


SCOPE OF APPLICATION All Project/Engineering		SHT/SHTS 1 / 79
Responsibility: Classic AUTOSAR Team	AUTOSAR KeyM User Manual	DOC. NO
AUTOSAR KeyM User Manual		

Document Change History				
Date (YYYY-MM-DD)	Ver.	Editor	Chap	Content (before revision-> after revision)
2021-01-15	1.0.0.0	JaeHyun Lim	All	Initial Version
2021-03-27	1.0.1.0	TamTV6	4.3	Update change log
2021-11-12	1.0.2.0	TamTV6	All	Applying change of company name Update 5.1.1, 8.1 chapter
2022-06-03	1.0.3.0	DienTC1	7.2, 8.1.4	7.2: - Update ERR109007 description for X509 extension type supported(Add AuthorityInfoAccess extension). 8.1.4: - Update Certificate parsing for X509 authorityKeyIdentifier extension and Un-known extension.
2022-07-01	1.0.3.1	DienTC1	4.3	Clarify copyright description Update incorrect company name
2022-08-23	1.0.4.0	DienTC1	4.3	Fix UNECE security coding violations
2022-10-06	1.0.5.0	DienTC1	4.3	- Move Integration_KeyM source to Reference_Code folder. - Support HANDLE-TERMINATION-AND-RESTART be configured in Swcd_Bsw_KeyM.arxml - Fix KEYM_USE_STBM generated(STD_ON/OFF) incorrectly in KeyM_Cfg.h
2022-12-15	1.0.6.0	DienTC1	4.3 8.1.5	- Support universal certificate configuration which has more cert elements that the real cert has. - Fix the method to set KEYM_USE_STBM on/off in

3 rd Edition Date: 2021-09-30	File Name KeyM_UM.pdf	Creation JH Lim 2021-01-15	Check YJ Yoon 2021-01-15	Approval JH Beak 2021-01-15
Document Management System				

				generated code. - Update illustrating image for removing Integration_KeyM source.
2023-02-10	1.0.6.1	DienTC1	4.3	- Update Change log
2023-04-21	1.0.7.0	PhuocLH9	4.3 7.2 8.1.2	- Update change log - Add more guide and generation error on setting of Certificate Element
2023-08-01	1.0.8.0	PhuocLH9	4.3 6.3.3	- Update change log - Add user function for certificate
2024-02-21	1.0.9.0	PhuocLH9	4.3	- Update change log

Document (DOC NO)	number	SHT/SHTS 3 / 79
----------------------	--------	--------------------

Table of Contents

1. OVERVIEW	7
2. REFERENCE	8
3. AUTOSAR SYSTEM	9
3.1 Overview of Software Layers	9
3.2 AUTOSAR Key Manager	10
4. PRODUCT RELEASE NOTES	11
4.1 Overview.....	11
4.2 Scope of the release	11
4.3 Change log	11
4.3.1 Version 1.0.0.0 (2021-01-15)	11
4.3.2 Version 1.0.1.0 (2021-03-27)	11
4.3.3 Version 1.0.2.0 (2021-11-12).....	12
4.3.4 Version 1.0.3.0 (2022-06-03)	12
4.3.5 Version 1.0.3.1 (2022-07-01)	14
4.3.6 Version 1.0.4.0 (2022-08-23)	14
4.3.7 Version 1.0.5.0 (2022-10-06)	14
4.3.8 Version 1.0.6.0 (2022-12-15)	15
4.3.9 Version 1.0.6.1 (2023-02-10)	16
4.3.10 Version 1.0.7.0 (2023-04-21)	16
4.3.11 Version 1.0.8.0 (2023-08-01)	18
4.3.12 Version 1.0.9.0 (2023-10-22)	18
4.4 Limitations	19
4.5 Deviation	20
5. CONFIGURATION GUIDE	21
5.1 KeyM Module.....	21
5.1.1 KeyMGeneral.....	21
5.1.2 KeyMCryptoKey	21
5.1.3 KeyMCertificate	22
5.1.4 KeyMCertificateElement	23
5.1.5 KeyMCertificateElementVerification	23
5.1.6 KeyMCertificateElementRule	23
5.1.7 KeyMCertificateElementCondition.....	24
5.1.8 KeyMCertificateElementConditionValue	24
5.1.9 KeyMCertificateElementConditionPrimitive	24
5.1.10 KeyMCertificateElementConditionArray	24
5.1.11 KeyMCertificateElementConditionArrayElement	24
5.1.12 KeyMCertificateElementConditionCertificateElement.....	25
5.1.13 KeyMCertificateElementConditionSenderReceiver	25
5.1.14 KeyMNvmBlock	25

5.2	System Configuration.....	25
5.2.1	ApplicationSwComponentType setting	25
5.2.2	EcuComposition settings	25
6.	APPLICATION PROGRAMMING INTERFACE (API).....	27
6.1	Type Definitions	27
6.1.1	KeyM_ConfigType	27
6.1.2	KeyM_KH_UpdateOperationType	27
6.1.3	KeyM_CertElementIteratorType	27
6.1.4	KeyM_CryptoKeyIdType.....	28
6.2	Macro Constants.....	28
6.3	Functions	28
6.3.1	General.....	28
6.3.1.1	KeyM_Init.....	28
6.3.1.2	KeyM_Deinit	29
6.3.1.3	KeyM_GetVersionInfo.....	29
6.3.2	Crypto key operation.....	29
6.3.2.1	KeyM_Start.....	30
6.3.2.2	KeyM_Prepare	30
6.3.2.3	KeyM_Update.....	31
6.3.2.4	KeyM_Finalize.....	32
6.3.2.5	KeyM_Verify	33
6.3.3	Certificate handling	34
6.3.3.1	KeyM_ServiceCertificate.....	34
6.3.3.2	KeyM_SetCertificate	36
6.3.3.3	KeyM_GetCertificate	36
6.3.3.4	KeyM_VerifyCertificates	37
6.3.3.5	KeyM_VerifyCertificate.....	38
6.3.3.6	KeyM_VerifyCertificateChain	39
6.3.3.7	KeyM_CertElementGet.....	40
6.3.3.8	KeyM_CertElementGetFirst	41
6.3.3.9	KeyM_CertElementGetNext.....	42
6.3.3.10	KeyM_CertGetStatus	43
6.3.3.11	KeyM_CertificateElementGetByIndex	44
6.3.3.12	KeyM_CertificateElementGetCount	45
6.3.3.13	KeyM_CertElementGetOIDHex	46
6.3.3.14	KeyM_CertGetDERPacketData.....	48
6.3.4	Scheduled Functions	49
6.3.4.1	KeyM_MainFunction.....	49
6.3.4.2	KeyM_MainBackgroundFunction	49
6.3.5	Expected Interfaces	50
6.3.5.1	KeyM_KH_Start	50
6.3.5.2	KeyM_KH_Prepare	51
6.3.5.3	KeyM_KH_Update	52
6.3.5.4	KeyM_KH_Finalize.....	53
6.3.5.5	KeyM_KH_Verify.....	54
6.3.5.6	KeyM_KH_ServiceCertificate	55
6.3.5.7	KeyM_CryptoKeyUpdateCallbackNotification.....	56
6.3.5.8	KeyM_CryptoKeyFinalizeCallbackNotification.....	57
6.3.5.9	KeyM_CryptoKeyVerifyCallbackNotification	57
6.3.5.10	KeyM_ServiceCertificateCallbackNotification	58
6.3.5.11	KeyM_CertificateVerifyCallbackNotification	59
6.4	Noted	59
6.4.1	In Communication with application SW-C	59

7. GENERATOR.....	60
7.1 Generator Option	60
7.2 Generator Error Messages	60
7.3 Warning Messages	65
7.4 Information Messages	65
8. APPENDIX	66
8.1 Design Notes.....	66
8.1.1 Certificate Chain in Hierachy.....	66
8.1.2 Certificate Formats	66
8.1.2.1 X509 and CRL formats	66
8.1.2.2 CVC format.....	68
8.1.3 Certificate status transitions	70
8.1.4 Certificate Parsing	70
8.1.5 Certificate Verification Steps	71
8.2 Setting Guide by Function	74
8.2.1 KeyM_Init.....	74
8.2.2 KeyM_Update	74
8.2.3 KeyM_Verify	74
8.2.4 KeyM_ServiceCertificate.....	75
8.2.5 KeyM_SetCertificate	76
8.2.6 Verify Certificate functions	76
8.2.7 Key Handler Functions.....	77
8.3 Bswmd (Bsw Module Description)	78
8.3.1 Bsw module event setting	78
8.4 Exclusive Areas	79
8.4.1 SchM Module Apis	79
8.4.2 Setting method.....	79

1. Overview

This document is based on the Autosar standard SRS/SWS. For more detailed functional description when using the module, refer to the reference document below.

The interpretation of the category related to setting is as follows

- Changeable (C) : Items that can be set by the user
- Fixed (F) : Items that cannot be changed by user
- NotSupported (N) : Items that not supported

2. Reference

Related Documentation

Sl. No.	Title	Version
1.	AUTOSAR_SWS_BSWGeneral.doc	4.4.0
2.	AUTOSAR_SWS_KeyManager.pdf	4.4.0
3.	AUTOSAR_SWS_CryptoServiceManager.pdf	4.4.0
4.	AUTOSAR_SRS_CryptoStack.pdf	4.4.0
5.	AUTOSAR_SWS_CryptoDriver.pdf	4.4.0
6.	AUTOSAR_SWS_SynchronizedTimeBaseManager.pdf	4.4.0
7.	AUTOSAR_SWS_NVRAMManager.pdf	4.4.0
8.	AUTOSAR_SWS_CryptoInterface.pdf	4.4.0

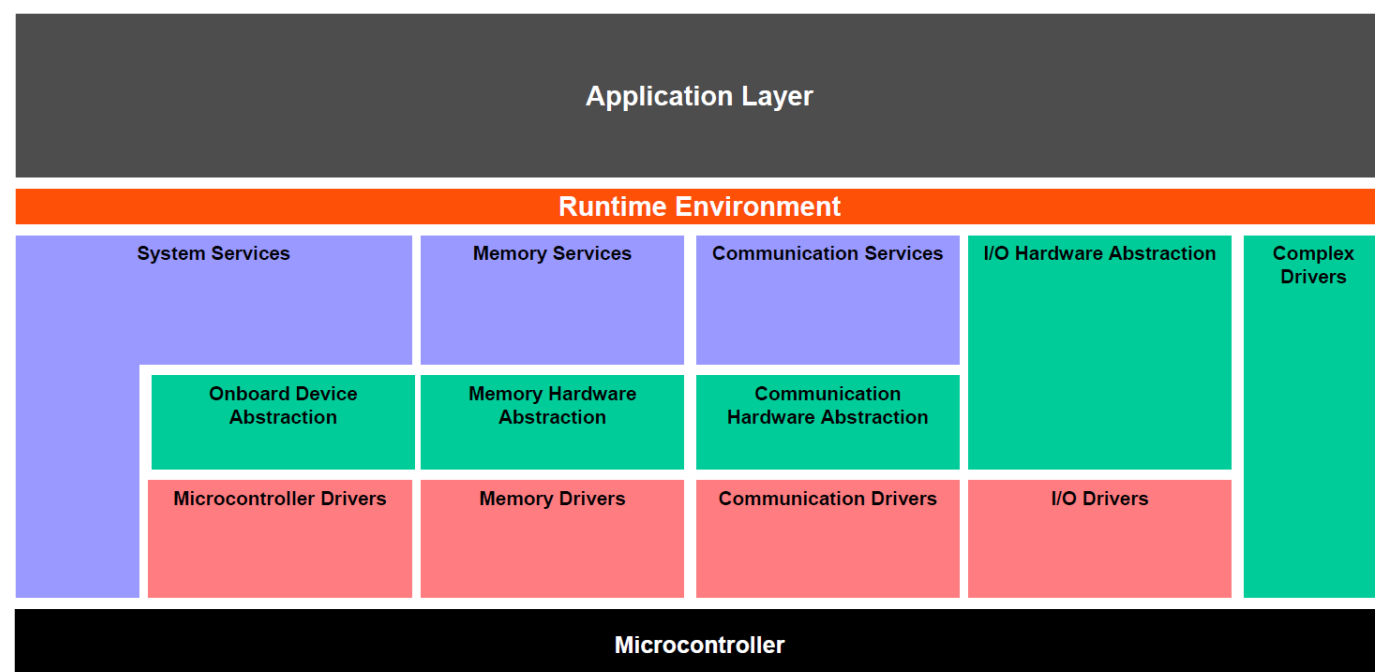
Acronyms and abbreviations

Abbreviation/ Acronym:	Description:
KeyM	KeyM Manager
PKI	Public Key Infrastructure
CSR	Certificate Signing Request
CSM	Crypto Service Manager
CRL	Certificate Revocation List
CA	Certificate Authority
OID	Object Identifier. A byte array that identifies a certificate element or group or list of certificate elements
NvM	NVRAM Manager
StbM	Synchronized Time Base Manager
CRYPTO	Crypto Driver
CRYIF	Crypto Interface

3. AUTOSAR System

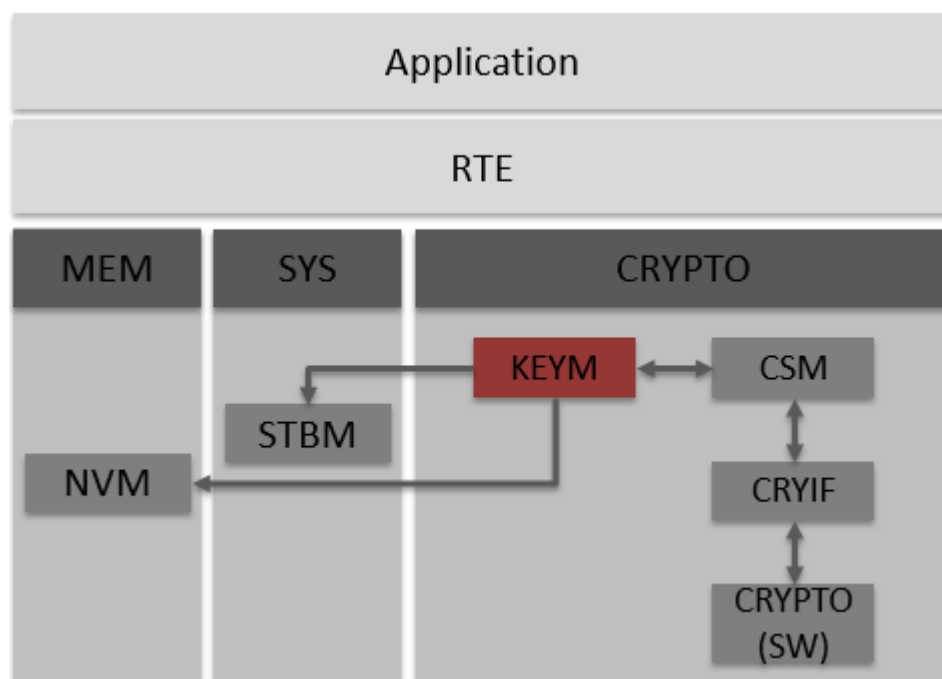
3.1 Overview of Software Layers

The Layered Architecture of the AUTOSAR platform is as follows. The AUTOSAR platform can be divided into Service Layer, ECU Abstraction Layer, Complex Device Drivers and Microcontroller Abstraction Layer.



3.2 AUTOSAR Key Manager

The Key Management module can roughly be divided into two parts: the crypto key sub module and the certificate sub module. The crypto key sub module is mainly used to interact with a key provisioning entity (key master) that initiates the generation or provides key material directly. These keys are assigned to crypto keys of the CSM and stored in dedicated NVM blocks or can be stored as keys of the respective crypto driver. The certificate sub module allows to configure certificates of a chain, providing interfaces to store and verify them.



4. Product Release Notes

4.1 Overview

The purpose of this chapter is to provide release-related content for Hyundai Autoever KeyM module and describes restrictions and specifics for KeyM Software product release version.

4.2 Scope of the release

All contents of this document are limited to the following Hyundai Autoever KeyM modules.

Module	Autosar version	Module version
KeyM	4.4.0	1.0.9.0

- ※ Module version means the Sw version of each module's BswModule Description (Bswmd) file.
Module release notes.

4.3 Change log

4.3.1 Version 1.0.0.0 (2021-01-15)

- **Version 1.0.0 기능**
 - Initial Version

Cause	Initial Version
Operation Impact	N/A
Configuration Impact	N/A
Required measure of ASW	N/A

4.3.2 Version 1.0.1.0 (2021-03-27)

- **Version 1.0.1**
 - Remove redundant condition in source code
 - Add NULL pointer checking in source code

Cause	There are redandunt conditions in source code
-------	---

	There are Null pointers that are not checked in source code
Operation Impact	N/A
Configuration Impact	N/A
Required measure of ASW	N/A

4.3.3 Version 1.0.2.0 (2021-11-12)

➤ Version 1.0.2

- Applying change of company name
- Add UserGetCurrentTime configuration to make KeyM_R44 compatible with R40 SWP.

Cause	Customer requests to support KeyM_R44 compatible with R40 SWP
Operation Impact	For R40, current time shall be get via UserGetCurrentTime function For R44, current time shall be get via STBM function
Configuration Impact	For R40, KeyMUserGetCurrentTimeFuc configuration shall be configured. For R44, KeyMCertTimebaseRef configuration shall be configured.
Required measure of ASW	N/A

4.3.4 Version 1.0.3.0 (2022-06-03)

➤ Change Request

- Support parsing and retrieving all X.509 certificate AuthorityKeyIdentifier extension sub-elements: keyIdentifier, authorityCertIssuer and authorityCertSerialNumber.

Cause	When X509 certificate AuthorityKeyIdentifier extension contains all possible sub-elements, the current KeyM only supports to parse and retrieve the first sub-element.
Operation Impact	N/A
Configuration Impact	KeyM Certificate extension element (See UM 5.1.4)
Required measure of ASW	N/A

➤ Change Request

- Support parsing X.509 AuthorityInfoAccess extension.

Cause	KeyM does not support this type of extension. So, Generator returns error if this extension is configured in KeyM Certificate.
Operation Impact	N/A
Configuration Impact	KeyM Certificate extension element (See UM 5.1.4)
Required measure of ASW	N/A

➤ Change Request

- Support parsing X.509 Unknown extension.

Cause	KeyM supports only some type of extension. So, Generator returns error if an unsupported extension is configured in KeyM Certificate.
Operation Impact	N/A
Configuration Impact	KeyM Certificate extension element (See UM 5.1.4)
Required measure of ASW	N/A

➤ Defect

- Fix compile error when KeyMCertificateManagerEnabled is configured as FALSE.

Cause	If KeyMCertificateManagerEnabled is configured as FALSE, compilation gets error because KeyM_CertificateIdType and KeyM_CertElementType are undefined.
Operation Impact	N/A
Configuration Impact	KeyM General KeyMCertificateManagerEnabled configuration (See UM 5.1.1)
Required measure of ASW	N/A

➤ Defect

- Fix missing KEYM_INTERNAL_BUFFER_LENGTH macro by Generator if no KeyMCryptoKey item is configuration.

Cause	When no KeyMCryptoKey item is configuration, KEYM_INTERNAL_BUFFER_LENGTH is not generated. However, KeyMCertificate needs to use this macro. This leads to compilation error.
Operation Impact	N/A
Configuration Impact	KeyM General KeyMCertificateManagerEnabled configuration (See UM 5.1.2, 5.1.3)

Required measure of ASW	N/A
-------------------------	-----

4.3.5 Version 1.0.3.1 (2022-07-01)

➤ Change Request

- Change the Copyright comment in the code
- DeliveryBoxHistory document template updates

Cause	The new Copyright comment is needed to update in the code. The new DeliveryBoxHistory document template is needed to update.
Operation Impact	N/A
Configuration Impact	N/A
Required measure of ASW	N/A

4.3.6 Version 1.0.4.0 (2022-08-23)

➤ Change Request

- Fix UNECE security coding violations

Cause	There are UNECE security coding violations in the source code. They need to be fixed.
Operation Impact	N/A
Configuration Impact	N/A
Required measure of ASW	N/A

4.3.7 Version 1.0.5.0 (2022-10-06)

➤ Change Request

- Move Integration_KeyM source to Reference_Code folder

Cause	Integration_KeyM source contains user-defined code that is possible to integrate into KeyM module. It is more reasonable to keep these code in Reference_Code folder.
Operation Impact	N/A
Configuration Impact	User can configure KeyMUserIncludeFiles sub-container and parameter in KeyMGeneral to use user-defined code for getting time instead of calling StbM service.
Required measure of ASW	N/A

➤ **Change Request**

- Support HANDLE-TERMINATION-AND-RESTART configurable in Swcd_Bsw_KeyM.arxml.

Cause	By AUTOSAR specs, HANDLE-TERMINATION-AND-RESTART is configurable in Swcd_Bsw_KeyM.arxml.
Operation Impact	N/A
Configuration Impact	N/A
Required measure of ASW	N/A

➤ **Defect**

- KEYM_USE_STBM is generated(STD_ON/OFF) incorrectly in KeyM_Cfg.h.

Cause	In some case, KEYM_USE_STBM is generated as STD_ON in KeyM_Cfg.h even STBM is not used. This issue might affect badly on RAM usage.
Operation Impact	N/A
Configuration Impact	N/A
Required measure of ASW	N/A

4.3.8 Version 1.0.6.0 (2022-12-15)

➤ **Improvement**

- Support universal configuration for certificate.

Cause	In the current version, certificate configuration must be exactly matched to the real certificate. If not, there may be error in cert parsing and verifying steps. That means the user who configures certificates must know the structure of real certificate. This is hard in some situations. So, KeyM needs to support certificate universal configuration which might be used for all possible (real) certificate.
Operation Impact	N/A
Configuration Impact	User can configure more certificate element(issuer/subject name, extensions) than the real certificate has.
Required measure of ASW	N/A

➤ **Defect**

- Compile error is occur if the user doesn't use the STBM Module because of the wrong generation of KEYM_USE_STBM value.

Cause	KeyM generator no check below conditions when generate the KEYM_USE_STBM value -KeyMCertificateManagerEnabled is set or not -no certificate is configured or not
Operation Impact	N/A
Configuration Impact	N/A
Required measure of ASW	N/A

4.3.9 Version 1.0.6.1 (2023-02-10)

➤ Improvement

- Improve the parsing logic of ECC Curve signature r and s (Normalization of different r, s on the same size)

Cause	Sometimes r and s are different size so that normalization is needed for Csm
Operation Impact	N/A
Configuration Impact	N/A
Required measure of ASW	N/A

➤ Improvement

- Allow ECC secp521r1 total Key size up to 136 byte
<KEYM_ECC_SIGNATURE_VALUE_MAX_LENGTH>

Cause	KeyM shall support ECC secp521r1 signature
Operation Impact	N/A
Configuration Impact	N/A
Required measure of ASW	N/A

4.3.10 Version 1.0.7.0 (2023-04-21)

➤ Improvement

- Improvement on Compilation warning of Certificate Services

Cause	Compilation warning is occurred.
Operation Impact	N/A
Configuration Impact	N/A
Required measure of ASW	N/A

➤ Improvement

- Remove declaration of variable in for loop (support C89 standard)

Cause	C89 compiler needs to be supported so that variable declaration is not allowed in the for loop.
Operation Impact	N/A
Configuration Impact	N/A
Required measure of ASW	N/A

➤ Improvement

- Support Sub-Element of SubjectPublicKeyInfo's PublicKeyAlgorithm and SubjectPublicKey

Cause	PublicKey Algorithm (Curve Type) and PublicKeyAlgorithm is needed for improving internal KeyM Validation.
Operation Impact	N/A
Configuration Impact	User also can configure two type. (see UM 8.1.2) 1) Single SubjectPublicKeyInfo configuration (contain all sub element) 2) Two sub-element configuration (SubjectPublicKey, PublicKeyAlgorithm)
Required measure of ASW	N/A

➤ Improvement

- Allow more elementOfStruct of KeyM_CertElementGet

Cause	Some elementofStruct can not be retrieved by KeyM_CertElementGet API
Operation Impact	N/A
Configuration Impact	N/A
Required measure of ASW	N/A

➤ Improvement

- Improve the error handling logic when wrong certificate is set to the KeyM

Cause	Several function is "void" type, user can't know the function executive succesfully or not
Operation Impact	N/A

Configuration Impact	N/A
Required measure of ASW	N/A

4.3.11 Version 1.0.8.0 (2023-08-01)

➤ Improvement

- Provide vendor specific APIs for user.

Cause	User want to get element OID in certificate.
Operation Impact	N/A
Configuration Impact	N/A
Required measure of ASW	N/A

Cause	User want to whole information in some specific fields in certificate.
Operation Impact	N/A
Configuration Impact	N/A
Required measure of ASW	N/A

4.3.12 Version 1.0.9.0 (2024-02-21)

➤ Bug

- Certificate with RAM storage cannot be verified if there is no certificate with NVM storage configuration.

Cause	Macro KEYM_STORAGE_IN_NVM_ENABLED has covered code for handling certificate stored in RAM in function KeyM_PrivPerformServiceCertificateOperation
Operation Impact	N/A
Configuration Impact	N/A
Required measure of ASW	N/A

➤ **Bug**

- Description in PDF is not match with Autosar specification.

Cause	Description for KeyMCertificateCsmKeyTargetRef in PDF is incorrect.
Operation Impact	N/A
Configuration Impact	N/A
Required measure of ASW	N/A

➤ **Improvement**

- Data race issue can be occurrence.

Cause	Some global variable can be rewrite/read when in used.
Operation Impact	N/A
Configuration Impact	N/A
Required measure of ASW	N/A

➤ **Bug**

- Functions are not allocated in KeyM section.

Cause	There's no section in code, so function is allocated in default section.
Operation Impact	N/A
Configuration Impact	N/A
Required measure of ASW	N/A

➤ **Improvement**

- Generate Exclusive Area Policy for TCG.

Cause	There is no Exclusive Area Policy
Operation Impact	N/A
Configuration Impact	N/A
Required measure of ASW	N/A

4.4 Limitations

- The Key Management module shall be used with a Crypto Service Manager and its underlying modules. Only a single KeyElement (with ID = 1) per CsmKey is currently supported.

4.5 Deviation

- In KeyMCertificate Client-Server-Interface(8.7.3.1), add new KeyMCertificateService Client-Server-Interface, this Client-Server-Interface has one operation of ServiceCertificate.
- In KeyMCertificate Client-Server-Interface(8.7.3.1), add new operations: VerifyCertificates, VerifyCertificateChain.
- For KeyM_ResultType, add new error values: KEYM_RT_PARAMETER_MISMATCH, KEYM_RT_KEY_CERT_SIZE_MISMATCH.
- KeyM_CertificateType type is removed. C-APIs and corresponding ClientServerInterfaces use KeyM_CertDataType type only.
- Add new APIs: KeyM_CertificateElementGetByIndex, KeyM_CertificateElementGetCount
- In KeyMGeneral, add new item KeyMUserGetCurrentTimeFunc

5. Configuration Guide

5.1 KeyM Module

5.1.1 KeyMGeneral

Refer to the following settings.

Parameter Name	Value	Category
KeyMCertificateChainMaxDepth	User Defined	C
KeyMCertificateManagerEnabled	User Defined	C
KeyMCryptoKeyHandlerPrepareEnabled	User Defined	C
KeyMCryptoKeyHandlerServiceCertificateEnabled	User Defined	C
KeyMCryptoKeyHandlerStartFinalizeEnabled	User Defined	C
KeyMCryptoKeyHandlerUpdateEnabled	User Defined	C
KeyMCryptoKeyHandlerVerifyEnabled	User Defined	C
KeyMCryptoKeyManagerEnabled	User Defined	C
KeyMCryptoKeyPrepareFunctionEnabled	User Defined	C
KeyMCryptoKeyStartFinalizeFunctionEnabled	User Defined	C
KeyMCryptoKeyVerifyAsyncMode	User Defined	C
KeyMCryptoKeyVerifyFunctionEnabled	User Defined	C
KeyMDevErrorDetect	User Defined	C
KeyMKeyCertNameMaxLength	User Defined	C
KeyMServiceCertificateFunctionEnabled	User Defined	C
KeyMMainFunctionPeriod	User Defined	C
KeyMUserGetCurrentTimeFunc	User Defined	C

5.1.2 KeyMCryptoKey

Refer to the following settings.

Parameter Name	Value	Category
KeyMCryptoCsmVerifyJobType	User Defined	C
KeyMCryptoKeyCryptoProps	User Defined	C
KeyMCryptoKeyGenerationInfo	User Defined	C
KeyMCryptoKeyGenerationType	User Defined	C

KeyMCryptoKeyId	User Defined	C
KeyMCryptoKeyMaxLength	User Defined	C
KeyMCryptoKeyName	User Defined	C
KeyMCryptoKeyStorage	User Defined	C
KeyMCryptoKeyCsmKeyTargetRef	User Defined	C
KeyMCryptoKeyCsmVerifyJobRef	User Defined	C
KeyMCryptoKeyNvmBlockRef	User Defined	C

5.1.3 KeyMCertificate

Refer to the following settings.

Parameter Name	Value	Category
KeyMCertAlgorithmType	User Defined	C
KeyMCertFormatType	User Defined	C
KeyMCertificateId	User Defined	C
KeyMCertificateMaxLength	User Defined	C
KeyMCertificateStorage	User Defined	C
KeyMCertificateName	User Defined	C
KeyMCertificateVerifyCallbackNotificationFunc	User Defined	C
KeyMServiceCertificateCallbackNotificationFunc	User Defined	C
KeyMCertCertificateElementRuleRef	User Defined	C
KeyMCertCsmSignatureGenerateJobRef	User Defined	C
KeyMCertCsmSignatureVerifyKeyRef	User Defined	C
KeyMCertPrivateKeyStorageCryptoKeyRef	User Defined	C
KeyMCertTimebaseRef	User Defined	C
KeyMCertUpperHierarchicalCertRef	User Defined	C
KeyMCertificateCsmKeyTargetRef	User Defined	C
KeyMCertificateNvmBlockRef	User Defined	C

5.1.4 KeyMCertificateElement

Refer to the following settings.

Parameter Name	Value	Category
KeyMCertificateElementHasIteration	User Defined	C
KeyMCertificateElementId	User Defined	C
KeyMCertificateElementMaxLength	User Defined	C
KeyMCertificateElementOfStructure	User Defined	C
KeyMCertificateElementObjectId	User Defined	C
KeyMCertificateElementObjectType	User Defined	C

5.1.5 KeyMCertificateElementVerification

Refer to the following settings.

Parameter Name	Value	Category
KeyMCertificateElementCondition	User Defined	C
KeyMCertificateElementRule	User Defined	C

5.1.6 KeyMCertificateElementRule

Refer to the following settings.

Parameter Name	Value	Category
KeyMLogicalOperator	User Defined	C
KeyMArgumentRef	User Defined	C

5.1.7 KeyMCertificateElementCondition

Refer to the following settings.

Parameter Name	Value	Category
KeyMCCertElementConditionType	User Defined	C
KeyMCCertificateElementRef	User Defined	C
KeyMCCertificateElementConditionValue	User Defined	C

5.1.8 KeyMCertificateElementConditionValue

Refer to the following settings.

Parameter Name	Value	Category
KeyMCCertificateElementConditionPrimitive	User Defined	C
KeyMCCertificateElementConditionArray	User Defined	C
KeyMCCertificateElementConditionCertificateElement	User Defined	C
KeyMCCertificateElementConditionSenderReceiver	User Defined	C

5.1.9 KeyMCertificateElementConditionPrimitive

Refer to the following settings.

Parameter Name	Value	Category
KeyMCCertificateElementConditionPrimitiveValue	User Defined	C

5.1.10 KeyMCertificateElementConditionArray

Refer to the following settings.

Parameter Name	Value	Category
KeyMCCertificateElementConditionArrayElement	User Defined	C

5.1.11 KeyMCertificateElementConditionArrayElement

Refer to the following settings.

Parameter Name	Value	Category
KeyMCertificateElementConditionArrayElementIndex	User Defined	C
KeyMCertificateElementConditionArrayElementValue	User Defined	C

5.1.12 KeyMCertificateElementConditionCerificateElement

Refer to the following settings.

Parameter Name	Value	Category
KeyMCertificateElementRef	User Defined	C

5.1.13 KeyMCertificateElementConditionSenderReceiver

Refer to the following settings.

Parameter Name	Value	Category
KeyMCertificateElementConditionSenderReceiver	User Defined	C

5.1.14 KeyMNvmBlock

Refer to the following settings.

Parameter Name	Value	Category
KeyMNvmBlockDescriptorRef	User Defined	C

5.2 System Configuration

5.2.1 ApplicationSwComponentType setting

Refer to AUTOSAR BSW Service API Guide.doc document.

5.2.2 EcuComposition settings

Refer to AUTOSAR BSW Service API Guide.doc document.

Document (DOC NO)	number	SHT/SHTS 26 / 79
----------------------	--------	---------------------

6. Application Programming Interface (API)

6.1 Type Definitions

6.1.1 KeyM_ConfigType

Name:	KeyM_ConfigType	
Type:	Structure	
Range:	Implementation specific	The content of this data structure is implementation specific
Description:	This structure is the base type to initialize the Key Manager module. A pointer to an instance of this structure will be used in the initialization of the Key Manager module.	
Available via:	KeyM.h	

6.1.2 KeyM_KH_UpdateOperationType

Name:	KeyM_KH_UpdateOperationType		
Type:	Enumeration		
Range:	KEYM_KH_UPDATE_KEY_UPDATE_REPE AT	0x01	Key handler has successfully performed the operation and provides new key data that shall be further operated by the update function of the key manager. A next call to key handler is requested.
	KEYM_KH_UPDATE_FINISH	0x02	Key handler has successfully performed all update operation. The update operation is finished and the result data can be provided back for a final result of the KeyM_Update operation.
Description:	Specifies the type of key handler update operation that was performed in the callback.		
Available via:	KeyM.h		

6.1.3 KeyM_CertElementIteratorType

Name:	KeyM_CertElementIteratorType	
Type:	Structure	
Range:	Implementation specific	The content of this data structure is implementation specific
Description:	This structure is used to iterate through a number of elements of a certificate.	
Available via:	KeyM.h	

6.1.4 KeyM_CryptoKeyIdType

Name:	KeyM_CryptoKeyIdType	
Type:	uint16	
Description:	Crypto key handle.	
Available via:	KeyM.h	

6.2 Macro Constants

None

6.3 Functions

6.3.1 General

6.3.1.1 KeyM_Init

Service name:	KeyM_Init	
Syntax:	<pre>void KeyM_Init(const KeyM_ConfigType* ConfigPtr)</pre>	
Service ID[hex]:	0x01	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	ConfigPtr	Pointer to the configuration set in VARIANT-POST-BUILD.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	This function initializes the key management module.	
Available via:	KeyM.h	

6.3.1.2 KeyM_Deinit

Service name:	KeyM_Deinit
Syntax:	void KeyM_Deinit(void)
Service ID[hex]:	0x02
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None
Parameters (inout):	None
Parameters (out):	None
Return value:	None
Description:	This function resets the key management module to the uninitialized state.
Available via:	KeyM.h

6.3.1.3 KeyM_GetVersionInfo

Service name:	KeyM_GetVersionInfo
Syntax:	void KeyM_GetVersionInfo(Std_VersionInfoType* VersionInfo)
Service ID[hex]:	0x03
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None
Parameters (inout):	None
Parameters (out):	VersionInfo Pointer to the version information of this module.
Return value:	None
Description:	Provides the version information of this module.
Available via:	KeyM.h

6.3.2 Crypto key operation

6.3.2.1 KeyM_Start

Service name:	KeyM_Start	
Syntax:	<pre>Std_ReturnType KeyM_Start(KeyM_StartType StartType, const uint8* RequestData, uint16 RequestDataLength, uint8* ResponseData, uint16* ResponseDataLength)</pre>	
Service ID[hex]:	0x04	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	StartType	Defines in which mode the key operation shall be executed.
	RequestData	Information that comes along with the request, e.g. signature
	RequestDataLength	Length of data in the RequestData array
Parameters (inout):	ResponseDataLength	In: Max number of bytes available in ResponseData Out: Actual number
Parameters (out):	ResponseData	Data returned by the function.
Return value:	Std_ReturnType	E_OK: Start operation successfully performed. Key update operations are now allowed. E_NOT_OK: Start operation not accepted. KEYM_E_PARAMETER_MISMATCH: Parameter do not match with expected value. KEYM_E_KEY_CERT_SIZE_MISMATCH: Parameter size doesn't match
Description:	This function is optional and only used if the configuration item KeyMCryptoKeyStartFinalizeFunctionEnabled is set to true. It intends to allow key update operation.	
Available via:	KeyM.h	

6.3.2.2 KeyM_Prepere

Service name:	KeyM_Prepere
----------------------	--------------

Syntax:	<pre>Std_ReturnType KeyM_Prepare(const uint8* RequestData, uint16 RequestDataLength, uint8* ResponseData, uint16* ResponseDataLength)</pre>	
Service ID[hex]:	0x05	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	RequestData	Information that comes along with the request
	RequestDataLength	Length of data in the RequestData array
Parameters (inout):	ResponseDataLength	In: Max number of bytes available in ResponseData Out: Actual number of bytes
	ResponseData	Data returned by the function.
Return value:	Std_ReturnType	<p>E_OK: Service has been accepted and will be processed internally. Results will be provided through a callback</p> <p>E_NOT_OK: Service not accepted due to an internal error.</p> <p>KEYM_E_PARAMETER_MISMATCH: Parameter do not match with expected value.</p> <p>KEYM_E_KEY_CERT_SIZE_MISMATCH: Parameter size doesn't match</p>
Description:	<p>This function is used to prepare a key update operation. The main intent is to provide information for the key operation to the key server. Other operations may start the negotiation for a common secret that is used further to derive key material.</p> <p>This function is only available if KeyMCryptoKeyPrepareFunctionEnabled is set to TRUE.</p>	
Available via:	KeyM.h	

6.3.2.3 KeyM_Update

Service name:	KeyM_Update
Syntax:	<pre>Std_ReturnType KeyM_Update(const uint8* KeyNamePtr, uint16 KeyNameLength, const uint8* RequestDataPtr, uint16 RequestDataLength, uint8* ResultDataPtr, uint16 ResultDataMaxLength)</pre>

)	
Service ID[hex]:	0x06	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	KeyNamePtr	Pointer to an array that defines the name of the key to be updated
	KeyNameLength	Specifies the number of bytes in keyName. The value 0 indicates that no keyName is provided within this function.
	RequestDataPtr	Information that comes along with the request
	RequestDataLength	Length of data in the RequestData array
	ResultDataMaxLength	Max number of bytes available in ResultDataPtr.
Parameters (inout):	None	
Parameters (out):	ResultDataPtr	Pointer to a data buffer used by the function to store results.
Return value:	Std_ReturnType	<p>E_OK: Service has been accepted and will be processed internally. Results will be provided through a callback</p> <p>E_NOT_OK: Service not accepted due to an internal error.</p> <p>E_BUSY: Service could not be accepted because another operation is already ongoing. Try next time.</p> <p>KEYM_E_PARAMETER_MISMATCH: Parameter do not match with expected value.</p> <p>KEYM_E_KEY_CERT_SIZE_MISMATCH: Parameter size doesn't match</p>
Description:	This function is used to initiate the key generation or update process.	
Available via:	KeyM.h	

6.3.2.4 KeyM_Finalize

Service name:	KeyM_Finalize
Syntax:	<pre>Std_ReturnType KeyM_Finalize(const uint8* RequestDataPtr, uint16 RequestDataLength, uint8* ResponseDataPtr, uint16 ResponseMaxDataLength</pre>

)	
Service ID[hex]:	0x07	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	RequestDataPtr	Information that comes along with the request
	RequestDataLength	Length of data in the RequestData array
Parameters (inout):	ResponseMaxDataLength	In: Max number of bytes available in ResponseData Out: Actual number of bytes in ResponseData or left untouched if service runs in asynchronous mode and function returns KEYM_E_OK.
Parameters (out):	ResponseDataPtr	Data returned by the function.
Return value:	Std_ReturnType	E_OK: Operation has been accepted and will be processed internally. Results will be provided through a callback E_NOT_OK: Operation not accepted due to an internal error. KEYM_E_BUSY: Validation cannot be performed yet. KeyM is currently busy with other jobs. KEYM_E_PARAMETER_MISMATCH: Parameter do not match with expected value. KEYM_E_KEY_CERT_SIZE_MISMATCH: Parameter size doesn't match
Description:	The function is used to finalize key update operations. It is typically used in conjunction with the KeyM_Start operation and returns the key operation into the idle mode. Further key prepare or update operations are not accepted until a new KeyM_Start operation has been initialized. This function is only available if KeyMCryptoKeyStartFinalizeFunctionEnabled is set to TRUE. In addition, updated key material will be persisted and set into valid state (calling Csm_KeySetValid).	
Available via:	KeyM.h	

6.3.2.5 KeyM_Verify

Service name:	KeyM_Verify
----------------------	-------------

Syntax:	<pre>Std_ReturnType KeyM_Verify(const uint8* KeyNamePtr, uint16 KeyNameLength, const uint8* RequestData, uint16 RequestDataLength, uint8* ResponseData, uint16* ResponseDataLength)</pre>	
Service ID[hex]:	0x08	
Sync/Async:	Synchronous Synchronous/Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	KeyNamePtr	Points to an array that defines the name of the key to be updated
	KeyNameLength	Specifies the number of bytes in KeyNamePtr. The value 0 indicates that no KeyNamePtr is provided within this function.
	RequestData	Information that comes along with the request
	RequestDataLength	Length of data in the RequestData array
Parameters (inout):	ResponseDataLength	In: Max number of bytes available in ResponseData Out: Actual number of bytes in ResponseData or left untouched if service runs in asynchronous mode and function returns KEYM_E_PENDING
Parameters (out):	ResponseData	Data returned by the function.
Return value:	Std_ReturnType	KEYM_E_PENDING: Operation runs in asynchronous mode, has been accepted and will be processed internally. Results will be provided through callback E_OK: Operation was successfully performed. Result information are available. E_NOT_OK: Operation not accepted due to an internal error. KEYM_E_BUSY: Validation cannot be performed yet. KeyM is currently busy with other jobs (for asynchronous mode).

6.3.3 Certificate handling

6.3.3.1 KeyM_ServiceCertificate

Service name:	KeyM_ServiceCertificate
----------------------	-------------------------

Syntax:	Std_ReturnType KeyM_ServiceCertificate(KeyM_ServiceCertificateType Service, const uint8* CertNamePtr, uint16 CertNameLength, const uint8* RequestData, uint16 RequestDataLength, uint8* ResponseData, uint16 ResponseDataLength) 	
Service ID[hex]:	0x09	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	Service	Provides the type of service the key manager has to perform.
	CertNamePtr	Points to an array that defines the name of the certificate to be updated
	CertNameLength	Specifies the number of bytes in CertNamePtr. The value 0 indicates that no CertNamePtr is provided within this function.
	RequestData	Information that comes along with the request
	RequestDataLength	Length of data in the RequestData array
	ResponseDataLength	Max number of bytes available in ResponseDataPtr.
Parameters (inout):	None	
Parameters (out):	ResponseData	Data returned by the function.
Return value:	Std_ReturnType	E_OK: Service data operation successfully accepted. E_NOT_OK: Operation not accepted due to an internal error. KEYM_E_PARAMETER_MISMATCH: Parameter do not match with expected value. KEYM_E_KEY_CERT_SIZE_MISMATCH: Parameter size doesn't match
Description:	The key server requests an operation from the key client. The type of operation is specified in the first parameter KeyM_ServiceCertificateType. Certificate operation requests are operated through this function. This function is only available if the configuration parameter KeyMServiceCertificateFunctionEnabled is set to TRUE.	
Available via:	KeyM.h	

6.3.3.2 KeyM_SetCertificate

Service name:	KeyM_SetCertificate	
Syntax:	<pre>Std_ReturnType KeyM_SetCertificate(KeyM_CertificateIdType CertId, const KeyM_CertDataType* CertificateDataPtr)</pre>	
Service ID[hex]:	0x0a	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	CertId	Holds the identifier of the certificate
	CertificateDataPtr	Pointer to a structure that provides the certificate data.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Certificate accepted. E_NOT_OK: Certificate could not be set. KEYM_E_PARAMETER_MISMATCH: Parameter do not match with expected value. KEYM_E_KEY_CERT_SIZE_MISMATCH: Parameter size doesn't match
Description:	This function provides the certificate data to the key management module to temporarily store the certificate.	
Available via:	KeyM.h	

6.3.3.3 KeyM_GetCertificate

Service name:	KeyM_GetCertificate	
Syntax:	<pre>Std_ReturnType KeyM_GetCertificate(KeyM_CertificateIdType CertId, KeyM_CertDataType* CertificateDataPtr)</pre>	
Service ID[hex]:	0x0b	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	CertId	Holds the identifier of the certificate

Parameters (inout):	CertificateDataPtr	Provides a pointer to a certificate data structure. The buffer located by the pointer in the structure shall be provided by the caller of this function. The length information indicates the maximum length of the buffer when the function is called. If E_OK is returned, the length information indicates the actual length of the certificate data in the buffer.
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK Certificate data available and provided. E_NOT_OK: Operation not accepted due to an internal error. KEYM_E_PARAMETER_MISMATCH: Certificate ID invalid. KEYM_E_KEY_CERT_SIZE_MISMATCH: Provided buffer for the certificate too small. KEYM_E_KEY_CERT_EMPTY: No certificate data available, the certificate slot is empty. KEYM_E_KEY_CERT_READ_FAIL: Certificate cannot be provided, access denied.
Description:	This function provides the certificate data	
Available via:	KeyM.h	

6.3.3.4 KeyM_VerifyCertificates

Service name:	KeyM_VerifyCertificates	
Syntax:	Std_ReturnType KeyM_VerifyCertificates(KeyM_CertificateIdType CertId, KeyM_CertificateIdType CertUpperId)	
Service ID[hex]:	0x0c	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	CertId	Holds the identifier of the lower certificate in the chain
	CertUpperId	Holds the identifier of the upper certificate in the chain
Parameters (inout):	None	
Parameters (out):	None	

Return value:	Std_ReturnType	<p>E_OK: Certificate verification request accepted. Operation will be performed in the background and response is given through a callback.</p> <p>E_NOT_OK: Operation not accepted due to an internal error.</p> <p>KEYM_E_BUSY: Validation cannot be performed yet. KeyM is currently busy with other jobs.</p> <p>KEYM_E_PARAMETER_MISMATCH: Certificate ID invalid.</p> <p>KEYM_E_KEY_CERT_EMPTY: One of the certificate slots are empty.</p> <p>KEYM_E_CERT_INVALID_CHAIN_OF_TRUST: An upper certificate is not valid.</p>
Description:	This function verifies two certificates that are stored and parsed internally against each other. The certificate referenced with CertId was signed by the certificate referenced with certUpperId. Only these two certificates are validated against each other.	
Available via:	KeyM.h	

6.3.3.5 KeyM_VerifyCertificate

Service name:	KeyM_VerifyCertificate	
Syntax:	Std_ReturnType KeyM_VerifyCertificate(KeyM_CertificateIdType CertId)	
Service ID[hex]:	0x0d	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	CertId	Holds the identifier of the certificate
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	<p>E_OK: Certificate verification request accepted. Operation will be performed in the background and response is given through a callback.</p> <p>E_NOT_OK: Operation not accepted due to an internal error.</p> <p>KEYM_E_BUSY: Validation cannot be performed yet. KeyM is currently busy with other jobs.</p> <p>KEYM_E_PARAMETER_MISMATCH: Certificate ID invalid.</p> <p>KEYM_E_KEY_CERT_EMPTY: One of the certificate slots are empty.</p>

		KEYM_E_CERT_INVALID_CHAIN_OF_TRUST: An upper certificate is not valid.
Description:	This function verifies a certificate that was previously provided with KeyM_SetCertificate() against already stored and provided certificates stored with other certificate IDs.	
Available via:	KeyM.h	

6.3.3.6 KeyM_VerifyCertificateChain

Service name:	KeyM_VerifyCertificateChain	
Syntax:	<pre>Std_ReturnType KeyM_VerifyCertificateChain(KeyM_CertificateIdType CertId, const KeyM_CertDataType[] certChainData, uint8 NumberOfCertificates)</pre>	
Service ID[hex]:	0x0e	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	CertId	Holds the identifier of the last certificate in the chain.
	certChainData	This is a pointer to an array of certificates sorted according to the order in the PKI.
	NumberOfCertificates	Defines the number of certificates stored in the CertChainData array.
Parameters (inout):	None	
Parameters (out):	None	

Return value:	Std_ReturnType	<p>E_OK: Certificate verification request accepted. Operation will be performed in the background and response is given through a callback.</p> <p>E_NOT_OK: Operation not accepted due to an internal error.</p> <p>KEYM_E_BUSY: Validation cannot be performed yet. KeyM is currently busy with other jobs.</p> <p>KEYM_E_PARAMETER_MISMATCH: Certificate ID invalid.</p> <p>KEYM_E_KEY_CERT_EMPTY: One of the certificate slots are empty.</p> <p>KEYM_E_CERT_INVALID_CHAIN_OF_TRUST: An upper certificate is not valid.</p>
Description:	<p>This function performs a certificate verification against a list of certificates. It is a pre-requisite that the certificate that shall be checked has already been written with KeyM_SetCertificate() and that the root certificate is either in the list or is already assigned to one of the other certificates.</p>	
Available via:	KeyM.h	

6.3.3.7 KeyM_CertElementGet

Service name:	KeyM_CertElementGet	
Syntax:	<pre>Std_ReturnType KeyM_CertElementGet(KeyM_CertificateIdType CertId, KeyM_CertElementIdType CertElementId, uint8* CertElementData, uint32* CertElementDataLength)</pre>	
Service ID[hex]:	0x0f	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	CertId	Holds the identifier of the certificate
	CertElementId	Specifies the ElementId where the data shall be read from.
Parameters (inout):	CertElementDataLength	In: Pointer to a value that contains the maximum data length of the CertElementData buffer.
		Out: The data length will be overwritten with the actual length of data placed to the buffer if the function returns E_OK. Otherwise, the it will be overwritten with the value zero.

Parameters (out):	CertElementData	Pointer to a data buffer allocated by the caller of this function. If available, the function returns E_OK and copies the data into this buffer.
Return value:	Std_ReturnType	E_OK: Element found and data provided in the buffer. E_NOT_OK: Element data not found. KEYM_E_PARAMETER_MISMATCH: Certificate ID or certificate element ID invalid. KEYM_E_KEY_CERT_SIZE_MISMATCH: Provided buffer for the certificate element too small. KEYM_E_KEY_CERT_EMPTY: No certificate data available, the certificate slot is empty. KEYM_E_KEY_CERT_INVALID: The certificate is not valid or has not yet been verified.
Description:	Provides the content of a specific certificate element. The certificate configuration defines how the certificate submodule can find the element, e.g. by providing the object identifier (OID). This function is used to retrieve this information if only one element is assigned to the respective OID.	
Available via:	KeyM.h	

6.3.3.8 KeyM_CertElementGetFirst

Service name:	KeyM_CertElementGetFirst	
Syntax:	<pre>Std_ReturnType KeyM_CertElementGetFirst(KeyM_CertificateIdType CertId, KeyM_CertElementIdType CertElementId, KeyM_CertElementIteratorType* CertElementIterator, uint8* CertElementData, uint32* CertElementDataLength)</pre>	
Service ID[hex]:	0x10	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant Reentrant for one iterator.	
Parameters (in):	CertId	Holds the identifier of the certificate
	CertElementId	Specifies the CertElementId where the data shall be read from.
Parameters	CertElementIterator	Pointer to a structure that is allocated and maintained by the caller. It shall not be destroyed or altered by the application until all elements have been retrieved through KeyM_CertElementGetNext().

(inout):	CertElementDataLength	In: Pointer to a value that contains the maximum data length of the CertElementData buffer. Out: The data length will be overwritten with the actual length of data placed to the buffer if the function returns E_OK.
Parameters (out):	CertElementData	Pointer to a data buffer allocated by the caller of this function. If available, the function returns E_OK and copies the data into this buffer.
Return value:	Std_ReturnType	E_OK: Element found and data provided in the buffer. The certElementIterator has been initialized accordingly. E_NOT_OK: Element data not found. CertElementIterator cannot be used for further calls. KEYM_E_PARAMETER_MISMATCH: Certificate ID or certificate element ID invalid. KEYM_E_KEY_CERT_SIZE_MISMATCH: Provided buffer for the certificate element too small. KEYM_E_KEY_CERT_EMPTY: No certificate data available, the certificate is empty. KEYM_E_CERT_INVALID: Certificate is not valid or not verified successfully
Description:	This function is used to initialize the iterative extraction of a certificate data element. It always retrieves the top element from the configured certificate element and initializes the structure KeyM_CertElementIterator so that consecutive data from this element can be read with KeyM_CertElementGetNext().	
Available via:	KeyM.h	

6.3.3.9 KeyM_CertElementGetNext

Service name:	KeyM_CertElementGetNext
Syntax:	Std_ReturnType KeyM_CertElementGetNext(KeyM_CertElementIteratorType* CertElementIterator, uint8* CertElementData, uint32* CertElementDataLength)
Service ID[hex]:	0x11
Sync/Async:	Synchronous

Reentrancy:	Reentrant Reentrant for one iterator.	
Parameters (in):	None	
Parameters (inout):	CertElementIterator	Pointer to a structure that is allocated by the caller and used by the function. It shall not be destroyed or altered by the application until all elements have been read from the list.
	CertElementDataLength	In: Pointer to a value that contains the maximum data length of the CertElementData buffer. Out: The data length will be overwritten with the actual length of data placed to the buffer if the function returns E_OK.
Parameters (out):	CertElementData	Pointer to a data buffer allocated by the caller of this function. If available, the function returns E_OK and copies the data into this buffer.
Return value:	Std_ReturnType	E_OK: Element found and data provided in the buffer. The CertElementIterator has been initialized accordingly. E_NOT_OK: Element data not found. CertElementIterator cannot be used for further calls. KEYM_E_PARAMETER_MISMATCH: Certificate ID or certificate element ID invalid. KEYM_E_KEY_CERT_SIZE_MISMATCH: Provided buffer for the certificate element too small. KEYM_E_KEY_CERT_EMPTY: No certificate data available, the certificate is empty. KEYM_E_CERT_INVALID: Certificate is not valid or not verified successfully
Description:	This function provides further data from a certificate element, e.g. if a set of data are located in one certificate element that shall be read one after another. This function can only be called if the function KeyM_CertElementGetFirst() has been called once before.	
Available via:	KeyM.h	

6.3.3.10 KeyM_CertGetStatus

Service name:	KeyM_CertGetStatus
----------------------	--------------------

Syntax:	Std_ReturnType KeyM_CertGetStatus(KeyM_CertificateIdType CertId, KeyM_CertificateStatusType* Status)	
Service ID[hex]:	0x12	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	CertId	Holds the identifier of the certificate
Parameters (inout):	None	
Parameters (out):	Status	Provides the status of the certificate.
Return value:	Std_ReturnType	E_OK
Description:	This function provides the status of a certificate.	
Available via:	KeyM.h	

6.3.3.11 KeyM_CertificateElementGetByIndex

Service name:	KeyM_CertificateElementGetByIndex	
Syntax:	Std_ReturnType KeyM_CertificateElementGetByIndex (KeyM_CertificateIdType CertId, KeyM_CertElementIdType CertElementId, uint32 Index, uint8* CertElementDataPtr, uint32*CertElementDataLengthPtr r)	
Service ID[hex]:	0x13	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	CertId	Identifier of the certificate where the element shall be read from.
	CertElementId	Specifies the ElementId where the data shall be read from.

	Index	Specifies the index to the element that shall be read (0..N).
Parameters (inout):	CertElementData LengthPtr	In: Pointer to a value that contains the maximum data length of the CertElementData buffer. Out: The data length will be overwritten with the actual length of data placed to the buffer if the function returns E_OK.
Parameters (out):	CertElementData Ptr	Pointer to a data buffer allocated by the caller of this function. If the function returns E_OK element data are copied into this buffer
Return value:	Std_ReturnType	E_OK: Element found and data provided in the buffer. The CertElementIterator has been initialized accordingly. E_NOT_OK: Element data not found. CertElementIterator cannot be used for further calls. KEYM_E_PARAMETER_MISMATCH: Certificate ID or certificate element ID invalid. KEYM_E_KEY_CERT_SIZE_MISMATCH: Provided buffer for the certificate element too small. KEYM_E_KEY_CERT_EMPTY: No certificate data available, the certificate is empty. KEYM_E_CERT_INVALID: Certificate is not valid or not verified successfully
Description:	This function provides the element data of a certificate. The function is used if an element type can have more than one parameter. The index specifies which element shall be read. The function works similar to the KeyM_CertElementGetFirst/KeyM_CertElementGetNext, but instead of the iteration, the individual elements can be accessed by index (like the operation in the service interface)	
Available via:	KeyM.h	

6.3.3.12 KeyM_CertificateElementGetCount

Service name:	KeyM_CertificateElementGetCount
Syntax:	Std_ReturnType KeyM_CertificateElementGetCount (KeyM_CertificateIdType CertId, KeyM_CertElementIdType CertElementId, uint16* CountPtr

)	
Service ID[hex]:	0x14	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	CertId	Identifier of the certificate where the element shall be read from.
	CertElementId	Specifies the ElementId where the data shall be read from.
Parameters (inout):	None	
Parameters (out):	CountPtr	Pointer to the buffer where the number of available data elements for this certificate element shall be copied to.
Return value:	Std_ReturnType	E_OK: Count value has been provided. E_NOT_OK: Unable to provide the count value. KEYM_E_PARAMETER_MISMATCH: Certificate ID or certificate element ID invalid resp. out of range.
Description:	This function provides the total number of data elements that are available for the specified certificate element. Typically, only one data element is available. But in some cases, several data elements can be assigned to one certificate element in a row. This function retrieves the total number of elements. The individual data elements can then accessed with KeyM_CertificateElementGetByIndex(). It is similar to the functions KeyM_CertElementGetFirst/KeyM_CertElementGetNext to retrieve a group of data elements of one certificate element.	
Available via:	KeyM.h	

6.3.3.13 KeyM_CertElementGetOIDHex

Service name:	KeyM_CertElementGetOIDHex
----------------------	---------------------------

Syntax:	Std_ReturnType KeyM_CertElementGetOIDHex (KeyM_CertificateIdType CertId, KeyM_CertElementIdType CertElementId, uint8* CertElementOidData, uint32* CertElementOidDataLength)	
Service ID[hex]:	None	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	CertId	Identifier of the certificate where the element shall be read from.
	CertElementId	Specifies the ElementId where the data shall be read from.
Parameters (inout):	CertElementOidDataLength	In: Pointer to a value that contains the maximum data length of the CertElementData buffer. Out: The data length will be overwritten with the actual length of data placed to the buffer if the function returns E_OK.
Parameters (out):	CertElementOidData Ptr	Pointer to a data buffer allocated by the caller of this function. If the function returns E_OK element data are copied into this buffer
Return value:	Std_ReturnType	E_OK: Element found and data provided in the buffer. The CertElementIterator has been initialized accordingly. E_NOT_OK: Element data not found. CertElementIterator cannot be used for further calls. KEYM_E_PARAMETER_MISMATCH: Certificate ID or certificate element ID invalid. KEYM_E_KEY_CERT_SIZE_MISMATCH: Provided buffer for the certificate element too small. KEYM_E_KEY_CERT_EMPTY: No certificate data available, the certificate is empty. KEYM_E_CERT_INVALID: Certificate is not valid

		or not verified successfully
Description:	This function provides the OID data of element in certificate. The function works similar to the KeyM_CertElementGet, but return value is hex value of OID not element data.	
Available via:	KeyM.h	

6.3.3.14KeyM_CertGetDERPacketData

Service name:	KeyM_ CertGetDERPacketData	
Syntax:	Std_ReturnType KeyM_ CertGetDERPacketData (KeyM_CertificateIdType CertId, uint8* ptrData, uint32* ptrData Length uint8 Typestruct)	
Service ID[hex]:	None	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	CertId	Identifier of the certificate where the element shall be read from.
	Typestruct	Specifies the type of which information should be read.
Parameters (inout):	PtrDataLength	In: Pointer to a value that contains the maximum data length of the CertElementData buffer. Out: The data length will be overwritten with the actual length of data placed to the buffer if the function returns E_OK.

Parameters (out):	PtrData	Pointer to a data buffer allocated by the caller of this function. If the function returns E_OK element data are copied into this buffer
Return value:	Std_ReturnType	E_OK: Element found and data provided in the buffer. The CertElementIterator has been initialized accordingly. E_NOT_OK: Element data not found. CertElementIterator cannot be used for further calls. KEYM_E_PARAMETER_MISMATCH: Certificate ID or certificate element ID invalid. KEYM_E_KEY_CERT_SIZE_MISMATCH: Provided buffer for the certificate element too small. KEYM_E_KEY_CERT_EMPTY: No certificate data available, the certificate is empty. KEYM_E_CERT_INVALID: Certificate is not valid or not verified successfully
Description:	This function provides the whole information data of specific information of certificate. The function works similar to the KeyM_CertElementGet, but return value is hex value in DER format.	
Available via:	KeyM.h	

6.3.4 Scheduled Functions

6.3.4.1 KeyM_MainFunction

Service name:	KeyM_MainFunction
Syntax:	void KeyM_MainFunction(void)
Service ID[hex]:	0x19
Description:	Function is called periodically according the specified time interval.
Available via:	SchM_KeyM.h

6.3.4.2 KeyM_MainBackgroundFunction

Service name:	KeyM_MainBackgroundFunction
Syntax:	void KeyM_MainBackgroundFunction(void)
Service ID[hex]:	0x1a

Description:	Function is called from a pre-emptive operating system when no other task operation is needed. Can be used for calling time consuming synchronous functions such as KeyM_KH_Update().
Available via:	SchM_KeyM.h

6.3.5 Expected Interfaces

6.3.5.1 KeyM_KH_Start

Service name:	KeyM_KH_Start	
Syntax:	<pre>Std_ReturnType KeyM_KH_Start(KeyM_StartType StartType, const uint8* RequestData, uint16 RequestDataLength, uint8* ResponseData, uint16* ResponseDataLength)</pre>	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	StartType	Defines in which mode the key operation shall be executed.
	RequestData	Information that comes along with the request, e.g. signature
	RequestDataLength	Length of data in the RequestData array
Parameters (inout):	ResponseDataLength	In: Max number of bytes available in ResponseData Out: Actual number of bytes in ResponseData if function returns E_OK.
Parameters (out):	ResponseData	Data returned by the function.
Return value:	Std_ReturnType	E_OK: Start operation successfully performed. Key update operations are now allowed. E_NOT_OK: Start operation not accepted. KEYM_E_PARAMETER_MISMATCH: Parameter do not match with expected value. KEYM_E_KEY_CERT_SIZE_MISMATCH: Parameter size doesn't match

Description:	If KeyMCryptoKeyStartFinalizeFunctionEnabled and KeyMCryptoKeyHandlerStartFinalizeEnabled is set to TRUE, this function will be called immediately when KeyM_Start gets called. The function shall return E_OK to switch the Key Manager into the active state for any key operation.
Available via:	KeyM_Externals.h

6.3.5.2 KeyM_KH_Prepare

Service name:	KeyM_KH_Prepare	
Syntax:	<pre>Std_ReturnType KeyM_KH_Prepare(const uint8* RequestData, uint16 RequestDataLength, uint8* ResponseDataPtr, uint16* ResponseDataLength)</pre>	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	RequestData	Information that comes along with the request
	RequestDataLength	Length of data in the RequestData array
Parameters (inout):	ResponseDataLength	In: Max number of bytes available in ResponseData Out: Actual number of bytes in ResponseData.
	ResponseDataPtr	Data returned by the function.
Return value:	Std_ReturnType	E_OK: Service has been accepted and will be processed internally. Results will be provided through a callback
		E_NOT_OK: Service not accepted due to an internal error. KEYM_E_PARAMETER_MISMATCH: Parameter do not match with expected value. KEYM_E_KEY_CERT_SIZE_MISMATCH: Parameter size doesn't match
Description:	If the configuration parameters KeyMCryptoKeyPrepareFunctionEnabled and KeyMCryptoKeyHandlerPrepareEnabled are both set to TRUE, then this function will be called immediately when KeyM_Prepare gets called. The function takes over the task to prepare a key management operation. The response data will be passed on as is to the caller of Key_Prepare.	
Available via:	KeyM_Externals.h	

6.3.5.3 KeyM_KH_Update

Service name:	KeyM_KH_Update	
Syntax:	<pre>Std_ReturnType KeyM_KH_Update(const uint8* KeyNamePtr, uint16 KeyNameLength, const uint8* RequestData, uint16 RequestDataLength, uint8* ResultDataPtr, uint16* ResultDataLengthPtr, KeyM_CryptoKeyIdType* KeyId, KeyM_KH_UpdateOperationType* UpdateOperation)</pre>	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	KeyNamePtr	Points to an array that defines the name of the key to be updated
	KeyNameLength	Specifies the number of bytes in KeyNamePtr. The value 0 indicates that no KeyNamePtr is provided within this function.
	RequestData	Information that comes along with the request
	RequestDataLength	Length of data in the RequestData array
Parameters (inout):	ResultDataLengthPtr	In: Max number of bytes available in ResultDataPtr Out: Actual number of bytes in ResultData or 0 if no data available. Unspecified or untouched if return value indicates a failure.
	KeyId	Provides a reference to the crypto key as an index to the crypto key table. In: Providing the key ID if a name was provided and a key was found. Returns 0xFFFFFFFF if no key was found. Out: Key ID of the key where the operation shall be performed to if updateOperation indicates a key operation.
Parameters (out):	ResultDataPtr	Data returned by the function.
	UpdateOperation	Provides information to the caller what operation has been performed and how to interpret the ResultData.
Return value:	Std_ReturnType	E_OK: Data returned by this function. E_NOT_OK: General error, no data provided. E_BUSY: Service could not be accepted because another operation is already ongoing. Try next time. KEYM_E_PARAMETER_MISMATCH: A parameter does

		not have expected value. Service discarded. KEYM_E_KEY_CERT_WRITE_FAIL: Key could not be
		written. KEYM_E_KEY_CERT_UPDATE_FAIL: General failure on updating a key.
Description:	If the configuration item KeyMCryptoKeyHandlerUpdateEnabled is set to TRUE, the KeyM_Update function will not perform any operation but will delegate the operation to the key handler. On return, the function provides the status of the key operation.	
Available via:	KeyM_Externals.h	

6.3.5.4 KeyM_KH_Finalize

Service name:	KeyM_KH_Finalize	
Syntax:	<pre>Std_ReturnType KeyM_KH_Finalize(const uint8* RequestData, uint16 RequestDataLength, uint8* ResponseData, uint16* ResponseDataLength)</pre>	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	RequestData	Information that comes along with the request
	RequestDataLength	Length of data in the RequestData array
Parameters (inout):	ResponseDataLength	In: Max number of bytes available in ResponseData Out: Actual number of bytes in ResponseData.
	ResponseData	Data returned by the function.
Return value:	Std_ReturnType	<p>E_OK: Operation has been accepted and will be processed internally. Results will be provided through a callback</p> <p>E_NOT_OK: Operation not accepted due to an internal error.</p> <p>KEYM_E_BUSY: Validation cannot be performed yet. KeyM is currently busy with other jobs.</p>

		<p>KEYM_E_PARAMETER_MISMATCH: Parameter do not match with expected value.</p> <p>KEYM_E_KEY_CERT_SIZE_MISMATCH: Parameter size doesn't match</p>
Description:	<p>If KeyMCryptoKeyStartFinalizeFunctionEnabled and KeyMCryptoKeyHandlerStartFinalizeEnabled is set to TRUE, this function will be called immediately when KeyM_Finalize gets called KeyM_Finalize() will not perform any operation but will call this key handler function to delegate the operation.</p>	
Available via:	KeyM_Externals.h	

6.3.5.5 KeyM_KH_Verify

Service name:	KeyM_KH_Verify	
Syntax:	<pre>Std_ReturnType KeyM_KH_Verify(const uint8* KeyNamePtr, uint16 KeyNameLength, const uint8* RequestData, uint16 RequestDataLength, uint8* ResponseData, uint16* ResponseDataLength)</pre>	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	KeyNamePtr	Pointer to an array that defines the name of the key to be updated
	KeyNameLength	Specifies the number of bytes in keyName. The value 0 indicates that no keyName is provided within this function.
	RequestData	Information that comes along with the request
	RequestDataLength	Length of data in the RequestData array
	ResponseDataLength	In: Max number of bytes available in ResponseData Out: Actual number of bytes in ResponseData.
Parameters (out):	ResponseData	Data returned by the function.

Return value:	Std_ReturnType	KEYM_E_PENDING: Operation runs in asynchronous mode, has been accepted and will be processed internally. Results will be provided through callback E_OK: Operation was successfully performed. Result
		information are available. E_NOT_OK: Operation not accepted due to an internal error. KEYM_E_BUSY: Validation cannot be performed yet. KeyM is currently busy with other jobs (for asynchronous mode). KEYM_E_PARAMETER_MISMATCH: Parameter do not match with expected value. KEYM_E_KEY_CERT_SIZE_MISMATCH: Parameter size doesn't match KEYM_E_KEY_CERT_INVALID: Key operation cannot be performed because the key name is invalid. KEYM_E_KEY_CERT_EMPTY: The key for this slot has not been set.
Description:	If KeyMCryptoKeyHandlerVerifyEnabled is set to TRUE, the KeyM_Verify function will not perform any operation but will delegate its operation to this service callback. The intention is to perform a verification of input data using the CSM job referenced with KeyMCryptoKeyCsmVerifyJobRef.	
Available via:	KeyM_Externals.h	

6.3.5.6 KeyM_KH_ServiceCertificate

Service name:	KeyM_KH_ServiceCertificate
Syntax:	Std_ReturnType KeyM_KH_ServiceCertificate(KeyM_ServiceCertificateType Service, const uint8* CertName, uint16 CertNameLength, const uint8* RequestData, uint16 RequestDataLength, uint8* ResponseData, uint16* ResponseDataLength)
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant

Parameters (in):	Service	Provides the type of service the certificate submodule has to perform.
	CertName	Points to an array that defines the name of the key to be updated
	CertNameLength	Specifies the number of bytes in keyName. The value 0 indicates that no keyName is provided within this function.
	RequestData	Information that comes along with the request
	RequestDataLength	Length of data in the RequestData array
Parameters (inout):	ResponseDataLength	In: Max number of bytes available in ResponseData Out: Actual number of bytes in ResponseData.
Parameters (out):	ResponseData	Data returned by the function.
Return value:	Std_ReturnType	E_OK: Service data operation successfully accepted. E_NOT_OK: Operation not accepted due to an internal error. KEYM_E_PARAMETER_MISMATCH: Parameter do not match with expected value. KEYM_E_KEY_CERT_SIZE_MISMATCH: Parameter size doesn't match
Description:	If KeyMCryptoKeyHandlerServiceCertificateEnabled is set to TRUE, this function will be called by KeyM_ServiceCertificate() to delegate the operation to this user specific service function.	
Available via:	KeyM_Externals.h	

6.3.5.7 KeyM_CryptoKeyUpdateCallbackNotification

Service name:	KeyM_CryptoKeyUpdateCallbackNotification	
Syntax:	<pre>void KeyM_CryptoKeyUpdateCallbackNotification(KeyM_ResultType Result, uint16 ResultDataLength, const uint8* ResultDataPtr)</pre>	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	Result	Contains information about the result of the operation.
	ResultDataLength	Contains the length of the resulting data of this operation if any.
	ResultDataPtr	Pointer to the data of the result.
Parameters	None	

(inout):	
Parameters (out):	None
Return value:	None
Description:	Notifies the application that a crypto key update operation has been finished. This function is used by the key manager.
Available via:	KeyM_Externals.h

6.3.5.8 KeyM_CryptoKeyFinalizeCallbackNotification

Service name:	KeyM_CryptoKeyFinalizeCallbackNotification	
Syntax:	<pre>void KeyM_CryptoKeyFinalizeCallbackNotification(KeyM_ResultType Result, uint16 ResultDataLength, uint8* ResultDataPtr)</pre>	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	Result	Contains information about the result of the operation.
	ResultDataLength	Contains the length of the resulting data of this operation.
	ResultDataPtr	Pointer to the data of the result (the data buffer that has been provided with the service function).
Return value:	None	
Description:	<p>Notifies the application that a crypto key finalize operation has been finished. The callback function is only called and needed if KeyMCryptoKeyStartFinalizeFunctionEnabled is set to TRUE.</p>	
Available via:	KeyM_Externals.h	

6.3.5.9 KeyM_CryptoKeyVerifyCallbackNotification

Service name:	KeyM_CryptoKeyVerifyCallbackNotification	
Syntax:	<pre>void KeyM_CryptoKeyVerifyCallbackNotification(KeyM_ResultType Result, uint32 KeyId, uint16 ResultDataLength,</pre>	

	uint8* ResultDataPtr)	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	Result	Contains information about the result of the operation.
	KeyId	The key identifier where this verification was started for.
	ResultDataLength	Contains the length of the resulting data of this operation if any.
	ResultDataPtr	Pointer to the data of the result.
Return value:	None	
Description:	Notifies the application that a crypto key verify operation has been finished. This function is used by the key manager.	
Available via:	KeyM_Externals.h	

6.3.5.10 KeyM_ServiceCertificateCallbackNotification

Service name:	KeyM_ServiceCertificateCallbackNotification	
Syntax:	void KeyM_ServiceCertificateCallbackNotification(KeyM_CertificateIdType CertId, KeyM_ResultType Result, uint16 ResultDataLength, uint8* ResultDataPtr)	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	CertId	The certificate identifier where this service was started for.
	Result	Contains information about the result of the operation.
	ResultDataLength	Contains the length of the resulting data of this operation if any.
	ResultDataPtr	Pointer to the data of the result.

Return value:	None
Description:	<p>Notifies the application that the certificate service operation has been finished.</p> <p>This function is used by the certificate submodule.</p> <p>This callback is only provided if KeyMServiceCertificateFunctionEnabled is set to TRUE. The function name is configurable by KeyMServiceCertificateCallbackNotificationFunc.</p>
Available via:	KeyM_Externals.h

6.3.5.11 KeyM_CertificateVerifyCallbackNotification

Service name:	KeyM_CertificateVerifyCallbackNotification	
Syntax:	<pre>Std_ReturnType KeyM_CertificateVerifyCallbackNotification(KeyM_CertificateIdType CertId, KeyM_CertificateStatusType Result)</pre>	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	CertId	The certificate identifier that has been verified.
	Result	Contains information about the result of the operation.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK
Description:	<p>Notifies the application that a certificate verification has been finished. The function name is configurable by KeyMCertificateVerifyCallbackNotificationFunc.</p>	
Available via:	KeyM_Externals.h	

6.4 Noted

6.4.1 In Communication with application SW-C

For the prototype of the RTE-based generated function, see the AUTOSAR BSW Service API Guide.doc document.

7. Generator

7.1 Generator Option

Options	Description
-G,--Generation	Symbolic parameters to be used for fore generation (skip validation).
-H,--Help	Display this help message.
-I,--Input <I>	ECU description file path of the module for which generation tool need to run.
-L,--Log	Symbolic parameters to be used for generation error log.
-M,--Module <M>	Specify module name and version to be generated code for.
-O,--Output <O>	Project-relative path to location where the generated code is to be placed.
-T,--Top_path <T>	Symbolic parameters to be used for set path of module.
-V,--Validate	Symbolic parameters to be used for invoking validation checks.

7.2 Generator Error Messages

ERR109001 : The value configured for parameter MODULE-ID in container BSW-MODULE-DESCRIPTION must be 109.

This error occurs, If value of Module Id in file BSWMD is not equal 109.

Report error

ERR109002 : The value configured for parameter VENDOR-ID in container BSW-IMPLEMENTATION must be 76.

This error occurs, If value of Vendor Id in file BSWMD is not equal 76.

Report error

ERR109003 : The parameter <Parameter Name> in the container <Container Name> should be configured.

This error occurs, if any of the below mentioned mandatory parameters are not configured.

Container Name	Parameter Name
BSW-IMPLEMENTATION	AR-RELEASE-VERSION
	VENDOR-ID
	SW-VERSION
BSW-MODULE-DESCRIPTION	MODULE-ID

ERR109004 : The value configured parameter <Parameter Name> in the container <Container Name> must have correct format (for example 1.0.0).

This error occurs, if the below parameter is configured incorrect format value.

Parameter Name	Container Name
SW-VERSION	BSW-IMPLEMENTATION

ERR109005 : The value of parameter <AR-RELEASE-VERSION> should be 4.4.0.

This error occurs, if the configured parameter <AR-RELEASE-VERSION> is not 4.4.0.

Report error

ERR109006 : The value of the parameter <Parameter Name> in the container <Container Name> mentioned in the below table must have start value of 0.

Container Name	Parameter Name
KeyMCertificate	KeyMCertificateId
KeyMCryptoKey	KeyMCryptoKeyId

This error occurs, If (KeyMCertificateId parameter of KeyMCertificate or KeyMCryptoKeyId parameter of KeyMCryptoKey have not start value of 0).

Report error

ERR109028 : The value of the parameter <Parameter Name> in the container <Container Name> mentioned in the below table must be continuous .

Container Name	Parameter Name
KeyMCertificate	KeyMCertificateId
KeyMCryptoKey	KeyMCryptoKeyId

This error occurs, If (value of KeyMCertificateId parameter of KeyMCertificate container or KeyMCryptoKeyId parameter of KeyMCryptoKey container is continuous).

Report error

ERR109029 : The value of the parameter KeyMCertificateElementId must be start from 0.

This error occurs, If (value of KeyMCertificateElementId parameter of KeyMCertificateElement container is not started from 0).

Report error

ERR109030 : The value of the parameter KeyMCertificateElementId must be continuous.

This error occurs, If (value of KeyMCertificateElementId parameter of KeyMCertificateElement container is not continuous).

Report error

ERR109008 : The value of the parameter KeyMCryptoKeyVerifyFunctionEnable must be TRUE if the parameter KeyMCryptoCsmVerifyJobType is configured.

This error occurs, If (value of KeyMCryptoKeyVerifyFunctionEnable parameter is FALSE and the parameter KeyMCryptoCsmVerifyJobType is configured)

Report error

ERR109009 : The parameter KeyMCryptoCsmKeySourceDeriveRef must be configured if the parameter KeyMCryptoKeyGenerationType is set to KEYM_DERIVED_KEY.

This error occurs, If (KeyMCryptoCsmKeySourceDeriveRef parameter is not configured and the parameter KeyMCryptoKeyGenerationType is set to KEYM_DERIVED_KEY)

Report error

ERR109007 : The value of parameter <Parameter Name> of <Element Name> element of a <Certificate Format

Type Name> certificate must be one of values <Supported Value>

Certificate Format Type Name	Parameter Name	Element Name	Supported Value
X509	KeyMCertificateElementObjectId	CertificateExtension	2.5.29.15, 2.5.29.17, 2.5.29.19, 2.5.29.37, 2.5.29.35, 2.5.29.14, 2.5.29.31, 1.3.6.1.5.5.7.1.1, Others*
CRL			2.5.29.20, 2.5.29.35, 2.5.29.28

This error occurs, If (value of KeyMCertificateElementObjectId parameter of CertificateExtension element is not supported valued).

Report error

*: are considered as Un-known extension types.

ERR109010 : The parameter KeyMArgumentRef in container KeyMCertificateElementRule must not configured to reference to other KeyMCertificateElementRule.

This error occurs, If (KeyMArgumentRef parameter is configured to refer to KeyMCertificateElementRule)

Report error

ERR109013 : For CVC certificate, the parameter KeyMCertificateElementObjectType of CertificateSubjectPublicKeyInfo_SubjectPublicKey element must be configured.

This error occurs, If (KeyMCertFormatType is CVC format and the parameter KeyMCertificateElementObjectType of not configured for CertificateSubjectPublicKeyInfo_SubjectPublicKey element is not configured)

Report error

ERR109011 : Except <Cert Element Name>, other elements must be configured once time only in <Cert Format Name> certificate.

Cert Format Name	Cert Element Name
X509	CertificateExtension, CertificateIssuerName, CertificateSubjectName
CRL	CertificateExtension, CertificateIssuerName, CertificateRevokedCertificates
CVC	CertificateExtension, CertificateSubjectPublicKeyInfo_SubjectPublicKey

This error occurs, If (In a <Cert Format Name> certificate, a element is configured more than once and the element is not <Cert Element Name>)

Report error

ERR109012 : For X509 certificate only refer to itself or other X509 certificate. For CVC certificate only refer to itself or other CVC certificate. For CRL certificate only refers X509 or CVC certificate.

This error occurs, If (the KeyMCertUpperHierachicalCertRef of X509 certificate refers to CVC or CRL)

Report error

This error occurs, If (the KeyMCertUpperHierachicalCertRef of CVC certificate refers to X509 or CRL)

Report error

This error occurs, If (the KeyMCertUpperHierachicalCertRef of CRL certificate refers to CRL)

Report error

ERR109014 : A KeyMNvMBlock configuration must not referenced by more than one Certificate or Key.

This error occurs, If (a KeyMNvMBlock configuraion is referenced by more than one Certificate or Key)

Report error

ERR109031 : A NvMBlockDescriptor configuration is referenced by more than KeyMNvMBlock.

This error occurs, If (a NvMBlockDescriptor configuration is referenced by more than one KeyMNvMBlock)

Report error

ERR109015 : When <Parameter2 Name> is configured to KEYM_STORAGE_IN_NVM, the <Parameter1 Name> must be configured.

Parameter1 Name	Parameter2 Name
KeyMCertificateNvmBlockRef	KeyMCertificateStorage
KeyMCryptoKeyNvmBlockRef	KeyMCryptoKeyStorage

This error occurs, If (<Parameter2 Name> is configured to KEYM_STORAGE_IN_NVM and <Parameter1 Name> is not configured)

Report error

ERR109018 : The parameter KeyMCertificateElementRef in KeyMCertificateElementCondition container is must be configured.

This error occurs, If (KeyMCertificateElementRef parameter in KeyMCertificateElementCondition container is not configured)

Report error

ERR109019 : The parameter CsmProcessingMode of CsmJob that is referenced by KeyCertCsmSignatureVerifyJobRef or KeyCertCsmSignatureGenerateJobRef must be configured as CRYPTO_PROCESSING_SYNC.

This error occurs, If (The parameter CsmProcessingMode of CsmJob that is referenced by KeyCertCsmSignatureVerifyJobRef or KeyCertCsmSignatureGenerateJobRef is configured as CRYPTO_PROCESSING_ASYNC)

Report error

ERR109020 : When certificate format is X509 certificate, if certificate element is CertificateExtension , KeyMCertificateElementObjectId must be configured.

This error occurs, If (certificate format is X509 certificate, if certificate element is CertificateExtension and

KeyMCertificateElementObjectId is not configured)

Report error

ERR109021 : The value of parameter KeyMCryptoKeyGenerationInfo must be correct format. E.g. "AA BB CC".

This error occurs, If (The value of parameter KeyMCryptoKeyGenerationInfo is not correct format)

Report error

ERR109022 : The length of value of parameter KeyMCertificateName or KeyMCryptoKeyName must be smaller than or equal value of KeyMKeyCertNameMaxLength.

This error occurs, If (The length of value of parameter KeyMCertificateName or KeyMCryptoKeyName is bigger than value of KeyMKeyCertNameMaxLength)

Report error

ERR109025 : one of 2 parameters (KeyMCryptoCsmVerifyJobType and KeyMCryptoKeyCsmVerifyJobRef) in container KeyMCryptoKey is configured and remaining parameter must be configured.

This error occurs, If (one of 2 parameters (KeyMCryptoCsmVerifyJobType and KeyMCryptoKeyCsmVerifyJobRef) in container KeyMCryptoKey is configured and remaining parameter is not configured.)

Report error

ERR109026 : The KeyMcertificateElementConditionSenderReceiver parameter must be referenced to a data element of R-Port.

This error occurs, If (The KeyMcertificateElementConditionSenderReceiver parameter is not referenced to a data element of R-Port.)

Report error

ERR109024 : The value of KeyMCertificateName and KeyMCryptoKeyName parameter must be unique.

This error occurs, If (The value of KeyMCertificateName and KeyMCryptoKeyName parameter is not unique)

Report error

ERR109023 : If configured, the value of KeyMCryptoKeyCryptoProps parameter must be value in the below table.

Parameter Name	Value
KeyMCryptoKeyCryptoProps	SECRET_KEY
	MASTER_ECU_KEY
	BOOT_MAC_KEY
	BOOT_MAC
	KEY_1
	KEY_2
	KEY_3
	KEY_4
	KEY_5
	KEY_6
	KEY_7
	KEY_8
	KEY_9
	KEY_10
	RAM_KEY

This error occurs, If (the value of KeyMCryptoKeyCryptoProps parameter is not value in the above table)

Report error

ERR109032 : If KeyMUserGetCurrentTimeFunc parameter is configured, the KeyMTimeBaseRef parameter must be not configured.

This error occurs, If (KeyMUserGetCurrentTimeFunc is configured and KeyMTimeBaseRef parameter is configured)

Report error

ERR109033 : If KeyMUserGetCurrentTimeFunc parameter is not configured, the KeyMTimeBaseRef parameter must be configured.

This error occurs, If (KeyMUserGetCurrentTimeFunc is not configured and KeyMTimeBaseRef parameter is not configured)

Report error

ERR109034: The number of structure CertificateSubjectPublicKeyInfo_PublicKeyAlgorithm maximum is 2 for ECC X509 and 1 for RSA X509 certificate.

Only for EC Certificate: + If number of CertificateSubjectPublicKeyInfo_PublicKeyAlgorithm is 2: Both OID of elements PublicKeyAlgorithm must be configured correctly

(for algorithm OID the value are:[1.2.840.10045.2.1; 1.3.132.1.12; 1.2.840.113549.1.1.1; 1.2.840.113549.1.1.10]

for curvename OID the value are:[1.3.132.0.33; 1.2.840.10045.3.1.7; 1.3.132.0.35; 1.3.132.0.8])

+ If number of CertificateSubjectPublicKeyInfo_PublicKeyAlgorithm is 1: OID of element PublicKeyAlgorithm must be blank.

This error occurs, If (element CertificateSubjectPublicKeyInfo_PublicKeyAlgorithm is not configured correctly)

Report error

7.3 Warning Messages

None

7.4 Information Messages

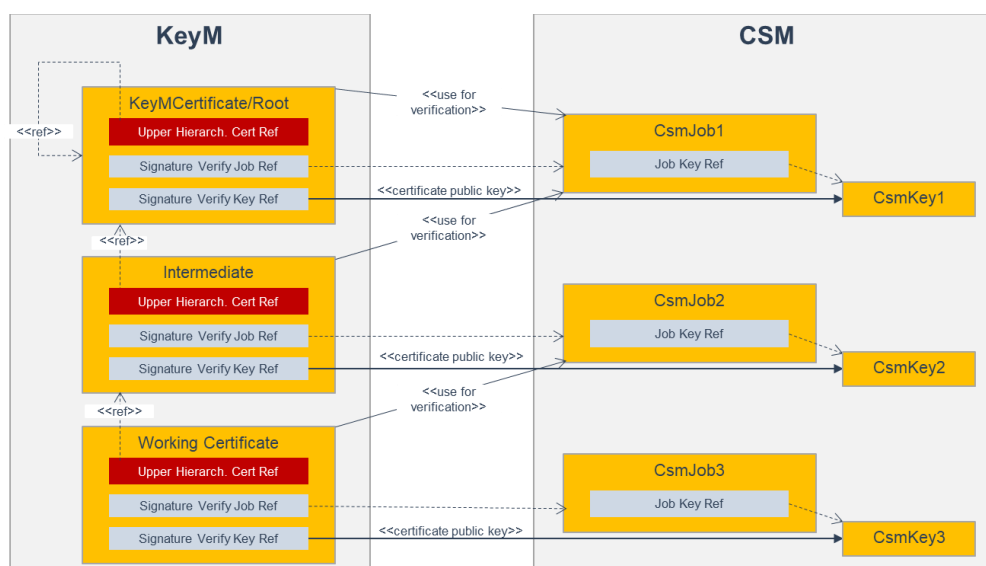
None

8. Appendix

8.1 Design Notes

8.1.1 Certificate Chain in Hierachy

The below figure illustrates the configuration of the CSM job and key and the references from the *KeyMCertificate*. This shows Signature Verify Job Ref and Signature Verify Key Ref of the upper certificate are used for signature verification of lower certificate.



8.1.2 Certificate Formats

Certificates can be represented in different formats. The configuration foresees three different formats, the X.509, CVC and CRL. Key elements that are assigned to certificates can be categorized into basic elements of the structure. This is configured with *KeyMCertificateElement/KeyMCertificateElementOfStructure*.

The following tables give correspondences of the enum values of *KeyMCertificateElementOfStructure* to the naming convention of the respective specifications.

8.1.2.1 X509 and CRL formats

Corresponding items of X.509 certificate elements from RFC5280 with element item configuration of *KeyMCertificateElement/KeyMCertificateElementOfStructure*

RFC 5280	KeyM Configuration of KeyMCertificateElement/ KeyMCertificateElementOfStructure
version	CertificateVersionNumber
serialNumber	CertificateSerialNumber
signature	CertificateSignatureAlgorithmID
issuer	CertificateIssuerName

notBeforeTime	CertificateValidityPeriodNotBefore
notAfterTime	CertificateValidityPeriodNotAfter
subject	CertificateSubjectName
subject Public Key Algorithm	CertificateSubjectPublicKeyInfo_PublicKeyAlgorithm
subject Public Key	CertificateSubjectPublicKeyInfo_SubjectPublicKey
issuerUniqueID	CertificateIssuerUniqueIdentifier
subjectUniqueID	CertificateSubjectUniqueIdentifier
extensions	CertificateExtension
signatureAlgorithm	CertificateSignatureAlgorithm
signatureValue	CertificateSignature

*Corresponding items of CRL elements from RFC5280 with element item configuration of
KeyMCertificateElement/KeyMCertificateElementOfStructure*

RFC 5280	KeyM Configuration of KeyMCertificateElement/ KeyMCertificateElementOfStructure
version	CertificateVersionNumber
signature	CertificateSignatureAlgorithmID
issuer	CertificateIssuerName
thisUpdate	CertificateValidityPeriodNotBefore
nextUpdate	CertificateValidityPeriodNotAfter
revokedCertificates	CertificateRevokedCertificates
crlExtensions	CertificateExtension
signatureAlgorithm	CertificateSignatureAlgorithm
signatureValue	CertificateSignature

For X509 and CRL certificate, the 'extensions' and 'crlExtensions' elements may include many extensions, for each extension must have a corresponding KeyMCertificateElement which has KeyMCertificateElementOfStructure parameter is CertificateExtension. Similar to the 'issuer', 'subject', 'revokedCertificates' element.

- The 'signatureAlgorithm' must be algorithm in the below table.

Algorithm	Algorithm ID
sha1WithRSAEncryption	1.2.840.113549.1.1.5
sha224WithRSAEncryption	1.2.840.113549.1.1.14
sha256WithRSAEncryption	1.2.840.113549.1.1.11
sha384WithRSAEncryption	1.2.840.113549.1.1.12
sha512WithRSAEncryption	1.2.840.113549.1.1.13
ecdsa-with-sha1	1.2.840.10045.4.1
ecdsa-with-sha224	1.2.840.10045.4.3.1
ecdsa-with-sha256	1.2.840.10045.4.3.2
ecdsa-with-sha384	1.2.840.10045.4.3.3
ecdsa-with-sha512	1.2.840.10045.4.3.4
rsassa-pss	1.2.840.113549.1.1.10

- The subjectPublicKey algorithm must be algorithm in the below table:

Algorithm	Algorithm ID
ecPublicKey	1.2.840.10045.2.1
ecdh	1.3.132.1.12
rsaEncryption	1.2.840.113549.1.1.1
rsassa-pss	1.2.840.113549.1.1.10

- The curve type (elementofStruct PublicKeyAlgorithm) of EC certificate must be algorithm in the below table:

Algorithm	Algorithm ID
Secp521r1	1.3.132.0.35
Secp256r1	1.2.840.10045.3.1.7
Secp224r1	1.3.132.0.33
Secp160r1	1.3.132.0.8

8.1.2.2 CVC format

Corresponding items of CVC elements as defined in BSI - Technical Guideline TR- 03110 with element item

configuration of KeyMCertificateElement/KeyMCertificateElementOfStructure

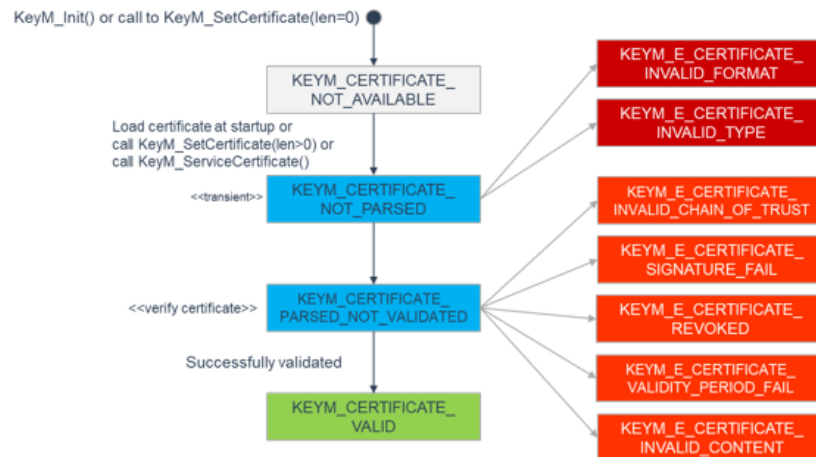
BSI - Technical Guideline TR-03110	KeyM Configuration of KeyMCertificateElement/ KeyMCertificateElementOfStructure
Certificate Profile Identifier	CertificateVersionNumber
Certificate Authority Reference	CertificateIssuerName
Public Key Object Identifier	CertificateSubjectPublicKeyInfo_PublicKeyAlgorithm
Public Key Domain Parameters	CertificateSubjectPublicKeyInfo_SubjectPublicKey
Certificate Holder Reference	CertificateSubjectName
Certificate Holder Authorization Template	CertificateSubjectAuthorization
Certificate Effective Date	CertificateValidityPeriodNotBefore
Certificate Expiration Date	CertificateValidityPeriodNotAfter
Certificate Extensions	CertificateExtension
Signature	CertificateSignature

For CVC certificate, the 'Public Key Domain Parameters' element may include many domain parameters, for each domain parameter must have a corresponding KeyMCertificateElement which has KeyMCertificateElementOfStructure parameter is CertificateSubjectPublicKeyInfo_SubjectPublicKey. Similar to 'Certificate Extensions' element. In addition, the 'Public Key Object Identifier' must be algorithm in the below table.

Algorithm	Algorithm ID
id-TA-RSA-v1-5-SHA-1	0.4.0.127.0.7.2.2.2.1.1
id-TA-RSA-v1-5-SHA-256	0.4.0.127.0.7.2.2.2.1.2
id-TA-RSA-PSS-SHA-1	0.4.0.127.0.7.2.2.2.1.3
id-TA-RSA-PSS-SHA-256	0.4.0.127.0.7.2.2.2.1.4
id-TA-RSA-v1-5-SHA-512	0.4.0.127.0.7.2.2.2.1.5
id-TA-RSA-PSS-SHA-512	0.4.0.127.0.7.2.2.2.1.6
id-TA-ECDSA-SHA-1	0.4.0.127.0.7.2.2.2.2.1
id-TA-ECDSA-SHA-224	0.4.0.127.0.7.2.2.2.2.2
id-TA-ECDSA-SHA-256	0.4.0.127.0.7.2.2.2.2.3
id-TA-ECDSA-SHA-384	0.4.0.127.0.7.2.2.2.2.4
id-TA-ECDSA-SHA-512	0.4.0.127.0.7.2.2.2.2.5

8.1.3 Certificate status transitions

A certificate has 11 status as defined in the KeyM_CertificateStatusType, each status will be entered by a defined state transition that is outlined in the below figure.



8.1.4 Certificate Parsing

After certificate data is set by KeyM_SetCertificate/KeyM_ServiceCertificate or load in the initialization phase, the certificate status becomes KEYM_CERTIFICATE_NOT_PARSED and certificate data shall be parsed in the MainBackground function. When parsing process has finished, if there is no error the certificate status shall be PARSED_NOT_VALIDATED, otherwise it can be INVALID_TYPE or INVALID_FORMAT.

- Certificate status becomes INVALID_FORMAT, if certificate parse operation detects violations to encoding rules such as ASN.1 or TLV (Tag Length Values) or basic elements of certificate are missing, please see the specification of each format certificate to get the detail. Note that, for extensions element in X509 and CRL:
 - The Subject Alternative Name extension only support *dnsName*.
 - The CRL Distribution Points and CRL Issuing Distribution Point extension only support *uniformResourceIdentifier* of *fullName* of *DistributionPointName*.
 - The Authority Key Identifier supports *keyIdentifier*, *authorityCertIssuer* and *authorityCertSerialNumber*.
 - Subject Key Identifier extension only support *keyIdentifier*.
 - For X509 Un-known extension type, it does not parse sub-element if any.
- Certificate status becomes INVALID_TYPE, if there is mismatch between certificate element configuration and certificate element data, for example:
 - The certificate data has version element data but there is not KeyMCertificateElement configuration of *CertificateVersionNumber* element and vice versa. It is similar to other elements such as *CertificateSerialNumber*, *CertificateSignature*, *CertificateSignatureAlgorithm*, *CertificateSignatureAlgorithmID*...
 - The certificate data has 3 extension elements data but there are not 3 KeyMCertificateElement configuration of *CertificateExtension* element. It is similar to other elements such as *CertificateIssuerName*, *CertificateSubjectName*, *CertificateRevokedCertificates*, *CertificateSubjectPublicKeyInfo*, *SubjectPublicKey*.
- Certificate status becomes INVALID_TYPE, if in the certificate data:

- Signature algorithm is not supported.
- The *signature* element and *signatureAlgorithm* element are not the same.
- The signature algorithm is not RSA-PSS and has *parameters* field is not absent or NULL.
- The public key algorithm is RSA and has *parameters* field is not absent or NULL.
- The public key algorithm is ECC and has *parameters* field is not absent or OID(Object Identifier).
- The validity date is not correct format or the year field of date is not in correct range. For example, in X509 or CRL, the date must has YYMMDDHHMMSSZ or YYYYMMDDHHMMSSZ and the year must not smaller than 1970. In CVC, th date has YYDDMM format and the year is from 2000 to 2099.

8.1.5 Certificate Verification Steps

- 1) The verification of a certificate starts with a check if all certificate that are linked with *KeyMCertUpperHierarchicalCertRef* are in the status KEYM_CERTIFICATE_VALID. If either of these referenced certificates are in the status KEYM_CERTIFICATE_NOT_PARSED or KEYM_CERTIFICATE_PARSED_NOT_VALIDATED, the parse and/respectively verification process shall be started for these linked certificates in the order from top (the last one of the linked list that either has no further link or links to itself) down to the bottom (the next certificate that is directly linked with *KeyMCertUpperHierarchicalCertRef* of the currently processed certificate). In this case, the status check of the linked certificates shall be re-done after all initiated certificate verifications have reached a final status.
- 2) If all certificates that are linked with *KeyMCertUpperHierarchicalCertRef* are in the status KEYM_CERTIFICATE_VALID, or *KeyMCertUpperHierarchicalCertRef* links to itself (self-signed certificates), the subject field of the currently validated certificate shall be verified with the issuer field of the next upper certificate (the one referenced by *KeyMCertUpperHierarchicalCertRef*). If one of the checks above have failed, the status KEYM_E_CERTIFICATE_INVALID_CHAIN_OF_TRUST shall be set and reported through callback function (if configured) and the validation process shall stop. Otherwise, the next step shall be performed
- 3) The signature of the certificate shall be verified by using the CSM verify job referenced with *KeyMCertUpperHierarchicalCertRef/ KeyMCertCsmSignatureVerifyJobRef*. If the signature verification fails, the certificate status KEYM_E_CERTIFICATE_SIGNATURE_FAIL shall be set and reported through callback function (if configured) and the validation process shall stop. Otherwise, the next step shall be performed.
- 4) If the certificate format type contains a time period, the KeyM module shall query the current time from configured time source (*KeyMCertTimebaseRef*) or by calling *UserGetCurrentTimeFunction* and then compare the current time if it is within the validity period of the certificate. If not, the certificate status KEYM_E_CERTIFICATE_VALIDITY_PERIOD_FAIL shall be set and reported through callback function (if configured) and the validation process shall stop. Otherwise, the next step shall be performed.
- 5) If the KeyM module maintains revocation lists, it shall check if the certificate under validation is part of the

revoked one. If so, the certificate status KEYM_E_CERTIFICATE_REVOKED shall be set and reported through callback function (if configured) and the validation process shall stop. Otherwise, the check next step shall be performed.

- 6) The contents of certificate elements shall be checked through by a check of all KeyMCertCertificateElementRuleRef. If one of the rules are violated, the certificate status KEYM_E_CERTIFICATE_INVALID_CONTENT shall be set and reported through callback function (if configured) and the validation process shall stop. Otherwise, the next step shall be performed.
- 7) If all verification steps have been performed and no error was detected during the verification, the public key of the certificate shall be set and validated (Csm_KeySetValid()) to the CSM key referenced by KeyMCertCsmSignatureVerifyKeyRef. The certificate status KEYM_CERTIFICATE_VALID shall be set and reported to the application if required by configured callback.

Note that:

- If calling to API of under modules(Csm, StBM) return NOT_OK, the verification process shall finished and the certificate status become NOT_VALID.
- At the step 4, SecOC module query the current time by calling StbM_GetCurrentTime function(based on KeyMTimeBaseRef) or by calling UserGetCurrentTimeFunction(based on KeyMUserGetCurrentTimeFunc). When UserGetCurrentTimeFunction is used, user must implement this function according to the below prototype:

Service name:	< UserGetCurrentTimeFunction>	
Syntax:	Std_ReturnType < UserGetCurrentTimeFunction>(uint64* currentTimePtr)	
Service ID[hex]:	NA	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	currentTimePtr	Pointer to the version information of this module.
Return value:	Std_ReturnType	E_OK: Operation was successfully performed. Current time are available. E_NOT_OK: Operation not accepted due to an internal error.
Description:	This function provides the current time. The current time is UTC time in second unit, E.g: 00:00:00 10/10/2020 UTC is converted to 0x5F80F980 seconds. Note that this function name is configuration	
Available via:	NA	

For example:

```

App_KeyM_User_Callout.c  App_KeyM_User_Callout.h  KeyM
31  **                               Global Data                               **
32  ****                               ****
33
34  /*****                               ****
35  **                               Function Prototypes                               **
36  ****                               ****
37  #define KEYM_START_SEC_CODE
38  #include "MemMap.h"
39
40  extern FUNC(Std_ReturnType, KEYM_USER_CODE) KeyM_User_GetCurrentTime(uint64 * currentTimePtr);
41
42  extern FUNC(void, KEYM_USER_CODE) KeyM_User_ConfigCurrentTime(uint64 curTime);
43
44  /*****                               ****
45  **                               Macros                               **
46  ****                               ****
47
48  #define KEYM_STOP_SEC_CODE
49  #include "MemMap.h"
50
51  #endif // APP_KEYM_USER_CALLOUT_H
52

```

AUTOSAR Explorer ARPackage Explorer

Build
Configuration
CryptoLib
Debug
Generated
References
Static_Code
 b_autosar_swc_IntegrationTest
 b_autosar_swc_QualityTest
 Integration_Code
 Modules
 Reference_Code
 inc
 App_KeyM_User_Callout.h
 src
 App_KeyM_User_Callout.c
 stub
 VT

```

KeyM  App_KeyM_User_Callout.c
1  #include "App_KeyM_User_Callout.h"
2
3  uint64 Global_Time = 1635774986;
4
5  Std_ReturnType KeyM_User_GetCurrentTime(uint64 * currentTimePtr)
6  {
7      *currentTimePtr = Global_Time;
8
9      return E_OK;
10 }
11
12 void KeyM_User_ConfigCurrentTime(uint64 curTime)
13 {
14     Global_Time = curTime;
15 }
16

```

App_KeyM_User_Callout.c App_KeyM_User_Callout.h KeyM

All Contents

Path: KeyM [KeyM] > KeyMGeneral [KeyMGeneral] > KeyMUserIncludeFiles [KeyMUserIncludeFiles]

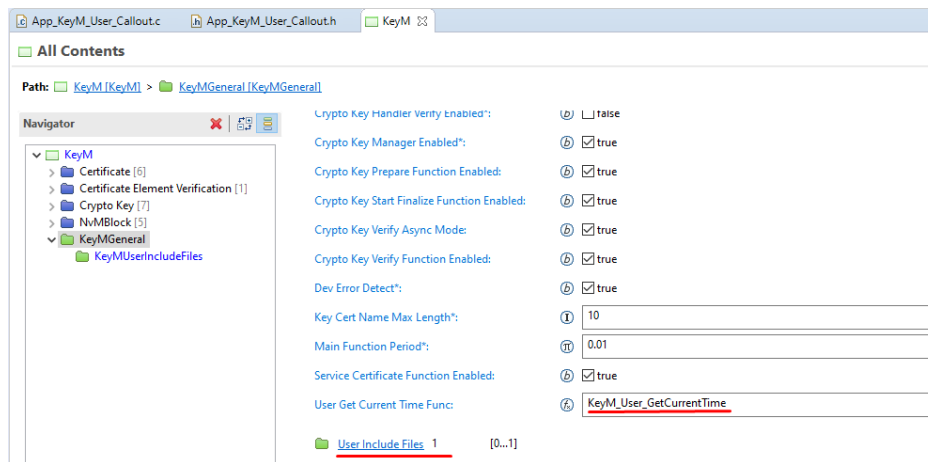
Navigator

- KeyM
 - Certificate [6]
 - Certificate Element Verification [1]
 - Crypto Key [7]
 - NvMBlock [5]
 - KeyMGeneral
 - KeyMUserIncludeFiles

Container Details - KeyMUserIncludeFiles

Short Name*: KeyMUserIncludeFiles

User Include Files: App_KeyM_User_Callout.h



8.2 Setting Guide by Function

8.2.1 KeyM_Init

- In the start up phase, KeyM_Init function shall be performed in the StartUpThree stage after executing NvM_ReadAll function in the StartUpTwo stage is finished. The NvM_ReadAll function reads all CryptoKey and Certificate data stored in NvM blocks to corresponding RAM slots in KeyM module.

8.2.2 KeyM_Update

- This function stores or derives key data to CryptoKeyElement regardless of KeyMCryptoKeyStorage parameter value. From user view there is no actual difference between KEYM_STORAGE_IN_RAM and KEYM_STORAGE_IN_CSM.
- The key data length must be smaller than or equal the value of CryptoKeyElementSize parameter of CryptoKeyElement. In the case of not equals, the CryptoKeyElementAllowPartialAccess parameter of CryptoKeyElement must be configured to TRUE to allow store key data to CryptoKeyElement.
- The KeyM module provide a uint8 array(RAM slot) for each KeyMCryptoKey, the length of array is value of KeyMCryptoKeyMaxLength parameter and the name of array is short name of KeyMCryptoKey configuration.
- If KeyMCryptoKeyStorage parameter is KEYM_STORAGE_IN_NVM, the value of NvMNvBlockLength of NvMBlock in the NvM module must equal value of KeyMCryptoKeyMaxLength parameter. And the input key data length must not be bigger than (KeyMCryptoKeyMaxLength - 3). In addition, the NvMRamBlockDataAddress parameter of NvMBlock must be configured to address of RAM slot of KeyMCryptoKey (E.g. &KeyMCryptoKey0[0]).
- If KeyMCryptoKeyStorage parameter is KEYM_STORAGE_IN_NVM, a key store job is considered as finished when the job of NvM write block has finished.

8.2.3 KeyM_Verify

- The function can be performed in synchronous or asynchronous mode by setting the KeyMCryptoKeyVerifyAsyncMode parameter. In the case asynchronous mode, if the parameter

KeyMCryptoKeyCsmVerifyJobRef is referenced to a asynchronous CsmJob, the parameter CsmJobPrimitiveCallbackRef of this CsmJob must be referenced to a CsmCallback container which has CsmCallbackFunc parameter is KeyM_PrvcsmCallback. The KeyM_PrvcsmCallback is KeyM call back function, this function is called by Csm to notify Csm verify job has finished.

8.2.4 KeyM_ServiceCertificate

- This function stores certificate data, the storage desination depends on KeyMCertificateStorage parameter value.
- If KeyMCertificateStorage parameter is KEYM_STORAGE_IN_CSM, the KeyMCertificateCsmKeyTargetRef parameter must be configured. The certificate data shall be stored to CryptoKeyElement Id 1 of CryptoKey which is referenced by the target CsmKey. The input certificate data length must be smaller than or equal value of CryptoKeyElementSize parameter of the CryptoKeyElement. In the case of not equals, the CryptoKeyElementAllowPartialAccess parameter of the CryptoKeyElement must be configured to TRUE to allow store certificate data to the CryptoKeyElement.
- The KeyM module provide a uint8 array(RAM slot) for each KeyMCertificate, the length of array is value of KeyMCertificateMaxLength parameter and the name of array is short name of KeyMCertificate configuration.
- If KeyMCryptoKeyStorage parameter is KEYM_STORAGE_IN_NVM, the value of NvMNvBlockLength of NvMBlock in the NvM module must equal value of KeyMCertificateMaxLength parameter. And the input certificate data length must not be bigger than (KeyMCertificateMaxLength - 3). In addition, the NvMRamBlockDataAddress parameter of NvMBlock must be configured to address of RAM slot of KeyMCertificate (E.g. &KeyMCertificate0[0]).
- For KEYM_SERVICE_CERT_REQUEST_CSR sub-service of this function, KeyM module shall based on KeyMcertificate configuration to generate certificate skeleton to provide to user and user shall modify the certificate skeleton to make actual certificate. The length of element of certificate skeleton is the value of KeyMCertificateElementMaxLength. In addition, for X509 and CVC format, in generating certificate skeleton processs, ECC keypair is generated and body of certificate skeleton is signed, thus KeyMCertPrivateKeyStorageCryptoKeyRef and KeyMCertCsmSignatureGenerateJobRef parameters must be configured correctly. For KeyMCertPrivateKeyStorageCryptoKeyRef, this parameter refers to a KeyMCryptoKey, the KeyMCryptoKey refers to CsmKey via CsmKeyTargetRef parameter, and the CsmKey refer to a CryptoKey. This CryptoKey must be configured correctly. For example:
 - If KeyMCertAlgorithmFormat is RSA, the CryptoKey must have atleast 2 key elements(Key Material and Algorithm) and the 'Key Material' key element must have CRYPTO_KE_FORMAT_BIN_RSA_PRIVATEKEY format.
 - If KeyMCertAlgorithmFormat is ECC, this CryptoKey and the CryptoKey which referred by KeyMCertCsmSignatureVerifyKeyRef must have atleast 2 key elements(Key Material and Algorithm) and the 'Key Material' key element must have CRYPTO_KE_FORMAT_BIN_OCTET format.
- For KEYM_SERVICE_CERT_UPDATE_ROOT sub-service, status of the certificate must be not be

NOT_PARSED, PARSED_NOT_VALIDATED, NOT_AVAILABLE, INVALID_TYPE, INVALID_FORMAT, otherwise operation result shall be KEYM_RT_KEY_CERT_INVALID. In addition, the new root and current root must have the same subject name and public key, otherwise operation result is KEYM_RT_KEY_CERT_UPDATE_FAIL.

8.2.5 KeyM_SetCertificate

- A certificate parsing and verification processes are performed in the main background, if these processes has not finished, call this function shall return E_NOT_OK. So user should not call this function when certificate status is KEYM_CERTIFICATE_NOT_PARSED or verify process has not finished.
- If KeyMCertificateStorage parameter is KEYM_STORAGE_IN_CSM or KEYM_STORAGE_IN_NVM, call this function shall return KEYM_E_PARAMETER_MISMATCH error and report KEYM_E_CONFIG_FAILURE to DET.

8.2.6 Verify Certificate functions

- In the certificate verification step 2, to verify the subject elements of the currently verified certificate, KeyM shall binary compare value of the subject elements with value of the issuer elements of the next upper certificate. Thus the encoding type of value of the subject elements must be the same encoding type of value of the issuer elements of the next upper certificate. Currently KeyM module supports printableString and utf8String encoding types only.
- In the certificate verification step 6, the contents of certificate elements shall be checked through by a check of all KeyMCertCertificateElementRuleRef. For a KeyMCertificateElementRule, If KeyMLogicalOperator parameter is KEYM_AND, logic expression of the rule is (*Rule = Condition_1 AND Condition_1 AND Condition_n*), thus the Rule is violated if there is at least one Condition is violated. If KeyMLogicalOperator is KEYM_OR, logic expression of the rule is (*Rule = Condition_1 OR Condition_1 OR Condition_n*), thus the Rule is violated if all Conditions are violated. For a Condition, value of the certificate element referenced by KeyMCertificateElementCondition/KeyMCertificateElementRef parameter shall be compared with ConditionValue. The KeyMCertElementConditionType parameter specifies how certificate element is compared, for example if parameter is KEYM_LESS_THAN and the value of certificate element not less than ConditionValue, the condition is violated. There are four ConditionValue types:
 - The KeyMCertificateElementConditionPrimitive is uint64 number, this should be used if the compared certificate element has only one element data(E.g. version, serialNumber, issuerUniqueID,...)
 - The KeyMCertificateElementConditionArray, each element of this array is a structure which contains 2 fields: elementIndex(uint16) and elementValue(uint64). The ConditionArray should be used if the compared certificate element has more than one element data(called sub-element) such as extensions, issuer, subject, revokedCertificate.... The elementIndex field specifies the index of sub-element in the compared certificate element and the elementValue specifies value compared with sub-element value.
 - The KeyMCertificateElementConditionCertificateElement, the value of compared certificate element

shall be compared with value of the elemen which is referenced by this.

- The KeyMCertificateElementConditionSenderReceiver, this parameter references a mode in a particular mode request port of a software component. This sender-receiver port must be R-Port and data type of the mode must be uint32 type. KeyM module shall call Rte_Read_Swc_Name_Port_Name_Data_Element_Name(uint32 * Data) function to get the mode value and then compare it with value of certificate element.
- In the certificate verification step 7, the CSM key referenced by KeyMCertCsmSignatureVerifyKeyRef must refer to a CryptoKey which has a key element ID 1, this key element must have correct format, for example:
 - Key element format must be CRYPTO_KE_FORMAT_BIN_RSA_PUBLICKEY when certificate public key algorithm is RSA.
 - Key element format must be CRYPTO_KE_FORMAT_BIN_OCTET when certificate public key algorithm is ECC.
- For KeyM_VerifyCertificateChain function, because there is no configuration for the intermediate certificates in the input list, so these intermediate certificates shall use the same configuration of the last certificate. Thus these intermediate certificates must have the same format and signature algorithm of the last certificate and format of the last certificate must not be CRL.
- If a CRL certificate has upper certificate is a CVC certificate and this CRL certificate is used for CVC certificate revocation, the issuer element of this CRL should have only one DN name and this DN name must have the same value of Certificate Holder Reference element of upper CVC certificate. In addition, the value of Certificate Holder Reference element of CVC certificate which shall be revoked must be included in the userCertificate of revokedCertificates of CRL.

8.2.7 Key Handler Functions

If key handler function is enabled, user must implement these functions by hand, for example if KeyMCryptoKeyHandlerUpdateEnabled is TRUE, user need to create implement of KeyM_KH_Update function, the below is an example of KeyM_KH_Update implementation used to update multiple keys.

```
key_update_handler.c x KeyM.h x KeyM_Externals.h x
#include "KeyM_Externals.h"

#define KEY_UPDATED_NUMBER 3u

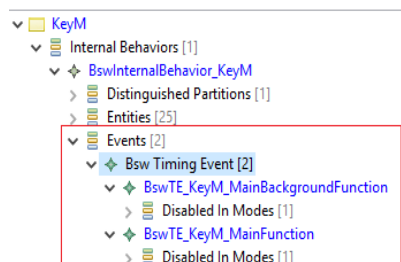
const uint8 key0_data[8u] = {0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01};
const uint8 key1_data[8u] = {0x02, 0x02, 0x02, 0x02, 0x02, 0x02, 0x02, 0x02};
const uint8 key2_data[8u] = {0x03, 0x03, 0x03, 0x03, 0x03, 0x03, 0x03, 0x03};
uint8 key_count = 0u;

FUNC(Std_ReturnType, KEYM_CODE) KeyM_KH_Update(
    P2CONST(uint8, AUTOMATIC, KEYM_APPL_CONST) KeyNamePtr,
    VAR(uint16, AUTOMATIC) KeyNameLength,
    P2CONST(uint8, AUTOMATIC, KEYM_APPL_CONST) RequestData,
    VAR(uint16, AUTOMATIC) RequestDataLength,
    P2VAR(uint8, AUTOMATIC, KEYM_APPL_DATA) ResultDataPtr,
    P2VAR(uint16, AUTOMATIC, KEYM_APPL_DATA) ResultDataLengthPtr,
    P2VAR(KeyM_CryptoKeyIdType, AUTOMATIC, KEYM_APPL_DATA) KeyId,
    P2VAR(KeyM_KH_UpdateOperationType, AUTOMATIC, KEYM_APPL_DATA) UpdateOperation)
{
    Std_ReturnType ret;
    uint16 index;
    const uint8 * keyDataPtr;
    uint16 keyDataLength;
    ret = E_OK;
    if (key_count == 0u)
    {
        keyDataPtr = &key0_data[0u];
        keyDataLength = 8u;
    }
    else if (key_count == 1u)
    {
        keyDataPtr = &key1_data[0u];
        keyDataLength = 8u;
    }
    else if (key_count == 2u)
    {
        keyDataPtr = &key2_data[0u];
        keyDataLength = 8u;
    }
    else
    {
        *UpdateOperation = KEYM_KH_UPDATE_FINISH;
        *ResultDataLengthPtr = 0u;
    }

    if (key_count < KEY_UPDATED_NUMBER)
    {
        key_count++;
        *UpdateOperation = KEYM_KH_UPDATE_KEY_UPDATE_REPEAT;
        /* Copy key data to the buffer of KeyM */
        for (index = 0u; index < keyDataLength; index++)
        {
            ResultDataPtr[index] = keyDataPtr[index];
        }
        *ResultDataLengthPtr = keyDataLength;
    }
}
```

8.3 Bswmd (Bsw Module Description)

8.3.1 Bsw module event setting



Filter: Search: 0 Loaded - 0 Shown - 0 Selected - [Custom: -- Table Default --]

Index	Short Name	Activation Rea...	Context Limita...	Starts On Event	Period
0	BswTE_KeyM_MainBackground...	◆	◆ BswDistingu...	◆ BswSE_KeyM_MainBackgroundFunction [/Bsw...	0.1
1	BswTE_KeyM_MainFunction	◆	◆ BswDistingu...	◆ BswSE_KeyM_MainFunction [/Bsw_KeyM/Key...	0.01

By default, there are two TimingEvents which are used to trigger OS tasks to perform KeyM_MainFunction and KeyM_MainbackgroundFuntion. Period of these TimingEvents is configured by Integrator.

In addition, for KeyM_MainbackgroundFuntion, user can use a BackgroundEvent instead of TimingEvent.

[illegible]