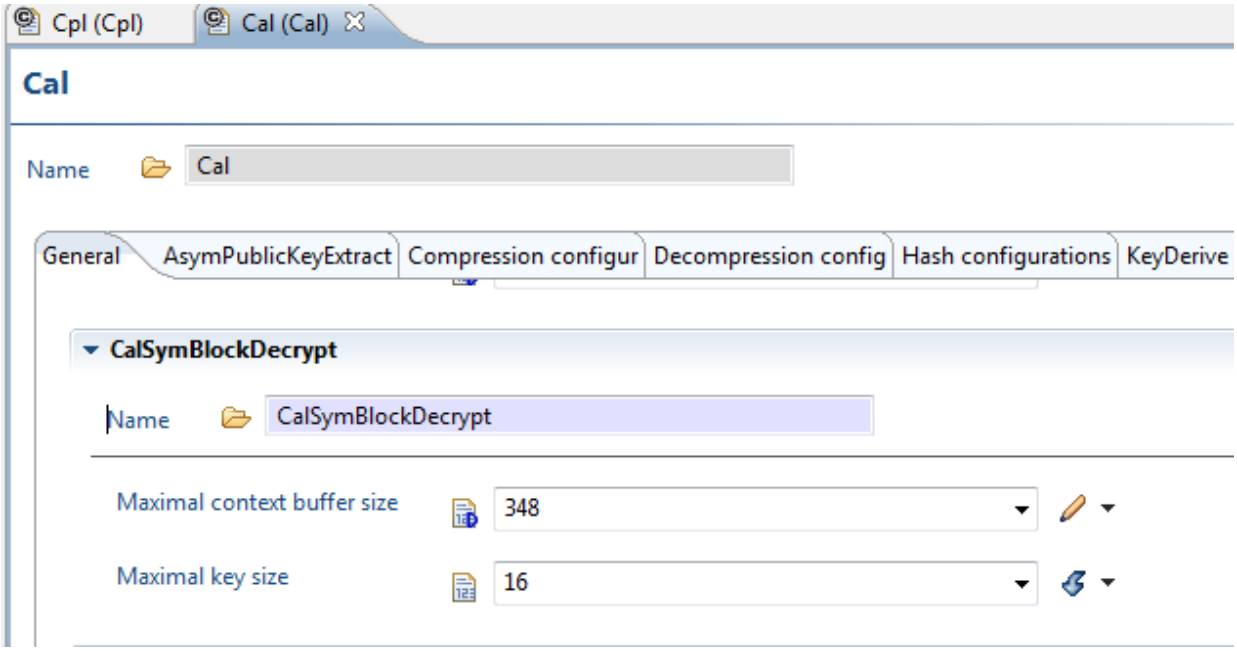

Figure 3.2. Cal General tab

- In the service container the *Maximal context buffer size* needs to be configured. This size is the maximum size of all context buffers (in bytes) of all `Cpl` primitves which are referenced by the service.

- Some service containers contain the parameter *Maximal key size*. If present, this parameter must be set to the maximum key size (in bytes) of all referenced `Cpl` primitives which implement this service.

4. Cal module references

4.1. Configuration parameters

Containers included		
Container name	**Multiplicity**	**Description**
CalAsymPublicKeyExtract	1..1	The configuration of AsymPublicKeyExtract services
CalCompress	1..1	The configuration of CalCompress services.
CalDecompress	1..1	The configuration of CalDecompress services.
CalGeneral	1..1	
CalHash	1..1	The Hash configuration.
CalKeyDerive	1..1	The configuration of KeyDerive services
CalKeyExchangeCalcPubVal	1..1	The configuration of KeyExchangeCalcPubVal services
CalKeyExchangeCalcSecret	1..1	The configuration of KeyExchangeCalcSecret services
CalMacGenerate	1..1	The configuration of MacGenerate primitives
CalMacVerify	1..1	The configuration of MacVerify services
CalRandomGenerate	1..1	The configuration of RandomGenerate services
CalRandomSeed	1..1	The configuration of RandomSeed services
CalSignatureVerify	1..1	The configuration of SignatureVerify services
CalSymBlockDecrypt	1..1	The SymBlockDecrypt configuration.
CalSymBlockEncrypt	1..1	The SymBlockEncrypt configuration.
CalSymDecrypt	1..1	The configuration of the SymDecrypt services.
CalSymKeyExtract	1..1	The configuration of SymKeyExtract services
CommonPublishedInformation	1..1	**Label:** Common Published Information Common container, aggregated by all modules. It contains published information about vendor and versions.

Parameters included	
Parameter name	**Multiplicity**

Parameters included	
IMPLEMENTATION_CONFIG_VARIANT	1..1

Parameter Name	IMPLEMENTATION_CONFIG_VARIANT
Label	Config Variant
Description	Select the configuration variant. Currently only PreCompile is supported.
Multiplicity	1..1
Type	ENUMERATION
Default value	VariantPreCompile
Range	VariantPreCompile

4.1.1. CalAsymPublicKeyExtract

Containers included		
Container name	Multiplicity	Description
CalAsymPublicKeyExtractConfig	0..32	

Parameters included	
Parameter name	Multiplicity
CalAsymPublicKeyExtractMaxCtxBufByteSize	1..1
CalAsymPublicKeyExtractMaxKeySize	1..1

Parameter Name	CalAsymPublicKeyExtractMaxCtxBufByteSize
Label	Maximal context buffer size
Description	Maximal size of all context buffers used for extraction of asymmetrical public keys. Must be at least the size, of the maximal required context buffer size of the primitives used for extraction of asymmetrical public keys.
Multiplicity	1..1
Type	INTEGER
Default value	100

Range	>=1	
	<=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CalAsymPublicKeyExtractMaxKeySize**	
Label	Maximal key size	
Description	Maximal size in bytes of all keys used in all CPL primitives which are used for extraction of asymmetrical public keys.	
Multiplicity	1..1	
Type	INTEGER	
Default value	1	
Range	>=1	
	<=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.2. CalAsymPublicKeyExtractConfig

Parameters included	
Parameter name	**Multiplicity**
CalAsymPublicKeyExtractInitConfiguration	1..1
CalAsymPublicKeyExtractPrimitiveName	1..1

Parameter Name	**CalAsymPublicKeyExtractInitConfiguration**	
Label	Primitive configuration	
Description	The configuration of the primitive	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile

Origin	AUTOSAR_ECUC

Parameter Name	**CalAsymPublicKeyExtractPrimitiveName**	
Label	Primitive name	
Description	Name of the Cpl primitive.	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.3. CalCompress

Containers included		
Container name	**Multiplicity**	**Description**
CalCompressConfig	0..32	

Parameters included	
Parameter name	**Multiplicity**
CalCompressMaxCtxBufByteSize	1..1

Parameter Name	**CalCompressMaxCtxBufByteSize**	
Label	Maximal context buffer size	
Description	The size of a context buffer (in bytes) used for compression. The maximal size must be at least the size of the largest context buffer of all the primitives used for compression.	
Multiplicity	1..1	
Type	INTEGER	
Default value	100	
Range	>=1	
	<=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile

Origin	AUTOSAR_ECUC

4.1.4. CalCompressConfig

Parameters included	
Parameter name	**Multiplicity**
CalCompressInitConfiguration	1..1
CalCompressPrimitiveName	1..1

Parameter Name	**CalCompressInitConfiguration**	
Label	Primitive configuration	
Description	The configuration of the primitive.	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CalCompressPrimitiveName**	
Label	Primitive name	
Description	Name of the Cpl primitive.	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.5. CalDecompress

Containers included		
Container name	**Multiplicity**	**Description**
CalDecompressConfig	0..32	

Parameters included	
Parameter name	**Multiplicity**
CalDecompressMaxCtxBufByteSize	1..1

Parameter Name	CalDecompressMaxCtxBufByteSize	
Label	Maximal context buffer size	
Description	The size of a context buffer (in bytes) used for decompression. The maximal size must be at least the size of the largest context buffer of all the primitives used for decompression.	
Multiplicity	1..1	
Type	INTEGER	
Default value	100	
Range	>=1 --- <=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.6. CalDecompressConfig

Parameters included	
Parameter name	**Multiplicity**
CalDecompressInitConfiguration	1..1
CalDecompressPrimitiveName	1..1

Parameter Name	CalDecompressInitConfiguration	
Label	Primitive configuration	
Description	The configuration of the primitive.	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile

Origin	AUTOSAR_ECUC

Parameter Name	**CalDecompressPrimitiveName**	
Label	Primitive name	
Description	Name of the Cpl primitive.	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.7. CalGeneral

Parameters included	
Parameter name	**Multiplicity**
CalMaxAlignScalarType	1..1
CalVersionInfoApi	1..1

Parameter Name	**CalMaxAlignScalarType**	
Label	Type with maximal alignment restrictions	
Description	Type with maximal alignment restrictions	
Multiplicity	1..1	
Type	STRING	
Default value	uint32	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CalVersionInfoApi**
Label	Version info API
Description	Version info API
Multiplicity	1..1
Type	BOOLEAN
Default value	false

Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.8. CalHash

Containers included		
Container name	**Multiplicity**	**Description**
CalHashConfig	0..32	

Parameters included	
Parameter name	**Multiplicity**
CalHashMaxCtxBufByteSize	1..1

Parameter Name	**CalHashMaxCtxBufByteSize**	
Label	Maximal context buffer size	
Description	Maximal size of all context buffers used for hashing. Must by at least the size, of the maximal required context buffer size of the primitives used for hashing.	
Multiplicity	1..1	
Type	INTEGER	
Default value	316	
Range	>=1 --- <=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.9. CalHashConfig

Parameters included	
Parameter name	**Multiplicity**
CalHashInitConfiguration	1..1

Parameters included	
CalHashPrimitiveName	1..1

Parameter Name	CalHashInitConfiguration	
Label	Primitive configuration	
Description	The configuration of the primitive	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	CalHashPrimitiveName	
Label	Primitive name	
Description	Name of the Cpl primitive.	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.10. CalKeyDerive

Containers included		
Container name	**Multiplicity**	**Description**
CalKeyDeriveConfig	0..32	

Parameters included	
Parameter name	**Multiplicity**
CalKeyDeriveMaxCtxBufByteSize	1..1
CalKeyDeriveMaxKeySize	1..1

Parameter Name	CalKeyDeriveMaxCtxBufByteSize
Label	Maximal context buffer size

Description	Maximal size of all context buffers used for key derivation. Must be at least the size, of the maximal required context buffer size of the primitives used for key derivation.	
Multiplicity	1..1	
Type	INTEGER	
Default value	100	
Range	>=1	
	<=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CalKeyDeriveMaxKeySize**
Label	Maximal key size
Description	Maximal size of all keys used in KeyDerive computations.
Multiplicity	1..1
Type	INTEGER
Default value	1
Range	>=1
	<=4294967295
Configuration class	**PreCompile:** VariantPreCompile
Origin	AUTOSAR_ECUC

4.1.11. CalKeyDeriveConfig

Parameters included	
Parameter name	**Multiplicity**
CalKeyDeriveInitConfiguration	1..1
CalKeyDerivePrimitiveName	1..1

Parameter Name	**CalKeyDeriveInitConfiguration**
Label	Primitive configuration

Description	The configuration of the primitive	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CalKeyDerivePrimitiveName**
Label	Primitive name
Description	Name of the Cpl primitive.
Multiplicity	1..1
Type	STRING
Configuration class	**PreCompile:** VariantPreCompile
Origin	AUTOSAR_ECUC

4.1.12. CalKeyExchangeCalcPubVal

Containers included		
Container name	**Multiplicity**	**Description**
CalKeyExchangeCalcPubValConfig	0..32	

Parameters included	
Parameter name	**Multiplicity**
CalKeyExchangeCalcPubValMaxBaseTypeSize	1..1
CalKeyExchangeCalcPubValMaxCtxBufByteSize	1..1
CalKeyExchangeCalcPubValMaxPrivateTypeSize	1..1

Parameter Name	**CalKeyExchangeCalcPubValMaxBaseTypeSize**
Label	Maximal base type size

Description	The maximum length, in bytes, of all base types used in all CPL primitives which implement a public value calculation.	
Multiplicity	1..1	
Type	INTEGER	
Default value	1	
Range	>=1 --- <=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CalKeyExchangeCalcPubValMaxCtxBufByteSize**	
Label	Maximal context buffer size	
Description	Maximal size of all context buffers used for calculation of the public value during a key exchange. Must be at least the size, of the maximal required context buffer size of the used primitives.	
Multiplicity	1..1	
Type	INTEGER	
Default value	100	
Range	>=1 --- <=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CalKeyExchangeCalcPubValMaxPrivateTypeSize**	
Label	Maximal private type size	
Description	The maximum length, in bytes, of all private types used in all CPL primitives which implement a public value calculation.	
Multiplicity	1..1	
Type	INTEGER	
Default value	1	
Range	>=1 --- <=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile

Origin	AUTOSAR_ECUC

4.1.13. CalKeyExchangeCalcPubValConfig

Parameters included	
Parameter name	**Multiplicity**
CalKeyExchangeCalcPubValInitConfiguration	1..1
CalKeyExchangeCalcPubValPrimitiveName	1..1

Parameter Name	**CalKeyExchangeCalcPubValInitConfiguration**	
Label	Primitive configuration	
Description	The configuration of the primitive	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CalKeyExchangeCalcPubValPrimitiveName**	
Label	Primitive name	
Description	Name of the Cpl primitive.	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.14. CalKeyExchangeCalcSecret

Containers included		
Container name	**Multiplicity**	**Description**
CalKeyExchangeCalcSecretConfig	0..32	

Parameters included	
Parameter name	**Multiplicity**
CalKeyExchangeCalcSecretMaxBaseTypeSize	1..1
CalKeyExchangeCalcSecretMaxCtxBufByteSize	1..1
CalKeyExchangeCalcSecretMaxPrivateTypeSize	1..1

Parameter Name	**CalKeyExchangeCalcSecretMaxBaseTypeSize**	
Label	Maximal base type size	
Description	The maximum length, in bytes, of all base types used in all CPL primitives which implement a public value calculation.	
Multiplicity	1..1	
Type	INTEGER	
Default value	1	
Range	>=1 \<=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CalKeyExchangeCalcSecretMaxCtxBufByteSize**	
Label	Maximal context buffer size	
Description	Maximal size of all context buffers used for calculation of the secret during a key exchange. Must be at least the size, of the maximal required context buffer size of the used primitives.	
Multiplicity	1..1	
Type	INTEGER	
Default value	100	
Range	>=1 \<=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	CalKeyExchangeCalcSecretMaxPrivateTypeSize	
Label	Maximal private type size	
Description	The maximum length, in bytes, of all private types used in all CPL primitives which implement a Secret calculation for a key exchange.	
Multiplicity	1..1	
Type	INTEGER	
Default value	1	
Range	>=1	
	<=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.15. CalKeyExchangeCalcSecretConfig

Parameters included	
Parameter name	**Multiplicity**
CalKeyExchangeCalcSecretInitConfiguration	1..1
CalKeyExchangeCalcSecretPrimitiveName	1..1

Parameter Name	CalKeyExchangeCalcSecretInitConfiguration	
Label	Primitive configuration	
Description	The configuration of the primitive	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	CalKeyExchangeCalcSecretPrimitiveName
Label	Primitive name
Description	Name of the Cpl primitive.
Multiplicity	1..1

Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.16. CalMacGenerate

Containers included		
Container name	**Multiplicity**	**Description**
CalMacGenerateConfig	0..32	

Parameters included	
Parameter name	**Multiplicity**
CalMacGenerateMaxCtxBufByteSize	1..1
CalMacGenerateMaxKeySize	1..1

Parameter Name	**CalMacGenerateMaxCtxBufByteSize**	
Label	Maximal context buffer size	
Description	Maximal size of all context buffers used for Mac generation. Must be at least the size, of the maximal required context buffer size of the primitives used for Mac generation.	
Multiplicity	1..1	
Type	INTEGER	
Default value	404	
Range	>=1 --- <=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CalMacGenerateMaxKeySize**
Label	Maximal key size
Description	Maximal size of all keys used in MAC computations

Multiplicity	1..1	
Type	INTEGER	
Default value	1	
Range	>=1 <=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.17. CalMacGenerateConfig

Parameters included	
Parameter name	**Multiplicity**
CalMacGenerateInitConfiguration	1..1
CalMacGeneratePrimitiveName	1..1

Parameter Name	**CalMacGenerateInitConfiguration**	
Label	Primitive configuration	
Description	The configuration of the primitive	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CalMacGeneratePrimitiveName**	
Label	Primitive name	
Description	Name of the Cpl primitive.	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.18. CalMacVerify

Containers included		
Container name	**Multiplicity**	**Description**
CalMacVerifyConfig	0..32	

Parameters included	
Parameter name	**Multiplicity**
CalMacVerifyMaxCtxBufByteSize	1..1
CalMacVerifyMaxKeySize	1..1

Parameter Name	**CalMacVerifyMaxCtxBufByteSize**	
Label	Maximal context buffer size	
Description	Maximal size of all context buffers used for Mac verification. Must be at least the size, of the maximal required context buffer size of the primitives used for Mac verification.	
Multiplicity	1..1	
Type	INTEGER	
Default value	404	
Range	>=1 --- <=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CalMacVerifyMaxKeySize**	
Label	Maximal key size	
Description	Maximal size of all keys used in MAC computations	
Multiplicity	1..1	
Type	INTEGER	
Default value	1	
Range	>=1 --- <=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile

Origin	AUTOSAR_ECUC

4.1.19. CalMacVerifyConfig

Parameters included	
Parameter name	**Multiplicity**
CalMacVerifyInitConfiguration	1..1
CalMacVerifyPrimitiveName	1..1

Parameter Name	**CalMacVerifyInitConfiguration**	
Label	Primitive configuration	
Description	The configuration of the primitive	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CalMacVerifyPrimitiveName**	
Label	Primitive name	
Description	Name of the Cpl primitive.	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.20. CalRandomGenerate

Containers included		
Container name	**Multiplicity**	**Description**
CalRandomGenerateConfig	0..32	

4.1.21. CalRandomGenerateConfig

Parameters included	
Parameter name	**Multiplicity**
CalRandomGenerateInitConfiguration	1..1
CalRandomGeneratePrimitiveName	1..1

Parameter Name	CalRandomGenerateInitConfiguration	
Label	Primitive configuration	
Description	The configuration of the primitive	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	CalRandomGeneratePrimitiveName	
Label	Primitive name	
Description	Name of the Cpl primitive.	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.22. CalRandomSeed

Containers included		
Container name	**Multiplicity**	**Description**
CalRandomSeedConfig	0..32	

Parameters included	
Parameter name	**Multiplicity**
CalRandomMaxCtxBufByteSize	1..1

Parameter Name	CalRandomMaxCtxBufByteSize	
Label	Maximal context buffer size	
Description	Maximal size of all context buffers used for random number generators. Must be at least the size, of the maximal required context buffer size of the primitives used for seeding and generating of random numbers.	
Multiplicity	1..1	
Type	INTEGER	
Default value	24	
Range	>=1	
	<=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.23. CalRandomSeedConfig

Parameters included	
Parameter name	**Multiplicity**
CalRandomSeedInitConfiguration	1..1
CalRandomSeedPrimitiveName	1..1

Parameter Name	CalRandomSeedInitConfiguration	
Label	Primitive configuration	
Description	The configuration of the primitive	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	CalRandomSeedPrimitiveName
Label	Primitive name
Description	Name of the Cpl primitive.
Multiplicity	1..1

Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.24. CalSignatureVerify

Containers included		
Container name	**Multiplicity**	**Description**
CalSignatureVerifyConfig	0..32	

Parameters included	
Parameter name	**Multiplicity**
CalSignatureVerifyMaxCtxBufByteSize	1..1
CalSignatureVerifyMaxKeySize	1..1

Parameter Name	**CalSignatureVerifyMaxCtxBufByteSize**	
Label	Maximal context buffer size	
Description	Maximal size of all context buffers used for signature verification. Must be at least the size, of the maximal required context buffer size of the primitives used for signature verification.	
Multiplicity	1..1	
Type	INTEGER	
Default value	100	
Range	>=1 --- <=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CalSignatureVerifyMaxKeySize**
Label	Maximal key size
Description	Maximal size in bytes of all keys used in all CPL primitives which are used for signature verification.

Multiplicity	1..1	
Type	INTEGER	
Default value	1	
Range	>=1 <=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.25. CalSignatureVerifyConfig

Parameters included	
Parameter name	**Multiplicity**
CalSignatureVerifyInitConfiguration	1..1
CalSignatureVerifyPrimitiveName	1..1

Parameter Name	**CalSignatureVerifyInitConfiguration**	
Label	Primitive configuration	
Description	The configuration of the primitive	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CalSignatureVerifyPrimitiveName**	
Label	Primitive name	
Description	Name of the Cpl primitive.	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.26. CalSymBlockDecrypt

Containers included		
Container name	**Multiplicity**	**Description**
CalSymBlockDecryptConfig	0..32	

Parameters included	
Parameter name	**Multiplicity**
CalSymBlockDecryptMaxCtxBufByteSize	1..1
CalSymBlockDecryptMaxKeySize	1..1

Parameter Name	**CalSymBlockDecryptMaxCtxBufByteSize**
Label	Maximal context buffer size
Description	Maximal size of all context buffers used in symmetrical decryption. Must be at least the size, of the maximal required context buffer size of the primitives used for symmetrical decryption.
Multiplicity	1..1
Type	INTEGER
Default value	348
Range	>=1 <=4294967295
Configuration class	**PreCompile:** VariantPreCompile
Origin	AUTOSAR_ECUC

Parameter Name	**CalSymBlockDecryptMaxKeySize**
Label	Maximal key size
Description	Maximal size of all keys used in symmetrical decryption
Multiplicity	1..1
Type	INTEGER
Default value	1
Range	>=1 <=4294967295

Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.27. CalSymBlockDecryptConfig

Parameters included	
Parameter name	**Multiplicity**
CalSymBlockDecryptInitConfiguration	1..1
CalSymBlockDecryptPrimitiveName	1..1

Parameter Name	**CalSymBlockDecryptInitConfiguration**	
Label	Primitive configuration	
Description	The configuration of the primitive	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CalSymBlockDecryptPrimitiveName**	
Label	Primitive name	
Description	Name of the Cpl primitive.	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.28. CalSymBlockEncrypt

Containers included		
Container name	**Multiplicity**	**Description**

Containers included		
CalSymBlockEncryptConfig	0..32	

Parameters included	
Parameter name	**Multiplicity**
CalSymBlockEncryptMaxCtxBufByteSize	1..1
CalSymBlockEncryptMaxKeySize	1..1

Parameter Name	**CalSymBlockEncryptMaxCtxBufByteSize**	
Label	Maximal context buffer size	
Description	Maximal size of all context buffers used in symmetrical encryption. Must by at least the size, of the maximal required context buffer size of the primitives used for symmetrical encryption.	
Multiplicity	1..1	
Type	INTEGER	
Default value	348	
Range	>=1 --- <=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CalSymBlockEncryptMaxKeySize**	
Label	Maximal key size	
Description	Maximal size of all keys used in symmetrical encryption	
Multiplicity	1..1	
Type	INTEGER	
Default value	16	
Range	>=1 --- <=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.29. CalSymBlockEncryptConfig

Parameters included	
Parameter name	**Multiplicity**
CalSymBlockEncryptInitConfiguration	1..1
CalSymBlockEncryptPrimitiveName	1..1

Parameter Name	CalSymBlockEncryptInitConfiguration	
Label	Primitive configuration	
Description	The configuration of the primitive	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	CalSymBlockEncryptPrimitiveName	
Label	Primitive name	
Description	Name of the Cpl primitive.	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.30. CalSymDecrypt

Containers included		
Container name	**Multiplicity**	**Description**
CalSymDecryptConfig	0..32	

Parameters included	
Parameter name	**Multiplicity**
CalSymDecryptMaxCtxBufByteSize	1..1

Parameters included	
CalSymDecryptMaxKeySize	1..1

Parameter Name	CalSymDecryptMaxCtxBufByteSize	
Label	Maximal context buffer size	
Description	Maximal size of all context buffers used for symmetrical decryption. Must be at least the size, of the maximal required context buffer size of the primitives used for symmetrical decryption.	
Multiplicity	1..1	
Type	INTEGER	
Default value	428	
Range	>=1 | <=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	CalSymDecryptMaxKeySize	
Label	Maximal key size	
Description	Maximal size of all keys used for symmetrical decryption.	
Multiplicity	1..1	
Type	INTEGER	
Default value	1	
Range	>=1 | <=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.31. CalSymDecryptConfig

Parameters included	
Parameter name	**Multiplicity**
CalSymDecryptInitConfiguration	1..1

Parameters included	
CalSymDecryptPrimitiveName	1..1

Parameter Name	CalSymDecryptInitConfiguration	
Label	Primitive configuration	
Description	The configuration of the primitive	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	CalSymDecryptPrimitiveName	
Label	Primitive name	
Description	Name of the Cpl primitive.	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.32. CalSymKeyExtract

Containers included		
Container name	**Multiplicity**	**Description**
CalSymKeyExtractConfig	0..32	

Parameters included	
Parameter name	**Multiplicity**
CalSymKeyExtractMaxCtxBufByteSize	1..1
CalSymKeyExtractMaxKeySize	1..1

Parameter Name	CalSymKeyExtractMaxCtxBufByteSize

Label	Maximal context buffer size	
Description	Maximal size of all context buffers used for symmetrical key extraction. Must be at least the size, of the maximal required context buffer size of the primitives used for symmetrical key extraction.	
Multiplicity	1..1	
Type	INTEGER	
Default value	72	
Range	>=1 --- <=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CalSymKeyExtractMaxKeySize**	
Label	Maximal key size	
Description	Maximal size of all keys used in SymKeyExtract computations	
Multiplicity	1..1	
Type	INTEGER	
Default value	1	
Range	>=1 --- <=4294967295	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.33. CalSymKeyExtractConfig

Parameters included	
Parameter name	**Multiplicity**
CalSymKeyExtractInitConfiguration	1..1
CalSymKeyExtractPrimitiveName	1..1

Parameter Name	**CalSymKeyExtractInitConfiguration**

Label	Primitive configuration	
Description	The configuration of the primitive	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**CalSymKeyExtractPrimitiveName**	
Label	Primitive name	
Description	Name of the Cpl primitive.	
Multiplicity	1..1	
Type	STRING	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.1.34. CommonPublishedInformation

Parameters included	
Parameter name	**Multiplicity**
ArMajorVersion	1..1
ArMinorVersion	1..1
ArPatchVersion	1..1
SwMajorVersion	1..1
SwMinorVersion	1..1
SwPatchVersion	1..1
ModuleId	1..1
VendorId	1..1
Release	1..1

Parameter Name	**ArMajorVersion**
Label	AUTOSAR Major Version
Description	Major version number of AUTOSAR specification on which the appropriate implementation is based on.

Multiplicity	1..1
Type	INTEGER_LABEL
Default value	1
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**ArMinorVersion**
Label	AUTOSAR Minor Version
Description	Minor version number of AUTOSAR specification on which the appropriate implementation is based on.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	2
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**ArPatchVersion**
Label	AUTOSAR Patch Version
Description	Patch level version number of AUTOSAR specification on which the appropriate implementation is based on.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**SwMajorVersion**
Label	Software Major Version
Description	Major version number of the vendor specific implementation of the module.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	1
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	SwMinorVersion
Label	Software Minor Version
Description	Minor version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	2
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	SwPatchVersion
Label	Software Patch Version
Description	Patch level version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	2
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	ModuleId
Label	Numeric Module ID
Description	Module ID of this module from Module List
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	206
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	VendorId
Label	Vendor ID
Description	Vendor ID of the dedicated implementation of this module according to the AUTOSAR vendor list
Multiplicity	1..1

Type	INTEGER_LABEL
Default value	1
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**Release**
Label	Release Information
Multiplicity	1..1
Type	STRING_LABEL
Default value	
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

4.2. Application programming interface (API)

4.2.1. Type definitions

4.2.1.1. Cal_AsymPublicKeyExtractConfigType

Purpose	Structure representing the configuration of the asymmetrical public key extraction algorithm.	
Type	struct	
Members	`Cal_ConfigIdType ConfigId`	Identifier for the current configuration.
	`Cal_ReturnType(* PrimitiveStartFct`	Pointer to the start function of the underlying Cpl primitive.
	`Cal_ReturnType(* PrimitiveUpdateFct`	Pointer to the update function of the underlying Cpl primitive.
	`Cal_ReturnType(* PrimitiveFinishFct`	Pointer to the finish function of the underlying Cpl primitive.
	`const void * PrimitiveConfigPtr`	Pointer to a Cpl configuration.

4.2.1.2. Cal_AsymPublicKeyExtractCtxBufType

Purpose	type representing the required context buffer for asymmetrical public key extraction.
Type	`Cal_AlignType[CAL_ASYMPUBKEYEXTRACT_CTX_BUF_SIZE]`

4.2.1.3. Cal_AsymPublicKeyType

Purpose	Type of asymmetrical public keys.	
Type	struct	
Members	`uint32 length`	Length information of the key data.
	`Cal_AlignType data`	key data.

4.2.1.4. Cal_CompressConfigType

Purpose	Structure representing the configuration of the compression algorithm.	
Type	struct	
Members	`Cal_ConfigIdType ConfigId`	Identifier for the current configuration.
	`Cal_ReturnType(* PrimitiveS-tartFct`	Pointer to the start function of the underly-ing Cpl primitive.
	`Cal_ReturnType(* PrimitiveUp-dateFct`	Pointer to the update function of the un-derlying Cpl primitive.
	`Cal_ReturnType(* PrimitiveFin-ishFct`	Pointer to the finish function of the under-lying Cpl primitive.
	`const void * PrimitiveConfigPtr`	Pointer to a Cpl configuration.

4.2.1.5. Cal_CompressCtxBufType

Purpose	Context buffer for the compression service.
Type	`Cal_AlignType[CAL_COMPRESS_CTX_BUF_SIZE]`

4.2.1.6. Cal_ConfigIdType

Purpose	The type of configuration IDs.

Type	`uint16`
Description	Every configuration of a CAL service has a number which uniquely identifies it among that service's configurations. The identifier is of this type.

4.2.1.7. Cal_DecompressConfigType

Purpose	Structure representing the configuration of the decompression algorithm.	
Type	struct	
Members	`Cal_ConfigIdType ConfigId`	Identifier for the current configuration.
	`Cal_ReturnType(* PrimitiveStartFct`	Pointer to the start function of the underlying Cpl primitive.
	`Cal_ReturnType(* PrimitiveUpdateFct`	Pointer to the update function of the underlying Cpl primitive.
	`Cal_ReturnType(* PrimitiveFinishFct`	Pointer to the finish function of the underlying Cpl primitive.
	`const void * PrimitiveConfigPtr`	Pointer to a Cpl configuration.

4.2.1.8. Cal_DecompressCtxBufType

Purpose	Context buffer for the decompression service.
Type	`Cal_AlignType[CAL_DECOMPRESS_CTX_BUF_SIZE]`

4.2.1.9. Cal_HashConfigType

Purpose	Structure which contains the configuration for a hash service.	
Type	struct	
Members	`Cal_ConfigIdType ConfigId`	Identifier for the current configuration.
	`Cal_ReturnType(* PrimitiveStartFct`	Pointer to the start function of the underlying Cpl primitive.
	`Cal_ReturnType(* PrimitiveUpdateFct`	Pointer to the update function of the underlying Cpl primitive.
	`Cal_ReturnType(* PrimitiveFinishFct`	Pointer to the finish function of the underlying Cpl primitive.
	`const void * PrimitiveConfigPtr`	Pointer to a Cpl configuration.

4.2.1.10. Cal_HashCtxBufType

Purpose	Defines the context buffer for the hash services.
Type	`Cal_AlignType[CAL_HASH_CONTEXT_BUFFER_SIZE]`

4.2.1.11. Cal_KeyDeriveConfigType

Purpose	Structure representing the configuration of the key derivation algorithm.	
Type	struct	
Members	`Cal_ConfigIdType ConfigId`	Identifier for the current configuration.
	`Cal_ReturnType(* PrimitiveS-tartFct`	Pointer to the start function of the underly-ing Cpl primitive.
	`Cal_ReturnType(* PrimitiveUp-dateFct`	Pointer to the update function of the un-derlying Cpl primitive.
	`Cal_ReturnType(* PrimitiveFin-ishFct`	Pointer to the finish function of the under-lying Cpl primitive.
	`const void * PrimitiveConfigPtr`	Pointer to a Cpl configuration.

4.2.1.12. Cal_KeyDeriveCtxBufType

Purpose	type representing the required context buffer for key derivation.
Type	`Cal_AlignType[CAL_KEYDERIVE_CONTEXT_BUFFER_SIZE]`

4.2.1.13. Cal_KeyExchangeBaseType

Purpose	Structure with base type information of the key exchange protocol.	
Type	struct	
Members	`uint32 length`	Length information of key exchange base.
	`Cal_AlignType data`	key exchange base data.

4.2.1.14. Cal_KeyExchangeCalcSecretCtxBufType

Purpose	type which specifies the array size of the context buffer required for key exchange protocol.

Type	`Cal_AlignType[CAL_KEYEXCHANGE_CONTEXT_BUFFER_SIZE]`

4.2.1.15. Cal_KeyExchangePrivateType

Purpose	Structure with private type information of the key exchange protocol.	
Type	struct	
Members	`uint32 length`	Length information of key exchange private value.
	`Cal_AlignType data`	key exchange private value data.

4.2.1.16. Cal_KeyExchangePubValConfigType

Purpose	Structure which contains the configuration for a public value calculation of a key exchange protocol.	
Type	struct	
Members	`Cal_ConfigIdType ConfigId`	Identifier for the current configuration.
	`Cal_ReturnType(* PrimitiveCalcPubValFct`	Pointer to the function of the underlying Cpl primitive which calculates the public value.
	`const void * PrimitiveConfigPtr`	Pointer to a Cpl configuration.
Description	Structure representing the configuration of a public value calculation of a key exchange protocol.	

4.2.1.17. Cal_KeyExchangeSecretConfigType

Purpose	Structure which contains the configuration for a secret calculation in a Key ExchangeSeed protocol.	
Type	struct	
Members	`Cal_ConfigIdType ConfigId`	Identifier for the current configuration.
	`Cal_ReturnType(* PrimitiveCalcSecretStartFct`	
	`Cal_ReturnType(* PrimitiveCalcSecretUpdateFct`	

	`Cal_ReturnType(* PrimitiveCalcSecretFinishFct`	
	`const void * PrimitiveConfigPtr`	Pointer to a Cpl configuration.
Description	Structure representing the configuration of a secret calculation in a Key ExchangeSeed protocol.	

4.2.1.18. Cal_MacGenerateConfigType

Purpose	Structure representing the configuration of the Mac generation algorithm.	
Type	struct	
Members	`Cal_ConfigIdType ConfigId`	Identifier for the current configuration.
	`Cal_ReturnType(* PrimitiveStartFct`	Pointer to the start function of the underlying Cpl primitive.
	`Cal_ReturnType(* PrimitiveUpdateFct`	Pointer to the update function of the underlying Cpl primitive.
	`Cal_ReturnType(* PrimitiveFinishFct`	Pointer to the finish function of the underlying Cpl primitive.
	`const void * PrimitiveConfigPtr`	Pointer to a Cpl configuration.

4.2.1.19. Cal_MacGenerateCtxBufType

Purpose	type which specifies the array size of the context buffer required for the Mac generation.
Type	`Cal_AlignType[CAL_MACGENERATE_CTX_BUF_SIZE]`

4.2.1.20. Cal_MacVerifyConfigType

Purpose	Structure representing the configuration of the Mac verification algorithm.	
Type	struct	
Members	`Cal_ConfigIdType ConfigId`	Identifier for the current configuration.
	`Cal_ReturnType(* PrimitiveStartFct`	Pointer to the start function of the underlying Cpl primitive.
	`Cal_ReturnType(* PrimitiveUpdateFct`	Pointer to the update function of the underlying Cpl primitive.

	`Cal_ReturnType(* PrimitiveFinishFct`	Pointer to the finish function of the underlying Cpl primitive.
	`const void * PrimitiveConfigPtr`	Pointer to a Cpl configuration.

4.2.1.21. Cal_MacVerifyCtxBufType

Purpose	type which specifies the array size of the context buffer required for the Mac verification.
Type	`Cal_AlignType[CAL_MACVERIFY_CONTEXT_BUFFER_SIZE]`

4.2.1.22. Cal_RandomCtxBufType

Purpose	type which specifies the array size of the context buffer required for random number seeding and generating.
Type	`Cal_AlignType[CAL_RANDOM_CONTEXT_BUFFER_SIZE]`

4.2.1.23. Cal_RandomGenerateConfigType

Purpose	Structure representing the configuration of the random generation algorithm.	
Type	struct	
Members	`Cal_ConfigIdType ConfigId`	Identifier for the current configuration.
	`Cal_ReturnType(* PrimitiveGenerateFct`	Pointer to the random generation function of the underlying Cpl primitive.
	`const void * PrimitiveConfigPtr`	Pointer to a Cpl configuration.

4.2.1.24. Cal_RandomSeedConfigType

Purpose	Structure representing the configuration of the random seed algorithm.	
Type	struct	
Members	`Cal_ConfigIdType ConfigId`	Identifier for the current configuration.
	`Cal_ReturnType(* PrimitiveSeedStartFct`	Pointer to the seed start function of the underlying Cpl primitive.
	`Cal_ReturnType(* PrimitiveSeedUpdateFct`	Pointer to the seed update function of the underlying Cpl primitive.

	`Cal_ReturnType(* PrimitiveSeedFinishFct`	Pointer to the seed finish function of the underlying Cpl primitive.
	`const void * PrimitiveConfigPtr`	Pointer to a Cpl configuration.

4.2.1.25. Cal_ReturnType

Purpose	Enumeration of the return values of the CAL.	
Type	enum	
Constants	CAL_E_OK	Operation successful.
	CAL_E_NOT_OK	Operation failed.
	CAL_E_SMALL_BUFFER	Result buffer is too small to hold the complete result.
	CAL_E_ENTROPY_EXHAUSTION	The pseudo random number generator cannot generate bytes at the moment.
	CAL_E_BUSY	API function isn't finished but yields execution focus.

4.2.1.26. Cal_SignatureVerifyConfigType

Purpose	Structure representing the configuration of the signature verification algorithm.	
Type	struct	
Members	`Cal_ConfigIdType ConfigId`	Identifier for the current configuration.
	`Cal_ReturnType(* PrimitiveStartFct`	Pointer to the start function of the underlying Cpl primitive.
	`Cal_ReturnType(* PrimitiveUpdateFct`	Pointer to the update function of the underlying Cpl primitive.
	`Cal_ReturnType(* PrimitiveFinishFct`	Pointer to the finish function of the underlying Cpl primitive.
	`const void * PrimitiveConfigPtr`	Pointer to a Cpl configuration.

4.2.1.27. Cal_SignatureVerifyCtxBufType

Purpose	type which specifies the array size of the context buffer required for signature verification.

Type	`Cal_AlignType[CAL_SIGNATUREVERIFY_CONTEXT_BUFFER_SIZE]`

4.2.1.28. Cal_SymBlockDecryptConfigType

Purpose	Structure representing the configuration of the symmetrical block decryption algorithm.	
Type	struct	
Members	`Cal_ConfigIdType ConfigId`	Identifier for the current configuration.
	`Cal_ReturnType(* PrimitiveStartFct`	Pointer to the start function of the underlying Cpl primitive.
	`Cal_ReturnType(* PrimitiveUpdateFct`	Pointer to the update function of the underlying Cpl primitive.
	`Cal_ReturnType(* PrimitiveFinishFct`	Pointer to the finish function of the underlying Cpl primitive.
	`const void * PrimitiveConfigPtr`	Pointer to a Cpl configuration.

4.2.1.29. Cal_SymBlockDecryptCtxBufType

Purpose	Context buffer for the symmetrical block decryption service.
Type	`Cal_AlignType[CAL_SYMBLOCKDECRYPT_CTX_BUF_SIZE]`

4.2.1.30. Cal_SymBlockEncryptConfigType

Purpose	Structure representing the configuration of the symmetrical block encryption algorithm.	
Type	struct	
Members	`Cal_ConfigIdType ConfigId`	Identifier for the current configuration.
	`Cal_ReturnType(* PrimitiveStartFct`	Pointer to the start function of the underlying Cpl primitive.
	`Cal_ReturnType(* PrimitiveUpdateFct`	Pointer to the update function of the underlying Cpl primitive.
	`Cal_ReturnType(* PrimitiveFinishFct`	Pointer to the finish function of the underlying Cpl primitive.
	`const void * PrimitiveConfigPtr`	Pointer to a Cpl configuration.

4.2.1.31. Cal_SymBlockEncryptCtxBufType

Purpose	Context buffer for the symmetrical block encryption service.
Type	`Cal_AlignType[CAL_SYMBLOCKENCRYPT_CTX_BUF_SIZE]`

4.2.1.32. Cal_SymDecryptConfigType

Purpose	Structure representing the configuration of the symmetrical decryption algorithm.	
Type	struct	
Members	`Cal_ConfigIdType ConfigId`	Identifier for the current configuration.
	`Cal_ReturnType(* PrimitiveS-tartFct`	Pointer to the start function of the underlying Cpl primitive.
	`Cal_ReturnType(* PrimitiveUp-dateFct`	Pointer to the update function of the underlying Cpl primitive.
	`Cal_ReturnType(* PrimitiveFin-ishFct`	Pointer to the finish function of the underlying Cpl primitive.
	`const void * PrimitiveConfigPtr`	Pointer to a Cpl configuration.

4.2.1.33. Cal_SymDecryptCtxBufType

Purpose	type representing the required context buffer for symmetrical decryption.
Type	`Cal_AlignType[CAL_SYMDECRYPT_CTX_BUF_SIZE]`

4.2.1.34. Cal_SymKeyExtractConfigType

Purpose	structure representing the configuration of the symmetrical key extraction algorithm.	
Type	struct	
Members	`Cal_ConfigIdType ConfigId`	
	`Cal_ReturnType(* PrimitiveS-tartFct`	
	`Cal_ReturnType(* PrimitiveUp-dateFct`	
	`Cal_ReturnType(* PrimitiveFin-ishFct`	

	`const void * PrimitiveConfigPtr`	

4.2.1.35. Cal_SymKeyExtractCtxBufType

Purpose	
Type	`Cal_AlignType[CAL_SYMKEYEXTRACT_CONTEXT_BUFFER_SIZE]`
Description	type representing the required context buffer for symmetrical key extraction.

4.2.1.36. Cal_SymKeyType

Purpose	Type of symmetrical keys.	
Type	struct	
Members	`uint32 length`	Length information of the key data.
	`Cal_AlignType data`	key data.

4.2.1.37. Cal_VerifyResultType

Purpose	Enumeration of the return values a signature verification.	
Type	enum	
Constants	CAL_E_VER_OK	Signature fits the given data.
	CAL_E_VER_NOT_OK	Signature does not fit the given data.

4.2.2. Objects

4.2.2.1. Cal_AsymPublicKeyExtractConfigs

Purpose	Array containing all AsymPublicKeyExtract service configurations.
Type	`const Cal_AsymPublicKeyExtractConfigType`

4.2.2.2. Cal_CompressConfigurations

Purpose	Array containing all existing compression service configurations.

Type	`const Cal_CompressConfigType`

4.2.2.3. Cal_DecompressConfigurations

Purpose	Array containing all existing decompression service configurations.
Type	`const Cal_DecompressConfigType`

4.2.2.4. Cal_HashConfigurations

Purpose	Array containg all Hash service configurations.
Type	`const Cal_HashConfigType`

4.2.2.5. Cal_KeyDeriveConfigurations

Purpose	Array containing all KeyDerive service configurations.
Type	`const Cal_KeyDeriveConfigType`

4.2.2.6. Cal_KeyExPubValConfigurations

Purpose	Array containing all configurations for the public value calculation of a key exchange protocol.
Type	`const Cal_KeyExchangePubValConfigType`

4.2.2.7. Cal_KeyExSecretConfigurations

Purpose	Array containing all configurations for the secret calculation of a key exchange protocol.
Type	`const Cal_KeyExchangeSecretConfigType`

4.2.2.8. Cal_MacGenerateConfigurations

Purpose	Array containg all configurations for the MacGenerate service.

Type	`const Cal_MacGenerateConfigType`

4.2.2.9. Cal_MacVerifyConfigurations

Purpose	Array containing all configurations of the MacVerify service.
Type	`const Cal_MacVerifyConfigType`

4.2.2.10. Cal_RandomGenConfigurations

Purpose	Random Generation configuration. An array containing the configurations of the RandomGenerate service.
Type	`const Cal_RandomGenerateConfigType`

4.2.2.11. Cal_RandomSeedConfigurations

Purpose	Random Seed configuration. An array containing the configurations of the RandomSeed service.
Type	`const Cal_RandomSeedConfigType`

4.2.2.12. Cal_SigVerifyConfigurations

Purpose	Signature verification configuration. An array containing the configurations of the signature verification service.
Type	`const Cal_SignatureVerifyConfigType`

4.2.2.13. Cal_SymBlockDecConfigurations

Purpose	Array containing all existing SymBlockDecrypt service configurations.
Type	`const Cal_SymBlockDecryptConfigType`

4.2.2.14. Cal_SymBlockEncConfigurations

Purpose	Array containg all existing configurations of the symmetrical block encryption service.

Type	`const Cal_SymBlockEncryptConfigType`

4.2.2.15. Cal_SymDecryptConfigurations

Purpose	Array containing all existing symmetrical decryption service configurations.
Type	`const Cal_SymDecryptConfigType`

4.2.2.16. Cal_SymKeyExtractConfigurations

Purpose	Array containing all SymKeyExtract service configurations.
Type	`const Cal_SymKeyExtractConfigType`

4.2.3. Functions

4.2.3.1. Cal_AsymPublicKeyExtractFinish

Purpose	Finish asymmetric public key extraction.	
Synopsis	`Cal_ReturnType` **`Cal_AsymPublicKeyExtractFinish`** `( Cal_ConfigIdType cfgId , Cal_AsymPublicKeyExtractCtxBufType contextBuffer , Cal_AsymPublicKeyType * keyPtr );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the the finishing of the asymmetric public key extraction shall be requested.
Parameters (in,out)	`contextBuffer`	A Pointer to a buffer which holds the context of the current configuration of the AsymPublicKeyExtract service.
Parameters (out)	`keyPtr`	A pointer to the buffer where the extracted key should be stored
Return Value	Error value.	
	`CAL_E_OK`	If the finish was successfully requested.
	`CAL_E_NOT_OK`	Otherwise.
Description	This function performs the finishing of the extraction of an asymmetrical public key and the storing of the key in the given buffer. The finish function of the configured primitive is called and its return value is returned.	

4.2.3.2. Cal_AsymPublicKeyExtractStart

Purpose	Start asymmetrical public key extraction.	
Synopsis	`Cal_ReturnType` **`Cal_AsymPublicKeyExtractStart`** `( Cal_ConfigIdType cfgId , Cal_AsymPublicKeyExtractCtxBufType contextBuffer );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the the start of the asymmetric public key extraction shall be requested.
Parameters (out)	`contextBuffer`	A Pointer to a buffer where the context of the current configuration of the AsymPublicKeyExtract service will be stored.
Return Value	Error value.	
	`CAL_E_OK`	If the start was successfully requested.
	`CAL_E_NOT_OK`	Otherwise.
Description	This function performs the start of the extraction of an asymmetrical public key for the given configuration. The start function of the configured primitive is called and its return value is returned.	

4.2.3.3. Cal_AsymPublicKeyExtractUpdate

Purpose	Update asymmetric public key extraction.	
Synopsis	`Cal_ReturnType` **`Cal_AsymPublicKeyExtractUpdate`** `( Cal_ConfigIdType cfgId , Cal_AsymPublicKeyExtractCtxBufType contextBuffer , const uint8 * dataPtr , uint32 dataLength );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the the update of the asymmetric public key extraction shall be requested.
	`dataPtr`	A pointer to the start of an array which contains a part of the key which should be extracted.
	`dataLength`	The amount of bytes of data.
Parameters (in,out)	`contextBuffer`	A Pointer to a buffer which holds the context of the current configuration of the AsymPublicKeyExtract service.

Return Value	Error value.	
	`CAL_E_OK`	If the update was successfully requested.
	`CAL_E_NOT_OK`	Otherwise.
Description	This function performs the update of the extraction of an asymmetrical public key for the given data. The update function of the configured primitive is called and its return value is returned.	

4.2.3.4. Cal_CompressFinish

Purpose	Finish compression computation.	
Synopsis	`Cal_ReturnType Cal_CompressFinish ( Cal_ConfigIdType cfgId , Cal_CompressCtxBufType contextBuffer , uint8 * oputBuf , uint32 * oputBufLen );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the start of the compression computation should be requested.
Parameters (in,out)	`oputBufLen`	Holds a pointer to a memory location in which the length information is stored. On calling this function this parameter shall contain the size of the buffer provided by oputBuf. On returning from this function the size of compressed data (the result) which was written to the buffer oputBuf shall be stored.
Parameters (out)	`contextBuffer`	Holds the pointer to the buffer in which the context of this service is stored.
	`oputBuf`	A pointer to the start of an array where the compressed data shall be stored.
Return Value	Error value.	
	`CAL_E_OK`	If the finish was successfully requested.
	`CAL_E_NOT_OK`	If no compression computation has been started via Cal_CompressStart(), yet.
	`CAL_E_SMALL_BUFFER`	The provided buffer oputBuf is too small to store the full result.
Description	This function requests the finishing of the compression computation. The finish function of the configured primitive is called and its return value is returned.	

4.2.3.5. Cal_CompressStart

Purpose	Start compression computation.	
Synopsis	`Cal_ReturnType Cal_CompressStart ( Cal_ConfigIdType cfgId , Cal_CompressCtxBufType contextBuffer );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the start of the compression computation should be requested.
Parameters (out)	`contextBuffer`	Holds the pointer to the buffer in which the context of this service is stored.
Return Value	Error value.	
	`CAL_E_OK`	If the start was successfully requested.
	`CAL_E_NOT_OK`	Otherwise.
Description	This function requests the start of the compression for the given configuration. The start function of the configured primitive is called and its return value is returned.	

4.2.3.6. Cal_CompressUpdate

Purpose	Update compression computation.	
Synopsis	`Cal_ReturnType Cal_CompressUpdate ( Cal_ConfigIdType cfgId , Cal_CompressCtxBufType contextBuffer , const uint8 * iputBuf , uint32 * iputBufLen , uint8 * oputBuf , uint32 * oputBufLen );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the update of the compression computation should be requested.
	`iputBuf`	Holds a pointer to the data that shall be compressed.
	`in/out]`	iputBufLen Holds a pointer to a memory location in which the length information is stored. On calling this function this parameter shall contain the size of the buffer provided by iputBuf. On returning from this function the length of data from buffer iputBuf that was already processed/compressed shall be stored.

Parameters (in,out)	`contextBuffer`	Holds the pointer to the buffer in which the context of this service is stored.
	`in/out]`	iputBufLen Holds a pointer to a memory location in which the length information is stored. On calling this function this parameter shall contain the size of the buffer provided by iputBuf. On returning from this function the length of data from buffer iputBuf that was already processed/compressed shall be stored.
	`oputBufLen`	Holds a pointer to a memory location in which the length information is stored. On calling this function this parameter shall contain the size of the buffer provided by oputBuf. On returning from this function the size of compressed data (the result) which was written to the buffer oputBuf shall be stored.
Parameters (out)	`oputBuf`	A pointer to the start of an array where the compressed data shall be stored.
Return Value	Error value.	
	`CAL_E_OK`	If the update was successfully requested.
	`CAL_E_NOT_OK`	If no compression computation has been started via Cal_CompressStart(), yet.
	`CAL_E_SMALL_BUFFER`	The provided buffer oputBuf is too small to store the full result. So not the full buffer iputBuf wasn't compressed, but only the first iputBufLen bytes (on returning).
Description	This function requests the update of the compression computation for the given data. The update function of the configured primitive is called and its return value is returned.	

4.2.3.7. Cal_DecompressFinish

Purpose	Finish decompression computation.
Synopsis	`Cal_ReturnType Cal_DecompressFinish ( Cal_ConfigIdType cfgId , Cal_DecompressCtxBufType contextBuffer , uint8 * oputBuf , uint32 * oputBufLen );`

Parameters (in)	`cfgId`	An identification of the configuration for which the start of the decompression computation should be requested.
Parameters (in,out)	`oputBufLen`	Holds a pointer to a memory location in which the length information is stored. On calling this function this parameter shall contain the size of the buffer provided by oputBuf. On returning from this function the size of decompressed data (the result) which was written to the buffer oputBuf shall be stored.
Parameters (out)	`contextBuffer`	Holds the pointer to the buffer in which the context of this service is stored.
	`oputBuf`	A pointer to the start of an array where the decompressed data shall be stored.
Return Value	Error value.	
	`CAL_E_OK`	If the finish was successfully requested.
	`CAL_E_NOT_OK`	If no decompression computation has been started via Cal_DecompressStart(), yet.
	`CAL_E_SMALL_BUFFER`	The provided buffer oputBuf is too small to store the full result.
Description	This function requests the finishing of the decompression computation. The finish function of the configured primitive is called and its return value is returned.	

4.2.3.8. Cal_DecompressStart

Purpose	Start decompression computation.	
Synopsis	`Cal_ReturnType Cal_DecompressStart ( Cal_ConfigIdType cfgId , Cal_DecompressCtxBufType contextBuffer );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the start of the decompression computation should be requested.
Parameters (out)	`contextBuffer`	Holds the pointer to the buffer in which the context of this service is stored.
Return Value	Error value.	

	CAL_E_OK	If the start was successfully requested.
	CAL_E_NOT_OK	Otherwise.
Description	This function requests the start of the decompression for the given configuration. The start function of the configured primitive is called and its return value is returned.	

4.2.3.9. Cal_DecompressUpdate

Purpose	Update decompression computation.	
Synopsis	`Cal_ReturnType Cal_DecompressUpdate ( Cal_ConfigIdType cfgId , Cal_DecompressCtxBufType contextBuffer , const uint8 * iputBuf , uint32 * iputBufLen , uint8 * oputBuf , uint32 * oputBufLen );`	
Parameters (in)	cfgId	An identification of the configuration for which the update of the decompression computation should be requested.
	iputBuf	Holds a pointer to the data that shall be decompressed.
	in/out]	iputBufLen Holds a pointer to a memory location in which the length information is stored. On calling this function this parameter shall contain the size of the buffer provided by iputBuf. On returning from this function the length of data from buffer iputBuf that was already processed/decompressed shall be stored.
Parameters (in,out)	contextBuffer	Holds the pointer to the buffer in which the context of this service is stored.
	in/out]	iputBufLen Holds a pointer to a memory location in which the length information is stored. On calling this function this parameter shall contain the size of the buffer provided by iputBuf. On returning from this function the length of data from buffer iputBuf that was already processed/decompressed shall be stored.
	oputBufLen	Holds a pointer to a memory location in which the length information is stored. On

		calling this function this parameter shall contain the size of the buffer provided by oputBuf. On returning from this function the size of decompressed data (the result) which was written to the buffer oputBuf shall be stored.
Parameters (out)	`oputBuf`	A pointer to the start of an array where the decompressed data shall be stored.
Return Value	Error value.	
	`CAL_E_OK`	If the update was successfully requested.
	`CAL_E_NOT_OK`	If no decompression computation has been started via Cal_DecompressStart(), yet.
	`CAL_E_SMALL_BUFFER`	The provided buffer oputBuf is too small to store the full result. So not the full buffer iputBuf wasn't decompressed, but only the first iputBufLen bytes (on returning).
Description	This function requests the update of the decompression computation for the given data. The update function of the configured primitive is called and its return value is returned.	

4.2.3.10. Cal_HashFinish

Purpose	Finish hash computation.	
Synopsis	`Cal_ReturnType Cal_HashFinish ( Cal_ConfigIdType cfgId , Cal_HashCtxBufType contextBuffer , uint8 * resultPtr , uint32 * resultLengthPtr , boolean truncationAllowed );`	
Parameters (in)	`cfgId`	an identification of the configuration for which the hash computation should be requested.
	`truncationAllowed`	If this flag is TRUE and the hash digest is longer than the given result buffer, the hash is truncated to the buffer length. If this flag is FALSE and the hash digest is longer than the given result buffer, an error code of CAL_E_SMALL_BUFFER is returned.

Parameters (in,out)	`contextBuffer`	A Pointer to a buffer which holds the context of the current hash computation.
	`resultLengthPtr`	a pointer to a variable which contains the maximal allowed length for the hash and where the actual length of the hash should be stored.
Parameters (out)	`resultPtr`	a pointer to the start of a buffer where the hash digest should be stored.
Return Value	error value	
	`CAL_E_OK`	If the finish was successfully requested.
	`CAL_E_NOT_OK`	If no hash computation has been started via Cal_HashStart(), yet.
Description	This function requests the finishing of the hash computation and the storing of the hash digest in the given result buffer. The finish function of the configured primitive is called and its return value is returned.	

4.2.3.11. Cal_HashStart

Purpose	Start hash computation.	
Synopsis	`Cal_ReturnType Cal_HashStart ( Cal_ConfigIdType cfgId , Cal_HashCtxBufType contextBuffer );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the start of the hash computation should be requested.
Parameters (out)	`contextBuffer`	A Pointer to a buffer where the contect of the current configuration will be stored.
Return Value	error value	
	`CAL_E_OK`	If the start was successfully requested.
	`CAL_E_NOT_OK`	If the service request failed.
Description	This function requests the start of the hash computation for the given configuration. The start function of the configured primitive is called and its return value is returned.	

4.2.3.12. Cal_HashUpdate

Purpose	Update hash computation.

Synopsis	`Cal_ReturnType Cal_HashUpdate ( Cal_ConfigIdType cfgId , Cal_HashCtxBufType contextBuffer , const uint8 * dataPtr , uint32 dataLength );`	
Parameters (in)	`cfgId`	an identification of the configuration for which the hash computation should be requested.
	`dataPtr`	a pointer to the start of an array which contains a part of the data for which the hash digest should be created.
	`dataLength`	the amount of bytes of data.
Parameters (in,out)	`contextBuffer`	A Pointer to a buffer which holds the context of the current hash computation.
Return Value	error value	
	`CAL_E_OK`	If the update was successfully requested.
	`CAL_E_NOT_OK`	If no hash computation has been started via Cal_HashStart(), yet.
Description	This function requests the update of the hash computation for the given data. The update function of the configured primitive is called and its return value is returned.	

4.2.3.13. Cal_KeyDeriveFinish

Purpose	Finish key derivation.	
Synopsis	`Cal_ReturnType Cal_KeyDeriveFinish ( Cal_ConfigIdType cfgId , Cal_KeyDeriveCtxBufType contextBuffer , Cal_SymKeyType * keyPtr );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the the finishing of the key derivation shall be requested.
Parameters (in,out)	`contextBuffer`	A Pointer to a buffer which holds the context of the current configuration of the KeyDerive service.
Parameters (out)	`keyPtr`	A pointer to the buffer where the derived key should be stored
Return Value	Error value.	
	`CAL_E_OK`	If the finish was successfully requested.
	`CAL_E_NOT_OK`	If the request failed.

Description	This function performs the finishing of the key derivation. The finish function of the configured primitive is called and its return value is returned.

4.2.3.14. Cal_KeyDeriveStart

Purpose	Start key derivation.	
Synopsis	`Cal_ReturnType Cal_KeyDeriveStart ( Cal_ConfigIdType cfgId , Cal_KeyDeriveCtxBufType contextBuffer , uint32 keyLength , uint32 iterations );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the the start of the key derivation shall be requested.
	`keyLength`	The length of the key to be derived.
	`iterations`	The number of iterations to be performed by the underlying key derivation primitive.
Parameters (out)	`contextBuffer`	A Pointer to a buffer where the context of the current configuration of the KeyDerive service will be stored.
Return Value	Error value.	
	`CAL_E_OK`	If the start was successfully requested.
	`CAL_E_NOT_OK`	Otherwise.
Description	This function performs the start of a key derivation for the given configuration. The start function of the configured primitive is called and its return value is returned.	

4.2.3.15. Cal_KeyDeriveUpdate

Purpose	Update key derivation.	
Synopsis	`Cal_ReturnType Cal_KeyDeriveUpdate ( Cal_ConfigIdType cfgId , Cal_KeyDeriveCtxBufType contextBuffer , const uint8 * passwordPtr , uint32 passwordLength , const uint8 * saltPtr , uint32 saltLength );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the the update of the key derivation shall be requested.

	passwordPtr	A pointer to the the password i.e. the original key from which to derive a new key.
	passwordLength	The length of the password in bytes.
	saltPtr	A pointer to the cryptographic salt i.e. a random number.
	saltLength	The length of the salt in bytes.
Parameters (in,out)	contextBuffer	A Pointer to a buffer which holds the context of the current configuration of the KeyDerive service.
Return Value	Error value.	
	CAL_E_OK	If the update was successfully requested.
	CAL_E_NOT_OK	If the request failed.
Description	This function performs the update of the key derivation for the given data. The update function of the configured primitive is called and its return value is returned.	

4.2.3.16. Cal_KeyExchangeCalcPubVal

Purpose	Calculate public value for the key exchange protocol.	
Synopsis	`Cal_ReturnType Cal_KeyExchangeCalcPubVal ( Cal_ConfigIdType cfgId , const Cal_KeyExchangeBaseType * basePtr , const Cal_KeyExchangePrivateType * privateValuePtr , uint8 * publicValuePtr , uint32 * publicValueLengthPtr );`	
Parameters (in)	cfgId	An identification of the configuration for which the public value calculation shall be requested.
	basePtr	A Pointer to the base information known to both users of the key exchange protocol.
	privateValuePtr	A pointer to the private information known only to the current user of the key exchange protocol.
Parameters (in,out)	publicValueLengthPtr	A pointer which holds the length information. On calling it holds the length f the buffer provided by publicValuePtr. On returning it holds the length of the calculated public value.

Parameters (out)	`publicValuePtr`	A pointer to the buffer where the public value shall be stored.
Return Value	error value	
	`CAL_E_OK`	If the public value calculation was successfully requested.
	`CAL_E_NOT_OK`	If the request failed.
	`CAL_E_SMALL_BUFFER`	If the provided buffer is too small to store the result.
Description	This function performs the calculation of a public value. The public value calculation function of the configured primitive is called and its return value is returned.	

4.2.3.17. Cal_KeyExchangeCalcSecretFinish

Purpose	Finish secret calculation.	
Synopsis	`Cal_ReturnType` **`Cal_KeyExchangeCalcSecretFinish`** `( Cal_ConfigIdType cfgId , Cal_KeyExchangeCalcSecretCtxBufType contextBuffer , uint8 * sharedSecretPtr , uint32 * sharedSecretLengthPtr , boolean TruncationIsAllowed );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the secret calculation finish shall be requested.
	`TruncationIsAllowed`	This parameter states whether the truncation of the calculated secret is allowed. TRUE: Truncation is allowed. FALSE: Truncation is not allowed.
Parameters (in,out)	`contextBuffer`	A Pointer to a buffer which holds the context of the current configuration.
	`sharedSecretLengthPtr`	Holds the Length information. On Calling the Pointer holds the length of the buffer provided by sharedSecretPtr. On returning the pointer holds the length of the shared secret.
Parameters (out)	`sharedSecretPtr`	A Pointer to a buffer which hold the calculated secret of the key exchange.
Return Value	error value	
	`CAL_E_OK`	If the secret of the key exchange was successfully calculated.

	CAL_E_NOT_OK	The request failed.
	CAL_E_SMALL_BUFFER	The provided buffer is too small to store the result and truncation is not allowed.
Description	This function performs the finishing of the secret calculation of a key exchange protocol.	

4.2.3.18. Cal_KeyExchangeCalcSecretStart

Purpose	Start calculation of the secret of a key exchange protocol.	
Synopsis	`Cal_ReturnType Cal_KeyExchangeCalcSecretStart ( Cal_ConfigIdType cfgId , Cal_KeyExchangeCalcSecretCtxBufType contextBuffer , const Cal_KeyExchangeBaseType * basePtr , const Cal_KeyExchangePrivateType * privateValuePtr );`	
Parameters (in)	cfgId	An identification of the secret calculation configuration for which the initialization shall be requested.
	basePtr	A Pointer to the base information known to both users of the key exchange protocol.
	privateValuePtr	A pointer to the private information known only to the current user of the key exchange protocol.
Parameters (out)	contextBuffer	A Pointer to a buffer where the context of the current configuration will be stored.
Return Value	error value	
	CAL_E_OK	If the secret calculation start was successfully requested.
	CAL_E_NOT_OK	If the request failed.
Description	This function performs the start of the secret calculation of a key exchange. The secret calculation start function of the configured primitive is called and its return value is returned.	

4.2.3.19. Cal_KeyExchangeCalcSecretUpdate

Purpose	Update secret calculation.

Synopsis	`Cal_ReturnType` **`Cal_KeyExchangeCalcSecretUpdate`** `( Cal_ConfigIdType cfgId , Cal_KeyExchangeCalcSecretCtxBufType contextBuffer , const uint8 * partnerPublicValuePtr , uint32 partnerPublicValueLength );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the secret calculation update shall be requested.
	`partnerPublicValuePtr`	A pointer to the public value data from the key exchange partner.
	`partnerPublicValueLength`	Holds the length of the public value data.
Parameters (in,out)	`contextBuffer`	A Pointer to a buffer which holds the context of the current configuration.
Return Value	error value	
	`CAL_E_OK`	If the secret calculation update was successfully requested.
	`CAL_E_NOT_OK`	Otherwise.
Description	This function performs the update of the KeyExchange secret calculation. The secret calculation update function of the configured primitive is called and its return value is returned.	

4.2.3.20. Cal_MacGenerateFinish

Purpose	Finish MAC generate computation.	
Synopsis	`Cal_ReturnType` **`Cal_MacGenerateFinish`** `( Cal_ConfigIdType cfgId , Cal_MacGenerateCtxBufType contextBuffer , uint8 * resultPtr , uint32 * resultLengthPtr , boolean truncationIsAllowed );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the finish of the MAC generate computation should be requested.
Parameters (in,out)	`contextBuffer`	Holds the pointer to the buffer in which the context of this service is stored.
	`resultLengthPtr`	Holds a pointer to a memory location in which the length information is stored. On calling this function this parameter shall contain the size of the buffer provided by resultPtr. When the request has finished,

		the actual length of the returned MAC shall be stored.
Parameters (out)	`resultPtr`	A pointer to the start of an array which will hold the generated MAC. If the result does not fit into the given buffer, and truncation is allowed, the result shall be truncated.
	`truncationIsAllowed`	A flag that states whether a truncation of the calculated Mac is allowed. TRUE = truncation is allowed. FALSE = truncation is not allowed.
Return Value	Error value.	
	`CAL_E_OK`	If the finish was successfully requested.
	`CAL_E_NOT_OK`	If no MAC generate computation has been started via Cal_MacGenerateStart(), yet.
	`CAL_E_SMALL_BUFFER`	If the provided buffer is too small to store the result and truncation was not allowed.
Description	This function performs the finishing of the MAC generate computation. The finish function of the configured primitive is called and its return value is returned.	

4.2.3.21. Cal_MacGenerateStart

Purpose	Start MAC generate computation.	
Synopsis	`Cal_ReturnType` **`Cal_MacGenerateStart`** `( Cal_ConfigIdType cfgId , Cal_MacGenerateCtxBufType contextBuffer , const Cal_SymKeyType * keyPtr );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the start of the MAC generate computation should be requested.
	`keyPtr`	A pointer to the key which should be used in the MAC generate computation.
Parameters (out)	`contextBuffer`	Holds the pointer to the buffer in which the context of this service can be stored.
Return Value	Error value.	
	`CAL_E_OK`	If the start was successfully requested.
	`CAL_E_NOT_OK`	If the request failed.
Description	This function performs the start of the MAC generate for the given configuration.	

4.2.3.22. Cal_MacGenerateUpdate

Purpose	Update MAC generate computation.	
Synopsis	`Cal_ReturnType Cal_MacGenerateUpdate ( Cal_ConfigIdType cfgId , Cal_MacGenerateCtxBufType contextBuffer , const uint8 * dataPtr , uint32 dataLength );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the update of the MAC generate computation should be requested.
	`dataPtr`	A pointer to the start of an array which contains the constant data for which a MAC shall be generated.
	`dataLength`	Length of the constant data in bytes.
Parameters (in,out)	`contextBuffer`	Holds the pointer to the buffer in which the context of this service is stored.
Return Value	Error value.	
	`CAL_E_OK`	If the update was successfully requested.
	`CAL_E_NOT_OK`	If no MAC generate computation has been started via Cal_MacGenerateStart(), yet.
Description	This function performs the update of the MAC generate computation for the given data. The update function of the configured primitive is called and its return value is returned.	

4.2.3.23. Cal_MacVerifyFinish

Purpose	Finish MAC verify computation.	
Synopsis	`Cal_ReturnType Cal_MacVerifyFinish ( Cal_ConfigIdType cfgId , Cal_MacVerifyCtxBufType contextBuffer , const uint8 * MacPtr , uint32 MacLength , Cal_VerifyResultType * resultPtr );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the finish of the MAC verify computation should be requested.
	`MacPtr`	A pointer to the start of an array which holds the MAC which shall be verified.

	MacLength	The length information of the Mac which shall be verified.
Parameters (in,out)	contextBuffer	Holds the pointer to the buffer in which the context of this service is stored.
Parameters (out)	resultPtr	A Pointer to the cerification result.
Return Value	Error value.	
	CAL_E_OK	If the finish was successfully requested.
	CAL_E_NOT_OK	If no MAC verify computation has been started via Cal_MacVerifyStart(), yet.
Description	This function performs the finishing of the MAC verify computation. The finish function of the configured primitive is called and its return value is returned.	

4.2.3.24. Cal_MacVerifyStart

Purpose	Start MAC verify computation.	
Synopsis	`Cal_ReturnType Cal_MacVerifyStart ( Cal_ConfigIdType cfgId , Cal_MacVerifyCtxBufType contextBuffer , const Cal_SymKeyType * keyPtr );`	
Parameters (in)	cfgId	An identification of the configuration for which the start of the MAC verify computation should be requested.
	keyPtr	A pointer to the key which should be used in the MAC verify computation.
Parameters (out)	contextBuffer	Holds the pointer to the buffer in which the context of this service can be stored.
Return Value	Error value.	
	CAL_E_OK	If the start was successfully requested.
	CAL_E_NOT_OK	If the request failed.
Description	This function performs the start of the MAC verification for the given configuration.	

4.2.3.25. Cal_MacVerifyUpdate

Purpose	Update MAC verify computation.

Synopsis	`Cal_ReturnType Cal_MacVerifyUpdate ( Cal_ConfigIdType cfgId , Cal_MacVerifyCtxBufType contextBuffer , const uint8 * dataPtr , uint32 dataLength );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the update of the MAC verify computation should be requested.
	`dataPtr`	A pointer to the start of an array which contains the constant data for which a MAC shall be verified.
	`dataLength`	Length of the constant data in bytes.
Parameters (in,out)	`contextBuffer`	Holds the pointer to the buffer in which the context of this service is stored.
Return Value	Error value.	
	`CAL_E_OK`	If the update was successfully requested.
	`CAL_E_NOT_OK`	If no MAC verify computation has been started via Cal_MacVerifyStart(), yet.
Description	This function performs the update of the MAC verify computation for the given data. The update function of the configured primitive is called and its return value is returned.	

4.2.3.26. Cal_RandomGenerate

Purpose	Generate pseudo random bytes.	
Synopsis	`Cal_ReturnType Cal_RandomGenerate ( Cal_ConfigIdType cfgId , Cal_RandomCtxBufType contextBuffer , uint8 * resultPtr , uint32 resultLength );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the byte generation should be requested.
	`resultLength`	Holds the amount of bytes which should be generated.
Parameters (in,out)	`contextBuffer`	A Pointer to a buffer where the context of the current configuration
Parameters (out)	`resultPtr`	A pointer to the start of a buffer where the generated pseudo random bytes should be stored.

Return Value	error value	
	`CAL_E_OK`	If the byte generation was successfully requested.
	`CAL_E_NOT_OK`	If the request failed.
	`CAL_E_ENTROPY_EXHAUSTION`	If the request failed, entropy of random number generator is exhausted.
Description	This function performs the generation of pseudo random bytes. The byte generation function of the configured primitive is called and its return value is returned.	

4.2.3.27. Cal_RandomSeedFinish

Purpose	Finish seeding.	
Synopsis	`Cal_ReturnType` **`Cal_RandomSeedFinish`** `( Cal_ConfigIdType cfgId , Cal_RandomCtxBufType contextBuffer );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the seeding finish shall be requested.
Parameters (in,out)	`contextBuffer`	A Pointer to a buffer where the context of the current configuration
Return Value	error value	
	`CAL_E_OK`	If the seeding finish was successfully requested.
	`CAL_E_NOT_OK`	If no seeding has been started via Cal_RandomSeedStart() yet.
Description	This function performs the finishing of the seeding.	

4.2.3.28. Cal_RandomSeedStart

Purpose	Start seeding.	
Synopsis	`Cal_ReturnType` **`Cal_RandomSeedStart`** `( Cal_ConfigIdType cfgId , Cal_RandomCtxBufType contextBuffer );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the initialization shall be requested.

Parameters (out)	`contextBuffer`	A Pointer to a buffer where the context of the current configuration will be stored.
Return Value	error value	
	`CAL_E_OK`	If the seeding start was successfully requested.
	`CAL_E_NOT_OK`	If the request failed.
Description	This function performs the initialization of the Random Seed. The initialization function of the configured primitive is called and its return value is returned.	

4.2.3.29. Cal_RandomSeedUpdate

Purpose	Update seeding.	
Synopsis	`Cal_ReturnType Cal_RandomSeedUpdate ( Cal_ConfigIdType cfgId , Cal_RandomCtxBufType contextBuffer , const uint8 * seedPtr , uint32 seedLength );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the seed update shall be requested.
	`seedPtr`	A pointer to the start of an array which contains the seed.
	`seedLength`	Holds the length of the seed array.
Parameters (in,out)	`contextBuffer`	A Pointer to a buffer where the context of the current configuration
Return Value	error value	
	`CAL_E_OK`	If the seeding update was successfully requested.
	`CAL_E_NOT_OK`	If no seeding has been started via Cal_RandomSeedStart() yet.
Description	This function performs the update of the Random seed. The seeding function of the configured primitive is called and its return value is returned.	

4.2.3.30. Cal_SignatureVerifyFinish

Purpose	Finish signature verification.

Synopsis	`Cal_ReturnType` **`Cal_SignatureVerifyFinish`** `( Cal_ConfigIdType cfgId , Cal_SignatureVerifyCtxBufType contextBuffer , const uint8 * signaturePtr , uint32 signatureLength , Cal_VerifyResultType * resultPtr );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the finish of the signature verification shall be requested.
	`signaturePtr`	A pointer to the start of an array where the signature to be verified is stored.
	`signatureLength`	The length of the signature in bytes.
Parameters (in,out)	`contextBuffer`	A pointer to a buffer which holds the context of the current SignatureVerify configuration.
Parameters (out)	`resultPtr`	A pointer to a variable where the result of the signature verification should be stored.
Return Value	error value	
	`CAL_E_OK`	If the finish was successfully requested.
	`CAL_E_NOT_OK`	If no signature verification has been started via Cal_SignatureVerifyStart() yet.
Description	This function performs the finishing of a signature verification. The finish function of the configured primitive is called and its return value is returned.	

4.2.3.31. Cal_SignatureVerifyStart

Purpose	Start signature verification.	
Synopsis	`Cal_ReturnType` **`Cal_SignatureVerifyStart`** `( Cal_ConfigIdType cfgId , Cal_SignatureVerifyCtxBufType contextBuffer , const Cal_AsymPublicKeyType * keyPtr );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the start of the signature verification shall be requested.
	`keyPtr`	A pointer to the key which should be used in the signature verification.
Parameters (out)	`contextBuffer`	A pointer to a buffer where the context of the current SignatureVerify configuration will be stored.

Return Value	error value	
	`CAL_E_OK`	If the start was successfully requested.
	`CAL_E_NOT_OK`	If the request failed.
Description	This function requests the start of the signature verification for the given configuration and key. The start function of the configured primitive is called and its return value is returned.	

4.2.3.32. Cal_SignatureVerifyUpdate

Purpose	Update signature verification.	
Synopsis	`Cal_ReturnType Cal_SignatureVerifyUpdate ( Cal_ConfigIdType cfgId , Cal_SignatureVerifyCtxBufType contextBuffer , const uint8 * dataPtr , uint32 dataLength );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the update of the signature verification shall be requested.
	`dataPtr`	A pointer to the start of an array which contains a part of the data for which the signature should be verified.
	`dataLength`	The amount of bytes of data.
Parameters (in,out)	`contextBuffer`	A pointer to a buffer which holds the context of the current SignatureVerify configuration.
Return Value	error value	
	`CAL_E_OK`	If the update was successfully requested.
	`CAL_E_NOT_OK`	If no signature verification has been started via Cal_SignatureVerifyStart()yet.
Description	This function performs the update of a signature verification for the given data. The update function of the configured primitive is called and its return value is returned.	

4.2.3.33. Cal_SymBlockDecryptFinish

Purpose	Finish symmetrical block decryption computation.
Synopsis	`Cal_ReturnType Cal_SymBlockDecryptFinish ( Cal_ConfigIdType cfgId , Cal_SymBlockDecryptCtxBufType contextBuffer );`

Parameters (in)	`cfgId`	An identification of the configuration for which the finishing of the symmetrical block decryption computation should be requested.
Parameters (in,out)	`contextBuffer`	Holds the pointer to the buffer in which the context of this service is stored.
Return Value	Error value.	
	`CAL_E_OK`	If the finish was successfully requested.
	`CAL_E_NOT_OK`	If no symmetrical block decryption computation has been started via Cal_SymBlockDecryptStart(), yet.
Description	This function requests the finishing of the symmetrical block decryption computation and the storing of the decrypted text in the given buffer. The finish function of the configured primitive is called and its return value is returned.	

4.2.3.34. Cal_SymBlockDecryptStart

Purpose	Start symmetrical block decryption computation.	
Synopsis	`Cal_ReturnType` **`Cal_SymBlockDecryptStart`** `( Cal_ConfigIdType cfgId , Cal_SymBlockDecryptCtxBufType contextBuffer , const Cal_SymKeyType * keyPtr );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the start of the symmetrical block decryption computation should be requested.
	`keyPtr`	A pointer to the key which should be used in the symmetrical block decryption computation.
Parameters (out)	`contextBuffer`	Holds the pointer to the buffer in which the context of this service can be stored.
Return Value	Error value.	
	`CAL_E_OK`	If the start was successfully requested.
	`CAL_E_NOT_OK`	Otherwise.
Description	This function requests the start of the symmetrical block decryption for the given configuration. The start function of the configured primitive is called and its return value is returned.	

4.2.3.35. Cal_SymBlockDecryptUpdate

Purpose	Update symmetrical block decryption computation.	
Synopsis	`Cal_ReturnType` **`Cal_SymBlockDecryptUpdate`** `( Cal_ConfigIdType cfgId , Cal_SymBlockDecryptCtxBufType contextBuffer , const uint8 * cipherTextPtr , uint32 cipherTextLength , uint8 * plainTextPtr , uint32 * plainTextLengthPtr );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the update of the symmetrical block decryption computation should be requested.
	`cipherTextPtr`	A pointer to the start of an array which contains the constant cipher text that shall be decrypted.
	`cipherTextLength`	Length of the constant cipher text in bytes.
Parameters (in,out)	`contextBuffer`	Holds the pointer to the buffer in which the context of this service is stored.
	`plainTextLengthPtr`	Holds a pointer to a memory location in which the length information is stored. On calling this function this parameter shall contain the size of the buffer provided by plainTextPtr. When the request has finished, the amount of data that has been decrypted shall be stored.
Parameters (out)	`plainTextPtr`	A pointer to the start of an array where the decrypted text will be stored.
Return Value	Error value.	
	`CAL_E_OK`	If the update was successfully requested.
	`CAL_E_NOT_OK`	If no symmetrical block decryption computation has been started via Cal_SymBlockDecryptStart(), yet.
Description	This function requests the update of the symmetrical block decryption computation for the given data. The update function of the configured primitive is called and its return value is returned.	

4.2.3.36. Cal_SymBlockEncryptFinish

Purpose	Finish symmetrical block encryption computation.

Synopsis	`Cal_ReturnType` **`Cal_SymBlockEncryptFinish`** `( Cal_ConfigIdType cfgId , Cal_SymBlockEncryptCtxBufType contextBuffer );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the finishing of the symmetrical block encryption computation should be requested.
Parameters (in,out)	`cipherTextLengthPtr`	Holds a pointer to a memory location in which the length information is stored. On calling this function this parameter shall contain the size of the buffer provided by cipherTextPtr. When the request has finished, the amount of data that has been encrypted shall be stored.
Return Value	Error value.	
	`CAL_E_OK`	If the finish was successfully requested.
	`CAL_E_NOT_OK`	If no symmetrical block encryption computation has been started via Cal_SymBlockEncryptStart(), yet.
Description	This function requests the finishing of the symmetrical block encryption computation and the storing of the encrypted text in the given buffer. The finish function of the configured primitive is called and its return value is returned.	

4.2.3.37. Cal_SymBlockEncryptStart

Purpose	Start symmetrical block encryption computation.	
Synopsis	`Cal_ReturnType` **`Cal_SymBlockEncryptStart`** `( Cal_ConfigIdType cfgId , Cal_SymBlockEncryptCtxBufType contextBuffer , const Cal_SymKeyType * keyPtr );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the start of the symmetrical block encryption computation should be requested.
	`keyPtr`	A pointer to the key which should be used in the symmetrical block encryption computation.
Parameters (out)	`contextBuffer`	Holds the pointer to the buffer in which the context of this service can be stored.

Return Value	Error value.	
	`CAL_E_OK`	If the start was successfully requested.
	`CAL_E_NOT_OK`	Otherwise.
Description	This function requests the start of the symmetrical block encryption for the given configuration. The start function of the configured primitive is called and its return value is returned.	

4.2.3.38. Cal_SymBlockEncryptUpdate

Purpose	Update symmetrical block encryption computation.	
Synopsis	`Cal_ReturnType` **`Cal_SymBlockEncryptUpdate`** `( Cal_ConfigIdType cfgId , Cal_SymBlockEncryptCtxBufType contextBuffer , const uint8 * plainTextPtr , uint32 plainTextLength , uint8 * cipherTextPtr , uint32 * cipherTextLengthPtr );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the update of the symmetrical block encryption computation should be requested.
	`plainTextPtr`	A pointer to the start of an array which contains the constant plain text that shall be encrypted.
	`plainTextLength`	Length of the constant plain text in bytes.
Parameters (in,out)	`contextBuffer`	Holds the pointer to the buffer in which the context of this service is stored.
	`cipherTextLengthPtr`	Holds a pointer to a memory location in which the length information is stored. On calling this function this parameter shall contain the size of the buffer provided by cipherTextPtr. When the request has finished, the amount of data that has been encrypted shall be stored.
Parameters (out)	`cipherTextPtr`	A pointer to the start of an array where the encrypted text will be stored.
Return Value	Error value.	
	`CAL_E_OK`	If the update was successfully requested.

	CAL_E_NOT_OK	If no symmetrical block encryption computation has been started via Cal_SymBlockEncryptStart(), yet.
Description	This function requests the update of the symmetrical block encryption computation for the given data. The update function of the configured primitive is called and its return value is returned.	

4.2.3.39. Cal_SymDecryptFinish

Purpose	Finish symmetrical decryption computation.	
Synopsis	`Cal_ReturnType Cal_SymDecryptFinish ( Cal_ConfigIdType cfgId , Cal_SymDecryptCtxBufType contextBuffer , uint8 * plainTextPtr , uint32 * plainTextLengthPtr );`	
Parameters (in)	cfgId	An identification of the configuration for which the finish of the symmetrical decryption computation should be requested.
Parameters (in,out)	contextBuffer	A buffer which holds the context for this SymDecrypt configuration.
	plainTextLengthPtr	Holds a pointer to a memory location in which the length information is stored. On calling this function this parameter shall contain the size of the buffer provided by plainTextPtr. When the request has finished, the amount of data that has been decrypted shall be stored.
Parameters (out)	plainTextPtr	A pointer to the start of an array where the decrypted text will be stored.
Return Value	Error value.	
	CAL_E_OK	If the finish was successfully requested.
	CAL_E_NOT_OK	If no symmetrical decryption computation has been started via Cal_SymDecryptStart(), yet.
	CAL_E_SMALL_BUFFER	If the provided buffer is to small to store the result.

Description	This function finishes the symmetrical decryption computation and the stores of the decrypted text in the given buffer. The finish function of the configured primitive is called and its return value is returned.

4.2.3.40. Cal_SymDecryptStart

Purpose	Start symmetrical decryption computation.	
Synopsis	`Cal_ReturnType Cal_SymDecryptStart ( Cal_ConfigIdType cfgId , Cal_SymDecryptCtxBufType contextBuffer , const Cal_SymKeyType * keyPtr , const uint8 * InitVectorPtr , uint32 InitVectorLength );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the start of the symmetrical decryption computation should be requested.
	`keyPtr`	A pointer to the key which should be used in the symmetrical decryption computation.
	`InitVectorPtr`	Holds a pointer to the initialization vector which has to be used during the symmetrical decryption computation.
	`InitVectorLength`	Holds the length of the initialisation vector which has to be used during the symmetrical decryption computation.
Parameters (out)	`contextBuffer`	A buffer which will hold the context for this SymDecrypt configuration.
Return Value	Error value.	
	`CAL_E_OK`	If the start was successfully requested.
	`CAL_E_NOT_OK`	Otherwise.
Description	This function performs the start of the symmetrical decryption for the given configuration. The start function of the configured primitive is called and its return value is returned.	

4.2.3.41. Cal_SymDecryptUpdate

Purpose	Update symmetrical decryption computation.

Synopsis	`Cal_ReturnType` **`Cal_SymDecryptUpdate`** `( Cal_ConfigIdType cfgId , Cal_SymDecryptCtxBufType contextBuffer , const uint8 * cipherTextPtr , uint32 cipherTextLength , uint8 * plainTextPtr , uint32 * plainTextLengthPtr );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the update of the symmetrical decryption computation should be requested.
	`cipherTextPtr`	A pointer to the start of an array which contains the constant cipher text that shall be decrypted.
	`cipherTextLength`	Length of the constant cipher text in bytes.
Parameters (in,out)	`contextBuffer`	A buffer which holds the context for this SymDecrypt configuration.
	`plainTextLengthPtr`	Holds a pointer to a memory location in which the length information is stored. On calling this function this parameter shall contain the size of the buffer provided by plainTextPtr. When the request has finished, the amount of data that has been decrypted shall be stored.
Parameters (out)	`plainTextPtr`	A pointer to the start of an array where the decrypted text will be stored.
Return Value	Error value.	
	`CAL_E_OK`	If the update was successfully requested.
	`CAL_E_NOT_OK`	If no symmetrical decryption computation has been started via Cal_SymDecryptStart(), yet.
	`CAL_E_SMALL_BUFFER`	If the provided buffer is to small to store the result.
Description	This function performs the update of the symmetrical decryption computation for the given data. The update function of the configured primitive is called and its return value is returned.	

4.2.3.42. Cal_SymKeyExtractFinish

Purpose	Finish symmetric key extraction.

Synopsis	`Cal_ReturnType` **`Cal_SymKeyExtractFinish`** `( Cal_ConfigIdType cfgId , Cal_SymKeyExtractCtxBufType contextBuffer , Cal_SymKeyType * keyPtr );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the the finishing of the symmetric key extraction shall be requested.
Parameters (in,out)	`contextBuffer`	A Pointer to a buffer which holds the context of the current configuration of the SymKeyExtract service.
Parameters (out)	`keyPtr`	A pointer to the buffer where the extracted key should be stored
Return Value	Error value.	
	`CAL_E_OK`	If the finish was successfully requested.
	`CAL_E_NOT_OK`	If no symmetric key extraction has been started via Cal_SymKeyExtractStart() yet.
Description	This function performs the finishing of the symmetric key extraction and the storing of the key in the given buffer. The finish function of the configured primitive is called and its return value is returned.	

4.2.3.43. Cal_SymKeyExtractStart

Purpose	Start symmetric key extraction.	
Synopsis	`Cal_ReturnType` **`Cal_SymKeyExtractStart`** `( Cal_ConfigIdType cfgId , Cal_SymKeyExtractCtxBufType contextBuffer );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the the start of the symmetric key extraction shall be requested.
Parameters (out)	`contextBuffer`	A Pointer to a buffer where the context of the current configuration of the SymKeyExtract service will be stored.
Return Value	Error value.	
	`CAL_E_OK`	If the start was successfully requested.
	`CAL_E_NOT_OK`	Otherwise.
Description	This function performs the start of the symmetric key extraction for the given configuration. The start function of the configured primitive is called and its return value is returned.	

4.2.3.44. Cal_SymKeyExtractUpdate

Purpose	Update symmetric key extraction.	
Synopsis	`Cal_ReturnType` **`Cal_SymKeyExtractUpdate`** `( Cal_ConfigIdType cfgId , Cal_SymKeyExtractCtxBufType contextBuffer , const uint8 * dataPtr , uint32 dataLength );`	
Parameters (in)	`cfgId`	An identification of the configuration for which the the update of the symmetric key extraction shall be requested.
	`dataPtr`	A pointer to the start of an array which contains a part of the key which should be extracted.
	`dataLength`	The amount of bytes of data.
Parameters (in,out)	`contextBuffer`	A Pointer to a buffer which holds the context of the current configuration of the SymKeyExtract service.
Return Value	Error value.	
	`CAL_E_OK`	If the update was successfully requested.
	`CAL_E_NOT_OK`	If no symmetric key extraction has been started via Cal_SymKeyExtractStart() yet.
Description	This function performs the update of the symmetric key extraction for the given data. The update function of the configured primitive is called and its return value is returned.	