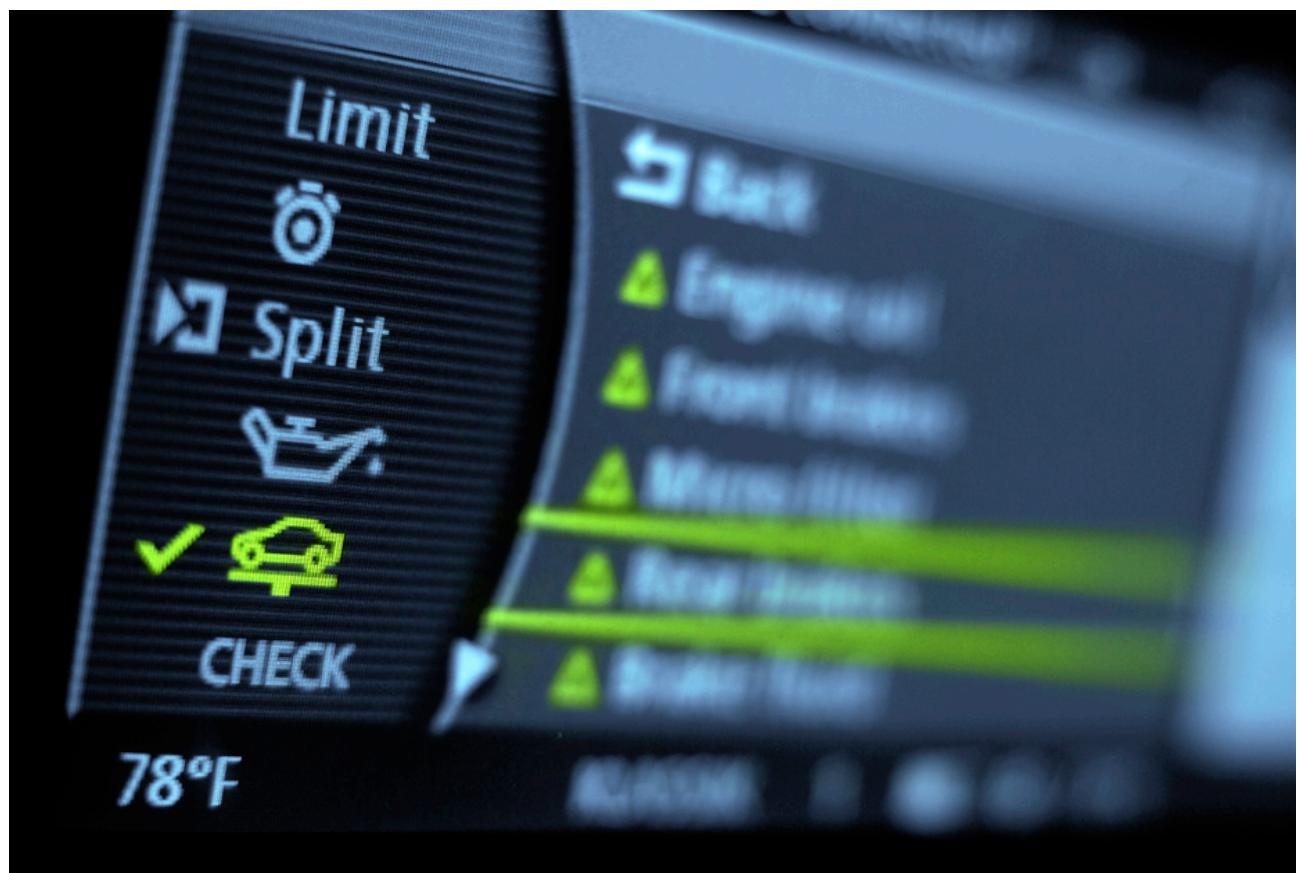




Elektrobit

EB tresos Bootloader for Essentials documentation





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

Technical support

<https://www.elektrobit.com/support>

Legal disclaimer

Confidential information.

ALL RIGHTS RESERVED. No part of this publication may be copied in any form, by photocopy, microfilm, retrieval system, or by any other means now known or hereafter invented without the prior written permission of Elektrobit Automotive GmbH.

All brand names, trademarks, and registered trademarks are property of their rightful owners and are used only for description.

Copyright 2021, Elektrobit Automotive GmbH.



Table of Contents

1. Overview of EB tresos Bootloader for Essentials documentation	36
2. BL for Essentials release notes	37
2.1. Overview	37
2.2. Scope of the release	37
2.2.1. Configuration tool	37
2.2.2. EB tresos Bootloader for Essentials modules	37
2.3. Module release notes	38
2.3.1. APP module release notes	38
2.3.1.1. Change log	38
2.3.1.2. New features	45
2.3.1.3. EB-specific enhancements	45
2.3.1.4. Deviations	46
2.3.1.5. Limitations	46
2.3.1.6. Open-source software	46
2.3.2. BLUpdater module release notes	46
2.3.2.1. Change log	46
2.3.2.2. New features	47
2.3.2.3. EB-specific enhancements	47
2.3.2.4. Deviations	48
2.3.2.5. Limitations	48
2.3.2.6. Open-source software	48
2.3.3. BM module release notes	48
2.3.3.1. Change log	48
2.3.3.2. New features	53
2.3.3.3. EB-specific enhancements	53
2.3.3.4. Deviations	53
2.3.3.5. Limitations	53
2.3.3.6. Open-source software	53
2.3.4. BIPduR module release notes	53
2.3.4.1. Change log	54
2.3.4.2. New features	59
2.3.4.3. EB-specific enhancements	59
2.3.4.4. Deviations	59
2.3.4.5. Limitations	59
2.3.4.6. Open-source software	59
2.3.5. Prog module release notes	59
2.3.5.1. Change log	60
2.3.5.2. New features	77
2.3.5.3. EB-specific enhancements	86



2.3.5.4. Deviations	87
2.3.5.5. Limitations	87
2.3.5.6. Open-source software	87
2.3.6. ProgOEMInd module release notes	87
2.3.6.1. Change log	87
2.3.6.2. New features	91
2.3.6.3. EB-specific enhancements	91
2.3.6.4. Deviations	91
2.3.6.5. Limitations	91
2.3.6.6. Open-source software	91
2.3.7. SA module release notes	92
2.3.7.1. Change log	92
2.3.7.2. New features	97
2.3.7.3. EB-specific enhancements	98
2.3.7.4. Deviations	98
2.3.7.5. Limitations	98
2.3.7.6. Open-source software	98
2.3.8. Uds module release notes	98
2.3.8.1. Change log	99
2.3.8.2. New features	105
2.3.8.3. EB-specific enhancements	106
2.3.8.4. Deviations	106
2.3.8.5. Limitations	106
2.3.8.6. Open-source software	107
3. BL for Essentials user guide	108
3.1. Overview	108
3.2. EB Tresos Bootloader user's guide for OEMInd	108
3.2.1. Introduction	108
3.2.2. Integration steps	108
3.2.3. General overview	110
3.2.3.1. Layer overview	110
3.2.3.2. plugin files structure	112
3.2.4. Response Pending scheduling	112
3.2.4.1. STM Timer ISR	112
3.2.5. Can Integration	113
3.2.5.1. CAN RX/TX interrupt priority levels	113
3.2.6. Crypto ASR v4.3 or higher version stack Integration	113
3.2.6.1. General Guidelines for the integration of Crypto ASR v4.3 stack	113
3.2.7. ReProgMemM Integration	113
3.2.7.1. Mandatory files for the integration	114
3.2.7.2. Constraints	114
3.2.7.3. Integration with asc_MemAcc module	114



3.2.8. Configuration	115
3.2.8.1. Notes	115
3.2.8.2. Prog Configuration	115
3.2.8.2.1. General	115
3.2.8.2.1.1. EraseState Callbacks	116
3.2.8.2.1.1.1. PROG_CustomIsFirstProgramming Callback	116
3.2.8.2.1.1.2. PROG_CustomDownloadNotification Callback	116
3.2.8.2.2. Crypto ASR 4.3 Stack generic configuration	117
3.2.8.2.3. Download data Signature verification	119
3.2.8.2.4. Download data Hash verification	120
3.2.8.2.5. Secure Checksum Computation	121
3.2.8.2.6. Download data Decryption	121
3.2.8.2.7. Memory	122
3.2.8.2.7.1. Prog memory configuration	122
3.2.8.2.8. Block and Segments	123
3.2.8.2.8.1. Addressing Modes in request download	126
3.2.8.2.9. Oem independent specific parameters	127
3.2.8.2.10. Download verification	128
3.2.8.2.11. Download Flash Routines	129
3.2.8.3. BM Configuration	129
3.2.8.4. BLUpdater Configuration	130
3.2.8.4.1. General	130
3.2.8.4.2. Bootloader Configuration	130
3.2.8.4.3. Memory	130
3.2.8.5. SA Configuration	130
3.2.8.6. UDS Configuration	131
3.2.8.6.1. General	131
3.2.8.6.2. Session	133
3.2.8.6.3. Service	133
3.2.8.6.4. Service_DID	133
3.2.8.6.5. Routines	133
3.2.8.7. BIPduR Configuration	133
3.2.8.7.1. Connection configuration	134
3.2.8.7.2. Multiple Receive Buffer Configuration	135
3.2.8.7.2.1. Behavior on reception of a Transfer Data request	135
3.2.8.7.3. Multiple Identifier Configuration	136
3.2.8.7.3.1. External Notification Configuration	136
3.2.8.7.3.2. Can Notification Configuration	137
3.2.8.8. Crypto Configuration	137
3.2.8.9. Ethernet Configuration	139
3.2.8.9.1. MemMap related	139
3.2.8.10. ReProgMemM Configuration	140



3.2.8.10.1. ReProgMemM configuration	140
3.2.8.10.1.1. Configuration	140
3.2.9. Callbacks	141
3.2.9.1. Programming sequences	142
3.2.9.2. Bootloader diagnostic Callbacks	145
3.2.9.2.1. UDS callbacks	145
3.2.9.2.2. Indication callbacks	146
3.2.9.3. Hardware Related Callbacks	147
3.2.9.4. BM Callbacks	148
3.2.9.4.1. BM_CustomHsmVerifyMac Callback	149
3.2.9.4.2. BM_CustomGetMacKey Callback	150
3.2.9.5. IBM Callbacks	150
3.2.9.5.1. BM_CheckProgRequest Callback	150
3.2.9.5.2. BM_CustomCheckValidAppl Callback	150
3.2.9.5.3. BM_CustomCheckValidBLU Callback	151
3.2.9.5.4. BM_CustomCheckValidBL Callback	151
3.2.9.5.5. BM_JumpToApplication Callback	151
3.2.9.5.6. BM_JumpToBLU Callback	152
3.2.9.5.7. BM_JumpToBL Callback	152
3.2.9.6. BM_CustomDualBankInit Callback	152
3.2.9.7. Startup callback	153
3.2.9.7.1. BM_CustomIsNormalStartup Callback	153
3.2.9.8. SA Callbacks	153
3.2.9.8.1. SA_CustomCalculateKey Callback	153
3.2.9.9. SA Antiscanning Callbacks	154
3.2.9.9.1. SA_CustomStoreAsRetryCnt Callback	154
3.2.9.9.2. SA_CustomRestoreAsRetryCnt Callback	154
3.2.9.10. Programming related callbacks	154
3.2.9.10.1. PROG_CheckProgRequest Callback	155
3.2.9.10.2. PROG_JumpToApplication Callback	155
3.2.9.10.3. PROG_isValidApplication Callback	155
3.2.9.10.4. PROG_InvalidateSection_BlockID Callback	156
3.2.9.10.5. PROG_SwitchApplicationModelInd Callback	156
3.2.9.10.6. PROG_GetSuppressBitFromAppli Callback	157
3.2.9.10.7. Fingerprint Callbacks	157
3.2.9.10.7.1. PROG_CustomWriteFingerprint Callback	157
3.2.9.10.7.2. PROG_CustomGetWriteFingerprintStatus Callback	158
3.2.9.10.8. Get Crypto Keys Callbacks	158
3.2.9.10.8.1. PROG_CustomGetAsymPublicKey Callback	158
3.2.9.10.8.2. PROG_CustomGetSymStaticKey Callback	159
3.2.9.10.9. PROG_CustomDecryptData Callback	159
3.2.9.10.10. MemoryAccessNotification Callback	159



3.2.9.10.10.1. PROG_CustomMemoryAccessNotification Callback	159
3.2.9.10.11. Custom Memory Access Callbacks	160
3.2.9.10.11.1. PROG_CustomMemoryErase Callback	160
3.2.9.10.11.2. PROG_CustomMemoryWrite Callback	160
3.2.9.10.11.3. PROG_CustomMemoryRead Callback	161
3.2.9.10.11.4. PROG_CustomMemGetJobStatus Callback	161
3.2.9.10.11.5. PROG_CustomGetNextSectorAddr Callback	162
3.2.9.10.12. Dual Memory Bank Callbacks	162
3.2.9.10.12.1. PROG_CustomCalcInactiveBankWriteAddr Callback	162
3.2.9.10.12.2. PROG_CustomCalcInactiveBankReadAddr Callback	162
3.2.9.10.12.3. PROG_CustomBankSwap Callback	163
3.2.9.10.12.4. PROG_CustomBankConfirmRollback Callback	163
3.2.9.10.13. Programming Counter Callbacks	163
3.2.9.10.13.1. PROG_CustomGetProgCounter Callback	164
3.2.9.10.13.2. PROG_CustomIncrementProgCounter Callback	164
3.2.9.10.14. Bootloader Updater Download Callbacks	165
3.2.9.10.14.1. PROG_CustomSetBLUDownloadInProgress Callback	165
3.2.9.10.14.2. PROG_CustomIsBLUDownloadInProgress Callback	165
3.2.9.10.14.3. PROG_CustomIsBLUPatternPresent Callback	165
3.2.9.10.14.4. PROG_CustomSetBLUVerificationSuccess Callback	166
3.2.9.10.15. Symetric Decryption Callbacks	167
3.2.9.10.15.1. PROG_CustomDecryptGetInitVector Callback	167
3.2.9.10.15.2. PROG_CustomGetSymDecryptionKey Callback	167
3.2.9.10.16. PROG_CustomCsmStrtPreproc Callback	168
3.2.9.11. BIPduR related callbacks	168
3.2.9.11.1. BIPduR_GetRxPduld Callback	168
3.2.9.11.2. BIPduR_StoreRxPduld Callback	169
3.2.9.11.3. BIPduR_Custom_Com_Init Callback	169
3.2.9.11.4. BIPduR_Custom_Com_Deactivate Callback	170
3.2.9.12. BLUpdater related callbacks	170
3.2.9.12.1. Bootloader Updater Callbacks	170
3.2.9.12.1.1. BLU_CustomSetValidityNewBootloader Callback	170
3.2.9.12.1.2. BLU_CustomSetValidityBLUpdater Callback	171
3.2.9.12.1.3. BLU_CustomGetPdulD Callback	171
3.2.9.12.1.4. BLU_CustomTriggerWatchdog Callback	171
3.2.9.12.1.5. BLU_Custom_Com_Init Callback	172
3.2.9.12.2. Bootloader Updater with Secure Boot Callbacks	172
3.2.9.12.2.1. BLU_CustomBootChecksumUpdate Callback	172
3.2.9.13. Communication related callbacks	173
3.2.9.13.1. BIPduR_GetGroupIdVal Callback	173
3.2.9.14. State machine guard callbacks	173
3.2.9.15. Response callbacks	173



3.2.9.16. ReProgMemM callbacks	174
3.2.9.16.1. ReProgMemM callbacks	174
3.2.9.16.1.1. ReProgMemM_FlashInit Callback	174
3.2.9.16.1.2. ReProgMemM_DualBank_Init Callback	174
3.2.9.16.1.3. ReProgMemM_CustomGetActiveBank Callback	174
3.2.9.16.1.4. ReProgMemM_CustomGetInactiveBankAddrOffset Callback	175
3.2.9.16.1.5. ReProgMemM_FlsDriver_JobStatus Callback	175
3.2.9.16.1.6. ReProgMemM_FlsExtDriver_JobStatus Callback	175
3.2.9.16.1.7. ReProgMemM_FlsDriver_Erase Callback	176
3.2.9.16.1.8. ReProgMemM_FlsExtDriver_Erase Callback	176
3.2.9.16.1.9. ReProgMemM_FlsDriver_Write Callback	176
3.2.9.16.1.10. ReProgMemM_FlsExtDriver_Write Callback	177
3.2.9.16.1.11. ReProgMemM_FlsDriver_Read Callback	177
3.2.9.16.1.12. ReProgMemM_FlsExtDriver_Read Callback	177
3.2.9.16.1.13. ReProgMemM_CustomGetOffset Callback	178
3.2.9.16.1.14. ReProgMemM_CustomGetPhysicalToLogicalAddress Call- back	178
3.2.10. Downloadable Flash driver Feature	178
3.2.10.1. Activation of the feature	179
3.2.10.2. Configuration of the feature	179
3.2.10.3. Integration of the feature	181
3.2.10.3.1. Linker file update	181
3.2.10.3.2. Available APIs	181
3.2.11. Compressed Flash driver Feature	181
3.2.11.1. Activation of the feature	182
3.2.11.2. Integration of the feature	182
3.2.11.2.1. Linker file update	182
3.2.12. Dual Memory Bank Feature	184
3.2.13. Secure Boot Feature	186
3.2.13.1. Activation of the feature	186
3.2.13.2. Secure Boot Checksum Calculation	186
3.2.13.3. Choice of the Checksum algorithm	187
3.2.13.4. Checksum computation without Secure Boot activated	187
3.2.13.5. Bootloader Checksum Verification	188
3.2.13.6. Secure Boot Information by Block	188
3.2.14. Secure Boot with HSM feature	189
3.2.14.1. Activation of the feature	190
3.2.14.2. Secure Boot Checksum Calculation	190
3.2.14.3. Choice of the Checksum algorithm	192
3.2.14.4. Bootloader Checksum Verification	192
3.2.14.5. Application Software Memory Block Information	193
3.2.15. Bootloader Updater Feature	194



3.2.15.1. Bootloader update sequence	194
3.2.15.2. Bootloader Updater process	195
3.2.15.3. Activation of the feature	197
3.2.15.4. Configuration of the feature	198
3.2.15.5. Integration of the feature	199
3.2.15.5.1. Initial Boot Manager Software	199
3.2.15.5.2. Bootloader Updater Software	199
3.2.15.5.3. Bootloader Software	199
3.2.15.6. Bootloader Updater Mandatory API to be called by the system	200
3.2.15.6.1. Initial Boot Manager Mandatory API	200
3.2.15.6.2. Bootloader Updater Mandatory API	200
3.2.16. Bootloader Updater Feature with Secure Boot	201
3.2.16.1. Activation of the feature	201
3.2.16.2. Configuration of the feature	201
3.2.16.3. Integration of the feature	202
3.2.16.3.1. Bootloader Updater Software	202
3.2.17. Mandatory API to be called by the system	202
3.2.18. General Bootloader Performance	203
3.2.19. Exception handling	204
3.2.20. Decryption Feature	204
3.2.20.1. Activation of the feature	204
3.2.21. Glossary	205
3.3. Annex: list of requirements	207
4. BL for Essentials module references	253
4.1. Overview	253
4.2. APP	253
4.2.1. Configuration parameters	253
4.2.1.1. General	253
4.2.1.2. Bootloader_Options	254
4.2.1.3. CommonPublishedInformation	254
4.2.1.4. PublishedInformation	257
4.2.2. Application programming interface (API)	257
4.2.2.1. Functions	257
4.2.2.1.1. APP_CalculateCrc	257
4.2.2.1.2. APP_Init	258
4.2.2.1.3. APP_Manage	258
4.2.2.1.4. APP_ReadBootFlag	258
4.2.2.1.5. APP_ReprogReqManage	259
4.2.2.1.6. APP_WriteBootFlag	259
4.2.2.1.7. PROG_WriteBootFlag	259
4.2.2.3. Integration notes	260
4.2.2.3.1. Exclusive areas	260



4.2.3.2. Production errors	260
4.2.3.3. Memory mapping	260
4.2.3.4. Integration requirements	260
4.3. BLUpdater	260
4.3.1. Configuration parameters	260
4.3.1.1. General	261
4.3.1.2. Bootloader_Configuration	263
4.3.1.3. Current_Bootloader	263
4.3.1.4. New_Bootloader	264
4.3.1.5. Memory	265
4.3.1.6. Security	268
4.3.1.7. SecureBoot	268
4.3.1.8. PduConnection	269
4.3.1.9. RxPdu	271
4.3.1.10. CommonPublishedInformation	273
4.3.1.11. PublishedInformation	276
4.3.2. Application programming interface (API)	276
4.3.2.1. Type definitions	276
4.3.2.1.1. tBLUBoolean	276
4.3.2.1.2. tBLUOperation	276
4.3.2.1.3. tBLUStatus	277
4.3.2.1.4. tCfgSegmentType	277
4.3.2.2. Macro constants	277
4.3.2.2.1. BLU_COMPARE	277
4.3.2.2.2. BLU_ERASE	277
4.3.2.2.3. BLU_E_BUSY	277
4.3.2.2.4. BLU_E_NOT_OK	278
4.3.2.2.5. BLU_E_OK	278
4.3.2.2.6. BLU_FALSE	278
4.3.2.2.7. BLU_FINISH	278
4.3.2.2.8. BLU_SECUREBOOT_UPDATE	278
4.3.2.2.9. BLU_TRUE	278
4.3.2.2.10. BLU_WRITE	278
4.3.2.3. Functions	279
4.3.2.3.1. BLU_CustomBootChecksumUpdate	279
4.3.2.3.2. BLU_CustomGetPdulID	279
4.3.2.3.3. BLU_CustomSetValidityBLUpdater	279
4.3.2.3.4. BLU_CustomSetValidityNewBootloader	280
4.3.2.3.5. BLU_CustomTriggerWatchdog	280
4.3.2.3.6. BLU_Custom_Com_Init	280
4.3.2.3.7. BLU_Init	280
4.3.2.3.8. BLU_Manage	281



4.3.2.3.9. BLU_Start	281
4.3.2.3.10. BLUpdater_PduRIfRxIndication	281
4.3.2.3.11. BLUpdater_PduRIfTxConfirmation	281
4.3.3. Integration notes	281
4.3.3.1. Exclusive areas	281
4.3.3.2. Production errors	282
4.3.3.3. Memory mapping	282
4.3.3.4. Integration requirements	282
4.4. BM	282
4.4.1. Configuration parameters	282
4.4.1.1. General	283
4.4.1.2. Security	286
4.4.1.3. BMCsmReferences	286
4.4.1.4. SecureBoot	288
4.4.1.5. CommonPublishedInformation	290
4.4.1.6. PublishedInformation	293
4.4.2. Application programming interface (API)	294
4.4.2.1. Functions	294
4.4.2.1.1. BM_CheckProgRequest	294
4.4.2.1.2. BM_CustomAllowTestApplExe	294
4.4.2.1.3. BM_CustomBckOperation	295
4.4.2.1.4. BM_CustomCheckValidAppl	295
4.4.2.1.5. BM_CustomCheckValidBL	295
4.4.2.1.6. BM_CustomCheckValidBLU	296
4.4.2.1.7. BM_CustomDualBankInit	296
4.4.2.1.8. BM_CustomGetComputedApplicationChecksum	296
4.4.2.1.9. BM_CustomGetExpectedApplicationChecksum	297
4.4.2.1.10. BM_CustomGetMacKey	297
4.4.2.1.11. BM_CustomHsmVerifyMac	297
4.4.2.1.12. BM_CustomIsNormalStartup	297
4.4.2.1.13. BM_CustomSetInvalidAppl	298
4.4.2.1.14. BM_CustomSetInvalidBlu	298
4.4.2.1.15. BM_CustomSetInvalidBoot	298
4.4.2.1.16. BM_DisableECCCheck	298
4.4.2.1.17. BM_EnableECCCheck	299
4.4.2.1.18. BM_GetTesterAddress	299
4.4.2.1.19. BM_HardwareInit	299
4.4.2.1.20. BM_InitialBootStartUp	299
4.4.2.1.21. BM_JumpToApplication	300
4.4.2.1.22. BM_JumpToBL	300
4.4.2.1.23. BM_JumpToBLU	300
4.4.2.1.24. BM_SoftwareInit	300



4.4.3. Integration notes	301
4.4.3.1. Exclusive areas	301
4.4.3.2. Production errors	301
4.4.3.3. Memory mapping	301
4.4.3.4. Integration requirements	301
4.5. BIPduR	301
4.5.1. Configuration parameters	302
4.5.1.1. General	302
4.5.1.2. SleepManagement	305
4.5.1.3. MultipleIdentifier	307
4.5.1.4. PduConnection	309
4.5.1.5. RxPdu	311
4.5.1.6. IDGroup	313
4.5.1.7. ConnectionReclist	314
4.5.1.8. CommonPublishedInformation	314
4.5.1.9. PublishedInformation	317
4.5.2. Application programming interface (API)	318
4.5.2.1. Functions	318
4.5.2.1.1. BIPduR_AllSlots	318
4.5.2.1.2. BIPduR_CopyRxData	318
4.5.2.1.3. BIPduR_CopyTxData	319
4.5.2.1.4. BIPduR_Custom_Com_Deactivate	320
4.5.2.1.5. BIPduR_Custom_Com_Init	320
4.5.2.1.6. BIPduR_GetGroupIdVal	320
4.5.2.1.7. BIPduR_GetNextBuffer	321
4.5.2.1.8. BIPduR_GetRxPduld	321
4.5.2.1.9. BIPduR_GetTpParameter	321
4.5.2.1.10. BIPduR_GroupIdFrameFilter	321
4.5.2.1.11. BIPduR_Init	322
4.5.2.1.12. BIPduR_Init1	322
4.5.2.1.13. BIPduR_Init2	322
4.5.2.1.14. BIPduR_IsNetworkSynchronized	322
4.5.2.1.15. BIPduR_IsTcpConnectionReestablished	323
4.5.2.1.16. BIPduR_LockBuffer	323
4.5.2.1.17. BIPduR_Manage	323
4.5.2.1.18. BIPduR_RetrieveConnectionInfo	323
4.5.2.1.19. BIPduR_RxIndication	324
4.5.2.1.20. BIPduR_SaveTesterAddress	324
4.5.2.1.21. BIPduR_SendMsgData	324
4.5.2.1.22. BIPduR_Send_TPFrame	324
4.5.2.1.23. BIPduR_SetTesterAddress	325
4.5.2.1.24. BIPduR_SimulateRxRequest	325



4.5.2.1.25. BIPduR_StartOfReception	325
4.5.2.1.26. BIPduR_StoreConnectionInfo	326
4.5.2.1.27. BIPduR_StoreRxPduld	326
4.5.2.1.28. BIPduR_TpChangeParameter	326
4.5.2.1.29. BIPduR_TpRxIndication	326
4.5.2.1.30. BIPduR_TpTxConfirmation	327
4.5.2.1.31. BIPduR_UnlockBuffer	327
4.5.2.1.32. LIN_ComLossInd	327
4.5.2.1.33. LIN_StatusInd	327
4.5.2.1.34. LIN_WakeUpInd	328
4.5.2.1.35. LTP_RxInd	328
4.5.2.1.36. LTP_TxConf	328
4.5.3. Integration notes	328
4.5.3.1. Exclusive areas	328
4.5.3.2. Production errors	328
4.5.3.3. Memory mapping	328
4.5.3.4. Integration requirements	329
4.6. Prog	329
4.6.1. Configuration parameters	329
4.6.1.1. CommonPublishedInformation	330
4.6.1.2. PublishedInformation	333
4.6.1.3. General	334
4.6.1.4. ProgCalReferences	346
4.6.1.5. DownloadVerification	347
4.6.1.6. DownloadFlashRoutines	351
4.6.1.7. SBLVerificationBlockTable	352
4.6.1.8. SBLVerificationStructure	353
4.6.1.9. Startup	353
4.6.1.10. CompleteAndCompatibleBlock	354
4.6.1.11. Segments	356
4.6.1.12. SecureBoot	361
4.6.1.13. Blocks	362
4.6.1.14. SecureBoot	364
4.6.1.15. Memory	366
4.6.1.16. GM	368
4.6.1.17. ProgCsmReferences	373
4.6.1.18. SignatureVerification	373
4.6.1.19. HashVerification	376
4.6.1.20. NRC78_Transmission	378
4.6.1.21. Security	379
4.6.1.22. ProgCsmReferences	382
4.6.1.23. SignatureVerification	383



4.6.1.24. HashVerification	386
4.6.1.25. OemInd	388
4.6.1.26. VAG	392
4.6.1.27. PSA	393
4.6.1.28. Decryption	394
4.6.2. Application programming interface (API)	397
4.6.2.1. Type definitions	397
4.6.2.1.1. ptAPPL_START_ADDR	397
4.6.2.1.2. ptCompleteCompatibleCallOut	397
4.6.2.1.3. ptSBL_StartUp_Code	397
4.6.2.1.4. tCryptoData	397
4.6.2.1.5. tDataBlockType	398
4.6.2.1.6. tDataBufferType	398
4.6.2.1.7. tDataLength	398
4.6.2.1.8. tMultipleBuffReprogInfo	398
4.6.2.1.9. tOperationType	398
4.6.2.1.10. tPageBuffer	399
4.6.2.1.11. tProgAccessType	399
4.6.2.1.12. tProgAddressType	399
4.6.2.1.13. tProgBoolean	399
4.6.2.1.14. tProgCompTimeoutStatus	399
4.6.2.1.15. tProgCompleteStatus	399
4.6.2.1.16. tProgCsmJobConf	400
4.6.2.1.17. tProgCsmNotification	400
4.6.2.1.18. tProgDownloadType	400
4.6.2.1.19. tProgEncryptOperation	400
4.6.2.1.20. tProgMemIdx	401
4.6.2.1.21. tProgMemMode	401
4.6.2.1.22. tProgMemType	401
4.6.2.1.23. tProgPECError	401
4.6.2.1.24. tProgPartitionType	401
4.6.2.1.25. tProgPsiValue	401
4.6.2.1.26. tProgRequestRoutineData	401
4.6.2.1.27. tProgRequestRoutineResult	402
4.6.2.1.28. tProgResCause	402
4.6.2.1.29. tProgRoutineResultInfo	402
4.6.2.1.30. tProgSigBypass	402
4.6.2.1.31. tProgStatus	402
4.6.2.1.32. tProgVerifAlgo	402
4.6.2.1.33. tProgVerificationInfo	403
4.6.2.1.34. tProgrammingFlags	403
4.6.2.1.35. tRDParam	403



4.6.2.1.36. tRegionType	404
4.6.2.1.37. tReprogInfo	404
4.6.2.1.38. tSegmentType	404
4.6.2.1.39. tSignatureHandlingStatus	405
4.6.2.1.40. tVDSSlayout	405
4.6.2.1.41. tWriteInfo	405
4.6.2.1.42. t_PROG_fctptr	406
4.6.2.1.43. t_secondary_bootloader_interface	406
4.6.2.1.44. tpulGetAddress	406
4.6.2.1.45. tpulVerifySectionCrc	406
4.6.2.1.46. tpulinvalidateSection	406
4.6.2.1.47. tpulisValidApplication	406
4.6.2.1.48. tpulskipPage	407
4.6.2.2. Macro constants	407
4.6.2.2.1. CRY_	407
4.6.2.2.2. PROG_1ST_PROGRAMMING_ERASE_CHECK	407
4.6.2.2.3. PROG_APPLICATION_PARTITION	407
4.6.2.2.4. PROG_AUTHENTICATED_REPROG	407
4.6.2.2.5. PROG_BLOCK_ERASE_CHECK	407
4.6.2.2.6. PROG_BLU_APP_PARTITION	408
4.6.2.2.7. PROG_BLU_CAL_PARTITION	408
4.6.2.2.8. PROG_BLU_PARTITION	408
4.6.2.2.9. PROG_BLU_PARTITION_MASK	408
4.6.2.2.10. PROG_BOOTLOADER_PARTITION	408
4.6.2.2.11. PROG_BOOT_DLS_SIZE	408
4.6.2.2.12. PROG_BOOT_MAX_PROT_PARTITIONS	408
4.6.2.2.13. PROG_BOOT_MODULE_ID_SIZE	409
4.6.2.2.14. PROG_BOOT_NB_MODULE_SIZE	409
4.6.2.2.15. PROG_BOOT_NUMBER_OF_MODULES	409
4.6.2.2.16. PROG_BOOT_PART_NUMBER_SIZE	409
4.6.2.2.17. PROG_BOOT_PRIMARY_MICRO_ID	409
4.6.2.2.18. PROG_BOOT_PROT_CALIB_NUMBER_SIZE	409
4.6.2.2.19. PROG_BOOT_PROT_CALIB_PARTITION_ID_SIZE	410
4.6.2.2.20. PROG_CALIBRATION_PARTITION	410
4.6.2.2.21. PROG_CIPHERED_MIN_DATA_SIZE	410
4.6.2.2.22. PROG_COMPLETECOMPATIBLE_END	410
4.6.2.2.23. PROG_COMPLETECOMPATIBLE_ERROR	410
4.6.2.2.24. PROG_COMPLETECOMPATIBLE_START	410
4.6.2.2.25. PROG_CRCDONE	410
4.6.2.2.26. PROG_CRYPTOFINISHDONE	411
4.6.2.2.27. PROG_CRYPTOSTARTUPDATEINPROGRESS	411
4.6.2.2.28. PROG_CRYPTOUPDATEDONE	411



4.6.2.2.29. PROG_DECRYPTION_IN_PROGRESS	411
4.6.2.2.30. PROG_DECRYPT_BUFFER_SIZE	411
4.6.2.2.31. PROG_DIGEST_LENGTH	411
4.6.2.2.32. PROG_DISABLED_ERASE_CHECK	412
4.6.2.2.33. PROG_DOWNLOAD_BY_ADDR	412
4.6.2.2.34. PROG_DOWNLOAD_BY_LOGICAL_BLOCK	412
4.6.2.2.35. PROG_DOWNLOAD_BY_LOGICAL_BLOCK_SEGMENT	412
4.6.2.2.36. PROG_ECU_ID_SIZE	412
4.6.2.2.37. PROG_ECU_NAME_SIZE	412
4.6.2.2.38. PROG_ERASEDONE	412
4.6.2.2.39. PROG_ERASESTART	413
4.6.2.2.40. PROG_ERR_APP_NBID	413
4.6.2.2.41. PROG_ERR_BCID	413
4.6.2.2.42. PROG_ERR_CAL_REGION	413
4.6.2.2.43. PROG_ERR_CCID	413
4.6.2.2.44. PROG_ERR_CERT	413
4.6.2.2.45. PROG_ERR_COMPRESSION	414
4.6.2.2.46. PROG_ERR_DATA_TYPE	414
4.6.2.2.47. PROG_ERR_ECU_ID	414
4.6.2.2.48. PROG_ERR_ECU_NAME	414
4.6.2.2.49. PROG_ERR_ERASE_CAL	414
4.6.2.2.50. PROG_ERR_ERASE_SW	414
4.6.2.2.51. PROG_ERR_FLASH_WRITE	414
4.6.2.2.52. PROG_ERR_GET_APP_INFO	415
4.6.2.2.53. PROG_ERR_GET_CAL_INFO	415
4.6.2.2.54. PROG_ERR_KEY_NBID	415
4.6.2.2.55. PROG_ERR_LENGTH_EXCEEDED	415
4.6.2.2.56. PROG_ERR_MD	415
4.6.2.2.57. PROG_ERR_MODULE_ID	415
4.6.2.2.58. PROG_ERR_MORE_DATA_EXPECTED	416
4.6.2.2.59. PROG_ERR_MSG_OUT_OF_SEQUENCE	416
4.6.2.2.60. PROG_ERR_PARTITION_ID	416
4.6.2.2.61. PROG_ERR_PER_DATA_TX_NOT_ALLOW	416
4.6.2.2.62. PROG_ERR_PROTECTEDCAL_NOT_DEFINED	416
4.6.2.2.63. PROG_ERR_REVOKE_CAL	416
4.6.2.2.64. PROG_ERR_REVOKE_SW	416
4.6.2.2.65. PROG_ERR_ROOT_SIGNATURE	417
4.6.2.2.66. PROG_ERR_SBA_CERT	417
4.6.2.2.67. PROG_ERR_SBA_ECU_ID	417
4.6.2.2.68. PROG_ERR_SBA_ECU_NAME	417
4.6.2.2.69. PROG_ERR_SBA_SIGNATURE	417
4.6.2.2.70. PROG_ERR_SIGNATURE	417



4.6.2.2.71. PROG_ERR SUBJECT_NAME	418
4.6.2.2.72. PROG_ERR SW NOT_PRESENT	418
4.6.2.2.73. PROG_ERR SW_REGION	418
4.6.2.2.74. PROG_ERR_UNDEFINED	418
4.6.2.2.75. PROG_ERR_UPDATE_PSI	418
4.6.2.2.76. PROG_ESS_PARTITION	418
4.6.2.2.77. PROG_E_BUSY	418
4.6.2.2.78. PROG_E_BYPASS	419
4.6.2.2.79. PROG_E_CHECK_FAILED	419
4.6.2.2.80. PROG_E_COHCHK_CORRECT	419
4.6.2.2.81. PROG_E_COHCHK_INCORRECT	419
4.6.2.2.82. PROG_E_COHCHK_INCORRECT_OTHER	419
4.6.2.2.83. PROG_E_COHCHK_INCORRECT_SW_HW	419
4.6.2.2.84. PROG_E_COHCHK_INCORRECT_SW_SW	420
4.6.2.2.85. PROG_E_COHPRECHK_CORRECT	420
4.6.2.2.86. PROG_E_COHPRECHK_INCORRECT_HW_SW	420
4.6.2.2.87. PROG_E_COHPRECHK_INCORRECT_SW_SW	420
4.6.2.2.88. PROG_E_COHPRECHK_INTERNAL_ERROR	420
4.6.2.2.89. PROG_E_ERASED	420
4.6.2.2.90. PROG_E_NOT_BUSY	420
4.6.2.2.91. PROG_E_NOT_ERASED	421
4.6.2.2.92. PROG_E_NOT_OK	421
4.6.2.2.93. PROG_E_OK	421
4.6.2.2.94. PROG_E_RFS_DRIVER_FAIL	421
4.6.2.2.95. PROG_E_RFS_VERSION_FAIL	421
4.6.2.2.96. PROG_E_TXCONF_OK	421
4.6.2.2.97. PROG_FALSE	422
4.6.2.2.98. PROG_FLASH_ROUTINES_PARTITION	422
4.6.2.2.99. PROG_HSM_PARTITION	422
4.6.2.2.100. PROG_LOGICAL_BLOCK_VALUE_INIT	422
4.6.2.2.101. PROG_MAX_LENGTH_CHECKMEMORY_ANSWER	422
4.6.2.2.102. PROG_MAX_PARTITION	422
4.6.2.2.103. PROG_MAX_RD_PER_BLOCK	422
4.6.2.2.104. PROG_MAX_REGION_ALLOWED	423
4.6.2.2.105. PROG_MEMORY_ASYNCNCHRONOUS	423
4.6.2.2.106. PROG_MEMORY_NB	423
4.6.2.2.107. PROG_MEMORY_NOTUSED	423
4.6.2.2.108. PROG_MEMORY_SYNCHRONOUS	423
4.6.2.2.109. PROG_MEM_ACCESS_TYPE_NONE	423
4.6.2.2.110. PROG_MEM_ACCESS_TYPE_READ	424
4.6.2.2.111. PROG_MEM_ACCESS_TYPE_READ_WRITE	424
4.6.2.2.112. PROG_MEM_ACCESS_TYPE_WRITE	424



4.6.2.2.113. PROG_MEM_OPERATION_TYPE_ERASE	424
4.6.2.2.114. PROG_MEM_OPERATION_TYPE_READ	424
4.6.2.2.115. PROG_MEM_OPERATION_TYPE_WRITE	424
4.6.2.2.116. PROG_MEM_TYPE_CUSTOM	424
4.6.2.2.117. PROG_MEM_TYPE_EEPROM	425
4.6.2.2.118. PROG_MEM_TYPE_FLASH	425
4.6.2.2.119. PROG_MEM_TYPE_FLASH_EXT	425
4.6.2.2.120. PROG_MEM_TYPE_INIT	425
4.6.2.2.121. PROG_MEM_TYPE_RAM	425
4.6.2.2.122. PROG_MEM_TYPE_SCRATCHPAD	425
4.6.2.2.123. PROG_MIN_VAL_TO_WRITE	426
4.6.2.2.124. PROG_NO_OPERATION_IN_PROGRESS	426
4.6.2.2.125. PROG_NO_REPROG_TYPE	426
4.6.2.2.126. PROG_PEC_NO_ERROR	426
4.6.2.2.127. PROG_PROT_CALIBRATION_PARTITION	426
4.6.2.2.128. PROG_PSI_INVALID	426
4.6.2.2.129. PROG_PSI_PROGRAMMED	426
4.6.2.2.130. PROG_PSI_REVOKED	427
4.6.2.2.131. PROG_RC_WITH_RI_LENGTH	427
4.6.2.2.132. PROG_REQUEST_ROUTINE_RESULT_FAIL	427
4.6.2.2.133. PROG_REQUEST_ROUTINE_RESULT_INIT	427
4.6.2.2.134. PROG_REQUEST_ROUTINE_RESULT_PENDING	427
4.6.2.2.135. PROG_REQUEST_ROUTINE_RESULT_SUCCESS	427
4.6.2.2.136. PROG_RESET_CAUSE_BLU	428
4.6.2.2.137. PROG_RESET_CAUSE_DSC01	428
4.6.2.2.138. PROG_RESET_CAUSE_DSC02	428
4.6.2.2.139. PROG_RESET_CAUSE_ER	428
4.6.2.2.140. PROG_RESET_CAUSE_S3_TIMEOUT	428
4.6.2.2.141. PROG_RESET_CAUSE_SBLACTIVEFAIL	428
4.6.2.2.142. PROG_SBA_OK	428
4.6.2.2.143. PROG_SBA_PARTITION	429
4.6.2.2.144. PROG_SEGMENT_NB	429
4.6.2.2.145. PROG_STANDARD_REPROG	429
4.6.2.2.146. PROG SUBJECT_NAME_SIZE	429
4.6.2.2.147. PROG_TRUE	429
4.6.2.2.148. PROG_VDS_SIZE	429
4.6.2.2.149. PROG_VERIFY_CRC	430
4.6.2.2.150. PROG_VERIFY_HASH	430
4.6.2.2.151. PROG_VERIFY_SIGNATURE	430
4.6.2.3. Objects	430
4.6.2.3.1. m_astProgCsmJobConf	430
4.6.2.3.2. m_aubBlockDownloadStatus	430



4.6.2.3.3. m_ubCheckMemoryStatus	430
4.6.2.3.4. m_ubFailedCheckMemoryCount	431
4.6.2.3.5. m_ubSimulateProgSessionWithResponse	431
4.6.2.3.6. m_ubValidateMemoryStatus	431
4.6.2.3.7. m_ulRsDelayTimer	431
4.6.2.3.8. tFirstCheckMemoryAnswerInfo	431
4.6.2.3.9. ubDiagStatus	431
4.6.2.3.10. ulLength	431
4.6.2.4. Functions	432
4.6.2.4.1. PROG_ACT_Check_Impl80	432
4.6.2.4.2. PROG_ActiveSBL	432
4.6.2.4.3. PROG_ActiveSBL_Check	432
4.6.2.4.4. PROG_AnswerSuccessiveCheckMemoryRequests	433
4.6.2.4.5. PROG_AutoControl	433
4.6.2.4.6. PROG_AutoControl_Process	433
4.6.2.4.7. PROG_BckdManage	433
4.6.2.4.8. PROG_CRC	433
4.6.2.4.9. PROG_CRC_Compare	434
4.6.2.4.10. PROG_CalcCrc16	434
4.6.2.4.11. PROG_CalcCrc32	434
4.6.2.4.12. PROG_CheckDecompHeaderStatus	434
4.6.2.4.13. PROG_CheckMemory	435
4.6.2.4.14. PROG_CheckPartialSegmentListCrc	435
4.6.2.4.15. PROG_CheckPartialSwCvnStatus	435
4.6.2.4.16. PROG_CheckProgRequest	435
4.6.2.4.17. PROG_CheckProgrammingCounter	436
4.6.2.4.18. PROG_CheckProgrammingDependencies	436
4.6.2.4.19. PROG_CheckProgrammingPreCondition	437
4.6.2.4.20. PROG_CheckProgrammingRequest	437
4.6.2.4.21. PROG_CheckSegmentListCrc	437
4.6.2.4.22. PROG_CheckValidAppl	437
4.6.2.4.23. PROG_CheckValidateApplicationFailed	437
4.6.2.4.24. PROG_CheckValidateApplicationFinish	437
4.6.2.4.25. PROG_Check_Prg_Dep_Check	438
4.6.2.4.26. PROG_CloseExtendedSession	438
4.6.2.4.27. PROG_CloseProgrammingSession	438
4.6.2.4.28. PROG_CommunicationControl	438
4.6.2.4.29. PROG_ComputeBlockHash	439
4.6.2.4.30. PROG_ComputeMessageDigest	439
4.6.2.4.31. PROG_ControlDTCSetting	439
4.6.2.4.32. PROG_CopySBATicket	440
4.6.2.4.33. PROG_CsmCancelActiveJobs	440



4.6.2.4.34. PROG_CsmCheckResult	440
4.6.2.4.35. PROG_CsmManage	441
4.6.2.4.36. PROG_CsmSetPreConditions	441
4.6.2.4.37. PROG_CustCheckProgPrecond	441
4.6.2.4.38. PROG_CustCheckProgPrecondList	441
4.6.2.4.39. PROG_CustomCalcInactiveBankReadAddr	442
4.6.2.4.40. PROG_CustomCalcInactiveBankWriteAddr	442
4.6.2.4.41. PROG_CustomCheckCertificateVerification	442
4.6.2.4.42. PROG_CustomCheckCompatibilityId	443
4.6.2.4.43. PROG_CustomCheckRollbackId	443
4.6.2.4.44. PROG_CustomCheckSigningInfo	443
4.6.2.4.45. PROG_CustomCheckTargetName	444
4.6.2.4.46. PROG_CustomCheckUuid	444
4.6.2.4.47. PROG_CustomCheckZIAvailableSpace	444
4.6.2.4.48. PROG_CustomChecksumCalc	445
4.6.2.4.49. PROG_CustomClearPartProgSegList	445
4.6.2.4.50. PROG_CustomCoherencyCheck	445
4.6.2.4.51. PROG_CustomCompatibilityCheck	446
4.6.2.4.52. PROG_CustomComputeCoherencyPreCheck	446
4.6.2.4.53. PROG_CustomCsmStrtPreproc	447
4.6.2.4.54. PROG_CustomCvnVerification	447
4.6.2.4.55. PROG_CustomCvnVerificationStatus	448
4.6.2.4.56. PROG_CustomDecryptData	448
4.6.2.4.57. PROG_CustomDecryptGetInitVector	449
4.6.2.4.58. PROG_CustomDownloadNotification	449
4.6.2.4.59. PROG_CustomGetAsymPublicKey	449
4.6.2.4.60. PROG_CustomGetComputedBootloaderChecksum	450
4.6.2.4.61. PROG_CustomGetDownloadedSegmentSize	450
4.6.2.4.62. PROG_CustomGetECUStatus	450
4.6.2.4.63. PROG_CustomGetEculd	451
4.6.2.4.64. PROG_CustomGetEraseStatus	451
4.6.2.4.65. PROG_CustomGetExpectedCrc	451
4.6.2.4.66. PROG_CustomGetMacKey	452
4.6.2.4.67. PROG_CustomGetNextSectorAddr	452
4.6.2.4.68. PROG_CustomGetPartProgSegList	452
4.6.2.4.69. PROG_CustomGetProgCounter	453
4.6.2.4.70. PROG_CustomGetResetCause	453
4.6.2.4.71. PROG_CustomGetResumeAddress	453
4.6.2.4.72. PROG_CustomGetSBLStartAddress	454
4.6.2.4.73. PROG_CustomGetSegmentList	454
4.6.2.4.74. PROG_CustomGetSymDecryptionKey	454
4.6.2.4.75. PROG_CustomGetTpBsValue	455



4.6.2.4.76. PROG_CustomGetTpStminValue	455
4.6.2.4.77. PROG_CustomGetVerificationParameters	455
4.6.2.4.78. PROG_CustomGetWriteJobResult	456
4.6.2.4.79. PROG_CustomHsmUpdateBlock	456
4.6.2.4.80. PROG_CustomHsmUpdateFinish	457
4.6.2.4.81. PROG_CustomHsmUpdateInitBlock	457
4.6.2.4.82. PROG_CustomIncrementProgCounter	458
4.6.2.4.83. PROG_CustomInvalidateBootloaderChecksum	458
4.6.2.4.84. PROG_CustomIsBLUDownloadInProgress	458
4.6.2.4.85. PROG_CustomIsBLUPatternPresent	459
4.6.2.4.86. PROG_CustomIsFirstProgramming	459
4.6.2.4.87. PROG_CustomIsValidBootloaderChecksum	459
4.6.2.4.88. PROG_CustomMemGetJobStatus	460
4.6.2.4.89. PROG_CustomMemoryAccessNotification	460
4.6.2.4.90. PROG_CustomMemoryErase	461
4.6.2.4.91. PROG_CustomMemoryRead	461
4.6.2.4.92. PROG_CustomMemoryWrite	462
4.6.2.4.93. PROG_CustomReadKeyAppli	462
4.6.2.4.94. PROG_CustomSetAppValidity	463
4.6.2.4.95. PROG_CustomSetApplicationChecksum	463
4.6.2.4.96. PROG_CustomSetBLUDownloadInProgress	463
4.6.2.4.97. PROG_CustomSetBLUVerificationSuccess	464
4.6.2.4.98. PROG_CustomSetBootloaderChecksum	464
4.6.2.4.99. PROG_CustomSetDownloadVerificationSuccess	464
4.6.2.4.100. PROG_CustomSetECUStatus	465
4.6.2.4.101. PROG_CustomSetEraseStatus	465
4.6.2.4.102. PROG_CustomSetOpenProgSession	466
4.6.2.4.103. PROG_CustomSetPartProgSegList	466
4.6.2.4.104. PROG_CustomStartChecksumCalc	466
4.6.2.4.105. PROG_CustomStoreDownloadedSegmentSize	467
4.6.2.4.106. PROG_CustomStoreResetCause	467
4.6.2.4.107. PROG_CustomStoreResumeAddress	467
4.6.2.4.108. PROG_CustomStoreSegmentList	468
4.6.2.4.109. PROG_CustomUpdateCertAsymPublicKey	468
4.6.2.4.110. PROG_CustomUpdateChecksumCalc	469
4.6.2.4.111. PROG_CustomUpdateLogSaveMarkingByte	470
4.6.2.4.112. PROG_CustomVDStable_update	470
4.6.2.4.113. PROG_CustomValidateBootloaderChecksum	471
4.6.2.4.114. PROG_CustomWriteCRC	471
4.6.2.4.115. PROG_CustomWriteKeyAppli	471
4.6.2.4.116. PROG_CustomWriteProgStatus	472
4.6.2.4.117. PROG_CustomWriteZI	472



4.6.2.4.118. PROG_DisableECCCheck	472
4.6.2.4.119. PROG_Do_CheckHash	473
4.6.2.4.120. PROG_Do_CheckPrgDependencies	473
4.6.2.4.121. PROG_Do_CheckSignature	473
4.6.2.4.122. PROG_Do_CheckValidateApplication	473
4.6.2.4.123. PROG_Do_CoherencyCheck	473
4.6.2.4.124. PROG_Do_CompareKey	473
4.6.2.4.125. PROG_Do_DecryptionFinish	474
4.6.2.4.126. PROG_Do_DecryptionUpdate	474
4.6.2.4.127. PROG_Do_GetSeed	474
4.6.2.4.128. PROG_Do_RTE_Impl80	474
4.6.2.4.129. PROG_Do_SelfCheck_Impl80	474
4.6.2.4.130. PROG_DrvDown_IsFlashRoutinesPresent	474
4.6.2.4.131. PROG_Dsc01Cbk	475
4.6.2.4.132. PROG_Dsc03Cbk	475
4.6.2.4.133. PROG_ERASE_Check	476
4.6.2.4.134. PROG_EcuReset	476
4.6.2.4.135. PROG_EnableECCCheck	476
4.6.2.4.136. PROG_Entry_ACT_Check_Impl80	476
4.6.2.4.137. PROG_Entry_ActiveSBL	477
4.6.2.4.138. PROG_Entry_Alive	477
4.6.2.4.139. PROG_Entry_AutoControl	477
4.6.2.4.140. PROG_Entry_BLU_Resume	477
4.6.2.4.141. PROG_Entry_CheckDependenciesFinish	477
4.6.2.4.142. PROG_Entry_CheckHash	477
4.6.2.4.143. PROG_Entry_CheckMemory	478
4.6.2.4.144. PROG_Entry_CheckMemoryCompute	478
4.6.2.4.145. PROG_Entry_CheckMemoryFinish	478
4.6.2.4.146. PROG_Entry_CheckMemoryFinish_Impl80	478
4.6.2.4.147. PROG_Entry_ChecksumByRange	478
4.6.2.4.148. PROG_Entry_CoherencyPreCheck	478
4.6.2.4.149. PROG_Entry_CompareKey	478
4.6.2.4.150. PROG_Entry_CompareKeyCheck	479
4.6.2.4.151. PROG_Entry_DecomPHeader	479
4.6.2.4.152. PROG_Entry_DecryptionFinish	479
4.6.2.4.153. PROG_Entry_DecryptionUpdate	479
4.6.2.4.154. PROG_Entry_DefaultSession	479
4.6.2.4.155. PROG_Entry_EcuReset	479
4.6.2.4.156. PROG_Entry_Erase	480
4.6.2.4.157. PROG_Entry_EraseCheck	480
4.6.2.4.158. PROG_Entry_EraseFinish	480
4.6.2.4.159. PROG_Entry_EraseNRC78	480



4.6.2.4.160. PROG_Entry_EraseTransmitNRC78	480
4.6.2.4.161. PROG_Entry_ExtendedSession	480
4.6.2.4.162. PROG_Entry_GetSeed	480
4.6.2.4.163. PROG_Entry_GetSeedCheck	481
4.6.2.4.164. PROG_Entry_HSMUpdate_TDFinish	481
4.6.2.4.165. PROG_Entry_INIT	481
4.6.2.4.166. PROG_Entry_LogicalBlockHash	481
4.6.2.4.167. PROG_Entry_PartialVerificationCrc	481
4.6.2.4.168. PROG_Entry_PreInit	481
4.6.2.4.169. PROG_Entry_ProgrammingSession	482
4.6.2.4.170. PROG_Entry_RD	482
4.6.2.4.171. PROG_Entry_RD_Finish	482
4.6.2.4.172. PROG_Entry_RD_Impl80	482
4.6.2.4.173. PROG_Entry_RD_Signature	482
4.6.2.4.174. PROG_Entry RTE	482
4.6.2.4.175. PROG_Entry_RTEFailed	482
4.6.2.4.176. PROG_Entry_RTEFinish	483
4.6.2.4.177. PROG_Entry RTE_Impl80	483
4.6.2.4.178. PROG_Entry_Reset	483
4.6.2.4.179. PROG_Entry_ResumeVerification	483
4.6.2.4.180. PROG_Entry_Resume_Finish	483
4.6.2.4.181. PROG_Entry_SblSynch	483
4.6.2.4.182. PROG_Entry_SecureChecksumFailed	484
4.6.2.4.183. PROG_Entry_SelfCheck_Impl80	484
4.6.2.4.184. PROG_Entry_SignatureCheck	484
4.6.2.4.185. PROG_Entry_SignatureVerify	484
4.6.2.4.186. PROG_Entry_Sleep	484
4.6.2.4.187. PROG_Entry_Streaming	484
4.6.2.4.188. PROG_Entry_TD	485
4.6.2.4.189. PROG_Entry_TD_Failed	485
4.6.2.4.190. PROG_Entry_TD_Header	485
4.6.2.4.191. PROG_Entry_TD_Impl80	485
4.6.2.4.192. PROG_Entry_UpdatePSI	485
4.6.2.4.193. PROG_Entry_ValidateSBASignature	485
4.6.2.4.194. PROG_Entry_ValidateSBASignerInfo	485
4.6.2.4.195. PROG_Entry_ValidateSignature	486
4.6.2.4.196. PROG_Entry_ValidateSignerInfo	486
4.6.2.4.197. PROG_Entry_Validate_Application	486
4.6.2.4.198. PROG_Entry_ValidationFailed	486
4.6.2.4.199. PROG_Entry_ValidationFinish	486
4.6.2.4.200. PROG_Entry_Write	487
4.6.2.4.201. PROG_Entry_WriteFingerprint	487



4.6.2.4.202. PROG_Entry_Write_Impl80	487
4.6.2.4.203. PROG_Erase	487
4.6.2.4.204. PROG_EraseFirstCheck_Impl80	487
4.6.2.4.205. PROG_EraseMemoryRequest	487
4.6.2.4.206. PROG_Erase_Guard	488
4.6.2.4.207. PROG_Exit_CheckMemory	488
4.6.2.4.208. PROG_Exit_CheckMemoryFinish	488
4.6.2.4.209. PROG_Exit_INIT	488
4.6.2.4.210. PROG_Exit_PartialVerificationCrc	489
4.6.2.4.211. PROG_Exit_RTE_Decrypt	489
4.6.2.4.212. PROG_Exit_TD_Write	489
4.6.2.4.213. PROG_FindBlockIndexInTable	489
4.6.2.4.214. PROG_FlashPage	489
4.6.2.4.215. PROG_GetActiveCurrentSession	490
4.6.2.4.216. PROG_GetBlockDowngradeFlagByIndex	490
4.6.2.4.217. PROG_GetBlockIdByIndex	491
4.6.2.4.218. PROG_GetComputedBootloaderChecksum	491
4.6.2.4.219. PROG_GetCurrentDiagApp	491
4.6.2.4.220. PROG_GetDidF0F3	492
4.6.2.4.221. PROG_GetDidF0F6	492
4.6.2.4.222. PROG_GetKeyNBIDValue	492
4.6.2.4.223. PROG_GetMacKey	492
4.6.2.4.224. PROG_GetNBIDValue	493
4.6.2.4.225. PROG_GetNetworkStatus	493
4.6.2.4.226. PROG_GetProgCntrLockVal	493
4.6.2.4.227. PROG_GetPseudoFlashDriverAddress	494
4.6.2.4.228. PROG_GetRequestRoutineResult_Impl80	494
4.6.2.4.229. PROG_GetRequestSeedCounter	494
4.6.2.4.230. PROG_GetRequestSeedTimer	494
4.6.2.4.231. PROG_GetRsDelayTimer	494
4.6.2.4.232. PROG_GetSBIFlagValue	495
4.6.2.4.233. PROG_GetSecurityLevel	495
4.6.2.4.234. PROG_GetSeed_Unlocked	495
4.6.2.4.235. PROG_GetSuppressBitFromAppli	495
4.6.2.4.236. PROG_Guard_Erase_Check_EraseFinish	496
4.6.2.4.237. PROG_Guard_RD_Check_RTEFinish	496
4.6.2.4.238. PROG_HSMLastDataReceived	496
4.6.2.4.239. PROG_HSMRequestUpdate	497
4.6.2.4.240. PROG_HSMStatusManage	497
4.6.2.4.241. PROG_HSMUpdate_TD	497
4.6.2.4.242. PROG_HsmManage	497
4.6.2.4.243. PROG_IMPL30_HSMEntry	498



4.6.2.4.244. PROG_IMPL30_SignatureHandling	498
4.6.2.4.245. PROG_Impl10_CheckDataBlocksResult	498
4.6.2.4.246. PROG_Impl10_CheckMemoryAllowed	498
4.6.2.4.247. PROG_Impl10_CompareDataBlockHash	498
4.6.2.4.248. PROG_Impl10_Do_CheckHashOfKey	499
4.6.2.4.249. PROG_Impl10_Do_HashMoreUnwrittenData	499
4.6.2.4.250. PROG_Impl10_Entry_CheckMemoryFailed	499
4.6.2.4.251. PROG_Impl10_Entry_CheckReceivedKey	499
4.6.2.4.252. PROG_Impl10_Entry_SignatureCheck	499
4.6.2.4.253. PROG_Impl10_Entry_WriteKeyFinished	499
4.6.2.4.254. PROG_Impl10_FinalizeHash	500
4.6.2.4.255. PROG_Impl10_GenerateMac	500
4.6.2.4.256. PROG_Impl90_CheckDataBlocksResult	500
4.6.2.4.257. PROG_Impl90_CompareDataBlockHash	500
4.6.2.4.258. PROG_Impl90_Do_CheckHashOfKey	500
4.6.2.4.259. PROG_Impl90_Do_HashMoreUnwrittenData	500
4.6.2.4.260. PROG_Impl90_Entry_CheckReceivedKey	501
4.6.2.4.261. PROG_Impl90_Entry_SignatureCheck	501
4.6.2.4.262. PROG_Impl90_Entry_ValidateFailed	501
4.6.2.4.263. PROG_Impl90_Entry_ValidateFinish	501
4.6.2.4.264. PROG_Impl90_Entry_WriteKeyFinished	501
4.6.2.4.265. PROG_Impl90_FinalizeHash	501
4.6.2.4.266. PROG_Init	502
4.6.2.4.267. PROG_InvalidateBlock	502
4.6.2.4.268. PROG_InvalidateSection	502
4.6.2.4.269. PROG_InvalidateSection_BlockID	503
4.6.2.4.270. PROG_IsValidApplication	503
4.6.2.4.271. PROG_JumpToApplication	504
4.6.2.4.272. PROG_JumpToSBL	504
4.6.2.4.273. PROG_LogicalBlockHash	504
4.6.2.4.274. PROG_LogicalBlockHashFinish	504
4.6.2.4.275. PROG_Manage	505
4.6.2.4.276. PROG_MessageDigestCheck	505
4.6.2.4.277. PROG_OpenProgrammingSession	505
4.6.2.4.278. PROG_Prelinit	505
4.6.2.4.279. PROG_RD_Check	505
4.6.2.4.280. PROG_RTE	505
4.6.2.4.281. PROG_RangeChecksumFinish	506
4.6.2.4.282. PROG_ReadKeyProgrammedStatus	506
4.6.2.4.283. PROG_RequestDownload	506
4.6.2.4.284. PROG_RequestSeed	507
4.6.2.4.285. PROG_RequestTransferExit	507



4.6.2.4.286. PROG_ResReprog_CheckSegmentListVerif	507
4.6.2.4.287. PROG_ReturnsIsReProgRequestFromAppli	508
4.6.2.4.288. PROG_SA2_RD_Check	508
4.6.2.4.289. PROG_SBASignatureCheck	508
4.6.2.4.290. PROG_SBASignerInfoCheck	508
4.6.2.4.291. PROG_SecurityComputeAppChecksum	509
4.6.2.4.292. PROG_SendKey	509
4.6.2.4.293. PROG_SendNRC78	509
4.6.2.4.294. PROG_Send_NRC	509
4.6.2.4.295. PROG_SetCryptoDataSizeDataAddr	510
4.6.2.4.296. PROG_SetKeyNBIDValue	510
4.6.2.4.297. PROG_SetNBIDValue	510
4.6.2.4.298. PROG_SetNetworkStatus	510
4.6.2.4.299. PROG_SetProgrammingStatus	511
4.6.2.4.300. PROG_SetRequestSeedCounter	511
4.6.2.4.301. PROG_SetRequestSeedTimer	511
4.6.2.4.302. PROG_SetRsDelayTimer	511
4.6.2.4.303. PROG_SetSBIFlagValue	511
4.6.2.4.304. PROG_SignatureCheck	512
4.6.2.4.305. PROG_SignerInfoCheck	512
4.6.2.4.306. PROG_SimulateExtendedSessionNoResponse	512
4.6.2.4.307. PROG_SimulateOpenProgSession	512
4.6.2.4.308. PROG_SkipPage	512
4.6.2.4.309. PROG_Streaming	513
4.6.2.4.310. PROG_StreamingFrameReceived	513
4.6.2.4.311. PROG_SwitchApplicationMode	513
4.6.2.4.312. PROG_SwitchApplicationModelInd	513
4.6.2.4.313. PROG_TD	514
4.6.2.4.314. PROG_TD_Impl80	514
4.6.2.4.315. PROG_TpRxInd	514
4.6.2.4.316. PROG_TpStartOfReceptionInd	514
4.6.2.4.317. PROG_TpTxConf	514
4.6.2.4.318. PROG_TransferData	515
4.6.2.4.319. PROG_TxConfNotification	515
4.6.2.4.320. PROG_UpdatePSI	515
4.6.2.4.321. PROG_VerificationOnTheFly	516
4.6.2.4.322. PROG_VerificationOnTheFly_Impl80	516
4.6.2.4.323. PROG_VerifySectionCrc	516
4.6.2.4.324. PROG_Write	517
4.6.2.4.325. PROG_WriteCheck	517
4.6.2.4.326. PROG_WriteCheck_Impl80	517
4.6.2.4.327. PROG_WriteFingerprintCheck	517



4.6.2.4.328. PROG_Write_Impl80	517
4.6.2.4.329. Prog_CustomGetAdditionalProgrammingConditionalFlags	517
4.6.2.4.330. Prog_CustomGetECUInternalProgrammingFlag	518
4.6.2.4.331. Prog_CustomGetProgrammingConditionsFlag	518
4.6.2.4.332. Prog_CustomGetProgrammingTolerantConditionsFlag	518
4.6.2.4.333. Prog_CustomIsProdKeyPresent	518
4.6.2.4.334. Prog_CustomReadKeyChecksum	519
4.6.2.4.335. Prog_CustomWriteKey	519
4.6.2.4.336. Prog_GetAPPL_VBTLength	519
4.6.2.4.337. Prog_GetEssApplicationStartAddress	520
4.6.2.4.338. Prog_GetEssLength	520
4.6.2.4.339. Prog_GetEssLogicalBlockId	520
4.6.2.4.340. Prog_GetEssLogicalBlockLength	521
4.6.2.4.341. Prog_GetEssLogicalBlockNbr	521
4.6.2.4.342. Prog_GetEssLogicalBlockStartAddr	522
4.6.2.4.343. Prog_GetEssLogicalBlockVerifTable	522
4.6.2.4.344. Prog_GetEssStartAddr	523
4.6.2.4.345. Prog_GetEssValidityStatus	523
4.6.2.4.346. Prog_GetEssVerifTable	523
4.6.2.4.347. Prog_GetEss_VBTLength	524
4.6.2.4.348. Prog_GetLogicalBlockSignature	524
4.6.2.4.349. Prog_GetLogicalBlockVerifStructure	524
4.6.2.4.350. Prog_HSMTransferData	525
4.6.3. Integration notes	525
4.6.3.1. Exclusive areas	525
4.6.3.2. Production errors	526
4.6.3.3. Memory mapping	526
4.6.3.4. Integration requirements	526
4.7. SA	526
4.7.1. Configuration parameters	526
4.7.1.1. CommonPublishedInformation	527
4.7.1.2. PublishedInformation	530
4.7.1.3. General	530
4.7.1.4. CsmRandomGenerate	538
4.7.1.5. CsmRandomSeed	538
4.7.1.6. SignatureVerify	539
4.7.1.7. CsmMACVerification_MsgAuth	540
4.7.1.8. CsmMACGenerate_MsgAuth	541
4.7.1.9. CsmAES_Encryption	542
4.7.1.10. CsmAES_Decryption	543
4.7.1.11. CsmMACVerification_Poo	544
4.7.1.12. CsmMACGenerate_Poo	545



4.7.2. Application programming interface (API)	546
4.7.2.1. Type definitions	546
4.7.2.1.1. tAntiscanInfo	546
4.7.2.1.2. tDecompressStateType	547
4.7.2.1.3. tLimitReqSeedInfo	547
4.7.2.1.4. tSACsmJobConf	547
4.7.2.1.5. tSA_AESCTRAlgo	547
4.7.2.1.6. tSA_AuthenticationState	548
4.7.2.1.7. tSA_ConcatenatePar	548
4.7.2.1.8. tSA_MACAlgo	548
4.7.2.1.9. tSaBoolean	549
4.7.2.1.10. tSaCsmState	549
4.7.2.1.11. tSaStatus	549
4.7.2.2. Macro constants	549
4.7.2.2.1. CSM_E_BUSY	549
4.7.2.2.2. CSM_E_NOT_OK	549
4.7.2.2.3. CSM_E_OK	550
4.7.2.2.4. CSM_E_VER_NOT_OK	550
4.7.2.2.5. CSM_E_VER_OK	550
4.7.2.2.6. Csm_ReturnType	550
4.7.2.2.7. Csm_VerifyResultType	550
4.7.2.2.8. LZSS_BREAK EVEN	550
4.7.2.2.9. LZSS_END_OF_STREAM	551
4.7.2.2.10. LZSS_INDEX_BIT_COUNT	551
4.7.2.2.11. LZSS_LENGTH_BIT_COUNT	551
4.7.2.2.12. LZSS_MOD_WINDOW	551
4.7.2.2.13. LZSS_WINDOW_SIZE	551
4.7.2.2.14. SA1_AESCTR_CTOFFSET_REQ	551
4.7.2.2.15. SA1_AESCTR_CTOFFSET_RES	551
4.7.2.2.16. SA1_AESCTR_IVOFFSET	552
4.7.2.2.17. SA1_AESCTR_IVOFFSET_RES	552
4.7.2.2.18. SA1_ENCRYPTION_DATALEN	552
4.7.2.2.19. SA1_MSGID	552
4.7.2.2.20. SA1_MSGREQ_SIZE	552
4.7.2.2.21. SA1_MSGRES_SIZE	552
4.7.2.2.22. SA1_MSGSIZE_WITHOUT_MACVALUE	553
4.7.2.2.23. SA1_RESPONSE_MSGID	553
4.7.2.2.24. SA1_RESSIZE_WITHOUT_MACVALUE	553
4.7.2.2.25. SA2_AESCTR_CTOFFSET	553
4.7.2.2.26. SA2_AESCTR_IVOFFSET	553
4.7.2.2.27. SA2_AESCTR_PTLEN	553
4.7.2.2.28. SA2_MAX_AWAIT_TIME	553



4.7.2.2.29. SA2_MSGID	554
4.7.2.2.30. SA2_MSGREQ_SIZE	554
4.7.2.2.31. SA2_MSGRES_SIZE	554
4.7.2.2.32. SA2_MSGSIZE_WITHOUT_MACVALUE	554
4.7.2.2.33. SA_AESCTR_BLOCKSIZE	554
4.7.2.2.34. SA_AESCTR_IVLEN	554
4.7.2.2.35. SA_AES_DECRYPT_FINISH	555
4.7.2.2.36. SA_AES_DECRYPT_START	555
4.7.2.2.37. SA_AES_DECRYPT_UPDATE	555
4.7.2.2.38. SA_AES_ENCRYPT_FINISH	555
4.7.2.2.39. SA_AES_ENCRYPT_START	555
4.7.2.2.40. SA_AES_ENCRYPT_UPDATE	555
4.7.2.2.41. SA_AM0001_IDLE	556
4.7.2.2.42. SA_AM0001_MAXBUF	556
4.7.2.2.43. SA_AM0001_PROCESSING_FAILED	556
4.7.2.2.44. SA_AM0001_STATE_BUSY	556
4.7.2.2.45. SA_AM0001_STATE FAILED	556
4.7.2.2.46. SA_ANTISCANNING_ENABLED	556
4.7.2.2.47. SA_AS_LOCK_TIMER	556
4.7.2.2.48. SA_AS_MAX_NB_RETRY	557
4.7.2.2.49. SA_AUTHENTICATION_0001	557
4.7.2.2.50. SA_AUTHENTICATION_METHOD	557
4.7.2.2.51. SA_AUTHENTICATION_OFF	557
4.7.2.2.52. SA_AWAITS2_TIMEEXPIRED	557
4.7.2.2.53. SA_AWAIT_SA2REQ	557
4.7.2.2.54. SA_AWAIT_SA2REQ_BUSY	558
4.7.2.2.55. SA_CHALLENGE_BIT	558
4.7.2.2.56. SA_CLIENT_POO_SIZE	558
4.7.2.2.57. SA_CLIENT_RNDNUM_SIZE	558
4.7.2.2.58. SA_COMPARE_KEY_AM0001	558
4.7.2.2.59. SA_COMPARE_KEY_STANDARD	558
4.7.2.2.60. SA_COMPARE_KEY_TYPE	558
4.7.2.2.61. SA_COMPARE_KEY_VERIFY_SIGNATURE	559
4.7.2.2.62. SA_COMPRESSION_DISABLED	559
4.7.2.2.63. SA_COMPRESSION_ENABLED	559
4.7.2.2.64. SA_COMPRESSION_STATE	559
4.7.2.2.65. SA_CRYPTO_AESDECRYPT_KEYELEID	559
4.7.2.2.66. SA_CRYPTO_AESDECRYPT_KEYSIZE	559
4.7.2.2.67. SA_CRYPTO_AESENCRYPT_KEYELEID	560
4.7.2.2.68. SA_CRYPTO_AESENCRYPT_KEYSIZE	560
4.7.2.2.69. SA_CRYPTO_MACGENPOO_KEYELEID	560
4.7.2.2.70. SA_CRYPTO_MACGENPOO_KEYSIZE	560



4.7.2.2.71. SA_CRYPTO_MACGEN_KEYELEID	561
4.7.2.2.72. SA_CRYPTO_MACGEN_KEYSIZE	561
4.7.2.2.73. SA_CRYPTO_MACVERIFYPOO_KEYELEID	561
4.7.2.2.74. SA_CRYPTO_MACVERIFYPOO_KEYSIZE	561
4.7.2.2.75. SA_CRYPTO_MACVERIFY_KEYELEID	561
4.7.2.2.76. SA_CRYPTO_MACVERIFY_KEYSIZE	562
4.7.2.2.77. SA_CRYPTO_SIGN_KEYELEID	562
4.7.2.2.78. SA_CRY_EXPONENT_ENABLED	562
4.7.2.2.79. SA_CSMASR43_USED	562
4.7.2.2.80. SA_CSM_AESCTR_DECRYPTION_USED	562
4.7.2.2.81. SA_CSM_AESCTR_DECRYPT_ID	563
4.7.2.2.82. SA_CSM_AESCTR_ENCRYPTION_USED	563
4.7.2.2.83. SA_CSM_AESCTR_ENCRYPT_ID	563
4.7.2.2.84. SA_CSM_AESDECRYPT_KEYID	563
4.7.2.2.85. SA_CSM_AESENCRYPT_KEYID	563
4.7.2.2.86. SA_CSM_ALLJOBS_COUNT	563
4.7.2.2.87. SA_CSM_CANCELJOB_ENABLED	564
4.7.2.2.88. SA_CSM_INDEX	564
4.7.2.2.89. SA_CSM_MACGENPOO_KEYID	564
4.7.2.2.90. SA_CSM_MACGEN_KEYID	564
4.7.2.2.91. SA_CSM_MACVERIFYPOO_KEYID	564
4.7.2.2.92. SA_CSM_MACVERIFY_KEYID	564
4.7.2.2.93. SA_CSM_MAC_GENERATION_USED	565
4.7.2.2.94. SA_CSM_MAC_MSGCODEGEN_ID	565
4.7.2.2.95. SA_CSM_MAC_MSGCODEVERIFY_ID	565
4.7.2.2.96. SA_CSM_MAC_POOGEN_ID	565
4.7.2.2.97. SA_CSM_MAC_POOVRFY_ID	565
4.7.2.2.98. SA_CSM_MAC_VERIFICATION_USED	565
4.7.2.2.99. SA_CSM_RANDOM_GENERATE_ID	566
4.7.2.2.100. SA_CSM_RANDOM_SEED_ID	566
4.7.2.2.101. SA_CSM_RANDOM_SEED_KEYID	566
4.7.2.2.102. SA_CSM_SETKEY_ENABLED	566
4.7.2.2.103. SA_CSM_SIG_VERIFY_ID	566
4.7.2.2.104. SA_CSM_SIG_VERIFY_KEYID	566
4.7.2.2.105. SA_CSM_STATE_INIT	567
4.7.2.2.106. SA_CUSTOMGETAESCTRKEY	567
4.7.2.2.107. SA_CUSTOMGETMACKEY	567
4.7.2.2.108. SA_CUSTOM_CSMSTARTPREPROCESS_ENABLED	567
4.7.2.2.109. SA_Csm_MainFunction	567
4.7.2.2.110. SA_DECOMP_COMPLETE	567
4.7.2.2.111. SA_DECOMP_COMPRESSLEN	568
4.7.2.2.112. SA_DECOMP_COMPRESSPOS	568



4.7.2.2.113. SA_DECOMP_FINISH	568
4.7.2.2.114. SA_DECOMP_INIT	568
4.7.2.2.115. SA_DECOMP_IN_PROGRESS	568
4.7.2.2.116. SA_DECOMP_OUT_BUF_SIZE	568
4.7.2.2.117. SA_DECOMP_UNCOMPRESSED	569
4.7.2.2.118. SA_DECRYPT_REQDATA_SA1	569
4.7.2.2.119. SA_DECRYPT_REQDATA_SA1_FAILED	569
4.7.2.2.120. SA_DECRYPT_REQDATA_SA2	569
4.7.2.2.121. SA_DECRYPT_REQDATA_SA2_FAILED	569
4.7.2.2.122. SA_ECB2CTR_ENABLED	569
4.7.2.2.123. SA_ERR_NULL_PTR	569
4.7.2.2.124. SA_E_BUSY	570
4.7.2.2.125. SA_E_NOK_AS_LIMIT_RS	570
4.7.2.2.126. SA_E_NOK_AS_LOCKED	570
4.7.2.2.127. SA_E_NOK_INVALID_KEY	570
4.7.2.2.128. SA_E_NOK_RS_DELAY_LOCKED	570
4.7.2.2.129. SA_E_NOT_OK	570
4.7.2.2.130. SA_E_OK	571
4.7.2.2.131. SA_E_STATUS_UNKNOWN	571
4.7.2.2.132. SA_E_WAITING_SA2_NOT_OK	571
4.7.2.2.133. SA_FALSE	571
4.7.2.2.134. SA_IDLE	571
4.7.2.2.135. SA_KEYUNLOCK	571
4.7.2.2.136. SA_KEYUNLOCK_FAILED	571
4.7.2.2.137. SA_KEYUNLOCK_SUCCESS	572
4.7.2.2.138. SA_KEY_LEN	572
4.7.2.2.139. SA_KEY_LEN_AM0001	572
4.7.2.2.140. SA_LIMIT_NB_REQUEST_SEED_ENABLED	572
4.7.2.2.141. SA_MACSIZE	572
4.7.2.2.142. SA_MAC_GENERATE_FINISH	572
4.7.2.2.143. SA_MAC_GENERATE_START	573
4.7.2.2.144. SA_MAC_GENERATE_UPDATE	573
4.7.2.2.145. SA_MAC_VERIFY_FINISH	573
4.7.2.2.146. SA_MAC_VERIFY_START	573
4.7.2.2.147. SA_MAC_VERIFY_UPDATE	573
4.7.2.2.148. SA_MANAGE_PERIOD	573
4.7.2.2.149. SA_MSGAUTHCODE_MACSIZE	573
4.7.2.2.150. SA_PUBLIC_KEY_LENGTH	574
4.7.2.2.151. SA_RANDOM_GEN_STATE_GENERATE	574
4.7.2.2.152. SA_RANDOM_GEN_STATE_STANDBY	574
4.7.2.2.153. SA_RANDOM_NUMBER_LENGTH	574
4.7.2.2.154. SA_REQUEST_SEED_DELAY_TIMER_ENABLED	574



4.7.2.2.155. SA_RS_DELAY_TIMER	574
4.7.2.2.156. SA_RS_LIMIT_COUNTER	575
4.7.2.2.157. SA_RS_LIMIT_TIMER	575
4.7.2.2.158. SA_SA2AWATINGTIME_EXPIRED	575
4.7.2.2.159. SA_SECUREDKEYS_STATUSREAD	575
4.7.2.2.160. SA_SECURITY_ALOGORITHM_CUSTOM	575
4.7.2.2.161. SA_SECURITY_ALOGORITHM_STANDARD	575
4.7.2.2.162. SA_SECURITY_ALOGORITHM_TYPE	575
4.7.2.2.163. SA_SECURITY_ALOGORITHM_UNDEFINED	576
4.7.2.2.164. SA_SEEDGEN_MSGAUTHCODE	576
4.7.2.2.165. SA_SEEDGEN_MSGAUTHCODE_FAILED	576
4.7.2.2.166. SA_SEEDGEN_MSGAUTHCODE_SUCCESS	576
4.7.2.2.167. SA_SEEDGEN_RNDIV	576
4.7.2.2.168. SA_SEEDGEN_RNDIV_FAILED	576
4.7.2.2.169. SA_SEEDGEN_RNDSERVER	577
4.7.2.2.170. SA_SEEDGEN_RNDSERVER_FAILED	577
4.7.2.2.171. SA_SEEDGEN_SERVERPOO	577
4.7.2.2.172. SA_SEEDGEN_SERVERPOO_ENCRYBLK2	577
4.7.2.2.173. SA_SEEDGEN_SERVERPOO_ENCRYPT	577
4.7.2.2.174. SA_SEEDGEN_SERVERPOO_ENCRYPT_FAILED	577
4.7.2.2.175. SA_SEEDGEN_SERVERPOO_FAILED	578
4.7.2.2.176. SA_SEEDGEN_SERVERPOO_SUCCESS	578
4.7.2.2.177. SA_SEED_CSM_RANDOM	578
4.7.2.2.178. SA_SEED_GEN_STATE_INIT	578
4.7.2.2.179. SA_SEED_GEN_STATE_START	578
4.7.2.2.180. SA_SEED_GEN_STATE_UPDATE	578
4.7.2.2.181. SA_SEED_LEN	578
4.7.2.2.182. SA_SEED_LEN_AM0001	579
4.7.2.2.183. SA_SEED_STANDARD	579
4.7.2.2.184. SA_SEED_TYPE	579
4.7.2.2.185. SA_SEND_SA1RES	579
4.7.2.2.186. SA_SEND_SA1RES_AWAIT_TXCONF	579
4.7.2.2.187. SA_SEND_SA1RES_FAILED	579
4.7.2.2.188. SA_SERVER_RNDNUM_SIZE	580
4.7.2.2.189. SA_SIGNATURE_CHECK_FINISH	580
4.7.2.2.190. SA_SIGNATURE_CHECK_STANDBY	580
4.7.2.2.191. SA_SIGNATURE_CHECK_START	580
4.7.2.2.192. SA_SIGNATURE_CHECK_UPDATE	580
4.7.2.2.193. SA_SIGNATURE_LENGTH	580
4.7.2.2.194. SA_STATIC_KEY_LEN	581
4.7.2.2.195. SA_STATIC_SEED_ENABLED	581
4.7.2.2.196. SA_TRUE	581



4.7.2.2.197. SA_UDS_ACK	581
4.7.2.2.198. SA_USE_CRYPTO	581
4.7.2.2.199. SA_VRFYMSG_AUTHCODE_SA1	581
4.7.2.2.200. SA_VRFYMSG_AUTHCODE_SA1_FAILED	581
4.7.2.2.201. SA_VRFYMSG_AUTHCODE_SA2	582
4.7.2.2.202. SA_VRFYMSG_AUTHCODE_SA2_FAILED	582
4.7.2.3. Objects	582
4.7.2.3.1. m_astCsmJobConf	582
4.7.2.3.2. m_aubSAPublicModulus	582
4.7.2.3.3. m_aubSaAesKeyData	582
4.7.2.3.4. m_aubSaMacKeyData	582
4.7.2.3.5. m_eSaCsmState	583
4.7.2.3.6. m_eSaStatus	583
4.7.2.3.7. m_pubSAReqResptr	583
4.7.2.3.8. m_stAESCTRpar	583
4.7.2.3.9. m_stLimitReqSeed	583
4.7.2.3.10. m_stMACpar	583
4.7.2.3.11. m_ubMacVerificationResult	584
4.7.2.3.12. m_ubSaAllKeysStatus	584
4.7.2.3.13. m_ulSAPublicExponent	584
4.7.2.4. Functions	584
4.7.2.4.1. SA_AM0001_GetState	584
4.7.2.4.2. SA_AM0001_SetState	584
4.7.2.4.3. SA_AM0001_SetStatus	585
4.7.2.4.4. SA_AuthMethod0001	585
4.7.2.4.5. SA_CompareKey	585
4.7.2.4.6. SA_ConcatenateAndStoreData	585
4.7.2.4.7. SA_CsmNotification	586
4.7.2.4.8. SA_CustomCsmStrtPreproc	586
4.7.2.4.9. SA_CustomGetAESCTRKey	586
4.7.2.4.10. SA_CustomGetAsymPublicKey	587
4.7.2.4.11. SA_CustomGetLastRandomNumber	587
4.7.2.4.12. SA_CustomGetMACKey	587
4.7.2.4.13. SA_CustomRestoreAsRetryCnt	587
4.7.2.4.14. SA_CustomStoreAsRetryCnt	588
4.7.2.4.15. SA_CustomStoreRandomNumber	588
4.7.2.4.16. SA_DecomplInputParamInit	588
4.7.2.4.17. SA_DecomprWriteDataConfirmation	588
4.7.2.4.18. SA_DecompressData	589
4.7.2.4.19. SA_DecompressInit	589
4.7.2.4.20. SA_DecompressManage	589
4.7.2.4.21. SA_GetDecompressedData	589



4.7.2.4.22. SA_GetSecuredKeysStatus	590
4.7.2.4.23. SA_GetSeed	590
4.7.2.4.24. SA_GetStatus	590
4.7.2.4.25. SA_Init	590
4.7.2.4.26. SA_Manage	591
4.7.2.4.27. SA_SetSecuredKeyStatus	591
4.7.3. Integration notes	591
4.7.3.1. Exclusive areas	591
4.7.3.2. Production errors	591
4.7.3.3. Memory mapping	591
4.7.3.4. Integration requirements	592
4.8. Uds	592
4.8.1. Configuration parameters	592
4.8.1.1. CommonPublishedInformation	593
4.8.1.2. PublishedInformation	596
4.8.1.3. General	596
4.8.1.4. Session	603
4.8.1.5. Service	604
4.8.1.6. Supplier_Services	609
4.8.1.7. Service_DID	611
4.8.1.8. Routine_Control	615
4.8.1.9. Service_OBD	620
4.8.2. Application programming interface (API)	623
4.8.2.1. Objects	623
4.8.2.1.1. m_astDiagSrvCfg1	623
4.8.2.1.2. m_astDiagSrvCfg2	623
4.8.2.1.3. m_astDiagSrvCfg3	623
4.8.2.1.4. m_astDiagSrvCfg5	624
4.8.2.2. Functions	624
4.8.2.2.1. UDS_CbkOnRxRequestInd	624
4.8.2.2.2. UDS_CustomIsBMCCountTimeout	624
4.8.2.2.3. UDS_CustomPositiveAnswerInd	625
4.8.2.2.4. UDS_CustomSupplier_BA	625
4.8.2.2.5. UDS_CustomSupplier_BB	626
4.8.2.2.6. UDS_CustomSupplier_BC	626
4.8.2.2.7. UDS_CustomSupplier_BD	627
4.8.2.2.8. UDS_CustomSupplier_BE	628
4.8.2.2.9. UDS_GetCurrentSession	628
4.8.2.2.10. UDS_Init	629
4.8.2.2.11. UDS_IsOBDSERVICE	629
4.8.2.2.12. UDS_IsPending	629
4.8.2.2.13. UDS_LongRequestEnd	630

4.8.2.2.14. UDS_LongRequestRespTxConf	630
4.8.2.2.15. UDS_LongRequestResponseInd	630
4.8.2.2.16. UDS_Manage	631
4.8.2.2.17. UDS_P2AboutToExpireInd	631
4.8.2.2.18. UDS_ReloadTStopDiag	631
4.8.2.2.19. UDS_ResponsePending_Manage	632
4.8.2.2.20. UDS_ResponsePending_TimCntManage	632
4.8.2.2.21. UDS_RxRequest	632
4.8.2.2.22. UDS_RxRequestWithAddrMode	633
4.8.2.2.23. UDS_SessionStatusInd	633
4.8.2.2.24. UDS_StopNRC78Timer	634
4.8.2.2.25. UDS_StopSessionTimer	634
4.8.3. Integration notes	634
4.8.3.1. Exclusive areas	634
4.8.3.2. Production errors	634
4.8.3.3. Memory mapping	635
4.8.3.4. Integration requirements	635
5. Bibliography	636



1. Overview of EB tresos Bootloader for Essentials documentation

Welcome to the EB tresos Bootloader for Essentials (BL for Essentials) product documentation.

This document provides:

- ▶ [Chapter 2, “BL for Essentials release notes”](#): release notes for the BL for Essentials modules
- ▶ [Chapter 3, “BL for Essentials user guide”](#): containing background information and instructions
- ▶ [Chapter 4, “BL for Essentials module references”](#): information about configuration parameters and the application programming interface

2. BL for Essentials release notes

2.1. Overview

This chapter provides the BL for Essentials specific release notes.

2.2. Scope of the release

2.2.1. Configuration tool

Your release of EB tresos Bootloader for Essentials is compatible with the release of the EB tresos Studio configuration tool:

- ▶ EB tresos Studio: 28.2.0 b211016-0103

2.2.2. EB tresos Bootloader for Essentials modules

The following table lists modules which are part of BL for Essentials release.

Module name	Module version	Supplier
APP	3.8.0	Elektrobit Automotive GmbH
BLUpdater	1.3.0	Elektrobit Automotive GmbH
BM	1.13.0	Elektrobit Automotive GmbH
BIPduR	0.23.0	Elektrobit Automotive GmbH
BundleBoot	0.1.23	Elektrobit Automotive GmbH
BundleBootOEMInd	0.0.23	Elektrobit Automotive GmbH
Prog	2.42.0	Elektrobit Automotive GmbH
ProgOEMInd	1.12.0	Elektrobit Automotive GmbH
SA	1.16.0	Elektrobit Automotive GmbH



Module name	Module version	Supplier
Uds	3.17.0	Elektrobit Automotive GmbH

Table 2.1. Modules specified by OEM specification

2.3. Module release notes

2.3.1. APP module release notes

- ▶ Module version: 3.8.0.BL3_B470986
- ▶ Supplier: Elektrobit Automotive GmbH

2.3.1.1. Change log

This chapter lists the changes between different versions.

Module version 3.8.0

2021-04-20

Module version 3.7.0

2020-10-22

Module version 3.6.9

2020-09-22

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 3.6.8

2019-12-09



- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 3.6.7

2018-10-25

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 3.6.6

2018-07-20

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 3.6.5

2017-04-03

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 3.6.4

2016-12-15

- ▶ Internal module improvement. This module version update does not affect module functionality
- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 3.6.3

2016-10-10

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 3.6.2

2016-08-12

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 3.6.1

2016-05-30



Module version 3.6.0

2016-03-29

Module version 3.5.6

2016-01-14

Module version 3.5.5

2015-10-26

- ▶ OSCAPP-352: Renamed file APP_Boot_VCC.c for APP_Boot.c in order to make it generic for all the OEMs.

Module version 3.5.4

2015-07-20

- ▶ OSCAPP-312: [BOOT_VCC_JLR] Add the PROG_SwitchApplicationModeInd callback to allow the integrator to do some actions before jumping to SBL.

Module version 3.5.3

2015-04-29

- ▶ OSCAPP-295: Remove EB_Init from APP_Init

Module version 3.5.2

2015-01-08

- ▶ Internal changes

Module version 3.5.1

2014-12-15



- ▶ OSCAPP-274: [BOOT_VCC_JLR] Add PROG_GetSBLInfo to allow customer to do some check before jumping in SBL.

Module version 3.5.0

2014-11-05

- ▶ OSCAPP-263: [BOOT_VCC_JLR] Add new Prog PROG_CustomerInit callback into APP_Boot_VCC.c
- ▶ OSCAPP-266: A TRESOS field list was created to allow to choose NvM MODE into APP. This field is only available when Dummy_Application is true. OSCAPP-262: - [PSA_CAN_LS] DummyCode in DEM_- StoreDtcCbk is updated. - [ALL] DummyCode in UDS_ClearDiagnosticInformation is updated.

Module version 3.4.0

2014-09-25

- ▶ OSCAPP-254: [PSA_CAN_HS_IS] DummyCode updated in APP_NmMonDefaultInd and APP_ComIf-BusStatusInd.
- ▶ OSCAPP-252: [BOOT_VCC_JLR] Add interfaces with ZF Framework

Module version 3.3.8

2014-09-15

- ▶ OSCAPP-243: [BOOT_VCC_JLR] A new integration part is generated in APP_TpTxConf if the network use is FR. It shall not be removed by customer.
- ▶ OSCAPP-245: [BOOT_VCC_JLR] Add two callbacks to manage the AlredyErase flag use to know if the ECU is already erase for the first reprogramming (when deliver to VCC)
- ▶ OSCAPP-244: [PSA_CAN_HS_IS] DummyCode in ComIf_ClearAllDTC and UDS_ClearDiagnosticInformation function updated.
- ▶ OSCAPP-247: [BOOT_VCC_JLR] To allow the correct asynchronous data retrieving in FR stack the buffer pass to TP_SetMsgData is now a global variable

Module version 3.3.7

2014-08-18

- ▶ OSCAPP-234: [BOOT_PSA] Callback implementation improved to better match implementation rules
- ▶ OSCAPP-235: [ALL] New UDS_SecurityCheck callback to check if ECU is locked (NRC33)



- ▶ OSCAPP-239: [BOOT_PSA] New callback PROG_ReprogStepCbk implemented
- ▶ OSCAPP-238: [ALL] Callback APP_GetUdsDataBufferInd have a new argument to notify the result of the UDS treatment

Module version 3.3.6

2014-07-09

- ▶ OSCAPP-227: [PSA_CAN_HS] Update about IE: callback IE_GetIdleModeFromNVM replaced by IE_SettleModeFromNVM API

Module version 3.3.5

2014-06-24

- ▶ OSCAPP-221: [PSA_CAN_HS] Improved to integrate new module IE

Module version 3.3.4

2014-06-06

- ▶ OSCAPP-220: [PSA_CAN_LS/PSA_CAN_HS] DummyCode in APP_UdsSessionStatusInd updated

Module version 3.3.3

2014-05-23

- ▶ OSCAPP-217: [PSA_CAN_HS_IS] Add callback APP_GetDiagnosticConditions
- ▶ OSCAPP-218: [RSA_CAN_HS] Simplify the Bootflag read management

Module version 3.3.2

2014-05-05

- ▶ OSCAPP-209: To be fully compliant with PSA HS LS specifications, APP_ComTxTimeout callback is used in PSA_CAN_HS_IS also.
- ▶ OSCAPP-210: [RSA_CAN_HS] Prototype of DEM_StoreDtcCbk changed (type of input parameter)
- ▶ OSCAPP-211: [VCC_JLR] Bug fix: Correction of the two following issue - The DSC 01 request shall not check the application validity for VCC. Only for JLR (when sleep management is enabled) - APP_ReprogReqManage shall be called in APP_Manage for VCC_JLR variant



- ▶ OSCAPP-211: [RSA_CAN_HS] Dummy code in APP_NmMonDefaultInd corrected

Module version 3.3.1

2014-04-10

- ▶ OSCAPP-206: APP_GetUdsDataBufferInd callback prototype added. This callback allows to indicate (and update if necessary) the data buffer status.
- ▶ OSCAPP-199: [RSA_CAN_HS] APP integration test code updated for MUTE and ABSENT DTC including DEM

Module version 3.3.0

2014-03-26

- ▶ OSCAPP-197: [BOOT_EB] Add APP_CalculateCrc function and plugins parameter Crc_Buffer_Size for CRC Algorithm
- ▶ OSCAPP-198: [BOOT_PSA_CAN_LS] Add initialization of eBootFromAppli in default case
- ▶ OSCAPP-200: [ALL] All src and includes files moved to generated/templates folder

Module version 3.2.21

2014-03-05

- ▶ OSCAPP-193: [BOOT_VCC_JLR]: Add callbacks to manage timeout error of CompleteAndCompatible callout

Module version 3.2.20

2014-02-27

- ▶ OSCAPP-184 : CDTCS (Inter Memo Def 0x85) could be integrated to APP
- ▶ OSCAPP-188: [BOOT_PSA_CAN_LS] Refresh Watchdog callback
- ▶ OSCAPP-189: Improvement: [OM_TYPE3] Update OM Callback APP_OmModifyInd When BSI life phase is received. LOM (LNI STACK) is set to NORMAL or SLEEP.

Module version 3.2.19

2014-02-18

- ▶ OSCAPP-182: Improvement: in case DEM is present it is possible to add the noMedAck (0xDF0000) in the DEM directly



- ▶ OSCAPP-181: Fixed known issue: [PSA]: JDD_Flush API is called when UDS clear DTC is received (instead of JDD_Init and JDD_Start)

Module version 3.2.18

2014-02-14

- ▶ OSCAPP-176: [BOOT_PSA_CAN_LS]: Bug fix: Switching APP to BOOT is done with UDS SA1 (SA1 response in BOOT)

Module version 3.2.17

2014-01-24

- ▶ OSCAPP-157: [BOOT_PSA_CAN_LS] Remove APP_TpTxConf implementation code as not used anymore
- ▶ OSCAPP-157: [BOOT] Remove unused implementation in APP_ReprogReqManage (APP_Boot_XXX.c)
- ▶ OSCAPP-159: New CAN_GetBaudRateIdx callback in case of multi baudrate configuration
- ▶ OSCAPP-168: Update APP_UdsSessionStatusInd callback adding new arguments (new session, old session and the reason of the changing session)
- ▶ OSCAPP-169: [BOOT_JLR] Update APP with new feature to manage the bootloader JLR sleep management

Module version 3.2.16

2013-12-17

- ▶ OSCAPP-152: Fixed known issue: [PSA_CAN_LS] redeclaration variables

Module version 3.2.15

2013-11-28

- ▶ OSCAPP-146: Fixed known issue: [PSA], Generate correctly the DEM callback for PSA_CAN_LS variants
- ▶ OSCAPP-148: Improvement: [DEM PSA], Automatically generate the DEM_Manage into APP_Manage
- ▶ OSCAPP-150: Fixed known issue: [PSA], TELE unlocking status check missing in integration test code
- ▶ OSCAPP-149: Fixed known issue: [PSA_CAN_LS], Management of GHD is not correct in case defect added is ABSENT_BSI (integration test code)
- ▶ OSCAPP-147: Improvement: [JLR], Adding variant JLR for bootloader management.



Module version 3.2.14

2013-11-14

- ▶ OSCAPP-144: Fixed known issue: [PSA], ComIf_NoMedAcqIndication have now two argument (DTCcode and Header value) to allow customer to remove the MED from memory

Module version 3.2.13

2013-10-24

- ▶ OSCAPP-137: Allow generation of boot source files in PSA_CAN_LS
- ▶ OSCAPP-138: Fixed known issue: include mechanism changed (EBCLG_Prj.h included instead of EBLIN_Prj.h if CLG is present)
- ▶ OSCAPP-141: Fixed known issue: [PSA_CAN_HS_IS], Add COM_SendFrame of version frame from EB_Init to APP_Init

Module version 3.2.12

2013-10-03

- ▶ OSCAPP-134: Fixed known issue: DEM_StartOperatingCycle is now called for OM Type3

2.3.1.2. New features

- ▶ Add new tresos field list to allow to choose NvM MODE. This field is only available when Dummy_Application is true.

Description:

New feature available since version 3.4.0 In all variant, it is now possible to configure NvM Mode. There are two modes : the asynchronous mode and the synchronous mode. The NvM mode influence DEM/GHD features. DTC erasing shall be completed before diagnostic response transmission and NM reactivation.

Date

2014-10-30

2.3.1.3. EB-specific enhancements

This module is not part of the AUTOSAR specification.



2.3.1.4. Deviations

This module is not part of the AUTOSAR specification.

2.3.1.5. Limitations

This chapter lists the limitations of the module. Refer to the module references chapter *Integration notes*, subsection *Integration requirements* for requirements on integrating this module.

- ▶ For this module no limitations are known.

2.3.1.6. Open-source software

osc_App module does not use open-source software.

2.3.2. BLUpdater module release notes

- ▶ Module version: 1.3.0.BL3_B470986
- ▶ Supplier: Elektrobit Automotive GmbH

2.3.2.1. Change log

This chapter lists the changes between different versions.

Module version 1.3.0

2021-10-25

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.2.0

2021-07-22

- ▶ Implemented Bootloader Updater for Impl10



Module version 1.1.0

2021-04-20

- ▶ Implemented HSM based SecureBoot Update

Module version 1.0.0

2020-11-27

- ▶ BLU erase the full old boot memory section, write the new boot, and check the write operation

2.3.2.2. New features

- ▶ Bootloader Updater no Communication support added

Description:

The bootloader Updater allows the update of bootloader image without tester communication.

Date

2021-07-02

- ▶ HSM SecureBoot Update

Description:

The bootloader Updater allows HSM based Secure Boot Update.

Date

2021-04-08

- ▶ Bootloader Updater

Description:

The Bootloader Updater is a software that allows to reprogram the bootloader without informing the Tester.

Date

2020-07-07

2.3.2.3. EB-specific enhancements

This module is not part of the AUTOSAR specification.



2.3.2.4. Deviations

This module is not part of the AUTOSAR specification.

2.3.2.5. Limitations

This chapter lists the limitations of the module. Refer to the module references chapter *Integration notes*, subsection *Integration requirements* for requirements on integrating this module.

- ▶ There is no limitation noticed for this plugin

2.3.2.6. Open-source software

osc_BLUpdater module does not use open-source software.

2.3.3. BM module release notes

- ▶ Module version: 1.13.0.BL3_B470986
- ▶ Supplier: Elektrobit Automotive GmbH

2.3.3.1. Change log

This chapter lists the changes between different versions.

Module version 1.13.0

2021-10-25

Module version 1.12.0

2021-07-23

- ▶ Support Crypto ASR-4.3



Module version 1.11.1

2021-05-26

Module version 1.11.0

2021-04-20

- ▶ OSCBM-299: Fixed known issue: Application is not verified by the SecureBoot feature when BM Timeout Check is activated
- ▶ Implemented HSM based SecureBoot for Impl60
- ▶ Implemented Test Application support in IBM
- ▶ Implemented HSM based SecureBoot in IBM
- ▶ Implemented Test application support in IBM for Impl 0

Module version 1.10.0

2020-11-27

- ▶ Internal module improvement. This module version update does not affect module functionality
- ▶ Internal module improvement. Impl03 and Impl04 Merged to trunk
- ▶ Implemented Initial Boot Manager variant
- ▶ Internal module improvement. Changes for CMAC generation with CryShe
- ▶ Implemented HSM based SecureBoot for Impl10
- ▶ Used Dual bank feature for Impl4
- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.8.0

2020-09-22

- ▶ Implemented HSM Secure Boot feature for Implementation 03 and Implementation 04.
- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.4.0

2020-03-24



- ▶ Implemented HSM Secure Boot feature for Implementation 30 and Implementation 31.

Module version 1.3.0

2019-11-29

- ▶ Implemented secure boot feature for profile 10.
- ▶ Implemented to support FlexRay protocol in BL 3.9.1 bootloader version.
- ▶ Implemented "Dual Memory Bank" feature.
- ▶ Changed the name of the API to perform a reset and go to sleep in order to be in line with new API design naming

Module version 1.1.6

2019-03-26

- ▶ Implemented support to crypto ASR 4.3 stack via the Demo_CSM_Wrapper.
- ▶ Implemented secure boot feature for profile31.
- ▶ Implemented the support of Demo_CSM wrapper for Crypto ASR 4.3.

Module version 1.1.5

2018-10-25

- ▶ Improved Time-out at startup.

Module version 1.1.4

2018-07-23

- ▶ Implemented secure boot feature for profile50.

Module version 1.1.3

2018-06-14

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.1.2

2018-03-22



- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.1.1

2017-12-18

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.1.0

2017-10-26

- ▶ Implemented Authenticated Boot feature for profile50.

Module version 1.0.9

2017-10-16

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.0.8

2017-07-03

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.0.7

2017-05-11

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.0.6

2017-04-03

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.0.5

2016-12-16



- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.0.4

2016-10-10

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.0.3

2016-08-12

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.0.2

2016-05-30

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.0.1

2016-03-29

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.0.0

2016-02-23

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 0.0.3

2015-11-27

- ▶ OSCBM-64 : Bug Fix : Get the diagnostic tester source address if programming is requested by application.
Otherwise, set it to EB_ALL_TESTER_ADDRESS

Module version 0.0.2

2015-07-15



- ▶ OSCBM-47 : Bug Fix : to avoid problem after long loss of synchronization All slot mode shall be set each time we go in BootMode

Module version 0.0.1

2015-04-29

- ▶ Module creation

2.3.3.2. New features

- ▶ No new features have been added since the last release.

2.3.3.3. EB-specific enhancements

This module is not part of the AUTOSAR specification.

2.3.3.4. Deviations

This module is not part of the AUTOSAR specification.

2.3.3.5. Limitations

This chapter lists the limitations of the module. Refer to the module references chapter *Integration notes*, subsection *Integration requirements* for requirements on integrating this module.

- ▶ For this module no limitations are known.

2.3.3.6. Open-source software

osc_BM module does not use open-source software.

2.3.4. BIPduR module release notes

- ▶ Module version: 0.23.0.BL3_B470986
- ▶ Supplier: Elektrobit Automotive GmbH



2.3.4.1. Change log

This chapter lists the changes between different versions.

Module version 0.23.0

2021-10-25

- ▶ OSCBPDUR-229: Fixed known issue: DiagnosticSessionControl(ProgrammingSession) is not responded after reset
- ▶ OSCBPDUR-231: Fixed known issue: Requests other than functional TesterPresent are not ignored when a response to DiagnosticSessionControl received in application is expected.
- ▶ OSCBPDUR-236: Fixed known issue: S3 Timer is reloaded if TesterPresent is received after Response Pending

Module version 0.20.0

2021-07-22

- ▶ OSCBPDUR-223: Fixed known issue: Functional Tester Present frame not acknowledged on Ethernet

Module version 0.18.0

2021-04-21

- ▶ Updated the init functionality of the memory access with the introduction of ReProgMemM module
- ▶ Implemented Sleep Management feature for NM Message.
- ▶ OSCBPDUR-209: Fixed known issue: LIN physical response corrupted by functional request

Module version 0.17.0

2020-11-27

- ▶ After poweron/soft reset if Bootloader receives architecture message, Bootloader will select a group of canID belongs to the architecture message.

Module version 0.16.0

2020-09-22

- ▶ OSCBPDUR-185: Fixed known issue: [FCA Atlantis High] S3 Timeout in Bootloader Extended Session does not Reset ECU



- ▶ Changed the variable definition from static to configurable type i.e PduLengthType which can be configured to u16 or u32.
- ▶ Added compatibility with BL-0

Module version 0.14.0

2020-03-24

- ▶ Implemented Standard CAN ID reception which can bypass tester filtering
- ▶ Implemented support to communicate over FlexRay protocol
- ▶ OSCBCLPDUR-156: Fixed known issue: No NRC=13(IMLOIF) response for TesterPresent with Suppress Response Bit
- ▶ Implemented streaming feature for Implementation-10 and Implementation-11
- ▶ Implemented the Queued Requests feature
- ▶ Internal module improvement. This module version update does not affect module functionality
- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 0.13.0

2019-06-12

Module version 0.12.0

2019-11-29

- ▶ Implemented the Queued Requests feature

Module version 0.11.0

2019-11-07

- ▶ OSCBCLPDUR-156: Fixed known issue: No NRC=13(IMLOIF) response for TesterPresent with Suppress Response Bit
- ▶ Implemented streaming feature for Implementation-10 and Implementation-11

Module version 0.10.0

2019-09-10



- ▶ Implemented support to communicate over FlexRay protocol

Module version 0.9.0

2019-07-24

- ▶ OSCBBLPDUR-137: Fixed known issue: Prevent from going into sleep mode in case of long treatment of frame.
- ▶ Implemented Standard CAN ID reception which can bypass tester filtering

Module version 0.8.0

2019-03-24

- ▶ Added a function in order to check an online ethernet connection for profile20

Module version 0.7.0

2019-03-22

- ▶ Improved the handling of functional tester present with physical requests
- ▶ OSCBBLPDUR-89: Fixed known issue: which leads to not releasing the connection on transmission confirmation when the transmission is done without reception first
- ▶ OSCBBLPDUR-102: Fixed known issue: return incorrect values for BS and STmin when reading them through RDBI

Module version 0.6.0

2018-10-25

- ▶ Implemented Tester Filtering in the start of request reception

Module version 0.5.0

2018-07-20

- ▶ Internal module improvement. This module version update does not affect module functionality
- ▶ Corrected the request reception simulation at startup
- ▶ Implemented the initialization and de-activation of Com Stack in the integration code instead of doing it in plugin.



Module version 0.4.1

2018-06-14

Module version 0.4.0

2018-03-23

- ▶ OSCBLLPDUR-80: Fixed known issue: which leads to the impossibility to activate the TP Change parameter feature
- ▶ OSCBLLPDUR-70: Fixed known issue: which leads to timeout with MultipleBuffers
- ▶ Improved functional Pdu handling for several connection

Module version 0.3.0

2017-12-18

- ▶ Implemented Dynamic reconfiguration of TP parameters

Module version 0.2.1

2017-10-26

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 0.2.0

2017-10-13

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 0.1.0

2017-09-11

- ▶ OSCBLLPDUR-44: Fixed known issue: which leads to a non response after reset to an ECU Reset and Programming Session request when multiple identifiers were used

Module version 0.0.6

2017-08-02



- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 0.0.5

2017-07-03

- ▶ OSCBLLPDUR-35: Fixed known issue: which leads to corruption of received data when multiple receive buffers were used

Module version 0.0.4

2017-06-12

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 0.0.3

2017-06-07

- ▶ OSCBLLPDUR-26: Fixed known issue: that stop physical segmentation when functional request are received at the same time
- ▶ Corrected Multiple Identifier feature
- ▶ Improved Lin management to allow full LIN feature
- ▶ OSCBLLPDUR-31: Fixed known issue: which led the ECU to go in exception on a reception of a frame with suppress positive response bit

Module version 0.0.2

2017-05-11

- ▶ OSCBLLPDUR-17: Fixed known issue: that lock connection when message with Suppress Positive Response Bit set is received
- ▶ Added management of Ethernet communication stack

Module version 0.0.1

2017-04-03

- ▶ Module creation
- ▶ Added management of multiple buffer
- ▶ Added management of multiple Identifier



- ▶ Added management of LIN slave routing

2.3.4.2. New features

- ▶ The variable definition changed to configurable type i.e PduLengthType

Description:

The variable definition changed from static to configurable type i.e PduLengthType which can be configured to u16 or u32.

Date

2020-09-04

2.3.4.3. EB-specific enhancements

This module is not part of the AUTOSAR specification.

2.3.4.4. Deviations

This module is not part of the AUTOSAR specification.

2.3.4.5. Limitations

This chapter lists the limitations of the module. Refer to the module references chapter *Integration notes*, subsection *Integration requirements* for requirements on integrating this module.

2.3.4.6. Open-source software

osc_BIPduR module does not use open-source software.

2.3.5. Prog module release notes

- ▶ Module version: 2.42.0.BL3_B470986



- ▶ Supplier: Elektrobit Automotive GmbH

2.3.5.1. Change log

This chapter lists the changes between different versions.

Module version 2.42.0

2021-10-25

- ▶ OSCPROG-2315: Fixed known issue: The bootloader sends NRC31 to the other communication types other than normalCommunicationMessages for Impl40.
- ▶ Implemented Configuration for the location of CRC in the check memory routine for IMPL 60.
- ▶ OSCPROGVAG-172 Fixed known issue: Some logical blocks are not excluded from downgrade protection verification
- ▶ OSCPROG-2408 Fixed known issue: Bootloader responds with NRC 70 for Request Download if the memory size of the block is invalid
- ▶ OSCPROG-2391 Fixed known issue: Bootloader does not respond to the TransferData request
- ▶ OSCPROG-2431 Fixed known issue: The configuration of the logical block 0xFE causes a generation error
- ▶ Implemented CheckValiApplication Routine Control 0x0304 for Signed SW for IMPL 4
- ▶ OSCPROG-2413 Fixed known issue: Bootloader does not erase all the logical blocks upon the erase of ESS for IMPL 10
- ▶ OSCPROG-2428 Fixed known issue: Decryption fails when the memory size specified in the Request-Download is the uncompressed data size
- ▶ OSCPROG-2406: Fixed known issue: No response to DiagnosticSessionControl(ProgrammingSession) after reset.
- ▶ Implemented support to manage dynamically the signature/CRC verification based on DFI for Signed-compressed and Signed-uncompressed data download for IMPL 40.
- ▶ Implemented One Level Bootloader i.e IMPL 05 for OEM JLR.

Module version 2.38.0

2021-07-27

- ▶ OSCPROG-2361: Fixed known issue: Reduce length of response for UDS request 10 02.
- ▶ OSCPROG-2353: Fixed known issue: The bootloader does not respond after RequestTransferExit request for IMPL 10 and IMPL 11



- ▶ Implemented Programming sequence for IMPL90 and IMPL91
- ▶ Implemented CRC 16 for IMPL90 and IMPL91
- ▶ Implemented Signature verification for IMPL90 and IMPL91
- ▶ Internal module improvement Merge IMPL90 and IMPL91 to trunk
- ▶ Internal module improvement for IMPL 10 and 11.
- ▶ Support of download verification using Hash for IMPL 60.
- ▶ OSCPROG-2382: Fixed known issue: No response to CheckMemory routine due to memory corruption for IMPL 10
- ▶ OSCPROG-2369: Fixed known issue: PBL does not stay in Programming Session on Activate SBL Routine 0224 Failure
- ▶ OSCPROG-2373: Fixed known issue: The bootloader does not respond to the RequestTransferExit request after downloading an invalid application
- ▶ Implemented support for Secure Boot with Csm for IMPL 60.
- ▶ Implemented support for Crypto vASR 4.3 or higher version integration with Bootloader.

Module version 2.36.0

2021-04-21

- ▶ Improved RTE implementation for IMPL3 and IMPL4
- ▶ OSCPROG-2256: Fixed known issue: Verify Partial Software Checksum routine verifies the wrong block
- ▶ Implemented to support different Signature algo usage, implemented to allow VDS storage from the Integration code and the Internal module improvements for IMPL3 and IMPL4
- ▶ OSCPROG-2266: Fixed known issue: [DAG] Block size value is updated in case of invalid STmin value
- ▶ Implemented to allow write to RAM only for Impl3
- ▶ Implemented PreErase feature for Impl4
- ▶ OSCPROG-2269: Fixed known issue: Ethernet Bootloader not responding for the request Open Programming Session received by the application.
- ▶ Implemented Decryption CBC AES 128 for Impl60
- ▶ OSCPROG-2300: Fixed known issue: Application considered as valid even if the verification has failed
- ▶ OSCPROG-2301: Fixed known issue: WriteFingerprint request fails in case of download by logical block
- ▶ Implemented HSM based SecureBoot for Impl60
- ▶ OSCPROG-2299: Fixed known issue: CheckMemory failure for compressed data download.
- ▶ OSCPROG-2280: Fixed known issue: Bootloader no longer responds if it is triggered 10 82 from the application and the parameter Dsc_Prog_Response is set to true



- ▶ OSCPROG-2296: Fixed known issue: Communication fails after flashing the application file with a size of 1 byte.
- ▶ OSCPROG-2297: Fixed known issue: Plain data type check is not done in the Bootloader.
- ▶ Implemented Bootloader Updater with HSM SecureBoot
- ▶ Added the support of ReProgMemM module for memory access
- ▶ OSCPROG-2282: Fixed known issue: Response after a reset not implemented in the Bootloader.
- ▶ Implemented the downgrade protection for Impl40

Module version 2.34.0

2020-11-30

- ▶ OSCPROG-2186: Fixed known issue: Prevent failure of Signature verification if a TransferData block is re-transmitted
- ▶ OSCPROG-2187: Fixed known issue: Wrong NRC priority: NRC73 is sent when NRC71 conditions apply
- ▶ OSCPROG-2214: Fixed known issue: Implementation does not obey the NRC 0x24 for a TransferData request
- ▶ Implemented new variants IMPL3 and IMPL4
- ▶ Implemented Verification Data Structure for IMPL3 and IMPL4
- ▶ Implemented callbacks support for CertificateUpdate routine handling for Impl3 and Impl4
- ▶ Changed CMAC generation with CryShe
- ▶ Implemented HSM based SecureBoot for Impl10
- ▶ OSCPROG-2215: Fixed known issue: Observed NRC31 for TD request with WrongBlockSequence counter value and invalid TDRecord Parameter instead of NRC73 violating NRC priority
- ▶ Implemented Bootloader Updater for Impl40
- ▶ Implemented Bootloader Updater for Impl60
- ▶ Internal module improvement. Merged IMPL3 and IMPL4 to trunk
- ▶ Added the feature Dual Memory Bank for IMPL4

Module version 2.31.0

2020-09-22

- ▶ OSCPROG-2118: Fixed known issue: [GM] Wrong response for programming session request when in programming session.
- ▶ OSCPROG-2162: Fixed known issue: [GM] Bootloader does not verify the calibration signature with the correct AsymPublicKey.



- ▶ Implemented Positive response managed by the Boot for the 10 02 request received by the application when Ethernet protocol used.
- ▶ Changed the variable definition from static to configurable type i.e PduLengthType which can be configured to u16 or u32.
- ▶ OSCPROG-2135: Fixed known issue: Send NRC 31 to a Request Download with segment size which provoked overflow
- ▶ OSCPROG-2151: Fixed known issue: State machine is blocked in case of no TCP reconnection.
- ▶ Implemented SA Generation2 Authentication Method (0x0001) feature for Implementation 10 and Implementation 11.
- ▶ Implemented SA Secured Key Programming Status read for Implementation 10 and Implementation 11.
- ▶ Implemented Limit Client RequestSeed for Implementation 10 and Implementation 11.
- ▶ Implemented SA Delay timer at startup for Implementation 10 and Implementation 11.
- ▶ Implemented Decryption feature for Impl40

Module version 2.28.0

2020-06-26

- ▶ Implemented callbacks support for CheckReprogramming Preconditions Handling for Impl40
- ▶ Implemented callbacks support for Check Programming Dependency Handling for Impl30
- ▶ OSCPROG-2120: Fixed known issue: [DAG/VWAG] Wrong address used on verification when DualBank is activated
- ▶ Improved FCA Mid - Callback for signature Verification

Module version 2.27.0

2020-04-29

- ▶ OSCBLPDUR-185: Fixed known issue: [FCA Atlantis High] S3 time-out in bootloader extended session does not reset the ECU
- ▶ Improved Decompression slicing feature to set slicing value lesser than 1000 to improve bootloader performance.

Module version 2.22.0

2020-04-01

- ▶ OSCPROG-2047: Fixed known issue: NRC 0x13 is not sent for Service 0x34 with an invalid message length



- ▶ OSCPROG-2048: Fixed known issue: Wrong NRC code for UpdatePSI routine
- ▶ OSCPROG-2003: Fixed compilation error on EraseCheck when it is set to First Programming Check
- ▶ OSCPROG-2054: Fixed known issue: Sleep Timer is activated for VCC and OEMInd.
- ▶ OSCPROG-2058: Fixed known issue: DiagnosticSessionControl request is accepted without security unlock.
- ▶ OSCPROG-2044: Fixed known issue: Bootloader does not compile on Ethernet if the parameter Dsc_Prog_Response is enabled.
- ▶ Implemented segment based configurable Signature Verification for Impl30.
- ▶ Implemented HSM based SecureBoot and Secure Application checksum verification for Impl30 and Impl31.
- ▶ OSCPROG-2060: Fixed known issue: CRC computation is failed because of wrong initialization conditions of the algorithm.
- ▶ Implemented Compressed Flash Driver feature for OEMIND.
- ▶ OSCPROG-1957: Fixed known issue: The bootloader sends an incorrect response for a CheckReprogrammingPreconditions routine.
- ▶ Improved verification buffer size so it can be configured via the variable "DownloadVerification/Verification Buffer size" in EB tresos Studio
- ▶ Implemented FCA standard reprogramming type with CRC in data and Logical block hash request

Module version 2.17.0

2019-12-06

- ▶ Implemented check to ensure Fr Network synchronization before transmission of response for profile 11 and improvement in the handling of multiple TP connections.
- ▶ OSCPROG-1918: Fixed known issue: Wrong response is sent when maximum reprogramming counter is reached for DAG.
- ▶ OSCPROG-1948: Fixed known issue: Wrong return code on download verification.
- ▶ Implemented HSM Software Update.
- ▶ OSCPROG-1872: Fixed known issue: Checksum generation errors are not detected if Authenticated/Secured Boot is enabled.
- ▶ OSCPROG-1931: Fixed known issue: Wrong NRC in case of a Transfer Data request with no data.
- ▶ OSCPROG-1930: Fixed known issue: Corruption flag of the flash driver is not correctly updated after a signature length check failure.
- ▶ OSCPROG-1929: Fixed known issue: Transition from Compare Key to Request Download is not rejected.
- ▶ OSCPROG-1917: Fixed known issue: [OemInd] Response to a DSC01 request is not sent after reset.



- ▶ OSCPROG-1906: Fixed known issue: [OemInd] The callback PROG_CustomSetCrcCompareSuccess() is not called if CRC verification is wrong.
- ▶ Implemented the activation of the anti-scanning feature for implementation 10.
- ▶ OSCPROG-1983: Fixed known issue: [GM] Response to a DSC01 request is not sent after reset
- ▶ OSCPROG-1961: Fixed known issue: Sleep Timer is deactivated once a request is received while in default session
- ▶ Changed the name of the API which performs a reset and go to sleep to be in line with new API design naming
- ▶ OSCPROG-1950: Fixed known issue: The bootloader does not respond after the reset to an open programming session request received by the application
- ▶ Implemented the feature "Dual Memory Bank"
- ▶ OSCPROG-1901: Fixed known issue: NRC78 response is not sent for an EcuReset (0x11)
- ▶ OSCPROG-1773: Fixed known issue: Receive size of data during TransferData is corrupted
- ▶ Implemented the Check Memory status feature
- ▶ OSCPROG-2026: Fixed known issue: NRC78 response is sent continuously when receiving Check Memory Request with wrong signature

Module version 2.14.0

2019-07-23

- ▶ Implemented a preprocessor check to unselect the Cry_LN.h header file when the Tresos parameter "Use_CSM_ASR430_DemoWrapper" is set to "TRUE"
- ▶ OSCPROG-1847: Fixed known issue: SecurityAccess subfunction parameter is not read correctly.
- ▶ Implemented management of two successive Check Memory request for profile 10 and 11
- ▶ Implemented the feature to perform Session change without SecurityLevel Reset across the transitions between Non-default sessions for implementation 31
- ▶ OSCPROG-1743: Fixed known issue: Bootloader returns a wrong memory check result because it does not check the FileSize value for implementation 31
- ▶ OSCPROG-1896: Fixed known issue: Correct wrong answer sent when second Check Memory request is correct for implementation 10
- ▶ Added preinit waiting function for profile20
- ▶ OSCPROG-1869: Fixed known issue: Programming can continue even if the writing of the fingerprint failed for implementation 31
- ▶ Implemented the replacement of the SBI flag variable by customer callbacks
- ▶ Implemented the feature of write public key for profile 10



- ▶ OSCPROG-1865: Fixed known issue: NRC78 response is not sent for a DiagnosticSessionControl(defaultSession) request with the SuppressPositiveResponse bit set
- ▶ OSCPROG-1829: Fixed known issue: Flash writing operation is not done correctly when signature verification is performed on received data

Module version 2.13.0

2019-05-16

- ▶ Implemented the feature "ECU Software Structure"
- ▶ Implemented the feature "Download verification using the verification block table"
- ▶ Implemented the feature "Secure boot using Mac"
- ▶ OSCPROG-1772: Fixed known issue: Sleep Timer Management in Default Session
- ▶ OSCPROG-1765: Fixed known issue: [OEMInd] Continuous pending response when two successive WriteFingerPrint requests are received and the first is wrong
- ▶ OSCPROG-1821: Fixed known issue: [VWAG/OEMInd] ECU does not return to Lock state when GetSeed is not allowed (NRC_37)
- ▶ Implemented a guard function to ensure that only two successive check memory are allowed for profile 10

Module version 2.12.1

2019-04-03

Module version 2.12.0

2019-03-26

- ▶ OSCPROG-1646: Fixed known issue: Read finger print service returns a wrong response
- ▶ OSCPROG-1715: Fixed known issue: Information of two bytes block identifier is not correctly retrieved when feature "Signature Verification with address and length from Request Download" is activated
- ▶ OSCPROG-1561: Fixed known issue: Wrong PEC value is stored when RequestTransferExit request is received before all transfer data requests are finished
- ▶ Updated "check programming dependencies" feature by using callbacks in this for GM.
- ▶ OSCPROGFCA-72: Fixed known issue: Continuous NRC78 on request Check memory 0xF000.
- ▶ OSCPROG-1712: Fixed known issue: In programming session in case of security access is unlocked a new programming session request resets the security level.
- ▶ Implemented support of fingerprint for the RAM segments and check to avoid erase sector by sector for RAM segements



- ▶ OSCPROG-1445: Fixed known issue: Erase routine rejected after asynchronous fingerprint writing.
- ▶ OSCPROG-1731: Fixed known issue: Compile error if routine Verify_partial_software_checksum is not configured.
- ▶ OSCPROG-1703: Fixed known issue: Activate feature "Programming Counter" for FCA Atlantis High
- ▶ OSCPROG-1211: Fixed known issue: The transition from diagnostic default session to default session causes the ECU to reset
- ▶ OSCPROG-2021: Fixed known issue: Sleep Timer is activated for FCA
- ▶ OSCPROG-1753: Fixed known issue: CheckProgram routine fails when block identifier configuration doesn't start from 0
- ▶ OSCPROG-1711: Fixed known issue: Activate feature "Erase Check" for FCA Atlantis High
- ▶ Implemented a returned status in API PROG_CustomIncrementProgCounter
- ▶ OSCPROG-1548: Fixed known issue: Continuous pending response on reception of an invalid signature
- ▶ Implemented the feature "Protected Calibration" for GM bootloader
- ▶ Implemented the support of Demo_CSM wrapper for Crypto ASR 4.3.
- ▶ OSCPROG-1758: Fixed known issue: Invalid Response length for "Adjust ISO 15765-2 BS and STmin Parameter" DID writing for Daimler and Volkswagen
- ▶ OSCPROG-1767: Fixed known issue: SPREC parameters are present in DSC02 although SPREC_IN_-RESPONSE is deactivated
- ▶ Implemented secure boot feature for profile31.

Module version 2.11.0

2018-10-25

- ▶ Implemented the All Custom Memory Access
- ▶ Modified programming counter which shall not be incremented if the blocks have already been completely erased
- ▶ Added implementation variant 31
- ▶ Added of new structure in order to use two private key for GM signature calculate
- ▶ Removing program failed function and adding the behavior of profile20 to TD failed and RTE failed entry functions instead
- ▶ Implemented Preliminary Erasing feature
- ▶ Added call of blpdur for the tester filtering feature
- ▶ Modified the response to the request \$F0 \$F0 in the case the application is invalid/revoked by adding the information of invalid calibration [GM]
- ▶ Added Compression on variant 30



- ▶ Added download by logical block only feature for profile 40
- ▶ Added PROG_Dsc02Cbk callback implementation for profile 40
- ▶ Fixed compilation error when sleep management is off
- ▶ Implemented the feature "Compressed Flash Driver"
- ▶ Implemented the feature "Resumable reprogramming"
- ▶ Fixed compilation error when macros are defined in C files
- ▶ Implemented the feature "Block header reading"
- ▶ Implemented the feature "OEMInd: support asynchronous memory access"
- ▶ Implemented the feature "Signature on Compressed data"
- ▶ Implemented the feature "Signature Verification with address and length from Request Download"

Module version 2.10.0

2018-07-24

Module version 2.9.0

2018-07-20

- ▶ Corrected bad behavior on signature check start, update and finish as well as CRC computation [GM]
- ▶ Internal module improvement. This module version update does not affect module functionality
- ▶ OSCPROG-1388: Fixed known issue: Correct bad behaviour when receiving successive CheckMemory requests
- ▶ Implemented the Impl40/Impl50 public key management for Impl20.
- ▶ Added of callback to notify once there is a synchronous memory access.
- ▶ Implemented the application checksum computation and write in non-volatile memory for Secure Boot feature
- ▶ OSCPROG-1591: Fixed known issue: Correct compilation error when "Diagnostic Reprogramming response" parameter is not set
- ▶ Corrected bad behavior when response to "open programming session" request is sent by application
- ▶ Corrected bad behavior on the sending of positive response for the Transfer Data in the last buffer when Multiple buffers is enabled (no more Rx buffers available)

Module version 2.8.2

2018-06-14



Module version 2.8.1

2018-06-11

- ▶ Implemented SecurityAccess in Application feature.
- ▶ Implemented Flash Erase All feature.
- ▶ OSCPROG-1251: Fixed known issue: The minimum length of a TransferData request is not correctly checked.
- ▶ OSCPROG-1255: Fixed known issue: The calculation of the end address in RequestDownload and Erase request can overflow.
- ▶ Corrected bad behaviour on Check Memory when receiving Request Download and no Erase requests
- ▶ Corrected integration issue between PROG and Flash modules when Prog is configured to Synchronous and Flash returns Busy state.

Module version 2.8.0

2018-03-23

- ▶ OSCPROG-1451: Fixed known issue : Compression format 0xA is not supported, compression algorithm is now configurable
- ▶ Implemented Data Decryption feature.
- ▶ OSCPROG-1448: Fixed known issue : Only first segment of asynchronous memory is erased
- ▶ Improved CRC calculation form OEMInd by adding the possibility to deactivate the calculation
- ▶ Improved Request download feature by adding new configuration parameter (Maximum RequestDownload Per Block) to size the storing of downloaded area for every logical block
- ▶ Improved FingerPrint reading and writing feature.
- ▶ Improved Erase feature : NRC78 can be transmitted by configuration before software invalidation on the Erase request reception.
- ▶ Improved Erase feature : Modification of configuration parameter Erase Check Type
- ▶ Corrected VerifyPartialSoftwareChecksum routine with Signature feature
- ▶ Implemented Downloading Flash Driver feature.
- ▶ OSCPROGDAG-74: Fixed known issue: [DAG] Randomly incorrect response to SecurityAccess request.

Module version 2.7.1

2018-02-06

- ▶ OSCPROG-1390: Fixed known issue : [OEMInd] Missing return value in PROG_CustomSetCrcCompa-reSuccess

**Module version 2.7.0**

2017-12-18

- ▶ Removed Csm_Init call (moved to BM plugin)
- ▶ OSCPROG-1384: Fixed known issue : Multiple buffer processing can be interrupted by new incoming requests
- ▶ Implemented Dynamic reconfiguration of TP parameters feature.
- ▶ OSCPROG-1357: Fixed known issue : The Verify_partial_software_checksum routine is rejected
- ▶ Implemented Support of signature check for Impl40 and Impl60
- ▶ OSCPROG-1424: Fixed known issue : Undefined API when only external flash is used

Module version 2.6.1

2017-10-26

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 2.6.0

2017-10-16

- ▶ OSCPROG-1277: Fixed known issue : Give the user the possibility to address block on two bytes
- ▶ OSCPROG-1345: Fixed known issue : When compression is activated, downloading a non compressed data doesn't work
- ▶ Implemented Several external memory with a different access modes feature
- ▶ Implemented Signature verification on the fly feature

Module version 2.5.0

2017-09-08

- ▶ Updated Multiple buffer feature by managing the last TDaccording to Daimler clarification
- ▶ OSCPROG-1269: Fixed known issue : Fix compare key design
- ▶ OSCPROG-1315: Fixed known issue : Jump from application to bootloader through ProgrammingRequest (DSC02) does not work
- ▶ Updated decompression feature in order to add slicing
- ▶ OSCPROG-1224: Fixed known issue : Block 0 is supported by bootloader when no bootloader partition is defined
- ▶ Improved response after reset feature by storing and retrieving connection context when the ECU shall reset



- ▶ OSCPROG-1214: Fixed known issue : Verification of partial software checksum failure

Module version 2.4.1

2017-08-02

- ▶ OSCPROG-1222: Fixed known issue : The key NBID is not updated in NVM if a more recent one is received
- ▶ OSCPROG-1223: Fixed known issue : Correct Response to Request 0xFF00 depending of type errors occurred during check of erase memory parameters

Module version 2.4.0

2017-07-11

- ▶ Implemented LZSS compression feature

Module version 2.3.3

2017-06-16

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 2.3.2

2017-06-12

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 2.3.1

2017-05-11

- ▶ OSCPROG-1145: Fixed known issue : Positive Response was not sent after Ecu Reset whereas option was activated
- ▶ OSCPROG-1149: Fixed known issue : Correct Response to Request 0xFF00 when errors occurred during check of erase memory parameters
- ▶ Added callbacks to get symmetrical/asymmetrical cryptography public keys

Module version 2.3.0

2017-04-03

- ▶ Improved sleep management by activating the feature for Impl40 and Impl50



- ▶ Improved decompression feature by managing correctly the status returned by decompression library
- ▶ Implemented Implementation 60 features
- ▶ Implemented Multiple receive buffer feature
- ▶ Improved Ecuid feature: Added possibility to use a callback to get ECUID
- ▶ Improved EraseMemory and CheckMemory responses in Impl50
- ▶ Improved management of ECCCheck for all memory access functions
- ▶ OSCPROG-982: Fixed known issue: Wrong management in case of asynchronous memory use
- ▶ Removed unused API
- ▶ Updated CRC feature by adding the possibility to compute CRC on a particular Block Id
- ▶ Implemented watchdog deactivation for Impl50
- ▶ OSCPROG-1020: Fixed known issue: The NRC sequence is not correctly checked for an RTE request
- ▶ Improved DiagnosticSessionControl feature : Manage DiagnosticSessionControl request with subservice ExtendedSession and reset security level when entering ExtendedSession
- ▶ Updated fingerprint feature to handle target with pages superior to 8 bytes
- ▶ Implemented Programming counter for logical blocks feature
- ▶ Implemented Partial software checksum feature
- ▶ OSCPROG-1093: Fixed known issue : Erasing by blockID could return NRC_72 if PROG software module is configured in asynchronous mode
- ▶ OSCPROG-1020: Fixed known issue: The NRC sequence is not correctly checked for a TD request
- ▶ Implemented Routine CheckProgrammingDependencies feature for Impl20
- ▶ OSCPROG-1120: Fixed known issue : When logical block feature is activated, the RoutineControl Erase-Memory 0xFF00 will respond with the wrong answer (fields routineInfo and routineStatusRecord)
- ▶ OSCPROG-1122: Fixed known issue : When CRC32 feature is activated, the CRC value is extracted at wrong index from the RC 0x0202
- ▶ OSCPROG-1121: Fixed known issue : When Coherency check feature is activated, the RoutineControl EraseMemory 0xFF01 will respond with the wrong answer (fields routineInfo and routineStatusRecord)
- ▶ OSCPROG-1141: Fixed known issue : The RoutineControl EraseMemory 0xFF00 responds with the wrong answer (routineInfo and routineStatusRecord fields)

Module version 2.2.0

2016-12-16

- ▶ Changed header calculation for GM by including AppSwInfo header in a region
- ▶ Improve Erase by block id feature by adding new callback PROG_InvalidateSection_BlockID



- ▶ Improved decompression feature by subtracting compression header size to have the real size of compressed data
- ▶ OSCPROG-888: Fixed known issue: Fix issue on CheckProgrammingRequest when using Flash_Ext
- ▶ Improved robustness for decompression errors management
- ▶ OSCPROG-939: Fixed known issue: Fix issue on CheckProgrammingRequest when using Flash_Ext (by RANGE)
- ▶ OSCPROG-914: Fixed known issue: Erasing is performed even if the memory is already erased
- ▶ Improved erase feature by calling of setting the erase status upon reception of the Request download.
- ▶ Improved DSC response
- ▶ OSCPROG-953: Fixed known issue: Routine CheckProgrammingDependencies is rejected for address range on several segments
- ▶ Improved design of PROG module
- ▶ Improved CRC calculation by providing the information of which segment is invalidated when using the callback PROG_CustomSetAppValidity
- ▶ Implemented coherency check feature
- ▶ OSCPROG-955: Fixed known issue: LZMA/ARLE decompression can lead to an infinite loop
- ▶ Improved calculation of the Programming Status
- ▶ Improved CRC range calculation

Module version 2.1.1

2016-10-10

- ▶ OSCPROG-854: Fixed known issue: Add a verification on open programming session request's callback
- ▶ Implemented LZMA decompression feature
- ▶ Improved CRC calculation allowing asynchronous management upon receiving the Request Transfer exist
- ▶ Updated data access error management

Module version 2.1.0

2016-08-12

- ▶ Implemented Programming pre-condition check
- ▶ Implemented Reset cause and DSC/ER response management
- ▶ Implemented Management of CheckMemory routine for CRC verification
- ▶ OSCPROG-731: Fixed known issue: Unexpected behavior when reading DID F0F0 and no application is present



- ▶ Implemented Logical block management and Download by logical address
- ▶ OSCPROG-794: Fixed known issue: Missing initialization variable in PROG_CheckDecompHeaderStatus
- ▶ OSCPROG-794: Fixed known issue: Tresos error when Sleep_Management_Type is Off and ProgGM is present
- ▶ OSCPROG-758: Fixed known issue: GM application verification fails when external Flash is used
- ▶ Internal module improvement. This module version update does not affect module functionality
- ▶ OSCPROG-780: Fixed known issue: Unexpected behavior when DFI from RequestDownload request is not aligned with datatype from TransferData request
- ▶ Improved GM BootInfo Block management by adding memmap section
- ▶ Corrected data overwriting issue in decompression buffer (ARLE decompression)
- ▶ OSCPROG-697: Fixed known issue: SBA DID reading shall not contain Datatype field
- ▶ Added APIs to get access to DID F0F3/F0F6

Module version 2.0.0

2016-05-31

- ▶ OSCPROG-663: Fixed known issue: Correction of a bug where the TxConf where not handle correctly in case of diag response received from functional addressing
- ▶ Improved DiagnosticSessionControl feature by transmitting NRC78 at init for DSC02 response
- ▶ OSCPROG-681: Fixed known issue: Correction of a bug where negative response is not sent in case of failure during TD reception
- ▶ Implemented ARLE decompression support
- ▶ Implemented Write Fingerprint management
- ▶ Implemented Anti-scaning management
- ▶ Implemented FCA Autocontrol management
- ▶ Implemented Programming status management
- ▶ Added F180 DID Service Management

Module version 1.0.0

2016-02-23

- ▶ OSCPROG-555: Fixed known issue : Fix erase routine response size
- ▶ OSCPROG-631: Fixed known issue: The new API called by state machine PROG_Erase_NRC78 has been added in the product



Module version 0.1.3

2016-01-26

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 0.1.2

2016-01-14

- ▶ OSCPROG-585: Fixed known issue : Correct decompression issue where in case of high compression ratio the length can be superior to u16 max value
- ▶ OSCPROG-591: Fixed known issue: The sleep timer is now correctly reloaded when a Diag frame is received
- ▶ OSCPROG-590: Fixed known issue: Correction of a bug where the TxConf where not handle correctly in case of diag response received from functional addressing
- ▶ Implemented management of flashPage and skippage in product for implementation 2
- ▶ Improved external Flash memory access management
- ▶ OSCPROG-601: Fixed known issue: The management for reception of the same TD several times is now correctly handled
- ▶ OSCPROG-607: Fixed known issue: The compression is now correctly handle in the error case of a unfinished segmentation
- ▶ Improved watchdog feature by calling it before starting the SBL software

Module version 0.1.1

2015-11-18

- ▶ Improved External flash feature by adding the management of an offset in the external Flash addressing
- ▶ OSCPROG-462: Fixed known issue : Remove patch done for management of different P2 and P2* by session
- ▶ Improved DiagnosticSessionControl feature by adding the possibility to respond to DSC 02 received in application from the bootloader

Module version 0.1.0

2015-10-30

- ▶ Improved support of external Flash for PSI reading
- ▶ Added SBA management at bootloader startup
- ▶ Added cryptography management with verification of signer info and signature.



- ▶ Updated PSI management to be able to write it twice without erasing the Flash Memory
- ▶ OSCPROG-462: Removed Fixing of known issue : P2 and P2* values handled per sessions.

Module version 0.0.3

2015-09-08

- ▶ OSCPROG-462: Fixed known issue : P2 and P2* value to return from DCS response are now managed by the PROG layer to allow returning a different value depending on the session required
- ▶ OSCPROG-444: Fixed known issue : S3 timeout is now correctly reloaded when NetworkStatus BOOT is received
- ▶ Improved JLR implementation
- ▶ Added Page buffer management to allow reception of TD not aligned on a Flash page size

Module version 0.0.2

2015-07-20

- ▶ OSCPROG-404: Fixed known issue: The BlockSequenceCounter of TD request is now correctly returned and is not incremented if the same TD is received two time in a row
- ▶ OSCPROG-406: Fixed known issue: When a writing error happened after the full reception of a TD, a NRC_72 is now correctly sent
- ▶ OSCPROG-429: Fixed known issue: Prog shall call EB_AllSlots() when network is synchronized
- ▶ OSCPROG-421: Fixed known issue: Prog shall go to ERASE_FINISH state in case of erase error
- ▶ OSCPROG-435: Fixed known issue: Wrong NRC was transmitted in CheckProgDependencies routine
- ▶ OSCPROG-415: Fixed known issue: Interruption are disabled twice in PROG_SwitchApplicationMode
- ▶ OSCPROG-430: Fixed known issue: Fix mistake in Design
- ▶ OSCPROG-407: Fixed known issue: When RTE is received without a full TD reception, positive response shall not be sent.
- ▶ OSCPROG-446: Fixed known issue: Check on argument shall be added in PROG_EraseMemory and PROG_ActiveSBL
- ▶ OSCPROG-465: Fixed known issue: TD positive response shall only be requested if no error where detected during the TD
- ▶ OSCPROG-471: Fixed known issue: Add "volatile" to some const to avoid that the compiler optimize the code and change the needed behaviour
- ▶ OSCPROG-470: Fixed known issue: Correction of the length check in PROG_GetSegmentByAddress API
- ▶ OSCPROG-477: Fixed known issue: Correction of the compression management
- ▶ Implemented asynchronous management of Autocontrol for Implementation1/2



- ▶ Added callback to notify upper layer that the ECU will switch to Application
- ▶ Corrected SBL header management to avoid issue with compiler the two variable for SBL header has been set into the same structure

Module version 0.0.1

2015-04-29

- ▶ Module creation

2.3.5.2. New features

- ▶ Configuration for the location of CRC

Description:

The bootloader allows the configuration for the location of CRC in the check memory routine for IMPL 60.

Date

2021-08-26

- ▶ Downgrade Protection

Description:

The bootloader allows the verification of the downgrade protection for Implementation40.

Date

2021-04-19

- ▶ HSM SecureBoot

Description:

The bootloader can allow HSM based Secure Boot and Secure Application Verification for Implementation60.

Date

2021-03-11

- ▶ Bootloader Updater for Impl40 and Impl60

Description:



The bootloader updater feature allows to erase the current bootloader and flashing a new bootloader during the download sequence

Date

2020-11-18

► JLR Verification Data Structure

Description:

Verification Data Structure for Implementation3 and 4 stores the correct values of block start address, size and hash.

Date

2020-07-16

► Decryption of received data for Impl40.

Description:

The Bootloader can decrypt received data before writing them into the flash.

Date

2020-09-17

► The variable definition changed to configurable type i.e PduLengthType.

Description:

The variable definition changed from static to configurable type i.e PduLengthType which can be configured to u16 or u32.

Date

2020-09-04

► FCA standard reprogramming type

Description:

FCA reprogramming type for Implementation31 using CRC16 checksum with CRC flashed in data, and Logical block hash request to check if the app is up to the date.

Date

2020-03-27

► HSM SecureBoot



Description:

The bootloader can allow HSM based Secure Boot and Secure Application Verification for Implementation30 and 31.

Date

2020-01-20

► Dual Memory Bank

Description:

The bootloader allows the reprogramming of a second memory bank (inactive memory bank) while software is running from the first one (active memory bank).

Date

2019-11-19

► HSM Software Update

Description:

The bootloader allows the reprogramming of HSM module.

Date

2019-08-30

► Allow session transitions without the SecurityLevel reset

Description:

The bootloader can allow the non-default session transitions without SecurityLevel reset for Impl31

Date

2019-07-09

► SBI flag Callbacks

Description:

The bootloader calls customer callbacks for set/get of the SBI flag which be located by the customer.

Date

2019-07-01

► Protected Calibration



Description:

The bootloader do not revoke protected calibrations on application erasing and do a compatibility check of the protected calibrations modules on application RTE.

Date

2019-03-08

- ▶ Programming counter

Description:

The bootloader now blocks the programming after a configurable counter lock value for Impl31

Date

2018-11-16

- ▶ Add Address and Length to the data to perform the signature verification

Description:

The Bootloader can verify the signature by including address and length extracted from RD to the data to verify.

Date

2018-10-23

- ▶ Signature on compressed data

Description:

The Bootloader can verify the signature on data passed through Transfer Data requests.

Date

2018-10-23

- ▶ Block header reading

Description:

The Bootloader can read a block header and use it to validate the block content.

Date

2018-10-22

- ▶ Resumable reprogramming



Description:

The Bootloader, in case of download interrupt, can restart the download from the beginning of the current memory segment.

Date

2018-10-08

► Compressed Flash Driver

Description:

The Flash Driver is compressed in bootloader binary and is decompressed in RAM after a successful Security Access.

Date

2018-10-08

► Security Access in Application

Description:

The bootloader can accept a Securitylevel change that was triggered by a SecurityAccess in Application.

Date

2018-04-04

► Flash Erase All

Description:

The bootloader can erase all the Flash memory with only one request (without parameters).

Date

2018-04-04

► Downloading Flash Driver

Description:

The bootloader can download the flash routines using UDS requests.

Date

2018-03-13

► Data Decryption



Description:

The bootloader can provide a callback for Data Decryption.

Date

2018-02-07

- ▶ Signature computation on the fly

Description:

The bootloader can manage the verification of cryptographics signature during the software download or after the software download for impl50

Date

2017-10-06

- ▶ Several external memory with a different access modes

Description:

The bootloader can manage several memories (1 internal, 1 external, 1 RAM) with a different access mode for each one

Date

2017-10-06

- ▶ Security access with crypto callbacks

Description:

The bootloader get callbacks for symmetrical cryptography key and for asymmetrical cryptography public key

Date

2017-04-28

- ▶ Partial software checksum

Description:

The bootloader now can receive and handle the partial software checksum request for Impl40

Date

2017-02-21



► Programming counter

Description:

The bootloader now blocks the programming after a configurable counter lock value for Impl40 and Impl50

Date

2017-02-20

► Multiple buffer management

Description:

The bootloader now handles multiple buffers, which allow to receive transfer data requests while writing the previous ones received.

Date

2017-01-07

► Add Implementation 60 features

Description:

The bootloader is now able to run with features for implementation 60

Date

2017-01-05

► Coherency check

Description:

The bootloader now handles the coherency check of the blocks existing in the ECU's flash memory.

Date

2016-11-30

► Logical block management and Download by logical address

Description:

The bootloader now handles logical blocks and the download by logical address.

Date

2016-06-27

► CheckMemory routine design



Description:

Management of the CheckMemory routine for CRC verification. Used for the validation of the downloaded application/calibration.

Date

2016-06-07

- ▶ Reset cause and DSC/ER response management

Description:

A way of knowing the reset cause is needed as well as if a response should be sent after reset and to which service.

Date

2016-06-06

- ▶ Programming pre-condition check

Description:

Before starting the programming the pre-conditions are checked.

Date

2016-06-02

- ▶ Add F180 DID Service Management

Description:

The bootloader now handle the Boot Information (Number of Modules, Boot Id, Part Number, DLS)

Date

2016-05-26

- ▶ Add ARLE decompression support

Description:

The bootloader now handle ARLE decompression using Autosar decompression module

Date

2016-04-18

- ▶ Programming status management



Description:

Programming status available at the end of each programming cycle.

Date

2016-03-02

- ▶ FCA Autocontrol management

Description:

Autocontrol of the programmed application/calibration validity (for FCA).

Date

2016-03-02

- ▶ Add of Anti-scaning management

Description:

In the case of an unlock request received by the ECU a key is needed. If this key is wrong a retry is accepted. If the key received by the ECU is wrong the second time also, the ECU will be locked for a configurable period of time.

Date

2016-03-02

- ▶ Write Fingerprint management

Description:

Writing of programmed application, calibration or bootloader fingerprint in ECU memory.

Date

2016-03-02

- ▶ Add management of flashPage and skippage in product for implementation 2

Description:

The bootloader now handle sending the two feature flashPage and skippage for the implementation 2

Date

2015-12-15

- ▶ Add the possibility to respond to DSC 02 received in application from the bootloader



Description:

The bootloader now handle sending the response from a reprogramming request received from the application layer

Date

2015-11-17

- ▶ Add the management of an offset in the external Flash addressing

Description:

The bootloader is now able to manage the external Flash that shall be addressed by a tool in with an address offset.

Date

2015-11-13

- ▶ Added new feature for full cryptography management

Description:

The bootloader now manage the validation of the signed header (signer + signature) and the hash calculation at the of the reprogramming sequence

Date

2015-09-28

- ▶ Added new feature for SBA check at bootloader startup

Description:

The bootloader now manage the SBA ticket validation at startup

Date

2015-09-17

2.3.5.3. EB-specific enhancements

This module is not part of the AUTOSAR specification.

2.3.5.4. Deviations

This module is not part of the AUTOSAR specification.

2.3.5.5. Limitations

This chapter lists the limitations of the module. Refer to the module references chapter *Integration notes*, subsection *Integration requirements* for requirements on integrating this module.

- ▶ The already erase feature is currently not available

Description:

The feature already erase that allow not to erase the memory if it is already erase is not available.

Rationale:

The memory erasing will be done in any case even if it is already erased

Requirements:

SwAD-ARCH-0218-1

2.3.5.6. Open-source software

osc_PROG module does not use open-source software.

2.3.6. ProgOEMInd module release notes

- ▶ Module version: 1.12.0.BL3_B470986
- ▶ Supplier: Elektrobit Automotive GmbH

2.3.6.1. Change log

This chapter lists the changes between different versions.

Module version 1.12.0

2021-10-25

- ▶ Internal module improvement. This module version update does not affect module functionality



Module version 1.11.0

2021-07-21

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.10.0

2021-04-21

- ▶ Implemented HSM SecureBoot
- ▶ Implemented Bootloader Updater with HSM SecureBoot

Module version 1.9.0

2021-02-25

- ▶ OSCPROGOEMIND-114 : Implemented Decryption for EB Essentials.
- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.8.0

2020-11-27

- ▶ OSCPROGOEMIND-108 : Implemented Bootloader Updater

Module version 1.4.0

2020-03-27

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.3.9

2020-02-03

- ▶ Internal module improvement. This module version update does not affect module functionality
- ▶ New feature: Compressed flash driver

Module version 1.3.8

2019-12-09

- ▶ New feature: Ethernet Sync after reset



Module version 1.3.7

2019-09-26

- ▶ OSCPROGOEMIND-81: Fixed known issue: [OEMInd] Unexpected call to callback PROG_CustomSetDownloadVerificationSuccess() when it returns PROG_E_NOT_OK

Module version 1.3.6

2019-07-23

- ▶ OSCPROG-1821: Fixed known issue: [OEMInd] ECU does not return to Lock state when GetSeed is not allowed (NRC_37)
- ▶ OSCPROGOEMIND-73: Fixed known issue: [OEMInd] Continuous pending response when some transitions are requested

Module version 1.3.5

2019-03-22

Module version 1.3.4

2018-10-25

- ▶ Internal module improvement. This module version update does not affect module functionality
- ▶ OSCPROG-1388: Fixed known issue: ECU is in infinite Response pending when receiving Check Memory Request after Check Programming dependencie
- ▶ OSCPROG-1388: Fixed known issue: Correct bad behaviour when receiving successive CheckMemory requests
- ▶ Internal module improvement. This module version update does not affect module functionality
- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.3.3

2018-06-14

Module version 1.3.2

2018-06-11



- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.3.1

2018-03-22

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.3.0

2017-12-18

- ▶ New feature: Add signature check support for CheckMemory

Module version 1.2.1

2017-10-26

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.2.0

2017-10-13

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.1.0

2017-09-08

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.0.2

2017-05-11

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.0.1

2017-04-03

- ▶ Improved sequence management



Module version 1.0.0

2016-11-24

- ▶ Implemented Hsm for OEM Independent

2.3.6.2. New features

- ▶ Bootloader Updater

Description:

The bootloader updater feature allows to erase the current bootloader and flashing a new bootloader during the download sequence

Date

2020-11-20

- ▶ Add Hsm for OEM Independent

2.3.6.3. EB-specific enhancements

This module is not part of the AUTOSAR specification.

2.3.6.4. Deviations

This module is not part of the AUTOSAR specification.

2.3.6.5. Limitations

This chapter lists the limitations of the module. Refer to the module references chapter *Integration notes*, subsection *Integration requirements* for requirements on integrating this module.

- ▶ For this module no limitations are known.

2.3.6.6. Open-source software

osc_Prog_OEMInd module does not use open-source software.



2.3.7. SA module release notes

- ▶ Module version: 1.16.0.BL3_B470986
- ▶ Supplier: Elektrobit Automotive GmbH

2.3.7.1. Change log

This chapter lists the changes between different versions.

Module version 1.16.0

2021-10-25

- ▶ Internal module improvement. This module version update does not affect module functionality.

Module version 1.15.0

2021-07-23

- ▶ Internal module improvement for IMPL90 and IMPL91
- ▶ Internal module improvement for IMPL90 and IMPL91
- ▶ Internal module improvement Merge IMPL90 and IMPL91 to trunk
- ▶ Implemented Crypto 4.3 support.

Module version 1.13.0

2021-04-21

- ▶ Internal module improvement for IMPL3 IMPL4 IMPL10 and IMPL11
- ▶ Internal module improvement for IMPL3 IMPL4 IMPL10 and IMPL11
- ▶ Internal module improvement. This module version update does not affect module functionality.

Module version 1.12.0

2020-11-27

- ▶ Internal module improvement. Merge IMPL3 and IMPL4 to trunk.
- ▶ Internal module improvement. Merge IMPL3 and IMPL4 to trunk.



Module version 1.10.0

2020-09-22

- ▶ Changed the variable definition from static to configurable type i.e PduLengthType which can be configured to u16 or u32.
- ▶ Implemented SA Delay timer at startup for Implementation 10 and Implementation 11.
- ▶ Implemented Limit Client RequestSeed for Implementation 10 and Implementation 11.
- ▶ Implemented SA Secured Key Programming Status read for Implementation 10 and Implementation 11.
- ▶ Implemented SA Generation2 Authentication Method (0x0001) feature for Implementation 10 and Implementation 11.

Module version 1.9.0

2020-07-15

- ▶ Implemented initialization of SA status at startup needed for CheckReprogramming Preconditions Request Handling in Impl40.

Module version 1.8.0

2020-03-27

- ▶ Corrected SA_GenerateRandomNumber API . Remove extra parentheses and update the check of SA CSM state under verify_signature configuration
- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.6.3

2019-11-29

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.6.2

2019-09-25

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.6.1

2019-07-17



- ▶ Implemented the feature "Standard OEM security algorithm" for profile 10
- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.6.0

2019-04-01

Module version 1.5.0

2019-03-21

- ▶ Implemented Static Seed management
- ▶ Implemented support to crypto ASR 4.3 stack via the Demo_CSM_Wrapper

Module version 1.4.4

2018-10-25

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.4.3

2018-07-24

- ▶ Added public key exponent support

Module version 1.4.2

2018-07-20

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.4.1

2018-06-14

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.4.0

2018-03-22



- ▶ Improved callback using to allow SA to use its own public key

Module version 1.3.0

2017-12-18

- ▶ Removed Csm initialization (managed by the BM plugin)

Module version 1.2.1

2017-10-26

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.2.0

2017-10-13

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.1.1

2017-08-02

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.1.0

2017-07-11

- ▶ OSCSA-182: Fixed known issues: No response to two consecutive seed requests
- ▶ OSCSA-183: Fixed known issues: No response to seed request after a failed unlocking attempt

Module version 1.0.7

2017-06-12

- ▶ OSCSA-173: Fixed known issues: Correct bug SecurityAccess anti-scanning timer cannot be set to value higher than 65s

Module version 1.0.6

2017-05-11



- ▶ New feature: Added crypto support with random seed generation and signature verification

Module version 1.0.5

2017-04-03

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.0.4

2016-12-16

- ▶ Improved tresos parameter for antiscanning now only available if antiscanning is enable
- ▶ Correct Seed generation in order to take into account Seed Size

Module version 1.0.3

2016-10-10

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.0.2

2016-08-12

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 1.0.1

2016-05-31

- ▶ New feature: Adding of Anti-scaning management

Module version 1.0.0

2016-03-29

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 0.0.3

2016-01-26



- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 0.0.2

2015-07-16

- ▶ OSCSA-71: Fixed known issues: Correct bug in get decompress data management where the decompress status where not correctly send back
- ▶ Improvement: remove input decompress buffer and use diag one instead

Module version 0.0.1

2015-04-29

- ▶ module creation

2.3.7.2. New features

- ▶ The variable definition changed to configurable type i.e PduLengthType

Description:

The variable definition changed from static to configurable type i.e PduLengthType which can be configured to u16 or u32.

Date

2020-09-04

- ▶ Added static seed support

Description:

In the case of consecutive GetSeed requests without CompareKey request (the first generated seed was not used), the response to the second GetSeed will contain the seed generated on the first seed request.

Date

2019-02-26

- ▶ Added crypto support

Description:

Added random seed generation and signature verification



Date

2017-05-05

- ▶ Adding of Anti-scaning management

Description:

In the case of an unlock request received by the ECU a key is needed. If this key is wrong a retry is accepted. If the key received by the ECU is wrong the second time also, the ECU will be locked for a configurable period of time.

Date

2016-05-02

2.3.7.3. EB-specific enhancements

This module is not part of the AUTOSAR specification.

2.3.7.4. Deviations

This module is not part of the AUTOSAR specification.

2.3.7.5. Limitations

This chapter lists the limitations of the module. Refer to the module references chapter *Integration notes*, subsection *Integration requirements* for requirements on integrating this module.

- ▶ For this module no limitations are known.

2.3.7.6. Open-source software

osc_SA module does not use open-source software.

2.3.8. Uds module release notes

- ▶ Module version: 3.17.0.BL3_B470986



- ▶ Supplier: Elektrobit Automotive GmbH

2.3.8.1. Change log

This chapter lists the changes between different versions.

Module version 3.17.0

2021-10-25

Module version 3.15.0

2030-07-02

- ▶ Internal module improvement Merge IMPL90 and IMPL91 to trunk
- ▶ Implemented API to check if the last response was response pending

Module version 3.14.0

2021-06-06

Module version 3.13.255

2031-12-31

- ▶ Implemented DSC02 wait at startup feature for Impl90
- ▶ Internal module improvement Merge IMPL90 and IMPL91 to trunk
- ▶ Implemented API to check if the last response was response pending

Module version 3.13.0

2021-04-21

- ▶ OSCUDS-478: Fixed known issue: NRC14 is not implemented in the bootloader

Module version 3.12.0

2020-11-27



- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 3.11.0

2020-09-18

- ▶ Internal module improvement. This module version update does not affect module functionality
- ▶ The variable definition changed from static to configurable type i.e PduLengthType which can be configured to u16 or u32.

Module version 3.9.2

2019-11-29

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 3.9.1

2019-07-24

- ▶ Added the callback to notify positive response to be sent.

Module version 3.9.0

2019-03-21

- ▶ OSCUDS-419: Fixed known issue: [ISO] Minimum size of the RequestDownload is not verified
- ▶ Added the callback to manage supplier services.
- ▶ Improved handling of response pending using hardware timer(external) with the aid of configuration parameter

Module version 3.8.4

2018-10-25

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 3.8.3

2018-06-15

- ▶ Internal module improvement. This module version update does not affect module functionality



Module version 3.8.2

2018-03-22

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 3.8.1

2017-12-18

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 3.8.0

2017-10-16

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 3.7.0

2017-09-11

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 3.6.0

2017-04-03

- ▶ Improved response pending management to be usable with BIPduR module and to be manage separately from the basic scheduling
- ▶ OSCUDS-381: S3 timeout is no more reloaded at each UDS request.
- ▶ OSCUDS-383: Fixed know issue: Minimal length for WDBI / IOCBI is now 4

Module version 3.5.5

2016-12-16

- ▶ Updated S3 timer management adding a new API to stop this timer.

Module version 3.5.4

2016-10-10

- ▶ Internal module improvement. This module version update does not affect module functionality



Module version 3.5.3

2016-08-12

- ▶ OSCUDS-328: Fixed known issue: UDS compilation error fix if Security check is activated an no ServiceDID or RoutineControl configured.

Module version 3.5.2

2016-05-30

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 3.5.1

2016-03-29

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 3.5.0

2015-11-18

- ▶ Improved of the NRC_78 timings management. It is now handled per sessions / only ISO variant

Module version 3.4.3

2015-07-15

- ▶ OSCUDS-286: [ISO] No NRC_7E or NRC_7F negative response to functionaly adressed request (ISO 14229-1:2013(E))

Module version 3.4.2

2015-04-29

- ▶ Improved of the NRC_78 timings management. It is now handled per sessions / only ISO variant

Module version 3.4.1

2015-04-07

- ▶ OSCUDS-259: [RSA_MIXED] Useless suppressPosRspMsgIndicationBit management removed



- ▶ OSCUDS-248: Fixed known issue: NRC_78 supported for DSC service

Module version 3.4.0

2015-03-06

- ▶ OSCUDS-233: [PSA_CAN_LS/RSA_MIXED] Fixed known issue: Security check callback (UDS_SecurityCheck) missing for RoutineControl service (NRC33)
- ▶ OSCUDS-261: Improvement: Security check feature managed by UDS

Module version 3.3.1

2014-11-06

- ▶ OSCUDS-239: Fixed known issue: HIS_STMT_Max metric error fixed (UDS_FilteringCfg2).

Module version 3.3.0

2014-10-01

- ▶ OSCUDS-237: Fixed known issue: Tresos fields "UDS_MAX_DID_MULTI_RDBI" range had been updated to prevent compiler warning (range = [0 ; 1000]).

Module version 3.2.0

2014-08-18

- ▶ OSCUDS-233: Fixed known issue: Security check callback (UDS_SecurityCheck) missing for some services (NRC33)

Module version 3.1.4

2014-06-05

- ▶ OSCUDS-220: Added Specific NRC definition in UDS_Types.h
- ▶ OSCUDS-229: [PSA_CAN_LS] Fixed known issue: Unpredictable NRC returned on RoutineControl request too short

Module version 3.1.3

2014-04-29

- ▶ Internal module improvement. This module version update does not affect module functionality



Module version 3.1.2

2014-04-24

- ▶ OSCUDS-207: [PSA_CAN_LS] Fixed known issue: Management of suppressPosRspMsgIndicationBit if response pending (NRC78) is activated
- ▶ OSCUDS-142: [PSA_CAN_LS] Add of a new API UDS_ForceRoutineControlStatus to force RoutineControl Status
- ▶ OSCUDS-220: Added Multi DID for RDBI UDS service

Module version 3.1.1

2014-02-13

- ▶ Internal module improvement. This module version update does not affect module functionality

Module version 3.1.0

2014-01-27

- ▶ OSCUDS-203: [ISO] Fixed known issue: Management of suppressPosRspMsgIndicationBit if response pending (NRC78) is activated
- ▶ OSCUDS-205: Update APP_UdsSessionStatusInd and UDS_SessionStatusInd callbacks adding new arguments (new session, old session and the reason of the changing session)

Module version 3.0.2

2013-11-12

- ▶ OSCUDS-193: Fixed known issue: Number max of DID allowed must be configurable in Tresos

Module version 3.0.1

2013-10-02

- ▶ OSCUDS-190: Add of a new API to get the current running session (UDS_GetCurrentSession)

Module version 3.0.0

2013-06-26

- ▶ OSCUDS-173: Variants management: The management of module multi-variants has been improved for ISO, PSA_CAN_LS and RSA_MIXED variants. Only the ordered variant is now visible.



- ▶ OSCUDS-178: Fixed known issue: Code may create warning if no OBD service configured.
- ▶ OSCUDS-180: Design document has been updated.
- ▶ OSCUDS-181: Fixed known issue: undefined reference to _UDS_LongRequestResponseInd, only for PSA_CAN_LS and RSA_MIXED variants.
- ▶ OSCUDS-183: Improvement: The variant name appears clearly into the header of all source files.

2.3.8.2. New features

- ▶ The variable definition changed to configurable type i.e PduLengthType.

Description:

The variable definition changed from static to configurable type i.e PduLengthType which can be configured to u16 or u32.

Date

2020-09-04

- ▶ [ISO]New handling of NRC_78 timings

Description:

The timings for P2/P2* are now handled per sessions. This impact a rework on the UDS plugin design. Introduction of a generic adjust value for P2/P2* as well. This is subtracted from P2/P2* timings configured on sessions.

Date

2015-10-21

- ▶ [ALL]Added security check management in UDS

Description:

Security check is now managed by UDS. The security level is configurable for each services configured in Service, Service_DID or Routine_Control. Careful: a callback to get the current security level shall be provided to UDS. Please refer to Userguide for details.

Date

2015-02-11

- ▶ [ALL]Added Multi DID for RDBI UDS service

Description:



Multi DID for RDBI UDS service is now available for all variants. Careful: *pulLen is not the received length but the current buffer size consumed. Please refer to Integration Manual for details.

Date

2014-04-17

- ▶ [PSA_CAN_LS] Added new API to force the RoutineControl status for specific RoutineIdentifier (UDS_-ForceRoutineControlStatus)

Description:

API can be used by application if you need to reset the status or RoutineControl after reset or session switching for example

Date

2014-04-07

- ▶ Added new API to get the current running session (UDS_GetCurrentSession)

Description:

Added new API to get the current running session (UDS_GetCurrentSession)

Date

2013-10-02

2.3.8.3. EB-specific enhancements

This module is not part of the AUTOSAR specification.

2.3.8.4. Deviations

This module is not part of the AUTOSAR specification.

2.3.8.5. Limitations

This chapter lists the limitations of the module. Refer to the module references chapter *Integration notes*, subsection *Integration requirements* for requirements on integrating this module.

- ▶ Usage of NRC_78 on RDBI request when MULIT-DID is activated



Description:

If MULTI-DID is activated and some RDBI requests use NRC_78, some DID informations will be missing in UDS answer. For example: * if DID#2 uses NRC_78 * tester send the request RDBI DID#1 DID#2 DID#3 The ECU will take into account only the request for DID#1 and DID#2; DID#3 request is lost.

2.3.8.6. Open-source software

osc_UDS module does not use open-source software.



3. BL for Essentials user guide

3.1. Overview

This chapter provides user guide for the BL for Essentials modules.

3.2. EB Tresos Bootloader user's guide for OEMInd

3.2.1. Introduction

This document present the different integration requirements of the EB tresos bootloader product

Three types of requirements will be described:

- ▶ Interface requirements between Bootloader and integration code
- ▶ Configuration requirement
- ▶ General performance requirement

3.2.2. Integration steps

In order to perform the Bootloader integration, it is adviced to follow the following steps:

1. Read EB tresos Bootloader Documentation:
 - ▶ Bootloader implementation matrixes:
 - identify for every OEM requirements if the requirement is fully/partially or not implemented in EB tresos Bootloader software module
 - allow integrator identifying the remaining work to do to have a full OEM compliant Bootloader
 - ▶ EB tresos Bootloader Generic Documentation: Bootloader environment documentation (Compiler,Base,Make, platform,demo)
 - ▶ EB tresos Bootloader for 'OEM' documentation (This document): Public Api definition, user guide
2. Run the delivered Demo software on evaluation board or adapt it to make it runs on project specific board (e.g real ECU)



3. Build the bootloader for your ECU/project:

- ▶ Replace or update, according to project/hardware needs, code and configuration from demo.
- ▶ Update Linkerfile according to your memory mapping
- ▶ Update Modules configuration (see advice after)
- ▶ Implement callbacks present in template directory (Project/hardware dependent)
- ▶ Check in implementation matrix requirements that are not implemented in Bootloader software modules (Project/Hardware specific) and implement them.

Here is some information about integration workspace folder structure:

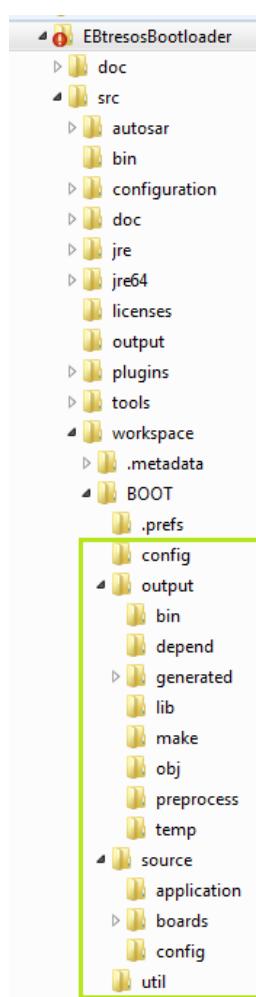


Figure 3.1.

config: plugins configurations file (see advice after)

output: output files of the build process:

- ▶ **bin:** Bootloader binary file
- ▶ **generated:** all files generated by Tresos (shall not be modified)
 - **include:** generated include files
 - **make:** generated makefile using for Bootloader software build
 - **src:** generated source files
 - **templates:** generated template files that shall be completed by integrator. The files from this folder are not compiled and shall be copied in application folder before being modified and compiled.

source: integration source code (shall be modified, completed by integrator)

- ▶ **application:** integration specific files that shall be implemented by integrator.
- ▶ **board:** hardware specific Api, startup code and linker files (shall be modified, completed by integrator)
- ▶ **config:** Autosar Compiler_Cfg.h and MemMap.h (shall be modified, completed by integrator)

util: workspace build scripts

Table 3.1.



Here is some advice for configuration management:

- ▶ Communication stack modules: Configure according to OEM requirements (use EcuExtract if provided by the OEM)
- ▶ UDS:
 - Update with additional DIDs for WDBI and RDBI service
 - Update with additional routines
 - Add supplier specific session (if needed)
- ▶ BM: use demo configuration
- ▶ CSM/CRY/CPL/CAL (if ordered): use demo configuration
- ▶ SA (if required): use demo configuration
- ▶ EB/BIPduR: use demo configuration and update it if additional Pdus are required
- ▶ APP: use demo configuration
- ▶ FLASH / ReProgMemM : use demo configuration
 - Only adapt sector configuration if ECU micro-controller memory size is different from delivered micro-controller configuration.
- ▶ PROG:
 - Update project specific parameters in General container
 - Configure Blocks according to application mapping, if OEM required logical block addressing for erasing/writing
 - Configure Segments according to application mapping

3.2.3. General overview

Important note: In order to run qualification process on a target, a startup code is required. This startup code is a sample created by Elektrobit or directly received from the hardware manufacturer. This is for EB internal testing purposes only. The startup code provided within Elektrobit release (startup assembly code and C implementation in board.c/h) is not tested nor qualified for series production use. The qualification statement does not refer to the startup code. The user shall not be entitled to use the startup code in connection with any series production. It shall be replaced by an operating system or a project and ECU specific startup code, which is qualified for mass production.

3.2.3.1. Layer overview

The following pictures show you the EB tresos Bootloader architecture.

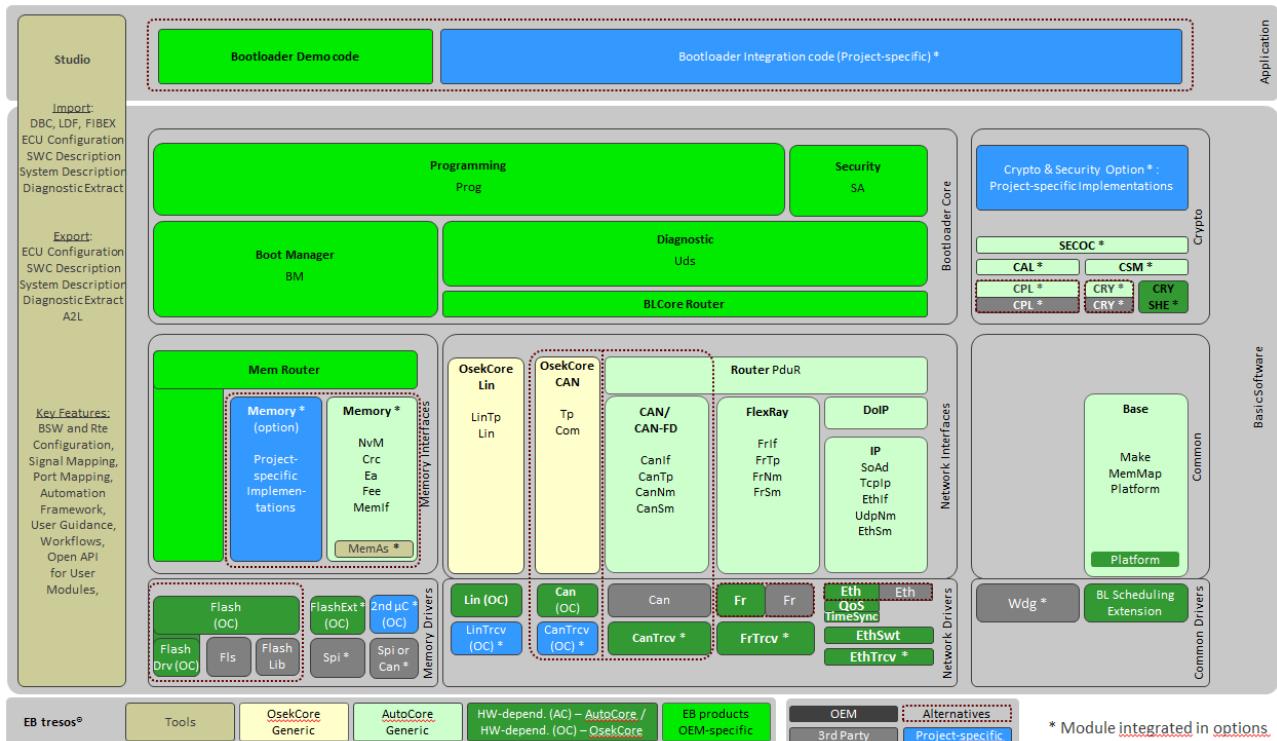


Figure 3.2. Bootloader architecture

The following software modules constitute the EB tresos Bootloader:

- ▶ BM: BootManager that is used at startup to route the software to bootloader or application software
- ▶ BLCore Router: Software router between all the communication stack and the bootloader core. Depending on the communication stack this router can be two different module:
 - ▶ If the communication stack is a OsekCore Can based the module BLCore Router will be managed by module EB
 - ▶ If the communication stack is an Autosar based (any network) the module BLCore Router will be managed by module BIPduR
- ▶ UDS : UDS ISO compliant for diagnostic service management
- ▶ PROG : Bootloader core software divided in two parts:
 - ▶ PROG<OEM> containing the bootloader state machine
 - ▶ PROG containing the programming features
- ▶ SA (optional) : Manage Security Access UDS request and the LZSS decompression algorithm
- ▶ FLASH or ReProgMemM: Flash driver called by the bootloader to write erase and read from Flash memory

The communication stack (CAN, FlexRay, Ethernet) shall be configured by the integrator except if the configuration package had been purchase.



If Autosar communication stack is used, configuration can be partially done by importing an ECU extract.

3.2.3.2. plugin files structure

Each plugin is composed with two kinds of files:

- ▶ Core files
- ▶ Generated files

The core files don't contain any configuration parameters, they are stored in the base folder "src" for the .c files and "include" for header files. These files don't have to be modified by the integrator.

The generated files are template source files completed by the EB tresos Studio tool, taking into account the module configuration. They are stored in "generate/src" for the .c files and "generate/include" for header files. These files don't have to be modified by the integrator.

The generated files to compile or include after generation are in the workspace of the EB tresos Studio of the project

NOTE: Some plugins doesn't contained any core files (e.g. EB) all the files are generated from EB tresos Studio.

The template files are template source files which have to be moved from the "generated/templates" folder to the "src/application" folder in the workspace of the EB tresos Studio of the project. They are stored in "generate/template" for the .c files and header files. These files can be modified by the integrator.

3.2.4. Response Pending scheduling

This chapter defines integration to STM(System) Timer Intialisation and interrupt configuration.

3.2.4.1. STM Timer ISR

The STM timer needs to be initialised and Interrupt generation on Timer Compare is enabled. Compare register to be loaded with value which corresponds to 1 millisecond based on the MCU clock.

If stm timer interrupt shall be used to schedule the NRC78 sending, the API UDS_ResponsePending_Tim-CntManage shall be used in timer ISR and parameter TIMER_RESPONSE_PENDING_CHECK shall be checked(enabled) in uds configuration(tresos). The use of the timer interrupt ensure to not miss sending of NRC 78 at P2 and P2 star time.



3.2.5. Can Integration

This chapter defines the integration constraints when using the CAN driver.

3.2.5.1. CAN RX/TX interrupt priority levels

Part of the services provided by the bootloader software is triggered upon the external events, such as the reception of certain CAN messages and the transmission of certain CAN messages. At the integration phase, if the CAN driver uses the interrupts for sending/receiving CAN messages, user shall configure CAN TX interrupt with a higher priority than that of CAN RX interrupt. So if both CAN TX interrupt flag and CAN RX interrupt flag are set, the ISR of CAN TX will be served first.

3.2.6. Crypto ASR v4.3 or higher version stack Integration

This chapter defines the integration constraints when using the Crypto ASR v4.3 stack.

3.2.6.1. General Guidelines for the integration of Crypto ASR v4.3 stack

This section describes the general guidelines for integrating the Crypto ASR v4.3 or higher stack for performing the crypto operations.

The Crypto ASR v4.3 or higher stack version consists of Csm, Crylf and one or more Crypto driver modules.

The integrator shall ensure to integrate and configure all the modules of the Cryptostack as per the user guidelines defined in the CryptoStack documentation.

All the Csm jobs that shall be configured for the Prog modules shall be Asynchronous type.

Csm mainfunction and the Crypto driver main function shall be configured as background tasks or higher periodicity tasks for the faster crypto operation.

Note: The demo-Csm wrapper for the integration of the Crypto ASR v4.3 or higher stack is no longer supported.

3.2.7. ReProgMemM Integration

This chapter defines the integration steps to perform when switching from the Flash wrapper to the ReProg-MemM module.



3.2.7.1. Mandatory files for the integration

- ▶ Replace Flash configuration file with the ReProgMemM configuration file and update the preferences file
- ▶ Update the Prog configuration file
- ▶ Call the ReProgMemM_MainFunction in your cyclic task. The memory operations are performed during the main function calls.
- ▶ Update the ReProgMemM callbacks in ReProgMemM_Interfaces.c, ReProgMemM_Interfaces.h and remove Flash_Cbk.c
- ▶ Add the DualBank callbacks file if the feature is enabled
- ▶ Dcm stubs are required

3.2.7.2. Constraints

No burst mode supported yet. The parameter REPROGMEMM_SIZE_PER_TURN should be the lowest possible.

Do not configure a memory if it is not going to be used in the upper layer (PROG module)

If the copy of the driver routines shall be performed by the ReProgMemM module, please, in the linker file, add the 4 following symbols:

- ▶ `__Flash_API_ROM_SECTION_START_ADDR`: must be mapped to the beginning of the flash driver area in ROM section
- ▶ `__Flash_API_ROM_SECTION_END_ADDR`: must be mapped to the end of the flash driver area in ROM section
- ▶ `__Flash_API_RAM_SECTION_START_ADDR`: must be mapped to the beginning of the flash driver area in RAM section
- ▶ `__Flash_API_RAM_SECTION_END_ADDR`: must be mapped to the end of the flash driver area in RAM section

3.2.7.3. Integration with asc_MemAcc module

The asc_MemAcc module shall be used if concurrent access to Flash banks can occur (ex : OTA client and Application).

A wrapper is necessary in order to integrate the asc_MemAcc module with a synchronous Flash driver.

Both plugins shall be integrated in addition to the ReProgMemM module.



The wrapper then the MemAcc mainfunctions APIs shall be called before the ReProgMemM mainfunction API.

3.2.8. Configuration

This chapter defines configuration parameters that are advised for the correct behaviour of the bootloader.

3.2.8.1. Notes

Configuration parameters that are not described in this document shall be adapt according to the project needs, they are not mandatory and their use is depending of the project integration. You can refer to the parameter description in EB tresos Studio.

Some software modules contains OEM dedicated configuration parameters, these parameters won't be available if you are not integrating a Bootloader for this OEM. Please ignore these parameters in such case.

Some configuration parameters can be automatically disabled depending of your configuration/integration and appear "grayed" in EB tresos configuration. Their value will be unused so please ignore these parameters in such case.

3.2.8.2. Prog Configuration

3.2.8.2.1. General

Id:	OSC-INTMAN-BOOTLOADER-0071
Version:	1
Description:	The integrator shall ensure that "Max_Bytes_in_TD" field is filled with a value multiple of "Min_Value_To_Flash" field value + 2.

Id:	OSC-INTMAN-BOOTLOADER-0124
Version:	1
Description:	<p>The integrator shall ensure that if Reset after S3 Timeout in Programming Session is required or not</p> <p>► The "Trigger Reset after S3 Timeout in Programming Session" checkbox is ticked/enabled</p>



Id:	OSC-INTMAN-BOOTLOADER-0125
Version:	1
Description:	<p>The integrator shall ensure that if Reset after S3 Timeout in Extended Session is required or not</p> <ul style="list-style-type: none"> ▶ The "Trigger Reset after S3 Timeout in Extended Session" checkbox is ticked/enabled

Id:	OSC-INTMAN-BOOTLOADER-0123
Version:	1
Description:	<p>The integrator shall ensure that if Dual Memory Bank feature shall be used:</p> <ul style="list-style-type: none"> ▶ The "Dual Memory Bank Used" checkbox is ticked/enabled

3.2.8.2.1.1. EraseState Callbacks

3.2.8.2.1.1.1. PROG_CustomIsFirstProgramming Callback

This callback is called on reception of Erase routine

It allows skipping the erasing if the ECU Flash has never been programmed and so is already fully erased. It allows saving time during ECU first download.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0114
Version:	2
Description:	<p>The integrator shall implement in PROG_CustomIsFirstProgramming callback software providing information if Flash was never programmed before and that erase shall be skipped. The callback should return PROG_TRUE if the Flash was never programmed and PROG_FALSE otherwise.</p>

3.2.8.2.1.1.2. PROG_CustomDownloadNotification Callback

This callback is called on reception of RequestDownload routine

It allows the update of the first programming flag to point that the ECU Flash has already been programmed.



Optionally, other treatments related to the starting of the Flash memory programming can be done by the integrator in this callback.

It could also be used to reset the application validity flag

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0159
Version:	1
Description:	The integrator shall implement in PROG_CustomDownloadNotification callback software by updating the information that Flash was already programmed before. This indicates that, for any other further programming, the Flash memory should be erased.

Id:	OSC-INTMAN-OEMIND-BOOTLOADER-0122
Version:	1
Description:	The integrator shall ensure to set the correct value for the Prog_RC_CrcOffset eg.: set this value as 0 if the CRC position in the UDS request will be right after the Routine Identifier in the RoutineControl Checkmemory service .

Id:	OSC-INTMAN-OEMIND-BOOTLOADER-0020
Version:	1
Description:	The integrator shall ensure that "Transmit_Response_Before_Reset" field is set.

Id:	OSC-INTMAN-OEMIND-BOOTLOADER-0022
Version:	1
Description:	The integrator shall ensure that "Transmit_Nrc78_On_SecurityAccess" field is set if the Security Access seed/key generation process is long (more than P2 time).

3.2.8.2.2. Crypto ASR 4.3 Stack generic configuration

Id:	OSC-INTMAN-CRYPTOASR43-BOOTLOADER-0001
Version:	1
Description:	The integrator shall ensure to enable Allow2Cancel_OngoingJobs to cancel the previously started or interrupted or any ongoing jobs, needed only when there are no concurrent jobs configured to run in parallel with this job operation in the



Bootloader.(for eg.: during the simultaneous Signature verification and Decryption of the downloaded data, job cancellation of this job shall be disabled). Note:

- This requirement is applicable only for the Crypto ASR 4.3 integration.

Id:	OSC-INTMAN-CRYPTOASR43-BOOTLOADER-0002
Version:	1
Description:	<p>(Optional)The integrator shall ensure to set the Block Slicing- BS_size parameter value > 0 to enable the feature for the corresponding job. This feature shall perform the slicing of large amount of data (to be processed by crypto stack) into smaller chunks so that the time for the asynchronous processing per call at the Crypto driver is reduced. Note:</p> <ul style="list-style-type: none"> ➤ This requirement is applicable only for the Crypto ASR 4.3 integration. ➤ It is recommended to call PROG_CsmManage() with high periodicity if this feature is supported. ➤ Disclaimer: Setting here a very small value shall significantly increase the overall crypto operation time

Id:	OSC-INTMAN-CRYPTOASR43-BOOTLOADER-0003
Version:	1
Description:	<p>The integrator shall ensure to enable Allow2SetCryptoKey to set the key for any Csm job that needs a key to be passed to the Crypto stack from the Bootlaoder, this is needed especially when the key is not stored in the Crypto Driver or the Hardware Security Module prior to the start of the Crypto operation. Note:</p> <ul style="list-style-type: none"> ➤ This requirement is applicable only for the Crypto ASR 4.3 integration.

Id:	OSC-INTMAN-CRYPTOASR43-BOOTLOADER-0004
Version:	1
Description:	<p>(Optional)The integrator shall ensure to enable Allow2CustomCsmStartPreprocess the Custom preprocessing feature only when project specific or the algorithm specific operations shall be performed before the start of the current crypto operation. For eg.: implementing the DER encoding of the plain key for performing RSASSA_PSS signature verification. Note:</p> <ul style="list-style-type: none"> ➤ This requirement is applicable only for the Crypto ASR 4.3 integration.

Id:	OSC-INTMAN-CRYPTOASR43-BOOTLOADER-0005
------------	--



Version:	1
Description:	<p>If enabled to set the key, the integrator shall ensure to configure the Crypto driver Key element parameter-CryptoKeyElementSize and access type for the corresponding Crypto key. Note:</p> <ul style="list-style-type: none"> ▶ This requirement is applicable only for the Crypto ASR 4.3 integration. ▶ Hint: normally the size must include total key size including the all the subelements of the Key. ▶ For more info refer to the integration guide of the integrated Crypto driver.

Id:	OSC-INTMAN-CRYPTOASR43-BOOTLOADER-0006
Version:	1
Description:	<p>The integrator shall ensure to enable AllowStreamStart configuration parameter when the Start operation for the Crypto operation cannot be completed with in a single Asynchronous main function call (eg.: for the Signature verification using EDDSA algo.). Note:</p> <ul style="list-style-type: none"> ▶ This requirement is applicable only for the Crypto ASR 4.3 integration. ▶ This feature can be enabled only if the Crypto driver supports the StreamStart mode of operation.

3.2.8.2.3. Download data Signature verification

Id:	OSC-INTMAN-SIGNATUREVERIFICATION-BOOTLOADER-0001
Version:	1
Description:	<p>The integrator shall ensure to select the appropriate configured Csm Job for the ProgCsmSignatureVerifyConfigId parameter under the SignatureVerification section.</p>

Id:	OSC-INTMAN-SIGNATUREVERIFICATION-BOOTLOADER-0002
Version:	1
Description:	<p>The integrator shall ensure to enable SignFinSendFRP to send response pending before the Crypto operation with the job mode as Finish is executed, this shall be enabled only when the Crypto driver blocks the other Bootlaoder tasks execution for longer duration than the tasks' periodicity while executing the Finish job operation. Note:</p> <ul style="list-style-type: none"> ▶ This requirement is applicable only for the Crypto ASR 4.3 integration.



Id:	OSC-INTMAN-SIGNATUREVERIFICATION-BOOTLOADER-0003
Version:	1
Description:	<p>If enabled to set the key, the integrator shall ensure to configure the Csm asymmetric key size- CsmAsymPublicKeyMaxLength. Note:</p> <ul style="list-style-type: none"> ➤ This requirement is applicable only for the Crypto ASR 4.3 integration.

Id:	OSC-INTMAN-SIGNATUREVERIFICATION-BOOTLOADER-0004
Version:	1
Description:	<p>The integrator shall ensure to configure the Csm asymmetric key size(CsmSignatureVerifyMaxKeySize). Note:</p> <ul style="list-style-type: none"> ➤ This requirement is applicable only for the integration of the Crypto stack version below v4.3.

3.2.8.2.4. Download data Hash verification

Id:	OSC-INTMAN-HASHVERIFICATION-BOOTLOADER-0001
Version:	1
Description:	The integrator shall ensure to select the appropriate configured Csm Job for the ProgCsmHashConfigId parameter under the HashVerification section.

Id:	OSC-INTMAN-HASHVERIFICATION-BOOTLOADER-0002
Version:	1
Description:	<p>The integrator shall ensure to enable HashFinSendFRP to send response pending before the Crypto operation with the job mode as Finish is executed. This shall be enabled only when the Crypto driver blocks the other Bootlaoder tasks execution for longer duration than the tasks' periodicity while executing the Finish mode job operation. Note:</p> <ul style="list-style-type: none"> ➤ This requirement is applicable only for the Crypto ASR 4.3 integration.

Id:	OSC-INTMAN-HASHVERIFICATION-BOOTLOADER-0003
Version:	1
Description:	<p>The integrator shall ensure to configure the Hash digest length(CsmHashResultLength) in the Csm primitives configuration. Note:</p> <ul style="list-style-type: none"> ➤ This requirement is applicable only for the Crypto ASR 4.3 integration.



3.2.8.2.5. Secure Checksum Computation

Id:	OSC-INTMAN-SECURECHECKSUMGEN-BOOTLOADER-0001
Version:	1
Description:	The integrator shall ensure to select the appropriate configured Csm Job for the ProgCsmSecureConfigId parameter in the PROG module.

Id:	OSC-INTMAN-SECURECHECKSUMGEN-BOOTLOADER-0002
Version:	1
Description:	<p>If enabled to set the key (eg.: for AES CMAC operation), the integrator shall ensure to configure the Csm symmetric key size- CsmSymKeyMaxLength. Note:</p> <ul style="list-style-type: none"> ▶ This requirement is applicable only for the Crypto ASR 4.3 integration.

Id:	OSC-INTMAN-SECURECHECKSUMGEN-BOOTLOADER-0003
Version:	1
Description:	<p>The integrator shall ensure to configure the Csm symmetric key size(CsmSymKeyExtractMaxKeySize).</p> <ul style="list-style-type: none"> ▶ This requirement is applicable only for the integration of crypto stack version below 4.3.

3.2.8.2.6. Download data Decryption

Id:	OSC-INTMAN-DECRYPTION-BOOTLOADER-0003
Version:	1
Description:	If "Use Symetric algorithm for decryption" is set to true, the Bootloader will be able to receive encrypted data and decrypt them before writing to Flash.

Id:	OSC-INTMAN-DECRYPTION-BOOTLOADER-0004
Version:	1
Description:	ProgCsmDecryptionConfigId id shall be used to point to the Csm service of the decryption algorithm. The current implementation supports only the CbcPkcs7Decrypt Cry Primitive.

Id:	OSC-INTMAN-DECRYPTION-BOOTLOADER-0005
Version:	1



Description:	<p>If enabled to set the key, the integrator shall ensure to configure the Csm symmetric key size- CsmSymKeyMaxLength.</p> <ul style="list-style-type: none"> ▶ The IV and the Decryption key element reference shall be positioned in the CryptoKeyElementRef table of the crypto driver such that the IV keyelementref shall be the placed at the first position or row and Cipher keyelementref shall be the placed at the second position or row. ▶ When the Signature verification on fly is enabled, ensure to configure Signature verification and the Decryption on separate Csm Queues (also separate driver objects), to allow concurrent jobs execution. ▶ This requirement is applicable only for the Crypto ASR 4.3 integration.
--------------	--

3.2.8.2.7. Memory

3.2.8.2.7.1. Prog memory configuration

The integrator shall take care of the following only if the FLASH wrapper module is used and not the ReProg-MemM module :

Id:	OSC-INTMAN-BOOTLOADER-0070
Version:	2
Description:	The integrator shall configure all memory used for the project. Only 1 memory of each type (FLASH, FLASH_EXT, RAM) can be configured.

Id:	OSC-INTMAN-BOOTLOADER-0077
Version:	1
Description:	For every memory, the integrator shall configure the memory type: Internal Flash (FLASH), external Flash (FLASH_EXT) or RAM memory (RAM).

Id:	OSC-INTMAN-BOOTLOADER-0073
Version:	1
Description:	For every memory, the integrator shall configure the Memory Mode parameter depending if the Flash driver support synchronous or asynchronous interface call. Synchronous means that when the Flash Api is called, it will returns only when the request operation is performed. Asynchronous means that when the Flash Api is called, it returns before performing the requested operation, Prog module will later call periodically a "GetJobStatus" Api to be informed when the operation is finished. In case synchronous mode is used Prog module will re-



quest the erasing of Flash sector per sector to avoid a too long block time is Flash call.

Id:	OSC-INTMAN-BOOTLOADER-0074
Version:	1
Description:	For every memory, the integrator shall ensure that "Minimum value to write" field is set to the minimum size that shall be write for the memory (Flash page size).

Id:	OSC-INTMAN-BOOTLOADER-0075
Version:	1
Description:	For every memory, the integrator shall configure the address offset to be used when accessing the memory. It's used to convert the logical address get from the diagnostic request to the physical address of the memory.

Id:	OSC-INTMAN-BOOTLOADER-0076
Version:	1
Description:	For every memory, the integrator shall configure the erase state value of the memory (0x00 or 0xFF depending of the memory architecture).

3.2.8.2.8. Block and Segments

Id:	OSC-INTMAN-BOOTLOADER-0072
Version:	1
Description:	The integrator shall ensure that the configuration of the memory areas manipulated by the flashloader is consistent against protected areas (hardware protection key, any other sections that shall not be changed)

Id:	OSC-INTMAN-BOOTLOADER-0078
Version:	1
Description:	The integrator shall ensure to map every segment to the correct memory.

Id:	OSC-INTMAN-OEMIND-BOOTLOADER-0120
Version:	1
Description:	<p>The integrator shall ensure that at least one Application segment is configured as follow:</p> <ul style="list-style-type: none"> ➤ Memory_Type set to FLASH ➤ Access_Type set to READ_WRITE



- ▶ Reprog_Start_Address is the start of Application area
- ▶ Reprog_End_Address is the end of Application area
- ▶ Erase_Start_Address is equal to Reprog_Start_Address
- ▶ Erase_End_Address is equal to Reprog_End_Address
- ▶ Partition Type is equal to PROG_APPLICATION_PARTITION

NOTE: Additional application or calibration segment can be configured.

Id:	OSC-INTMAN-OEMIND-BOOTLOADER-0121
Version:	1
Description:	<p>The integrator shall perform the configuration of all logical blocks that shall be supported by the Bootloader (at least one), to download the Application and Calibration in Blocks, the following information need to be configured</p> <ul style="list-style-type: none"> ▶ BlockId: Identifier of the logical block, it is automatically generated in the order of appearance in Tresos starting from 0. Only one Block for Bootloader update can be configured and it shall be the first of the list (BlockId 0). It shall match the one that will be provided in Erase Routine and Request-Download request. ▶ First segment: reference to the first segment (defined in Segment container) of the logical block If a Block for Bootloader Update has been configured, it shall reference a segment of type Bootloader Updater ▶ Number of segments: number of segments in the logical block ▶ Block programming counter max: The maximum number of times a logical block can be programmed. If set to 0, no limit will applied. <p>NOTE: The segment list shall be ordered according to their belonging to the logical blocks. (e.g: if a block reference segment 0 as its first segment and is configured with 4 segments, the segments 0 to 3 are considered as belonging to this block).</p>

In case Logical Block is chosen, for write and erase requests, access is performed using a BlockId and a SegmentId:

- Segment are contiguous memory range aligned on Flash sector.
- Logical block allow regrouping several segments.

Erase request always provides a logical BlockId and the erasing will be performed on all segments that belong to this logical Block. It is possible to configure each logical block following three area types: - Bootloader memory area - Application memory area - Calibration memory area Only one logical block can be defined for Bootloader memory area. If a logical block is defined for the Bootloader memory area, please note that BlockId 0 will be



reserved for this logical block. If no logical block is defined for the Bootloader memory area, then BlockId 0 will not be reserved anymore and will be used to address either logical block defined for Application memory area or logical block defined for Calibration memory area.

Write request always provide a BlockId and a SegmentId, the SegmentId is the index of the segment inside the logical block to be written. Write address is the start address of the Segment identified in the request.

The following picture provide you an example of memory structure:

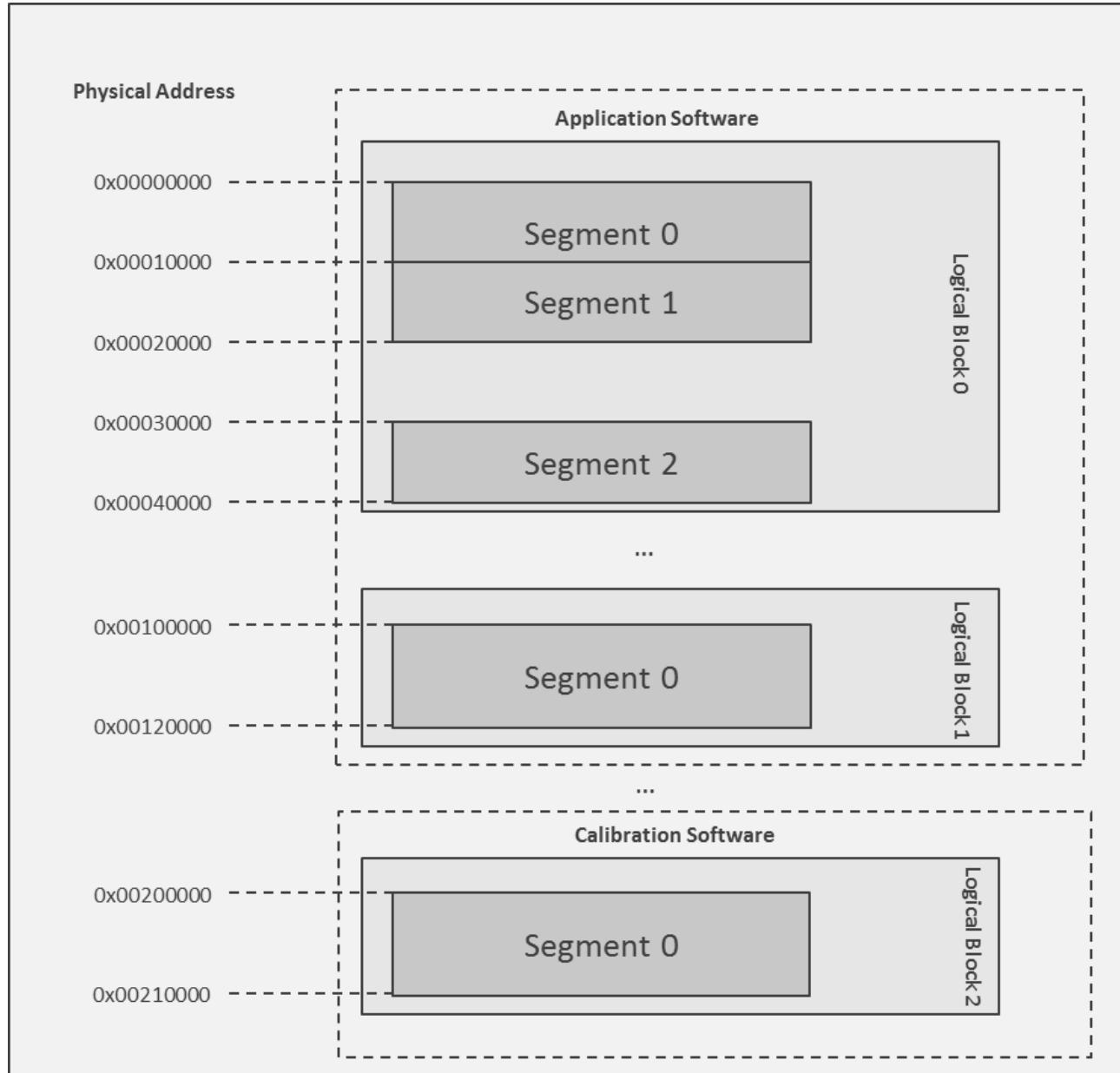


Figure 3.3. Memory structure example

For this example the associated Tresos configuration would be:



The screenshot shows two tables in the EB Bootloader configuration interface:

- Segments Table:**

Index	Name	Memory...	Access_T...	Reprog_S...	Reprog_E...	Erase_Sta...	Erase_En...	Partition Type
0	ApplicationBlock0Seg_0	FLASH	READ_WRITE	0x0	0xffff	0x0	0xffff	PROG_APPLICATION_PARTITION
1	ApplicationBlock0Seg_1	FLASH	READ_WRITE	0x10000	0x1ffff	0x10000	0x1ffff	PROG_APPLICATION_PARTITION
2	ApplicationBlock0Seg_2	FLASH	READ_WRITE	0x30000	0x3ffff	0x30000	0x3ffff	PROG_APPLICATION_PARTITION
3	ApplicationBlock1Seg0	FLASH	READ_WRITE	0x100000	0x1ffff	0x100000	0x1ffff	PROG_APPLICATION_PARTITION
4	CalibrationBlock2Seg_0	FLASH	READ_WRITE	0x200000	0x2ffff	0x200000	0x2ffff	PROG_CALIBRATION_PARTITION
- Blocks Table:**

Index	Name	First Segment	Number of segments	Prog...
0	AppliBlock0	@ /Prog/Prog/ApplicationBlock0Seg_0	0x3	
1	AppliBlock1	@ /Prog/Prog/ApplicationBlock1Seg0	0x1	
2	CalibBlock2	@ /Prog/Prog/CalibrationBlock2Seg_0	0x1	

Figure 3.4. Memory structure example

Id:	OSC-INTMAN-OEMIND-BOOTLOADER-0001
Version:	1
Description:	<p>The BLUpdater software index in Block and Segments tables is depending on the Addressing Modes in request download configuration</p> <ul style="list-style-type: none"> ➤ In case of Logical Block : The index of the BLUpdater block in the blocks table must have index 0 ➤ In other case : The BLUpdater software must be in downloaded in one segment with index 0 in the segments table

3.2.8.2.8.1. Addressing Modes in request download

The **Request Download Addressing Mode** parameter is used to specify the content of the RequestDownload address. There is two option for this parameter: Download by logical block and Download by logical block and segment.

Download by logical block and segment means that in the RequestDownload both block ID and the segment ID will be sent in the request along with the segment length. This is the default value of the parameter.

Download by logical block means that in the RequestDownload only the block ID will be sent along with the block lenght

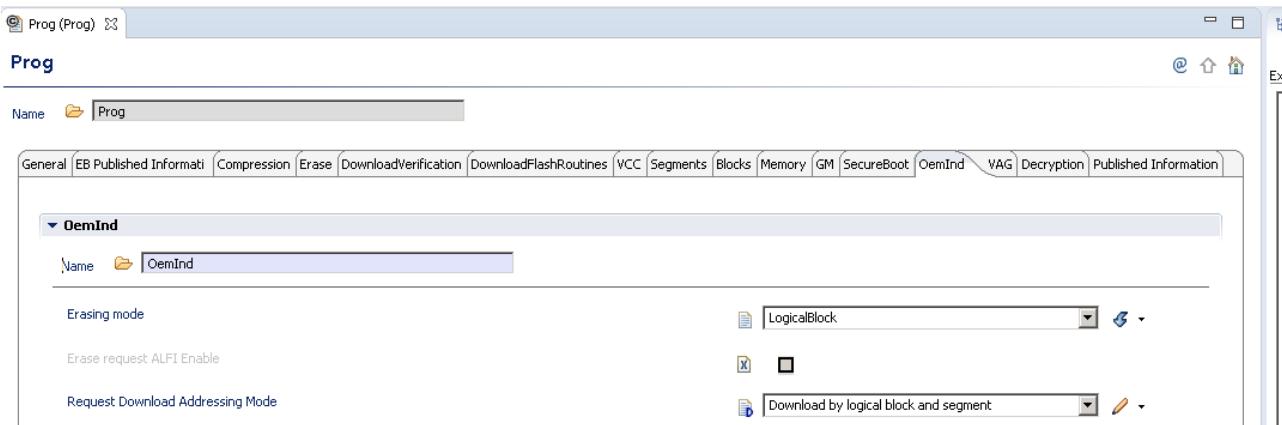


Figure 3.5. Request Download Addressing Mode Parameter

Id:	OSC-INTMAN-OEMIND-BOOTLOADER-0127
Version:	1
Description:	In case of Download by logical block is used, The block shall have only one segment

3.2.8.2.9. Oem independent specific parameters

Id:	OSC-INTMAN-OEMIND-BOOTLOADER-0100
Version:	1
Description:	<p>The integrator shall configure Erasing Mode parameter according to the expected erasing request. The following values are possible:</p> <ul style="list-style-type: none"> ▶ All: No information are provided in Erase request, on reception of the erase request, all the memory segments will be erased. ▶ Address: The Erase request contains the address range to be erased, only 1 segment can be erased with this request. Note that in this case the checksum will be computed on a single segment. ▶ LogicalBlock: The Erase request contains the logical block to be erased.

Id:	OSC-INTMAN-OEMIND-BOOTLOADER-0101
Version:	1
Description:	The integrator shall configure the "Erase request ALFI Enable" parameter to indicate if the Erase request contains the UDS ALFI field.

Id:	OSC-INTMAN-OEMIND-BOOTLOADER-0102
------------	-----------------------------------



Version:	1
Description:	The integrator shall configure the "Application validity" parameter to indicate if Elektrobit algorithm or customer specific algorithm shall be used. If customer specific, the integrator shall implement the PROG_InvalidateSection and PROG_CustomSetCrcCompareSuccess callbacks to manage the application validity status.
Id:	OSC-INTMAN-OEMIND-BOOTLOADER-0103
Version:	1
Description:	The integrator shall configure the "CRC algorithm" parameter to indicate which checksum shall be used to verify the software download.
Id:	OSC-INTMAN-OEMIND-BOOTLOADER-0104
Version:	1
Description:	The integrator shall configure the "FingerPrint Enable" parameter to indicate if a fingerprint shall be managed for the software download.
Id:	OSC-INTMAN-OEMIND-BOOTLOADER-0105
Version:	1
Description:	If FingerPrint is enable, the integrator shall configure the "Size_Of_FingerPrint" parameter to indicate the expected FingerPrint size.

3.2.8.2.10. Download verification

Id:	OSC-INTMAN-BOOTLOADER-2011
Version:	1
Description:	The integrator shall configure the verification buffer size in parameter "Verification Buffer size". This buffer is used to temporary locate the data read from memory in order to compute the CRC/Hash/Signature on them. Note that more the buffer is big, more the blocking time for verification computation will be high.
Id:	OSC-INTMAN-OEMIND-VERIF-BOOTLOADER-0001
Version:	1
Description:	The integrator shall configure (parameter "Maximum RequestDownload Per Block") the maximum number of RequestDownload request that can be received, by the Bootloader, for a single logical block. This value will be used to size the RAM structure storing the downloaded memory area that shall be used to perform the signature/CRC verification.



3.2.8.2.11. Download Flash Routines

Id:	OSC-INTMAN-OEMIND-FLASHROUTINES-BOOTLOADER-0126
Version:	1
Description:	The integrator shall ensure that "Compressed Flash Driver" feature is activated.

3.2.8.3. BM Configuration

Id:	OSC-INTMAN-OEMIND-BOOTLOADER-0010
Version:	1
Description:	If the feature is enabled integrator shall ensure that "BM_TIMEOUT_CHECK" field is set.

Id:	OSC-INTMAN-OEMIND-BOOTLOADER-0011
Version:	1
Description:	If the feature is enabled integrator shall ensure that "BM_SOURCE_ADDRESS_CHECK" field is set.

Id:	OSC-INTMAN-OEMIND-BOOTLOADER-0012
Version:	1
Description:	If the feature is enabled integrator shall ensure that "BM_TIMEOUT_VALUE" is configured

Id:	OSC-INTMAN-OEMIND-BOOTLOADER-0013
Version:	1
Description:	<p>The integrator shall ensure to select the Boot Manager to generate in "Boot_Manager_Variant" field</p> <ul style="list-style-type: none"> ▶ Initial Boot Manager : Checks if BootManager is integrated in a Initial Boot Manager software. ▶ Boot Manager : Checks if Boot Manager is integrated in a Bootloader software.

Id:	OSC-INTMAN-OEMIND-BOOTLOADER-0014
Version:	1
Description:	If Initial Manager is selected, the integrator shall ensure that "Initial_Check_Application" field is set if the application validy check need to be done in the Initial Boot Manager.



3.2.8.4. BLUpdater Configuration

3.2.8.4.1. General

Id:	OSC-INTMAN-BOOTLOADER-0200
Version:	1
Description:	The integrator shall configure the period for sending NRC78 to manage the sending of NRC78 after a definite interval of time.

3.2.8.4.2. Bootloader Configuration

Id:	OSC-INTMAN-BOOTLOADER-0201
Version:	1
Description:	For current bootloader, the integrator shall configure the start address, end address of the bootloader.

Id:	OSC-INTMAN-BOOTLOADER-0202
Version:	1
Description:	For new bootloader, the integrator shall configure the start address, end address of the bootloader.

3.2.8.4.3. Memory

Id:	OSC-INTMAN-BOOTLOADER-0203
Version:	1
Description:	For every memory, the integrator shall configure the memory type: Internal Flash (FLASH), external Flash (FLASH_EXT) or RAM memory (RAM).

3.2.8.5. SA Configuration

Id:	OSC-INTMAN-OEMIND-BOOTLOADER-0070
Version:	1



Description:	The integrator shall ensure that the parameter "Security_Access_Seed_Length" is set to 4
--------------	--

Id:	OSC-INTMAN-OEMIND-BOOTLOADER-0071
Version:	1
Description:	The integrator shall ensure that the parameter "Security_Access_Key_Length" is set to 4

Id:	OSC-INTMAN-OEMIND-BOOTLOADER-0072
Version:	1
Description:	If the Static Seed (response with precedent seed in the case of successive GetSeed requests without a respective key received) is required for the project, the integrator shall ensure that "Enable_Static_Seed" field is set.

3.2.8.6. UDS Configuration

3.2.8.6.1. General

The parameter "External Response Pending Manage Call" allow the integrator to call the scheduling function [UDS_ResponsePending_Manage](#) by itself outside of the rest of the stack scheduling (e.g. in a task with higher priority). This API manage the timer for P2 and P2*.

If stm timer interrupt shall be used to schedule the NRC78 sending, the API `UDS_ResponsePending_TimCntManage` shall be used instead and parameter `TIMER_RESPONSE_PENDING_CHECK` shall be checked(enabled) in uds configuration(tresos). The use of the timer interrupt ensure to not miss sending of NRC 78 at P2 and P2 star time.

Note: This API shall be called at the configured period of `UDS_MANAGE_PERIOD`

Id:	OSC-INTMAN-BOOTLOADER-0104
Version:	1
Description:	The integrator shall ensure that "Standard" field is set to "ISO"

Id:	OSC-INTMAN-BOOTLOADER-0105
Version:	1
Description:	The integrator shall ensure that "UDS_MANAGE_PERIOD" field is set to the call period of the <code>UDS_Manage</code> Api



Id:	OSC-INTMAN-BOOTLOADER-0100
Version:	1
Description:	The integrator shall ensure that "SecurityCheck" field is set and that the "SecurityFunction" is set to "PROG_GetSecurityLevel"

Id:	OSC-INTMAN-BOOTLOADER-0106
Version:	1
Description:	The integrator shall ensure that "RC_NRC_IMPLEMENTATION " field is set to 0x31

Id:	OSC-INTMAN-BOOTLOADER-0107
Version:	1
Description:	The integrator shall ensure that "DID_NRC_IMPLEMENTATION " field is set to 0x31

Id:	OSC-INTMAN-BOOTLOADER-0101
Version:	1
Description:	The integrator shall ensure that "RESPONSE_PENDING" field is set

Id:	OSC-INTMAN-BOOTLOADER-0102
Version:	1
Description:	The integrator shall ensure that "SPREC_IN_RESPONSE" field is set

Id:	OSC-INTMAN-BOOTLOADER-0103
Version:	1
Description:	The integrator shall ensure that "RELOAD_TSTOPDIAG" field is set

Id:	OSC-INTMAN-BOOTLOADER-0108
Version:	1
Description:	If for integration reason the NRC78 response pending shall be send before the end of the configured P2/P2star time, the integrator can configure the P2/P2star adjust parameter. In this case the UDS module will trig the NRC78 transmission at a time equal to (P2/P2star - P2/P2star adjust)

Id:	OSC-INTMAN-BOOTLOADER-0126
Version:	1



Description:	The integrator shall ensure that the max allowed response length for every DID in the RDBI service request "RDBI_MAX_RESPONSE_LENGTH" must be less than the configured reception buffer length "RxPhysicalBufferSize" in BIPduR
--------------	---

Id:	OSC-INTMAN-BOOTLOADER-0127
Version:	1
Description:	The integrator shall ensure that "UDS_TIMEOUT_CHECK" is set to true if "BM_TIMEOUT_CHECK" feature is enabled

3.2.8.6.2. Session

Id:	OSC-INTMAN-BOOTLOADER-0120
Version:	1
Description:	The integrator shall configure every of following session: DEFAULT, PROGRAMMING, EXTENDED, SUPPLIER, OTHER_01, OTHER_02, OTHER_03, OTHER_04

3.2.8.6.3. Service

The integrator shall configure in Service configuration window all service (except ReadDataByIdentifier, WriteDataByIdentifier and RoutineControl) that shall be supported by the ECU

3.2.8.6.4. Service_DID

The integrator shall configure in Service_DID configuration window all DIDs that shall be supported by the ECU

3.2.8.6.5. Routines

The integrator shall configure in Routine_Control configuration window all routines that shall be supported by the ECU

3.2.8.7. BIPduR Configuration

This configuration is only available if the project is based on a ASR communication stack



3.2.8.7.1. Connection configuration

A connection define the reception PDUs and transmission PDU that shall be used to communication with a diagnostic tester. In order to ensure Bootloader communication, at least one connection shall be defined. If the Bootloader shall be able to communicate with several testers, several connections shall be defined.

For every connection, the following parameters shall be configured:

Id:	OSC-INTMAN--BLPDUR-BOOTLOADER-0001
Version:	1
Description:	The integrator shall configure the TxPDU (TxPdu Reference) to be used for response transmission. It shall reference a valid ECUC PDU.
Id:	OSC-INTMAN--BLPDUR-BOOTLOADER-0002
Version:	1
Description:	The integrator shall configure the TxPDU confirmation identifier (TxPdu Identifier) to be used by PduR module to confirm the transmission. The TxPdu Identifiers shall be unique and consecutives.
Id:	OSC-INTMAN--BLPDUR-BOOTLOADER-0003
Version:	1
Description:	The integrator shall configure the Tester Address associated to this connection.
Id:	OSC-INTMAN--BLPDUR-BOOTLOADER-0004
Version:	1
Description:	The integrator shall indicate if this connection is associated to a LIN communication (Lin Connection) .
Id:	OSC-INTMAN--BLPDUR-BOOTLOADER-0005
Version:	1
Description:	The integrator shall indicate if this connection shall re-use a functional RxPduld already defined in another connection (Share Functional Id).
Id:	OSC-INTMAN--BLPDUR-BOOTLOADER-0006
Version:	1
Description:	If "Share Functional Id" is enabled, the integrator shall configured the shared Pdu Reference.
Id:	OSC-INTMAN--BLPDUR-BOOTLOADER-0007
Version:	1



Description:	The integrator shall configure all the RxPDU to be used for request reception. It shall reference a valid ECUC PDU, specify if the Pdu is functional or physical and the associated Pdu Identifier to be used by PduR module. The RxPdu Identifiers shall be unique and consecutive.
Id:	OSC-INTMAN--BLPDUR-BOOTLOADER-0008
Version:	1
Description:	A connection shall contain only one functional RxPdu or one reference functional RxPdu.

3.2.8.7.2. Multiple Receive Buffer Configuration

The Multiple Receive buffer allows the bootloader to receive a Transfer Data request while it writes the previous one received.

This feature allows to improve downloading performance but will greatly increase the RAM size usage.

In case the feature is enabled, the maximum number of buffer allowed is 4.

Warning: This feature shall be used only if Hardware/MCAL is able to perform a Flash write in less than a CAN frame duration (Depending of the Hardware/MCAL a flash page writing can be superior to CAN frame duration or the writing of a single Flash page is not possible).

Otherwise there is a risk to lose CAN frame reception.

Warning: This feature can not be used if the feature Queued Requests is enabled.

3.2.8.7.2.1. Behavior on reception of a Transfer Data request

Be aware that if the Multiple Buffer feature is activated, behavior will no more be compliant with ISO-14229. Bootloader will always acknowledge the first received Transfer Data. Answers sent by the bootloader after the reception of Transfer Data requests will always concern the Transfer Data request previously received.

This behavior can be explained by the fact that bootloader needs to send quickly an answer to tester tool to trigger the transmission from tester tool of the next Transfer Data request.

This answer is done:

- ▶ Before the current Transfer Data request is processed.
- ▶ Regardless the true state of the Transfer Data request currently processed.



3.2.8.7.3. Multiple Identifier Configuration

The Multiple Identifier feature is only available for Can network. It allows the bootloader select a group ID that contains one or multiple connections. The received Can frame configured will then be filtered depending on the current group activated

In order to select a group ID, an "Architecture Frame" (which identifier is configured), shall be sent at startup before a delay (which is also configured). It will contain the information of group ID to select.

Anyway, the "Architecture Frame" is not taken into account at each restart of the ECU. So it is not possible to select a group ID each time that ECU restarts.

The "Architecture Frame" will:

- ▶ be taken into account each time that the previous connection context is not needed.
- ▶ not be taken into account each time that the previous connection context is needed. It means that the previous group ID and the previous active connection before the restart will be retrieved.

Examples where previous connection context is not retrieved:

- ▶ if the feature "Send Response After Reset" is deactivated, then each request which makes the ECU restart will be responded before the restart. After restart, as no frame need to be transmitted before the timeout P2, no group ID and active connection will be set and the "Architecture Message" will be able to select one.
- ▶ if the feature "Send Response After Reset" is activated, and positive response bit is present then each request which makes the ECU restart will not be responded even after the restart. After restart, as no frame need to be transmitted before the timeout P2, no group ID and active connection will be set and the "Architecture Message" will be able to select one.

Example where previous connection context is retrieved:

- ▶ if the feature "Send Response After Reset" is activated, then each request which makes the ECU restart will be responded after the restart. After restart, as frames need to be transmitted on the same group ID and same active connection before the timeout P2, group ID and active connection will be retrieved and the "Architecture Message" will not be able to select different one.

The [multiple identifier](#) configuration can be used following two differents configurations

- ▶ EXTERNAL_NOTIFICATION: The group ID will be retrieve at startup by the bootloader by calling a callback
- ▶ CAN_NOTIFICATION: The Bootloader wait at startup the reception of the CAN frame that has one byte containing the architecture ID of a group ID

3.2.8.7.3.1. External Notification Configuration

With this configuration the customer shall only ensure to correctly fill the [BIPduR_GetGroupIdVal](#) callback



3.2.8.7.3.2. Can Notification Configuration

With this configuration the customer shall ensure to configure correctly the following element

In MultipleIdentifier container:

- ▶ Multiple Identifier Timeout: value of the timeout after which if the CAN frame has not been received the default group ID will be used
- ▶ ID group PDU reference: Reference to the PDU that contain the Architecture ID
- ▶ ID group PDU Id: Pdu Id that shall be used by PduR to call notify the PDU reception
- ▶ ID group Byte Number: Byte where the Architecture information is stored in the PDU (0 is LSB, 7 is MSB)

In IDGroup container:

- ▶ Default ID group: One default group ID shall be selected
- ▶ Architecture Id: In every group ID the architecture value shall be defined

3.2.8.8. Crypto Configuration

If signature verification is used for the project the following configuration shall be performed (required configuration is for RSA-PSS 2048 bits with SHA256) .

Id:	OSC-INTMAN-OEMIND-CRY-BOOTLOADER-0001
Version:	1
Description:	<p>The integrator shall configure in CRY module a CryRsaSsaPssVerify configuration with the following configuration:</p> <ul style="list-style-type: none">▶ CryRsaSsaPssVerifyUseTimeSlices (Use time slicing for RSASSA-PSS signature verification) set to true▶ CryRsaSsaPssVerifyNumberOfTimeSlices (Number of RsaSsaPss time slices) set to 10▶ CryRsaSsaPssVerifyUseCbk (Use configured callback function which returns maximum number of time slices) set to false▶ CryRsaSsaPssVerifyImmediateRestartEnabled (Enable the cancelation of an ongoing calculation regardless of the configuration ID) set to true▶ CryRsaSsaPssVerifyHashCfgRef (Hash configuration) set to CsmHashConfig_0▶ CryRsaSsaPssVerifyKeyLength (Key Length) set to 256▶ CryRsaSsaPssVerifySaltLength (Salt Length) set to 0



- ▶ CryRsaSsaPssVerifyB64Encoded (Base64 Encoded) set to false
- ▶ CryRsaSsaPssVerifyUseBarrett (Barrett reduction) set to false
- ▶ CryRsaSsaPssVerifySupportRestart (Enable the cancelation of ongoing requests) set to true

Id:	OSC-INTMAN-OEMIND-CRY-BOOTLOADER-0002
Version:	1
Description:	<p>The integrator shall configure in CRY module the following general parameters:</p> <ul style="list-style-type: none"> ▶ CrySHAOneAndTwoImplementation (Implementation variant) set to CRY_SHAONEANDTWO_INTERRUPTABLE ▶ CryInterruptableLN (Interruptable LN operations) set to true

Id:	OSC-INTMAN-OEMIND-CRY-BOOTLOADER-0003
Version:	1
Description:	<p>The integrator shall configure in CRY module a CrySHA2 configuration with the following configuration:</p> <ul style="list-style-type: none"> ▶ CrySHA2ImmediateRestartEnabled (Enable the cancelation of an ongoing calculation regardless of the configuration ID) set to true ▶ CrySHA2Type (Prime) set to CRY_SHA_256 ▶ CrySHA2IterationsPerMain (Number of iterations per MainFunction) set to 1 ▶ CrySHA2SupportRestart (Enable the cancelation of ongoing requests) set to true

Id:	OSC-INTMAN-OEMIND-CRY-BOOTLOADER-0004
Version:	1
Description:	<p>The integrator shall configure in CSM module a CsmSignatureVerify configuration with the following configuration:</p> <ul style="list-style-type: none"> ▶ CsmCallbackSignatureVerify set to "PROG_CsmNotification" ▶ CsmSignatureVerifyMaxKeySize set to 524 ▶ CsmSignatureVerifyInitConfiguration set to CryRsaSsaPssVerifyConfig_0 (configuration done in OSC-INTMAN-OEMIND-CRY-BOOTLOADER-0001) ▶ CsmSignatureVerifyEnableRteInterface (Enable Rte Interface) set to false ▶ CsmSignatureVerifyEnableRestart (Enable the cancelation of ongoing requests) set to true ▶ CsmSignatureVerifyUsePriorities (Csm priorities handling) set to true



Id:	OSC-INTMAN-OEMIND-CRY-BOOTLOADER-0005
Version:	1
Description:	<p>The integrator shall configure a CsmHash configuration (CsmHashConfig_0) with the following configuration:</p> <ul style="list-style-type: none"> ➤ CsmCallbackHash set to Cry_RsaSsaPssVerifyCallback ➤ CsmHashInitConfiguration set to CrySHA2Config_0 (configuration done in OSC-INTMAN-OEMIND-CRY-BOOTLOADER-0001) ➤ CsmHashPrimitiveName set to SHA2 ➤ CsmHashEnableRteInterface (Enable Rte Interface) set to false ➤ CsmHashEnableRestart (Enable the cancelation of ongoing requests) set to true ➤ CsmHashUsePriorities (Csm priorities handling) set to true

3.2.8.9. Ethernet Configuration

This information is only suitable if the project is using the EB ethernet communication stack on ACG-8 or newer versions.

3.2.8.9.1. MemMap related

From versions ACG-8 or newer, two new buffers have been added in ethernet driver: Eth_BufferRamRx and Eth_BufferRamTx. These two needs to be aligned on 8 bytes in memory to be handled correctly. To do so, a specific MemMap definition should be added, for each buffer, in project MemMap's configuration file (MemMap.-xdm). The absence of these two new definitions could prevent the ethernet driver from any communication possibilities. Example of MemMapAddressingMode:

```
Eth_BufferRamRx:

#if (defined EthBufferRx_ETH_BUFFER_RX_START_SEC_Common)
    #if (!defined ETH_MEMMAP_BUFFERRX_PRAGMA_EXECUTED)
#define ETH_MEMMAP_BUFFERRX_PRAGMA_EXECUTED
    #if defined _GREENHILLS_C_PPC_
        #pragma alignvar (8)
    #elif defined _DIABDATA_C_PPC_
        #pragma section ETH_BUFFER ".ETH_BUFFER_SEG" ".ETH_BUFFER_SEG"
        #pragma use_section ETH_BUFFER Eth_BufferRamRx
    #elif defined _TASKING_C_TRICORE_
```



```

        #pragma align 8
#endif
#endif
#undef EthBufferRx_ETH_BUFFER_RX_START_SEC_Common
#undef MEMMAP_ERROR
#elif (defined EthBufferRx_ETH_BUFFER_RX_STOP_SEC_Common)
#ifndef _GREENHILLS_C_PPC_
#pragma ghs section bss=default
#elif defined _TASKING_C_TRICORE_
#pragma align 0
#endif
#undef EthBufferRx_ETH_BUFFER_RX_STOP_SEC_Common
#undef MEMMAP_ERROR

```

3.2.8.10. ReProgMemM Configuration

This information is only suitable if the project is using the EB ReProgMemM module for the memory operations.

3.2.8.10.1. ReProgMemM configuration

For the integration of the ReProgMemM module, the integrator shall take care of the following configuration parameters :

3.2.8.10.1.1. Configuration

Id:	OSC-INTMAN-BOOTLOADER-0210
Version:	1
Description:	In the "Memory Configuration" tab, the integrator shall configure all memory used for the project. Only 1 memory of each type (FLASH, FLASH_EXT, RAM, CUSTOM) can be configured.

Id:	OSC-INTMAN-BOOTLOADER-0211
Version:	1
Description:	For every memory, the integrator shall configure the memory type: Internal Flash (FLASH), external Flash (FLASH_EXT), RAM memory (RAM) or Custom memory (CUSTOM).

Id:	OSC-INTMAN-BOOTLOADER-0212
------------	----------------------------



Version:	1
Description:	For every memory, the integrator shall ensure that "Minimum value to write" field is set to the minimum size that shall be written for the corresponding memory (Flash page size).

Id:	OSC-INTMAN-BOOTLOADER-0213
Version:	1
Description:	For every memory, the integrator shall configure the address offset to be used when accessing the corresponding memory. It's used to convert the logical address get from the diagnostic request to the physical address of the memory.

Id:	OSC-INTMAN-BOOTLOADER-0214
Version:	1
Description:	For every memory, the integrator shall configure the erase state value of the memory (0x00 or 0xFF depending of the memory architecture).

Id:	OSC-INTMAN-BOOTLOADER-0215
Version:	1
Description:	For every memory, the integrator shall configure the corresponding memory start address and entire length.

Id:	OSC-INTMAN-BOOTLOADER-0216
Version:	1
Description:	In the "Flash Sectors" tab, the integrator shall configure all the flash sectors (can be grouped into FlashBanks) with their corresponding start address, length and protection information (Programmable sector or not).

3.2.9. Callbacks

The callbacks are pieces of code that shall be implemented by integrator.

The callbacks are divided in different types:

- ▶ The bootloader diagnostic service callbacks.
- ▶ The hardware related callbacks.
- ▶ The BootManager callbacks.

- ▶ The Security callbacks.
 - ▶ The Programming callbacks.
 - ▶ The Communication callbacks.

The callback implementation execution time shall be as short as possible in order not to block other Bootloader tasks.

Id:	OSC-INTMAN-CBK-0001
Version:	1
Description:	The integrator shall ensure that the callback execution doesn't block the processing of Bootloader MainFunction/Manage cyclic task (e.g BIPduR_Manage)

3.2.9.1. Programming sequences

The following sequences show the common download sequence and indicate for every diagnostic request if a callback is called. These callbacks shall be implemented by integrator and help him identifying when the callback is called during the programming sequence.

When Initial Boot Manager (IBM) is used, the [Initial Boot Manager Sequence](#) comes before the Programming sequence, and this last one is executed when the Initial Boot Manager (IBM) starts the Bootloader.

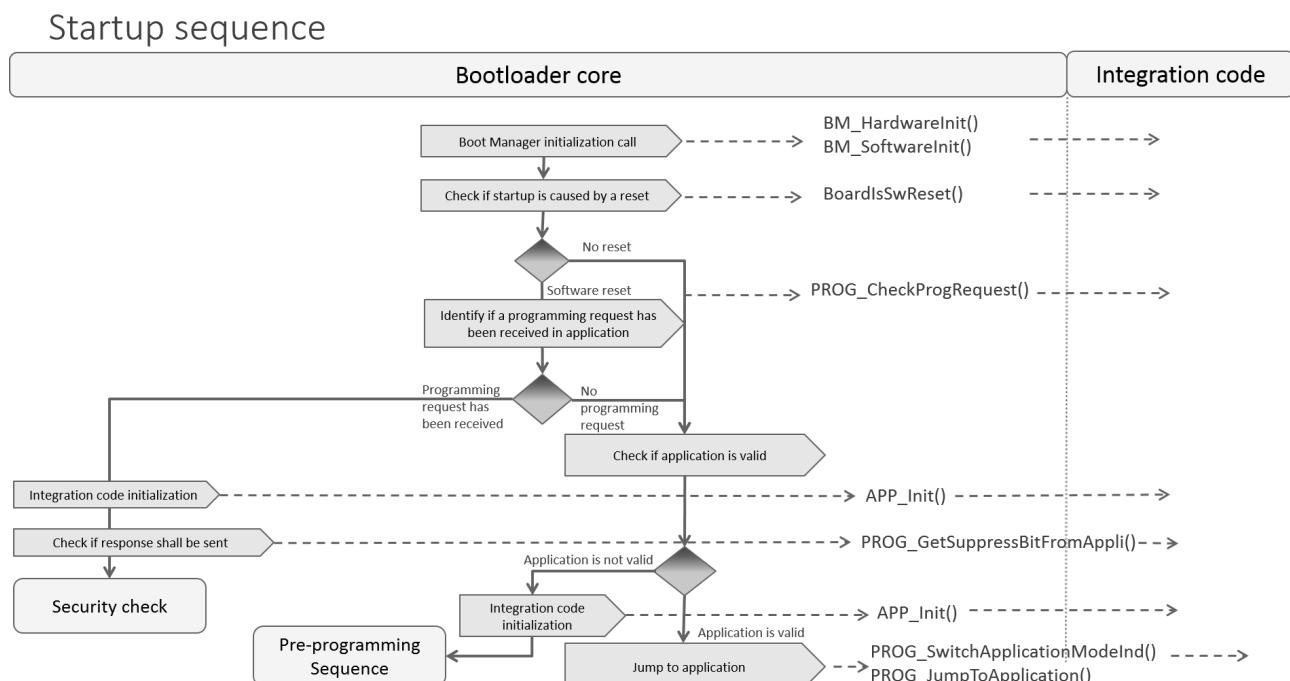


Figure 3.6. Startup sequence

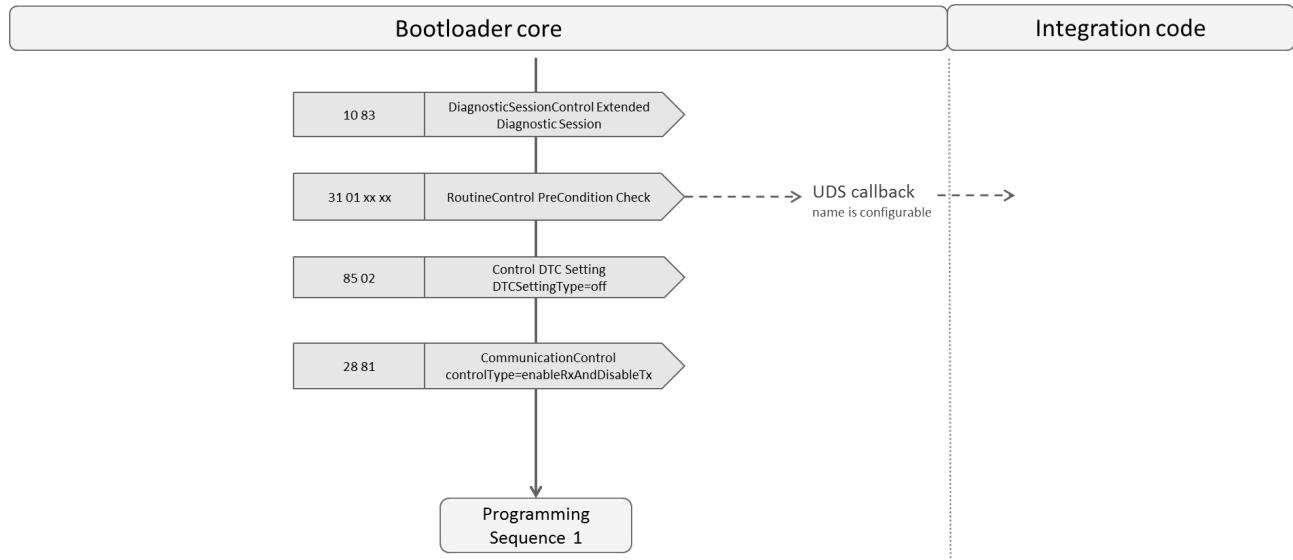


Figure 3.7. Pre-Programming sequence

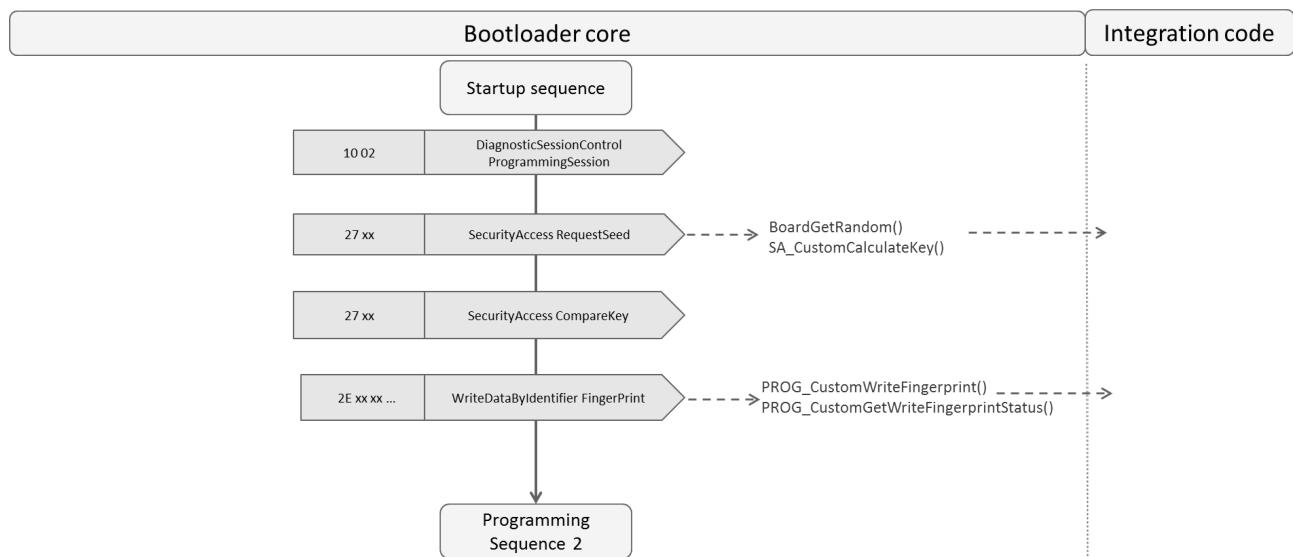


Figure 3.8. Reprogramming sequence



Reprogramming sequence for Erasing by BlockId

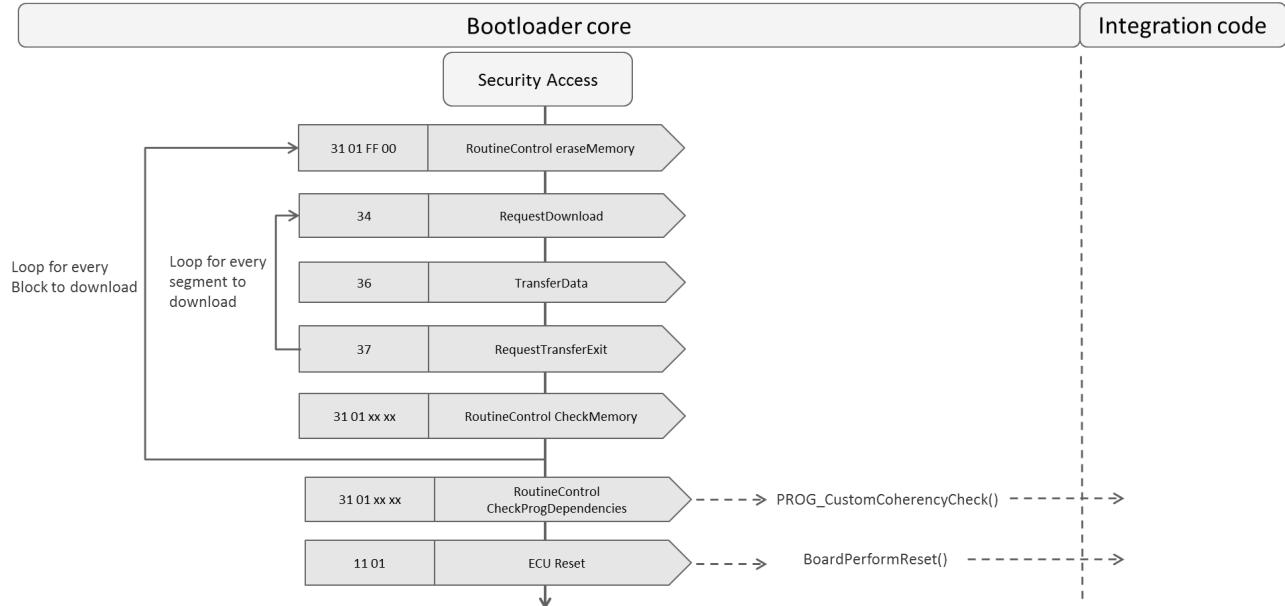


Figure 3.9. Reprogramming sequence

Application/calibration download with logical block

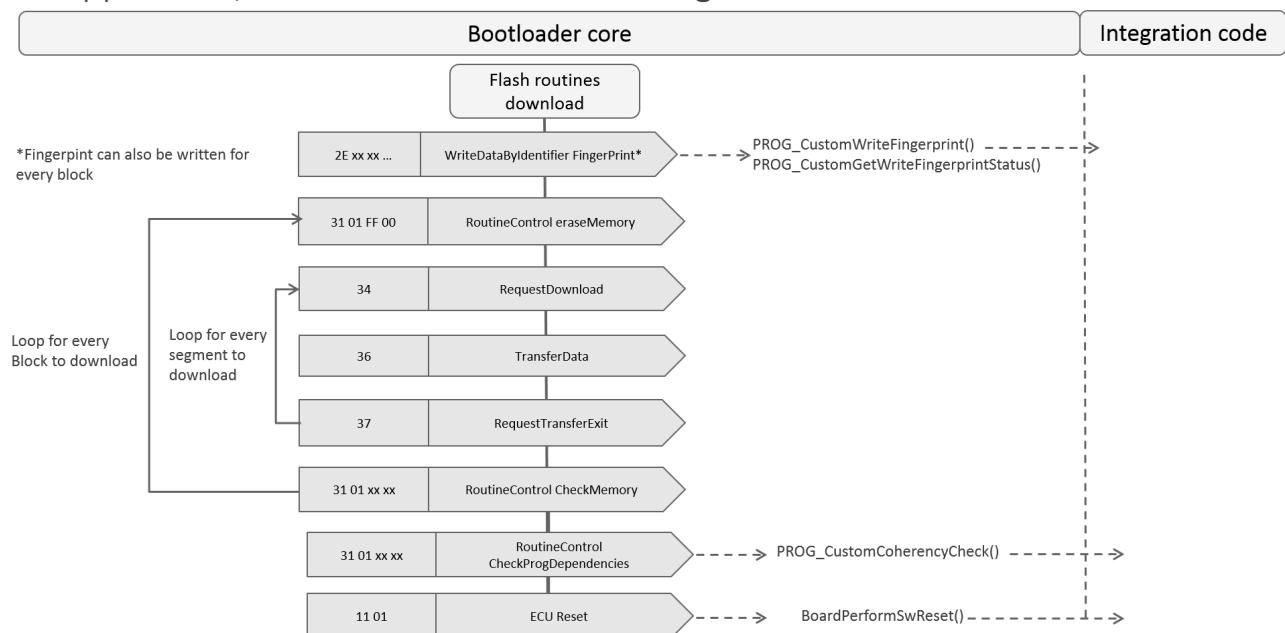


Figure 3.10. Reprogramming sequence



3.2.9.2. Bootloader diagnostic Callbacks

Bootloader diagnostic callbacks are generated by the UDS module, based on customer configuration, for all not required service of the bootloader (e.g. specific DID (RDBI services) for the project).

Mandatory services are described in the [OEM UDS configuration](#).

These callbacks shall be configured and fully managed by the integrator of the bootloader.

Id:	OSC-INTMAN-BOOTLOADER-0001
Version:	1
Description:	The integrator shall configure and fill all callbacks out of scope of the main bootloader purpose.

3.2.9.2.1. UDS callbacks

All UDS callbacks shall have the following prototype: tUdsStatus *CallbackName*(PduLengthType *pulLen, u8 *aubUdsData), with

tUdsStatus: status of the diagnostic service processing (UDS_ACK: positive response can be sent / UDS_NRC_ErrorCode)

pulLen: Request length(IN) / Response length(OUT)

aubUdsData: Request data(IN) / Response data(OUT)

Important Note: For service RDBI *pulLen is not the received length but the current buffer size consumed. This length is useful for customer in case of MULTI DID to check if response exceeds TP buffer size (NRC shall be returned in this case)

One implementation example of RDBI callback:

```
tUdsStatus UDS_RDBI_ECUSerialNumberDataIdentifier_f18c(PduLengthType *pulLen,
                                                       u8 *aubUdsData)
{
    tUdsStatus ubUdsStatus;
    u16 ubExpectedCallSize;
    u16 ubIdx;

    /* Here calculate the expected return size */
    /* including SID (1) DID (2) + data           */
    ubExpectedCallSize = 0x38;
```



```

if ((*pulLen + ubExpectedCallSize) > TP_DIAG_MSG_DATA_MAX)
{
    ubUdsStatus = UDS_NRC_XX;
}
else
{
    ubUdsStatus = UDS_ACK;

    /* Here fill the Data buffer from the third byte */
    for(ubIdx = 3U; ubIdx < ubExpectedCallSize; ubIdx++)
    {
        aubUdsData[ubIdx] = 0xEBU;
    }

    /* Return the expected size to the UDS layer */
    *pulLen = ubExpectedCallSize;
}

return ubUdsStatus;
}

```

3.2.9.2.2. Indication callbacks

Additional indications are provided by the bootloader at some important point of its processing (e.g. Segmented message reception). It is the integrator choice to fulfil them or not depending on the project need.

Id:	OSC-INTMAN-BOOTLOADER-0030
Version:	1
Description:	<p>The integrator shall ensure that the following callbacks are implemented:</p> <ul style="list-style-type: none"> ▶ UDS_CustomPositiveAnswerInd(): This callback is called to give possibility to the user to execute an action before the positive answer transmission. ▶ APP_GetUdsDataBufferInd(): This callback is called when UDS reponse is available but not yet transmitted. The buffer can be updated if necessary. ▶ APP_UdsSessionStatusInd(): Notification for diagnostic session transition. ▶ APP_TpRxInd(): This callback is called when a message reception is completed, successfully or not. ▶ APP_TpTxConf(): This callback is called when a message transmission is completed, successfully or not. ▶ UDS_P2AboutToExpireInd(): Notification just before the P2/P2_STAR time-out



3.2.9.3. Hardware Related Callbacks

These callbacks are used by the bootloader to get information regarding hardware register or perform some hardware control.

They are managed in the board.c file as an example but can moved and reworked by the integrator.

For more details please refer to the [Bootloader_Generic_documentation.pdf](#) file.

Id:	OSC-INTMAN-BOOT-HWCBK-0010
Version:	1
Description:	<p>The integrator shall ensure that the following callbacks are compiled and implemented:</p> <ul style="list-style-type: none">➤ BoardSetSleepState()➤ BoardPerformSwReset()➤ BoardIsSwReset()➤ BoardEnableInterrupts()➤ BoardDisableInterrupts()➤ BoardGetRandom()

BoardDisableInterrupts and BoardEnableInterrupts APIs can be called by any layer of the bootloader to disable and enable the interruption.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOT-HWCBK-0011
Version:	1
Description:	The integrator shall ensure that the API BoardDisableInterrupts is implemented and allow disabling the interruption.

Id:	OSC-INTMAN-BOOT-HWCBK-0012
Version:	1
Description:	The integrator shall ensure that the API BoardEnableInterrupts is implemented and allow enabling the interruption. Note: For Bootloader software that does not use the interruption this API shall stays empty.

The BoardIsSwRese API is called by the PROG layer to know if a software reset has been done and if it shall be evaluate if a programming request has been done.

It allows Bootloader identifying if the Bootloader start is caused by a power-on or by a software reset (e.g by application when receiving a programming event)



Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOT-HWCBK-0016
Version:	2
Description:	The integrator shall ensure that the API BoardIsSwRese is implemented and provide the cause of the previous reset.

The BoardSetSleepState API is called by the PROG layer to set the microcontroller in a sleep state

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOT-HWCBK-0013
Version:	2
Description:	The integrator shall ensure that the API BoardSetSleepState is implemented and allow setting the ECU in sleep state.

The BoardPerformSwReset API is called by the PROG layer to perform the reset the microcontroller.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOT-HWCBK-0014
Version:	2
Description:	The integrator shall ensure that the API BoardPerformSwReset is implemented and allows performing a software reset.

The BoardGetRandom API is called by the SA layer to get a pseudo random value for seed use.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOT-HWCBK-0015
Version:	1
Description:	The integrator shall ensure that the API BoardGetRandom is implemented and return a 32bit pseudo random value.

3.2.9.4. BM Callbacks

The BootManager has its own callback. They will be the first function call at the ECU startup and allow integrator to initialize its Hardware with [BM_HardwareInit](#) and its software with [BM_SoftwareInit](#)

Id:	OSC-INTMAN-BOOTLOADER-0020
-----	----------------------------



Version:	1
Description:	The integrator shall ensure that the API BM_HardwareInit is implemented and if necessary contains some specific hardware initialization.
Id:	OSC-INTMAN-BOOTLOADER-0021
Version:	1
Description:	The integrator shall ensure that the API BM_SoftwareInit is implemented and if necessary contains some specific software initialization.

3.2.9.4.1. BM_CustomHsmVerifyMac Callback

This callback is called when the bootloader software verifies the CMAC value of a given memory block occupied by the application software.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0250
Version:	1
Description:	<p>This callback must be implemented by the integrator if the feature "Secure Boot with HSM" is enabled. If EB HSM firmware is used, the integrator shall do the following steps:</p> <ul style="list-style-type: none"> ▶ The integrator shall call the API eb_hsm_mem_block_verify() to start the CMAC value verification asynchronously on HSM channel 0 if the job is not yet started. ▶ The integrator shall call the API eb_hsm_poll_channel() to get the status of HSM channel 0 every time this callback is entered. ▶ The integrator shall call the API eb_hsm_get_result_channel() to get the job result of HSM channel 0 when the job is finished. ▶ The integrator shall return BM_HSM_JOB_OK only if the CMAC verification job is finished without any error. ▶ The integrator shall return BM_HSM_JOB_FAILED if the CMAC verification job cannot be started or is finished with some error. ▶ The integrator shall return BM_HSM_JOB_PENDING if the CMAC verification job is started but not yet finished. <p>If a third-party HSM firmware is used, the integrator shall call the right API to do the CMAC check of the given memory block asynchronously and report the result.</p>



3.2.9.4.2. BM_CustomGetMacKey Callback

This function is called at startup.

It allows the integrator to provide the MAC key and keylength.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0251
Version:	1
Description:	The integrator shall implement in BM_CustomGetMacKey callback to provide the MAC key and keylength. If the SHE module is present, the SHE key format shall be used.

3.2.9.5. IBM Callbacks

The Initial Boot Manager is the first program executed at the ECU startup, checks the validity flags to decide to jump to the Application or the Bootloader Updater or the Bootloader with the callbacks described below.

3.2.9.5.1. BM_CheckProgRequest Callback

This callback is called to check if a programming request has been received by the application.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTLOADER-0022
Version:	1
Description:	The integrator shall implement in BM_CheckProgRequest callback software allowing getting information from application if a programming request has been received (e.g: read a flag from noinit RAM shared between Bootloader and Application). Return TRUE if a programming request has been received, return FALSE if no programming request has been received.

3.2.9.5.2. BM_CustomCheckValidAppl Callback

This callback is called to check if the application is valid or not.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTLOADER-0023
Version:	1



Description:	The integrator shall implement in BM_CustomCheckValidAppl callback software performing a check of the full software (application and calibration) to identify if application software is in a state where it can be started (valid and coherent). This that e.g (integration dependent) been done by checking that validity flag of every block/segment that are part the application software and check that different block/segment are all coherent with e.g version check. Return TRUE if the application is valid, return FALSE if the application is not valid.
--------------	---

3.2.9.5.3. BM_CustomCheckValidBLU Callback

This callback is called to check if the Bootloader Updater is valid or not.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTLOADER-0024
Version:	1
Description:	The integrator shall in BM_CustomCheckValidBLU callback software allowing getting information to identify if the Bootloader Updater software is in a state where it can be started (e.g: read a flag from non-volatile memory shared between Pre-Boot Manager and Bootloader Updater). Return TRUE if the Bootloader Updater is valid, return FALSE if the Bootloader Updater is not valid.

3.2.9.5.4. BM_CustomCheckValidBL Callback

This callback is called to check if the Bootloader is valid or not.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTLOADER-0025
Version:	1
Description:	The integrator shall in BM_CustomCheckValidBL callback software allowing getting information to identify if the Bootloader/New Bootloader software is in a state where it can be started (e.g: read a flag from non-volatile memory shared between Pre-Boot Manager and Bootloader Updater). Return TRUE if the Bootloader is valid, return FALSE if the Bootloader is not valid.

3.2.9.5.5. BM_JumpToApplication Callback

This callback is called to perform a jump to the application.



Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTLOADER-0026
Version:	1
Description:	The integrator shall implement in PROG_JumpToApplication callback software allowing jumping to application start address.

3.2.9.5.6. BM_JumpToBLU Callback

This callback is called to perform a jump to the bootloader updater.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTLOADER-0027
Version:	1
Description:	The integrator shall implement in BM_JumpToBLU callback software allowing jumping to bootloader updater start address.

3.2.9.5.7. BM_JumpToBL Callback

This callback is called to perform a jump to the bootloader.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTLOADER-0028
Version:	1
Description:	The integrator shall implement in BM_JumpToBL callback software allowing jumping to bootloader start address.

3.2.9.6. BM_CustomDualBankInit Callback

This callback is called at Bootloader start

It is used to configure and initialize the dual memory bank.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0238
-----	-------------------------



Version:	1
Description:	The integrator shall implement in BM_CustomDualBankInit callback the configuration of the hardware for the use of dual memory banks. Also, this callback shall initialize all the needed data for the dual memory banks usage.

3.2.9.7. Startup callback

3.2.9.7.1. BM_CustomIsNormalStartup Callback

This callback is called at Bootloader start

It's used to determine if a normal or abnormal reset happened.

In case of abnormal startup, Bootloader will wait during a configurable time-out before jumping to a valid application. During this time Bootloader waits the reception of DiagnosticSessionControl(Programming) request. If the request is not received before time-out, Bootloader jumps to application (if valid). If received the Bootloader will stay in Bootloader mode.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0179
Version:	1
Description:	The integrator shall implement in BM_CustomIsNormalStartup callback software identifying if a normal or abnormal startup happened. Criteria are project/hardware specific (e.g power on reset can be considered as abnormal startup)

3.2.9.8. SA Callbacks

3.2.9.8.1. SA_CustomCalculateKey Callback

This callback is called on reception of SecurityAccess service to perform computation of the security key based on a random seed.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTSACBK-0001
Version:	1



Description:	In SA_CustomCalculateKey callback, the integrator shall implement the key computation based on the provided random value. Computation shall be done using the algorithm required for the project.
--------------	---

3.2.9.9. SA Antiscanning Callbacks

3.2.9.9.1. SA_CustomStoreAsRetryCnt Callback

This callback is called on reception of SecurityAccess service in case anti-scanning feature is activated.

It allows integrator storing in non-volatile memory the retry counter value used for anti-scanning feature.

Bootloader uses callback SA_CustomRestoreAsRetryCnt at startup to get the stored value.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTSACBK-0002
Version:	1
Description:	In SA_CustomStoreAsRetryCnt callback, the integrator shall implement the storage in non-volatile memory of the provided counter value.

3.2.9.9.2. SA_CustomRestoreAsRetryCnt Callback

This callback is called at Bootloader startup (if anti-scanning feature is activated) to get the retry counter value from non-volatile memory.

Bootloader uses callback SA_CustomStoreAsRetryCnt to store the counter value.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTSACBK-0003
Version:	1
Description:	In SA_CustomRestoreAsRetryCnt callback, the integrator shall implement the get from non-volatile memory of the counter value. It shall be ensure that if value has never been written, the return value is 0.

3.2.9.10. Programming related callbacks

Callbacks from the following sections shall be implemented during integration task



3.2.9.10.1. PROG_CheckProgRequest Callback

This callback is called at Bootloader startup to know if a programming request has been received in Application.

If a programming request has been received in Application, Bootloader won't check application validity and stay in Bootloader mode and wait programming event from the network.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0100
Version:	1
Description:	The integrator shall implement in PROG_CheckProgRequest callback software allowing getting information from application if a programming request has been received (e.g: read a flag from noinit RAM shared between Bootloader and Application). PROG_BOOT_REPROG value shall be returned if a programming request has been received. PROG_BOOT_NO_REPROG shall be returned if no programming request has been received

3.2.9.10.2. PROG_JumpToApplication Callback

This callback is called at Bootloader startup if application is valid/coherent and shall be executed (no programming request received from application).

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0101
Version:	1
Description:	The integrator shall implement in PROG_JumpToApplication callback software allowing jumping to application start address.

3.2.9.10.3. PROG_isValidApplication Callback

This callback is called when application validity check is required

This can e.g happen at Bootloader startup to identify if Bootloader mode or application mode shall be processed.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0104
Version:	1



Description:	The integrator shall implement in PROG_isValidApplication callback software performing a check of the full software (application and calibration) to identify if application software is in a state where it can be started (valid and coherent). This that e.g (integration dependent) been done by checking that validity flag of every block/segment that are part the application software and check that different block/segment are all coherent with e.g version check. Return TRUE if application is valid, return FALSE if application is not valid.
--------------	---

3.2.9.10.4. PROG_InvalidateSection_BlockID Callback

This callback is called on Erase routine reception

Bootloader notify that an erasing will be performed on a logical block and allow integrator performing actions before an erasing (e.g invalidate application status)

Integration software is responsible to manage the application validity status. This callback can be used by integrator to manage the validity status (shall invalidate the block status with this callback).

If the feature Preliminary Erasing is activated, when the Erase request is received with the max block identifier, it shall perform the correspondent action (e.g. invalidate application status) in all segments.

Integrator has also the possibility to reject the erasing by returning a PROG_E_NOT_OK value (PROG_E_OK shall be returned if accepted)

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0107
Version:	1
Description:	The integrator shall implement in PROG_InvalidateSection_BlockID callback software performing operation that can be required by integration software before erasing and invalidating the logical block that will be erased. PROG_E_OK shall be returned if erasing is allowed, PROG_E_NOT_OK in other case.

3.2.9.10.5. PROG_SwitchApplicationModelInd Callback

This callback is called before Bootloader perform a jump to application

Bootloader notify that a jump to application will be performed and allow integrator performing actions before the jump

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0108
-----	-------------------------



Version:	1
Description:	The integrator shall implement in PROG_SwitchApplicationModeInd callback software performing operation that can be required before jumping to application.

3.2.9.10.6. PROG_GetSuppressBitFromAppli Callback

This callback is called when Bootloader shall send a response to a request that has been received in application

When ECU is running in application mode and receive a programming request, depending on configuration, the response can be sent by the Bootloader after resetting the ECU. In this case the Bootloader needs to know the UDS suppressPositiveResponse bit value from the request to identify if a response shall be sent or not.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0113
Version:	1
Description:	The integrator shall implement in PROG_GetSuppressBitFromAppli callback software getting from application information if the suppressPositiveResponse bit was set in the received request (e.g: read a flag from noinit RAM shared between Bootloader and Application)

3.2.9.10.7. Fingerprint Callbacks

3.2.9.10.7.1. PROG_CustomWriteFingerprint Callback

This callback is called on reception of WriteDataByIdentifier service for Fingerprint DID

It allows integrator performing the Fingerprint data validity check and perform its storing in non-volatile memory.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0116
Version:	1
Description:	The integrator shall implement in PROG_CustomWriteFingerprint callback software checking the validity of Fingerprint data and performing the writing in non-volatile memory of the Fingerprint data (pubRamBuffer points on the dataIdentifier field of the WriteDataByIdentifier, allowing integrator identifying the fingerprint using the DID identifier value) request Asynchronous management can be im-



plemented, in this PROG_E_BUSY value is returned and further call to PROG_CustomGetWriteFingerprintStatus will allow Bootloader to get writting status.

3.2.9.10.7.2. PROG_CustomGetWriteFingerprintStatus Callback

This callback is called after PROG_CustomWriteFingerprint returns PROG_E_BUSY, this callback is called periodically until getting a status different from PROG_E_BUSY

It allows integrator implementing writting of the fingerprint.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0117
Version:	1
Description:	The integrator shall implement in PROG_CustomGetWriteFingerprintStatus callback software providing status of the fingerprint writting

3.2.9.10.8. Get Crypto Keys Callbacks

3.2.9.10.8.1. PROG_CustomGetAsymPublicKey Callback

This callback is called in order to get the cryptographic asymetrical public key from user defined location.

PROG_CustomGetAsymPublicKey has the following parameters:

- ▶ The pointer to the public key's modulus array
- ▶ The pointer to the public key's exponent

It allows integrator to fetch the asymetrical public key and give it to the bootloader.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0142
Version:	1
Description:	The integrator shall implement in PROG_CustomGetAsymPublicKey callback the fetching of the asymetrical public key.

Id:	OSC-INTMAN-BOOTCBK-0242
Version:	1



Description:	If required, the integrator shall check if the production key is written or not. If it is written, the production key shall be returned, otherwise return the development key.
--------------	--

3.2.9.10.8.2. PROG_CustomGetSymStaticKey Callback

This callback is called in order to get the cryptographic symmetrical static key from user defined location.

It allows integrator to fetch the symmetrical static key and give it to the bootloader.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0143
Version:	1
Description:	The integrator shall implement in PROG_CustomGetSymStaticKey callback the fetching of the symmetrical static key.

3.2.9.10.9. PROG_CustomDecryptData Callback

In order to use this callback the Tresos parameter "Enable_Decryption" shall be set to "TRUE" and parameter Enable_Csm_Decryption shall be set to "FALSE".

This callback is called on reception of a Transfer Data (and before decompression if activated)

It allows the integrator to implement the decryption algorithm.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0161
Version:	1
Description:	The integrator shall implement PROG_CustomDecryptData callback copying decrypted data at the same location than the encrypted one.

3.2.9.10.10. MemoryAccessNotification Callback

3.2.9.10.10.1. PROG_CustomMemoryAccessNotification Callback

This callback is called after a successful memory data access.

It allows the customers to place their routines.



However the callback is called only on synchronous memory access.

PROG_CustomMemoryAccessNotification has the following parameters:

- ▶ Memory type (e.g. RAM, Flash or Flash Ext).
- ▶ Operation type (e.g. Read, Write or Erase).
- ▶ Start address of the memory.
- ▶ Data length.
- ▶ Data buffer.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0162
Version:	1
Description:	The integrator shall implement in PROG_CustomMemoryAccessNotification callback the procedure desired like an subsystem update.

3.2.9.10.11. Custom Memory Access Callbacks

3.2.9.10.11.1. PROG_CustomMemoryErase Callback

This callback is called upon receipt of an erasing request.

It allows the customer to implement his/her own erase routine.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0170
Version:	1
Description:	The integrator shall implement in PROG_CustomMemoryErase callback the desired erase routine. The callback should return one of the following macros PROG_E_OK, PROG_E_NOT_OK or PROG_E_BUSY, according to its memory access status. It is recommended to implement it asynchronously using the callback PROG_CustomMemGetJobStatus if it is a slow operation.

3.2.9.10.11.2. PROG_CustomMemoryWrite Callback

This callback is called upon receipt of an writing request.



It allows the customer to implement his/her own write routine.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0171
Version:	1
Description:	The integrator shall implement in PROG_CustomMemoryWrite callback the desired write routine. The callback should return one of the following macros PROG_E_OK, PROG_E_NOT_OK or PROG_E_BUSY, according to its memory access status. It is recommended to implement it asynchronously using the callback PROG_CustomMemGetJobStatus, if it is a slow operation.

3.2.9.10.11.3. PROG_CustomMemoryRead Callback

This callback is called upon receipt of an reading request.

It allows the customer to implement his/her own reading routine.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0172
Version:	1
Description:	The integrator shall implement in PROG_CustomMemoryRead callback the desired read routine. The callback should return one of the following macros PROG_E_OK, PROG_E_NOT_OK or PROG_E_BUSY, according to its memory access status. It is recommended to implement it asynchronously using the callback PROG_CustomMemGetJobStatus, if it is a slow operation.

3.2.9.10.11.4. PROG_CustomMemGetJobStatus Callback

This callback is called after an asynchronous memory access operation.

It allows the customer to get the status of the memory job.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0173
Version:	1
Description:	The integrator shall implement in PROG_CustomMemGetJobStatus callback the routine to get the memory job status. After a custom memory access this call-



back shall be called periodically until getting a status different from PROG_E_-BUSY.

3.2.9.10.11.5. PROG_CustomGetNextSectorAddr Callback

This callback is called after an synchronous memory access operation.

It allows the customer to get the address of the next memory sector.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0174
Version:	1
Description:	The integrator shall implement in PROG_CustomGetNextSectorAddr callback the routine to get the next sector memory address.

3.2.9.10.12. Dual Memory Bank Callbacks

3.2.9.10.12.1. PROG_CustomCalcInactiveBankWriteAddr Callback

This callback is called whenever an erase or write to the inactive memory bank is needed.

It allows the integrator to perform the calculation of the address in the inactive bank where the erase/write will be done.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0239
Version:	1
Description:	The integrator shall implement in PROG_CustomCalcInactiveBankWriteAddr callback the calculation of the erase or write address in the inactive bank based on the offset between banks and the current active memory bank.

3.2.9.10.12.2. PROG_CustomCalcInactiveBankReadAddr Callback

This callback is called whenever a read from the inactive memory bank is needed.

It allows the integrator to perform the calculation of the address in the inactive bank from where the read be done.



Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0240
Version:	1
Description:	The integrator shall implement in PROG_CustomCalcInactiveBankReadAddr callback the calculation of the read address in the inactive bank based on the offset between banks and the current active memory bank.

3.2.9.10.12.3. PROG_CustomBankSwap Callback

This callback if required shall be called on reception of Routine control 0xDF00 to perform the Back swap.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0273
Version:	1
Description:	The integrator shall implement in PROG_CustomCalcInactiveBankReadAddr callback UDS processing of Routine request 0xDF00 and functions for Bank swap if download and verification of application is successful in inactive memory bank

3.2.9.10.12.4. PROG_CustomBankConfirmRollback Callback

This callback if required shall be called on reception of Routine control 0x022A to perform the confirmation or Rollback of Back swap.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0274
Version:	1
Description:	In PROG_CustomBankConfirmRollback , the integrator shall implment UDS processing of the Routine request and Confirmation of Bank swap after download and verification of application is successful in inactive memory bank or Rollback of Swap. No further swap possible until a new Valid application is download. And this shall be checked at each swap request

3.2.9.10.13. Programming Counter Callbacks

In order to use those callbacks the Tresos parameter "Programming Counter Use" shall be checked.



Callbacks from the following sections shall be implemented during integration task for the Programming Counter feature.

The bootloader will get the stored value of the programming counter through the PROG_CustomGetProgCounter callback and will check it again the programming counter lock value in order to allow or reject the programming of a logical block. If the programming is allowed, the bootloader will call the PROG_CustomIncrementProgCounter to increment and store the programming counter for the programmed logical block.

If the blocks have already been completely erased, the programming counter is not incremented. Then, both callbacks PROG_CustomGetProgCounter and PROG_CustomIncrementProgCounter are not called.

3.2.9.10.13.1. PROG_CustomGetProgCounter Callback

This callback is called on reception of EraseMemory routine.

It allows the integrator to fetch, from non-volatile memory, the value of the programming counter for the logical block that needs to be programmed and to pass it to the bootloader. The callback is called on each erase routine.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0136
Version:	1
Description:	The integrator shall retrieve from the non-volatile memory the value of the programming counter. PROG_CustomGetProgCounter has the following parameter: The block Id for which the information is fetched

3.2.9.10.13.2. PROG_CustomIncrementProgCounter Callback

This callback is called by PROG_CheckProgrammingCounter if the programming is allowed.

It allows the integrator to increment and store the value of the programming counter for the logical block that needs to be programmed. The callback is called on each allowed erase routine.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0137
Version:	1
Description:	The integrator shall increment the value of the programming counter in the non-volatile memory. PROG_CustomIncrementProgCounter has the following parameter: The block Id for which the information is stored



3.2.9.10.14. Bootloader Updater Download Callbacks

3.2.9.10.14.1. PROG_CustomSetBLUDownloadInProgress Callback

This callback is called when the Bootloader Updater Program download is complete, and need to reset and switch to the Bootloader updater to update the bootloader.

It indicates that the bootloader is being updated in order that the new bootloader restore the state as kept before the reset by the old bootloader.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0231
Version:	1
Description:	The integrator shall implement in PROG_CustomSetBLUDownloadInProgress callback software updating the flag that indicates the Bootloader Updater Program download is in progress and the bootloader is being updated

3.2.9.10.14.2. PROG_CustomIsBLUDownloadInProgress Callback

This callback is called during the bootloader sequence to know if the bootloader has been updated, to restore the state kept before the reset by the old bootloader and continue the sequence.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0225
Version:	1
Description:	The integrator shall implement in PROG_CustomIsBLUDownloadInProgress callback software to get the flag that indicates if the Bootloader Updater Program download is in progress and the bootloader is being updated

3.2.9.10.14.3. PROG_CustomIsBLUPatternPresent Callback

This callback is called at CRC verification to check the presence of Bootloader Updater Pattern

It allows the integrator to compare an area of the flashed datas with a known pattern to check if the current logical block or segment is a Bootloader Updater software (e.g An array of bytes defined in the Bootloader Updater software at a known or relative address).



Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0228
Version:	1
Description:	<p>The integrator shall implement in PROG_CustomIsBLUPatternPresent callback software a read on the flashed data at the area of the Bootloader Updater pattern and check if it matches the known Bootloader Updater pattern. The address and size of the pattern can be decided in integration, the address is depending on the Addressing Modes in request download configuration</p> <ul style="list-style-type: none"> ▶ In case Logical Block : The pattern need to be inside the logical block, the address of pattern can be fixed or relative to the logical block memory area ▶ In other case : The pattern need to be inside the last segment of the Bootloader Updater, the address of pattern can be fixed or relative to the last segment memory area the Bootloader Updater <p>This function take as parameters :</p> <ul style="list-style-type: none"> ▶ ubLogicalBlockId : the current logical block Id, used for Logical Block case ▶ ubLogicalSegmentId : the current segment Id, used for other case <p>This function shall return :</p> <ul style="list-style-type: none"> ▶ PROG_TRUE if the Bootloader Updater pattern is present ▶ PROG_FALSE if the Bootloader Updater pattern is absent <p>Specific solution :</p> <ul style="list-style-type: none"> ▶ In case Logical Block : If the BLUpdater block id is fixed, check whether the current block corresponds to the BLUpdater block or not can be done only with comparing the logical block id, no need to define the pattern. ▶ In other case : If the BLUpdater last segment is fixed, check whether the current segment corresponds to the last one of BLUpdater or not can be done only with comparing the segment id, no need to define the pattern.

3.2.9.10.14.4. PROG_CustomSetBLUVerificationSuccess Callback

This callback is called after a successful CRC verification to validate the Bootloader Updater

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0229
------------	-------------------------



Version:	1
Description:	<p>The integrator shall implement in PROG_CustomSetDownloadVerificationSuccess callback software updating the Bootloader Updater validity status. This function shall return :</p> <ul style="list-style-type: none"> ▶ PROG_E_OK if no error occurs. ▶ PROG_E_NOT_OK in any error occurs.

3.2.9.10.15. Symetric Decryption Callbacks

3.2.9.10.15.1. PROG_CustomDecryptGetInitVector Callback

This callback is called on the start of the decryption data process.

PROG_CustomDecryptGetInitVector has the following parameters:

- ▶ The pointer to the init vector's array
- ▶ The pointer to the init vector's length

It allows getting the decryption IV and its length and give it to the bootloader.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0261
Version:	1
Description:	The integrator shall implement in PROG_CustomDecryptGetInitVector callback the fetching of the IV used for the decryption.

3.2.9.10.15.2. PROG_CustomGetSymDecryptionKey Callback

This callback is called on the start of the decryption data process.

PROG_CustomGetSymDecryptionKey has the following parameters:

- ▶ The pointer to the decryption key's array
- ▶ The pointer to the decryption key's length

It allows getting the decryption key and its length.

Callback definition can be found in [module reference chapter](#)



Id:	OSC-INTMAN-BOOTCBK-0262
Version:	1
Description:	The integrator shall implement in PROG_CustomGetSymDecryptionKey callback the fetching of the symetrical decryption key.

3.2.9.10.16. PROG_CustomCsmStrtPreproc Callback

This callback is called before the start of the CSM ASR v4.3(or higher) mode start operation

It allows the integrator to perform project specific or the algorithm specific implementation.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0275
Version:	1
Description:	This is an optional callback implementation, the integrator shall implement in PROG_CustomCsmStrtPreproc callback the project specific or the algorithm specific operations. For eg.: implementing the DER encoding of the plain key needed for the RSASSA_PSS signature verification algo.

3.2.9.11. BIPduR related callbacks

Callbacks from the following sections shall be implemented during integration task

3.2.9.11.1. BIPduR_GetRxPduld Callback

This callback is called each time the ECU reset and need to retrieve connection context in order to respond to a request received before this reset (received in application or bootloader mode).

It allows Bootloader to retrieve the Rx Pdu Id that shall be used to send a response to a diagnostic request received before the reset.

This value is completly project dependent and it is up to the integrator to define its own way to find the correct Rx Pdu ID

The value shall be get either from a previous call (before reset) to callback BIPduR_StoreRxPduld or shall be get from application.

Callback definition can be found in [module reference chapter](#).



Id:	OSC-INTMAN-BOOTCBK-0145
Version:	1
Description:	The integrator shall implement in callback BIPduR_GetRxPduld software getting the Rx Pdu Id to be used to respond to the request received before the reset.

3.2.9.11.2. BIPduR_StoreRxPduld Callback

This callback is called each time the Bootloader will reset and need to store connection context in order to retrieve it later.

It allows Bootloader or application to identify the Tx Pdu Id to be used to send the response after a reset.

Callback definition can be found in [module reference chapter](#).

Id:	OSC-INTMAN-BOOTCBK-0146
Version:	1
Description:	The integrator shall implement in callback BIPduR_StoreRxPduld the storage of the Rx Pdu Id to be used by ECU in order to send a response after a reset.

3.2.9.11.3. BIPduR_Custom_Com_Init Callback

This callback is called inside BIPduR_Init1 function, it initializes all the modules of Communication stack

It allows Bootloader to initialize all the modules of the communication stack

Initialization of communication stack is project dependent and it is up to the integrator to define which Communication protocol stack to be initialized based on OEM and their requirement (e.g. CAN or FR or Eth or Lin)

This function will return void

Callback definition can be found in [module reference chapter](#).

Id:	OSC-INTMAN-BOOTCBK-0168
Version:	2
Description:	The Integrator will add the initialization of all the modules in the communication stack and it is up to the integrator to define which Communication protocol stack to be initialized based on OEM and their requirement (e.g. CAN or FR or Eth or Lin). The integrator shall ensure that the PDUID of the BIPdul connection is initialized at the startup if it is not a software reset.



3.2.9.11.4. BIPduR_Custom_Com_Deactivate Callback

This callback is called when Bootloader wants to disable the communication or switch the state machine to NO communication mode

It allows Bootloader to Deactive or switch to communication mode

Deactivation/Disabling of communication is project dependent and it is up to the integrator to define which Communication protocol stack to be Disabled based on OEM and their requirement (e.g. CAN or FR or Eth or Lin)

This function will return void

Callback definition can be found in [module reference chapter](#).

Id:	OSC-INTMAN-BOOTCBK-0169
Version:	1
Description:	The Integrator will add disabling the communication or switch the state machine to NO communication mode code and it is up to the integrator to define which Communication protocol stack to be disabled based on OEM and their requirement (e.g. CAN or FR or Eth or Lin)

3.2.9.12. BLUpdater related callbacks

Callbacks from the following sections shall be implemented during integration task

3.2.9.12.1. Bootloader Updater Callbacks

3.2.9.12.1.1. BLU_CustomSetValidityNewBootloader Callback

This callback is called when we need to update the validity flag for the update of new bootloader image.

It allows the integrator to set a validity flag when the new bootloader image is successfully updated.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0243
Version:	1
Description:	The integrator shall implement in BLU_CustomSetValidityNewBootloader callback the marker to set up the validity flag when the new bootloader image is successfully updated.



3.2.9.12.1.2. BLU_CustomSetValidityBLUpdater Callback

This callback is called when we need to update the invalidity flag to invalidate the bootloader updater.

It allows the integrator to set an invalidity flag to invalidate the bootloader updater such that new application can be updated there.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0244
Version:	1
Description:	The integrator shall implement in BLU_CustomInvalidateBLUpdaterStatus callback the marker to set up the invalidity flag to invalidate the bootloader updater.

3.2.9.12.1.3. BLU_CustomGetPdulID Callback

This callback is called when we need to get the PdulID from the bootloader

It allows the integrator to obtain a PdulID from the bootloader

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0245
Version:	1
Description:	The integrator shall implement in BLU_CustomGetPdulID callback to obtain the PdulID from the bootloader.

3.2.9.12.1.4. BLU_CustomTriggerWatchdog Callback

This callback is called when we need to trigger the watchdog timer.

It allows the integrator to trigger watchdog timer.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0246
Version:	1
Description:	The integrator shall implement in BLU_CustomTriggerWatchdog callback to trigger the watchdog timer.



3.2.9.12.1.5. BLU_Custom_Com_Init Callback

This callback is called to initialize all modules of communication stack

It allows the integrator to initialize all modules of communication stack

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0247
Version:	1
Description:	The integrator shall implement in BLU_CustomTriggerWatchdog callback to initialize all modules of communication stack.

3.2.9.12.2. Bootloader Updater with Secure Boot Callbacks

This callback is called to update the secure boot after bootloader update success.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0278
Version:	1
Description:	<p>The integrator shall implement in BLU_CustomBootChecksumUpdate callback the sequence to update the secure boot of the new boot :</p> <ul style="list-style-type: none">▶ Retrieve the current BOOT_MAC_KEY for authorization purpose.▶ The integrator shall call the API eb_hsm_load_key() to load the BOOT_MAC_KEY, a new one can be provided, or the current can be reused.▶ Generate the checksum of the new boot with following steps:<ul style="list-style-type: none">▶ The integrator shall call the API eb_hsm_secure_boot() to start the job of updating the CMAC value of a given bootloader range asynchronously by using HSM channel 0 if the job is not yet started.▶ The integrator shall call the API eb_hsm_poll_channel() to get the status of HSM channel 0 every time this callback is entered.▶ The integrator shall call the API eb_hsm_get_result_channel() to get the job result of HSM channel 0 when the job is finished.▶ The integrator shall return BLU_E_OK only if the CMAC value of the given bootloader range is updated without any error.



- ▶ The integrator shall return BLU_E_NOT_OK if the job cannot be started or is finished with some error.
- ▶ The integrator shall return BLU_E_BUSY if the job is started but not yet finished.

3.2.9.13. Communication related callbacks

Callbacks from the following sections shall be implemented during integration task

3.2.9.13.1. BIPduR_GetGroupIdVal Callback

This callback is called at ECU initialization if the feature [multiple identifier](#) is configured to External Notification

It allows Bootloader retrieving the Group Id that shall be used by the ECU. Bootloader will only accept identifiers belonging to this group, other won't be responded.

This value is completely project dependent and it is up to the integrator to define its own way to find the correct group ID (e.g. from an I/O value, from a NVM data store,...)

In case the value returned is out of range the default configured group ID will be selected

Callback definition can be found in [module reference chapter](#).

Id:	OSC-INTMAN-BOOTCBK-0138
Version:	1
Description:	If the feature multiple identifier is configured to External Notification the callback BIPduR_GetGroupIdVal shall be filled to return the group ID to be used

3.2.9.14. State machine guard callbacks

Callbacks from PROG_Guard.c shall be implemented during integration task as defined in section "Programming sequence adaptation" from Bootloader_OEMInd_specification document.

3.2.9.15. Response callbacks

Callbacks from PROG_Responses.c shall be implemented during integration task as defined in section "Response management" from Bootloader_OEMInd_specification document.



3.2.9.16. ReProgMemM callbacks

Callbacks from the following sections shall be implemented during integration task.

3.2.9.16.1. ReProgMemM callbacks

Callbacks from the following sections shall be implemented during integration task when the ReProgMemM module is used.

3.2.9.16.1.1. ReProgMemM_FlashInit Callback

This callback is called at ReProgMemM initialization at startup in order to initialize the flash driver(s).

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0285
Version:	1
Description:	The integrator shall implement in ReProgMemM_FlashInit callback software calling the initialization APIs of the flash drivers (internal and/or external).

3.2.9.16.1.2. ReProgMemM_DualBank_Init Callback

If the DualBank feature is enabled, this callback is called at ReProgMemM initialization at startup in order to initialize the DualBank process.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0286
Version:	1
Description:	The integrator shall implement in ReProgMemM_DualBank_Init callback software initializing the DualBank functionality based on the target.

3.2.9.16.1.3. ReProgMemM_CustomGetActiveBank Callback

If the DualBank feature is enabled, this callback is called at ReProgMemM initialization at startup in order to get the current active bank.

It will be used to update the address for memory access operations

Callback definition can be found in [module reference chapter](#)



Id:	OSC-INTMAN-BOOTCBK-0288
Version:	1
Description:	The integrator shall implement in ReProgMemM_CustomGetActiveBank callback software that will get the information of the current active bank based on the target.

3.2.9.16.1.4. ReProgMemM_CustomGetInactiveBankAddrOffset Callback

If the DualBank feature is enabled, this callback is called at ReProgMemM initialization at startup in order to get the inactive bank address offset.

It will be used to update the address for memory access operations.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0287
Version:	1
Description:	The integrator shall implement in ReProgMemM_CustomGetInactiveBankAddrOffset callback software that will get the information of the inactive bank address offset based on the target.

3.2.9.16.1.5. ReProgMemM_FIsDriver_JobStatus Callback

This callback is called cyclically to get the flash driver operation status in order to set a global status for the ReProgMemM module.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0289
Version:	1
Description:	The integrator shall implement in ReProgMemM_FIsDriver_JobStatus callback software that will get the information on the current status of the flash driver. In case of asc_MemAcc module integration, the integrator shall get the driver current status from the MemAcc_GetJobResult API.

3.2.9.16.1.6. ReProgMemM_FIsExtDriver_JobStatus Callback

This callback is called cyclically to get the external flash driver operation status in order to set a global status for the ReProgMemM module.



Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0290
Version:	1
Description:	The integrator shall implement in ReProgMemM_FlsExtDriver_JobStatus callback software that will get the information on the current status of the external flash driver. In case of asc_MemAcc module integration, the integrator shall get the driver current status from the MemAcc_GetJobResult API.

3.2.9.16.1.7. ReProgMemM_FlsDriver_Erase Callback

This callback is called cyclically for the memory erasing operation in flash memory.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0291
Version:	1
Description:	The integrator shall implement in ReProgMemM_FlsDriver_Erase callback software that will call the flash driver erase routine. If the DualBank feature is not used and the erase operation happens on the same bank, the flash driver erase routine shall be copied and executed in RAM memory. In case of asc_MemAcc module integration, the integrator shall call the MemAcc_Erase API.

3.2.9.16.1.8. ReProgMemM_FlsExtDriver_Erase Callback

This callback is called cyclically for the memory erasing operation in external flash memory.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0292
Version:	1
Description:	The integrator shall implement in ReProgMemM_FlsExtDriver_Erase callback software that will call the external flash driver erase routine. In case of asc_MemAcc module integration, the integrator shall call the MemAcc_Erase API.

3.2.9.16.1.9. ReProgMemM_FlsDriver_Write Callback

This callback is called cyclically for the memory writing operation in flash memory.



Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0293
Version:	1
Description:	The integrator shall implement in ReProgMemM_FIsDriver_Write callback software that will call the flash driver write routine. If the DualBank feature is not used and the write operation happens on the same bank, the flash driver write routine shall be copied and executed in RAM memory. In case of asc_MemAcc module integration, the integrator shall call the MemAcc_Write API.

3.2.9.16.1.10. ReProgMemM_FIsExtDriver_Write Callback

This callback is called cyclically for the memory writing operation in external flash memory.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0294
Version:	1
Description:	The integrator shall implement in ReProgMemM_FIsExtDriver_Write callback software that will call the external flash driver write routine. In case of asc_MemAcc module integration, the integrator shall call the MemAcc_Write API.

3.2.9.16.1.11. ReProgMemM_FIsDriver_Read Callback

This callback is called cyclically for the memory reading operation in flash memory.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0295
Version:	1
Description:	The integrator shall implement in ReProgMemM_FIsDriver_Read callback software that will call the flash driver read routine. In case of asc_MemAcc module integration, the integrator shall call the MemAcc_Read API.

3.2.9.16.1.12. ReProgMemM_FIsExtDriver_Read Callback

This callback is called cyclically for the memory reading operation in external flash memory.



Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0296
Version:	1
Description:	The integrator shall implement in ReProgMemM_FlsExtDriver_Read callback software that will call the external flash driver read routine. In case of asc_MemAcc module integration, the integrator shall call the MemAcc_Write API.

3.2.9.16.1.13. ReProgMemM_CustomGetOffset Callback

This callback is called awhen a memory operation request is received in order to update the operation address.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0297
Version:	1
Description:	The integrator shall implement in ReProgMemM_CustomGetOffset callback software that will return the offset to be added in the request operation address.

3.2.9.16.1.14. ReProgMemM_CustomGetPhysicalToLogicalAddress Callback

This callback is called cyclically for every memory operation in order to get the logical address to be used for the operation.

Callback definition can be found in [module reference chapter](#)

Id:	OSC-INTMAN-BOOTCBK-0298
Version:	1
Description:	The integrator shall implement in ReProgMemM_CustomGetPhysicalToLogical-Address callback software that will convert, depending on the target, the physical address contained in the memory operation request into a logical address to be processed by the flash drivers.

3.2.10. Downloadable Flash driver Feature

Gives the possibility to download flash routines to the RAM.



The Flash routines are the functions responsible to write/erase the flash, so the functions are required to be executed in RAM to modify the FLASH.

3.2.10.1. Activation of the feature

The following steps describe the feature activation.

Id:	OSC-INTMAN-FLSDOWN-0170
Version:	1
Description:	<p>If downloading of the flash routines is required for the project the following shall be performed:</p> <ul style="list-style-type: none"> ▶ Activate the feature in PROG module by setting "Download FFlash driver" field .

Id:	OSC-INTMAN-FLSDOWN-0171
Version:	1
Description:	<p>If the rejection of any new attempt of the flash routines after a failed attempt is required for the project the following shall be performed:</p> <ul style="list-style-type: none"> ▶ Activate the feature in PROG module by setting "Download FFlash driver" field .

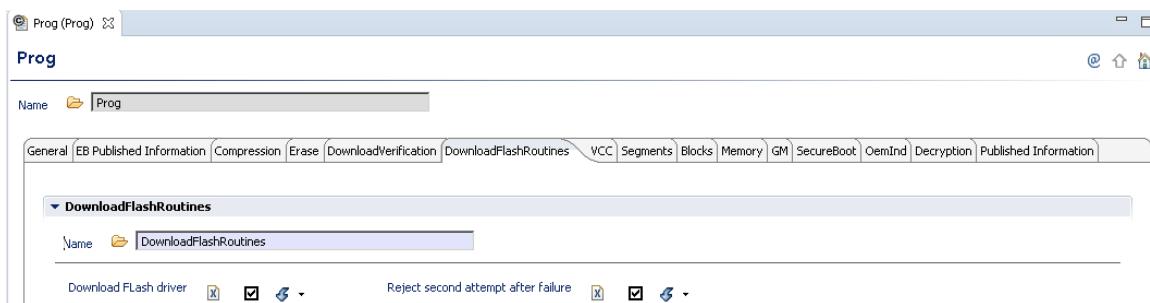


Figure 3.11. Activation of the feature Downloadable Flash driver

3.2.10.2. Configuration of the feature

The following steps describe the feature configuration.

Id:	OSC-INTMAN-FLSDOWN-0172
Version:	1
Description:	If not already existing,add a new Ram memory in the "Memory" tab.



Prog

I.	Name	Memory Type	Access Type	Min. Value	Address	Erase Value
0	Memory_FLASH	FLASH	synchronous	0x100	0	0x0
1	Memory_RAM	RAM	synchronous	0x8	0	0x0

Figure 3.12. Adding new RAM memory

Id:	OSC-INTMAN-FLSDOWN-0173
Version:	1
Description:	Add a new (and unique) segment for the flash driver routines, and put the addresses where these routines are mapped.
Id:	OSC-INTMAN-FLSDOWN-0174
Version:	1
Description:	Set the partition type of the segment to "PROG_FLASH_ROUTINES_PARTITION".
Id:	OSC-INTMAN-FLSDOWN-0175
Version:	1
Description:	Map the segment to the RAM memory create previously.
Id:	OSC-INTMAN-FLSDOWN-0176
Version:	1
Description:	Deactivate the validity check for this segment, or the generation will fail.

Prog

I.	Name	Memory	Access Type	Reprog_Start_Ad...	Reprog_End_Ad...	Erase_Start_Ad...	Erase_End_Ad...	Partition Type
0	FLASH_Driver_Seg	@/Prog/Prog/...	READ_WRITE	0x70014000	0x70015000	0x70014000	0x70015000	PROG_FLASH_ROUTINES_PARTITION

Figure 3.13. Adding new segment for flash routines

Id:	OSC-INTMAN-FLSDOWN-0177
Version:	1
Description:	Add a new block for the flash driver routines and set the "First segment" field to the created segment in the previous step.



Prog

I.	Name	First Segment	Number o...	Programm...	Block Ide...
0	Appli	@ /Prog/Prog/Application_Seg0	0x3	0xc	0x0
1	Calib	@ /Prog/Prog/Calibration_Seg0	0x2	0xc	0x1
2	Flash_Driver	@ /Prog/Prog/FLASH_Diver_Seg	0x1	0xc	0x2

Figure 3.14. Adding new block for flash routines

3.2.10.3. Integration of the feature

3.2.10.3.1. Linker file update

The following steps shall be done in the linker file in order to correctly map the Flash routines to the RAM:

Id:	OSC-INTMAN-FLSDOWN-0178
Version:	1
Description:	Add a linker file section in RAM to map all the code from MemMap section "FLASH_FLS_START_SEC_CODE".

3.2.10.3.2. Available APIs

The following APIs are available for the integrator :

- ▶ PROG_DrvDown_IsFlashRoutinesPresent: Returns the value of m_ubFlashRoutinesPresent, that represents the presence of the flash routines in RAM.
PROG_TRUE Flash routines are present in RAM.
PROG_FALSE Flash routines are not present in RAM.

3.2.11. Compressed Flash driver Feature

Gives the possibility to decompress flash routines to the RAM.

The Flash routines are the functions responsible to write/erase the flash, so the functions are required to be executed in RAM to modify the FLASH.



It is assumed that the binary flashed into the ROM of the ECU contains compressed flash routines.

3.2.11.1. Activation of the feature

The following steps describe the feature activation.

Id:	OSC-INTMAN-FLSCOMP-0000
Version:	1
Description:	If decompression of the flash routines is required for the project the following shall be performed: ► Activate the feature in PROG module by setting "Decompress Flash driver" field.

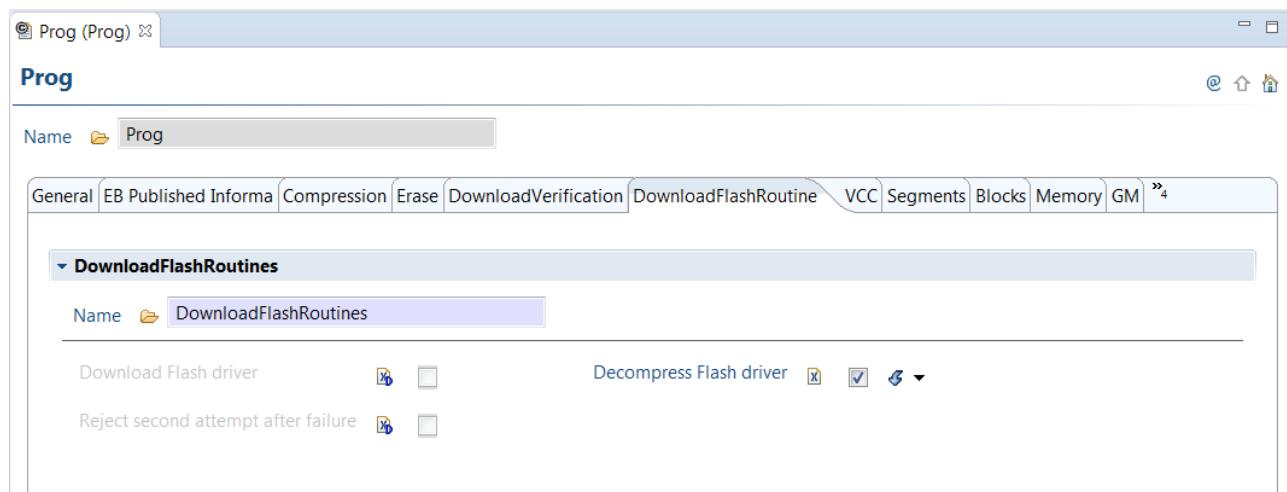


Figure 3.15. Activation of the feature Compressed Flash driver

3.2.11.2. Integration of the feature

3.2.11.2.1. Linker file update

The following steps shall be done in the linker file in order to correctly map the Flash routines to the RAM:

Id:	OSC-INTMAN-FLSCOMP-0001
Version:	1
Description:	► Add a linker file section in RAM to map all the code from MemMap section "FLASH_FLS_START_SEC_CODE".



- ▶ The flash driver area in ROM section shall be located in a contiguous space area.
- ▶ The flash driver area in RAM section shall be located in a contiguous space area.

Id:	OSC-INTMAN-FLSCOMP-0002
Version:	1
Description:	<p>In the linker file, add the 4 following symbols:</p> <ul style="list-style-type: none"> ▶ <code>__FLASH_DRIVER_ROM_START_ADDRESS</code>: must be mapped to the beginning of the flash driver area in ROM section ▶ <code>__FLASH_DRIVER_ROM_END_ADDRESS</code>: must be mapped to the end of the flash driver area in ROM section ▶ <code>__FLASH_DRIVER_RAM_START_ADDRESS</code>: must be mapped to the beginning of the flash driver area in RAM section ▶ <code>__FLASH_DRIVER_RAM_END_ADDRESS</code>: must be mapped to the end of the flash driver area in RAM section

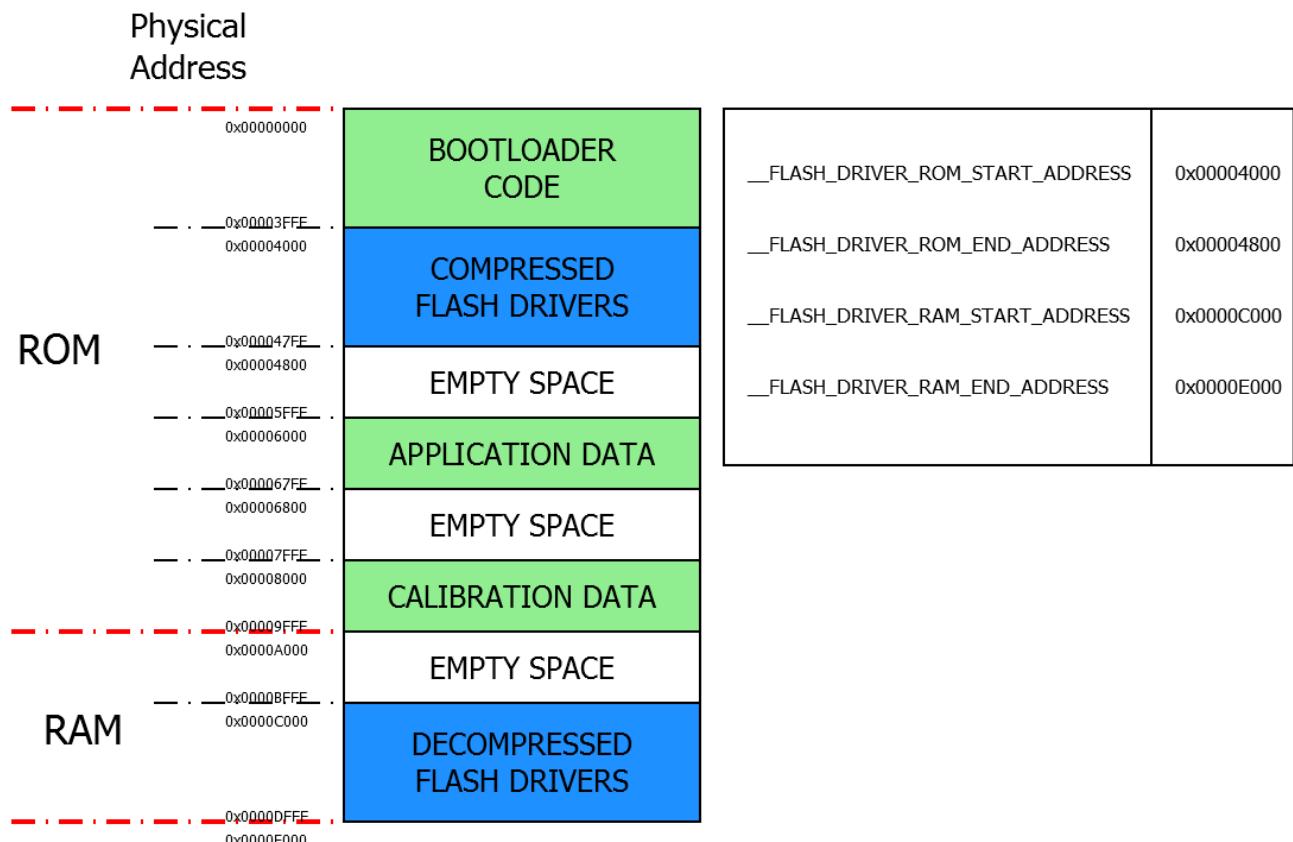


Figure 3.16. Example of Mapping of symbols

Id:	OSC-INTMAN-FLSCOMP-0003
Version:	1
Description:	Size of the compressed flash routines in ROM shall be provided in the last 4 bytes of the flash driver area in ROM section.

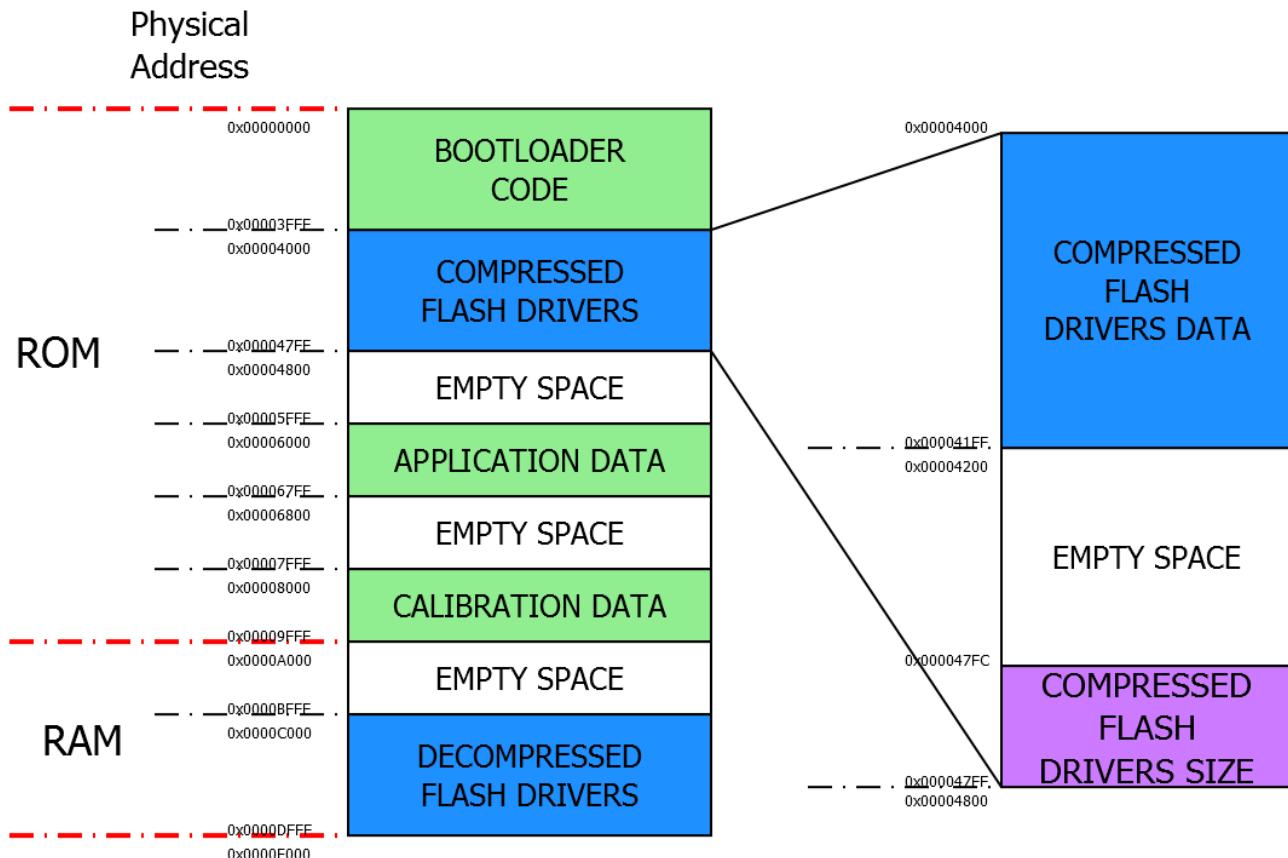


Figure 3.17. Location of Size of Compressed Flash Routines

3.2.12. Dual Memory Bank Feature

The dual memory bank feature allows the download of a new application in a different memory bank (inactive) while the bootloader and/or the application is running on the current memory bank (active).

The activation of the feature is done by checking the "Dual Memory Bank Used" checkbox in the "General" tab under Tresos Studio.

When the feature is activated, the EB tresos Bootloader will always download the new application/calibration in the inactive memory bank.

If required, A swap request, and Confirmation or Rollback needs to be done by the tester. The implementation of the banks swap, confirmation/rollback shall be done in integration code.



If required, A swap request shall only be allowed if the downloaded application is Valid (successful checks : Hash and Signature)

If required, For Commit and Rollback, no further swap possible until a new Valid application is downloaded.
This shall be checked at each swap request

The activation of the freshly programmed memory bank is done after the ECU reset.

After the banks swap, the software execution is done from the active bank. This means that the EB tresos Bootloader needs to be flashed in both banks to assure the startup and the application validity checks.

Following integration requirements apply when using Dual Memory Bank feature.

The following callbacks shall be implemented by integrator when feature is used:

- ▶ **BM_CustomDualBankInit:** Shall implement the activation of the dual bank capability of the hardware and the initialization of the needed data for the memory banks manipulation.
- ▶ **PROG_CustomCalcInactiveBankWriteAddr:** Shall implement the calculation of the addresses to be erased or written in the inactive memory bank.
- ▶ **PROG_CustomCalcInactiveBankReadAddr:** Shall implement the calculation of the addresses to be read from the inactive memory bank.
- ▶ **PROG_CustomSetDownloadVerificationSuccess:** Shall be used to identify, a successful download verification, if required

Id:	OSC-INTMAN-DUALBANK-0001
Version:	1
Description:	If dual memory bank feature is required, the integrator shall program the EB tresos Bootloader in both active and inactive memory banks.

Id:	OSC-INTMAN-DUALBANK-0002
Version:	1
Description:	If dual memory bank feature is required, the integrator shall implement and manage the swap request (UDS service configuration and handler).

Id:	OSC-INTMAN-DUALBANK-0003
Version:	1
Description:	If dual memory bank feature is required, the integrator shall assure that the memory banks configurations are identical in order to have similar performances of the software on both banks.

Id:	OSC-INTMAN-DUALBANK-0004
-----	--------------------------



Version:	1
Description:	If dual memory bank feature is required, the integrator shall implement and manage the Commit or Rollback Software (UDS service configuration and handler).

3.2.13. Secure Boot Feature

Secure Boot: Bootloader software verify the authenticity of Application/Bootloader software before their execution. This is to avoid to execute corrupted code. This verification is performed by a checksum computation (Hash/Mac) on Bootloader/Application software and a comparison with expected values (for the bootloader, a flashed checksum by the tester and for the application, an internal calculated checksum value on downloaded application data). If HSM SecureBoot feature is enabled please also refer to the section Secure_Boot_Hsm.

3.2.13.1. Activation of the feature

Following steps describe the activation of the feature.

Id:	OSC-INTMAN-BOOTCBK-0265
Version:	1
Description:	<p>If secure bootloader is required for the project the following shall be performed:</p> <ul style="list-style-type: none"> ▶ Activate the feature in BM module by setting "Authenticated / Secure Boot" field to "Secure".

3.2.13.2. Secure Boot Checksum Calculation

Following steps shall be performed for the calculation of the Bootloader checksum regarding the feature.

Note: Not applicable if HSM SecureBoot feature is enabled.

Id:	OSC-INTMAN-BOOTCBK-0266
Version:	1
Description:	<p>If secure bootloader is required for the project the following shall be performed:</p> <ul style="list-style-type: none"> ▶ Flash the bootloader without symbols. ▶ Flash the binary file containing the checksum [1] ▶ Then load the bootloader symbols



- ▶ [1] The binary file containing the checksum of the bootloader is calculated as follows :
 - ▶ Calculate the hash/Mac of the blocksize of the bootloader that needs to be authenticated
 - ▶ Write that value inside a binary file to be flashed at a specific address outside the bootloader area
 - ▶ Note: The hash should be calculated on a contiguous range of the bootloader without a gap. In the linker File, set a rom area at that specific address for the variable "m_aubBootloaderChecksum" that will have the checksum value. In the Memory mapping file, define the section of the variable.

3.2.13.3. Choice of the Checksum algorithm

The checksum calculation can be done with either Hash or Mac algorithms. Following steps shall be performed to select the correct Checksum algorithm. If HSM SecureBoot is enabled only MAC Algo is configurable.

Id:	OSC-INTMAN-BOOTCBK-0267
Version:	1
Description:	<p>To select the correct algorithm to compute the Secure Boot checksum, following steps shall be performed:</p> <ul style="list-style-type: none"> ▶ Set "BMCsmChecksumConfigId" in BM module to the correct configuration ▶ Set "ProgCsmSecureConfigId" in PROG module to the correct configuration ▶ In the case Mac algorithm is selected the configuration in PROG and BM plugin shall refer to a Mac configuration, otherwise an error will be mentioned

3.2.13.4. Checksum computation without Secure Boot activated

The checksum calculation of application can be done even if Secure Boot feature is deactivated.

Note: Not applicable if "Use HSM" is enabled.

Id:	OSC-INTMAN-BOOTCBK-0268
Version:	1
Description:	If Checksum computation for Secure Boot shall be done when the Secure Boot feature is deactivated, following steps shall be performed:



- ▶ "Authenticated / Secure Boot" is set to "OFF" in BM module.
- ▶ "Checksum computation" is activated in PROG module.

3.2.13.5. Bootloader Checksum Verification

Even if the feature Secure Boot is activated, the verification of checksum of the Bootloader may not be done at startup. If HSM Secureboot feature is enabled, the Boot verification happens at every startup and this field is not configurable.

Id:	OSC-INTMAN-BOOTCBK-0269
Version:	1
Description:	<p>If Bootloader checksum verification is not required at startup, the following steps shall be performed:</p> <ul style="list-style-type: none"> ▶ "Bootloader verified/ Bootloader not verified" is set to "OFF" in BM module.

3.2.13.6. Secure Boot Information by Block

If the feature Secure Boot is activated, the following information by Block/Segment shall be set (Start address, length, Block will be verified , Block can be blocker for the execution of the application).

Id:	OSC-INTMAN-BOOTCBK-0270
Version:	1
Description:	<p>If secure bootloader is required for the project the following information shall be set:</p> <ul style="list-style-type: none"> ▶ "Verified in Secure Boot": The checksum will be verified at startup. ▶ "Blocker for Software execution": In case the checksum controlled at startup is different from the expected one the Application won't be executed. ▶ "Start Address for the Secure Boot Verification": Start address of the area on which the checksum is computed. ▶ "Length of the Block area for the Secure Boot Verification": Length of the area on which the checksum is computed. ▶ Please note: The Block/Segment that shall be verified by HSM, should not have gaps/empty areas/non-filled areas within the memory area on which verification shall be performed by HSM.

3.2.14. Secure Boot with HSM feature

The Secure Boot feature can also be implemented by using HSM if that module is supported by the microcontroller. And "EB zenture HSM Firmware" is ordered.

1. The bootloader software can configure HSM firmware once so that HSM will calculate the checksum related to the **bootloader software** before starting its execution at each startup time. If the calculated value does not match with the value stored in HSM, the execution of the bootloader software will be blocked.
2. The bootloader software can ask HSM firmware to verify the checksum related to the **application software** (and/or its calibrations). If the verification fails, the bootloader software may reject launching the application software.
3. The bootloader software can ask HSM firmware to update the checksum related to the application software (and/or its calibrations) during CheckMemory routine control or Submit Signature Routine based on the OEM.

The communication between the bootloader software and the HSM firmware is ensured by EB HSM Proxy. So that plugin must also be integrated into the project. Please also refer to "EB zenture HSM Firmware" user manual for more information.

Note: this section and the section "Secure Boot Feature" are mutually exclusive.

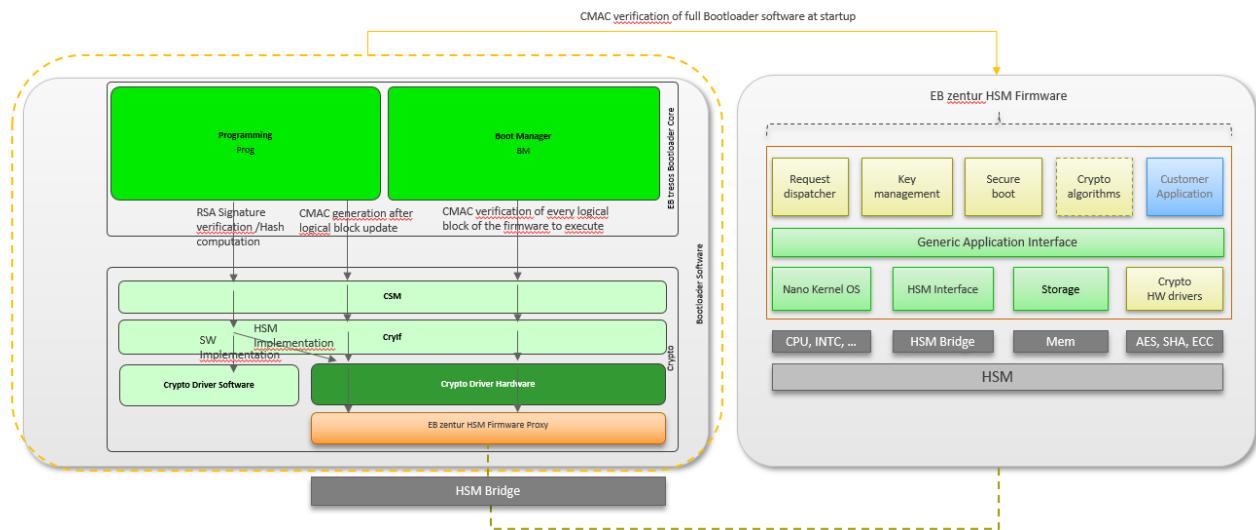


Figure 3.18. Layered view: Bootloader and HSM firmware

Id:	OSC-INTMAN-BOOTCBK-0257
Version:	1
Description:	To have a functional EB HSM Proxy, the integrator shall implementing the following functions in "eb_hsm_integ.c" in the bootloader project:



- ▶ eb_hsm_integ_GetCounterValue: get the tick value of a free-running timer.
- ▶ eb_hsm_integ_GetElapsedValue: get the number of ticks between the current tick value and a previous tick value.
- ▶ eb_hsm_integ_TICKS2MS: convert ticks to milliseconds.
- ▶ If Crypto ASR 4.3 higher version is used, add HSM Crypto Driver initialization before the call to BM_Startup API

3.2.14.1. Activation of the feature

Following steps describe the activation of the feature.

Id:	OSC-INTMAN-BOOTCBK-0255
Version:	1
Description:	<p>Do the following steps in order to enable the feature "Secure Boot with HSM" feature:</p> <ul style="list-style-type: none"> ▶ Activate the feature in BM module by setting "Authenticated / Secure Boot" field to "Secure". ▶ Check the box "Use HSM".

3.2.14.2. Secure Boot Checksum Calculation

HSM firmware is capable of calculating the checksum of a given memory area. But before doing that calculation, the firmware needs to be setup by some utility program (the details of this topic are out of the scope of the bootloader software). The following elements are needed:

- ▶ The master ECU key which is the source of authentication.
- ▶ The memory area (its start address and length in bytes).
- ▶ The key with which to calculate the checksum of the given memory area.

These elements as well as different checksums will always be safely kept inside HSM. And they cannot be exported. Incase of CMAC verification of Application or other configured Blocks using Crypto stack 4.3 and above, the CMAC shall not be stored in HSM and it will be the responsibility of the Integrator to configure to store in a secured non-volatile memory.

After the reset, the HSM firmware starts to run first. It systematically verifies the checksum of the bootloader software before giving the control to it. Then the bootloader software starts to run. Bootloader verifies (via HSM firmware) the checksum of the application software before giving the control to it.

ECU startup

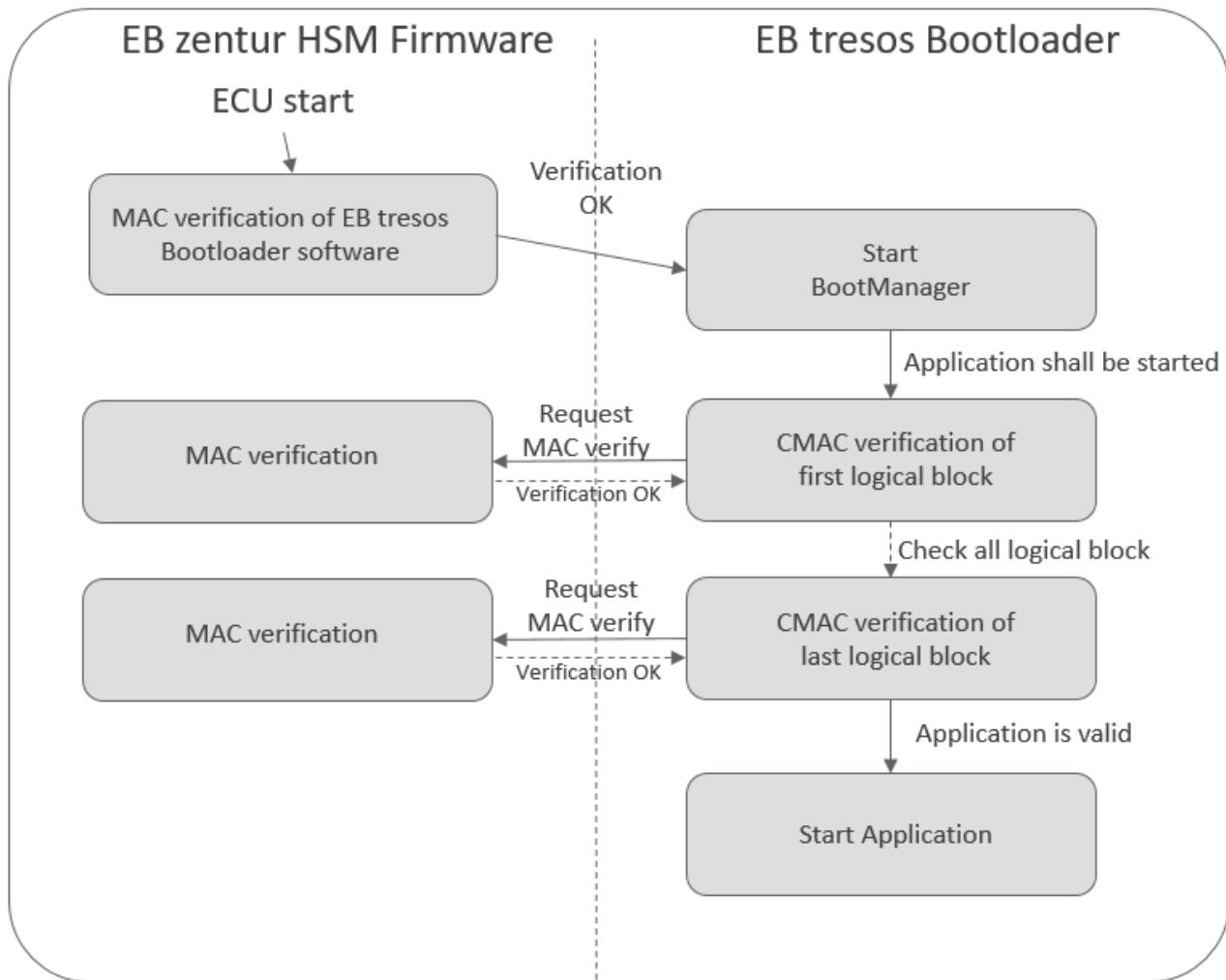


Figure 3.19. Startup sequence with HSM firmware integrated

The bootloader software is capable of updating the application software when it is asked to do so. It relies on the HSM firmware to update the checksum(s) of the application software during **CheckMemory routine control** or **Submit Signature Routine** based on the OEM.

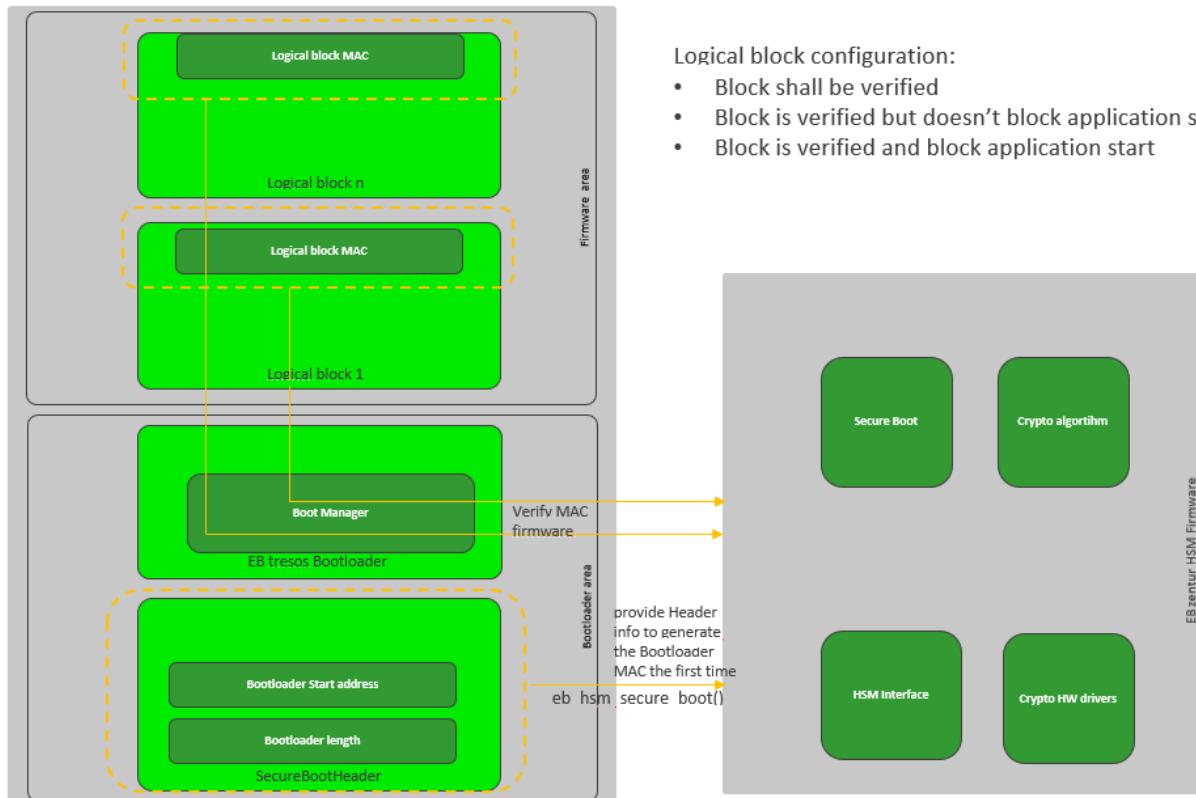


Figure 3.20. Checksum calculation by using HSM firmware

3.2.14.3. Choice of the Checksum algorithm

In the current version of the bootloader software, it only uses CMAC (Cipher based Message Authentication Code) algorithm to calculate the checksum.

3.2.14.4. Bootloader Checksum Verification

If the "Secure Boot with HSM" feature is enabled, the verification of the checksum of the bootloader software shall be performed during the startup process of the HSM firmware.

Id:	OSC-INTMAN-BOOTCBK-0258
Version:	1
Description:	During the startup process of the bootloader software, the integrator can call the function eb_hsm_get_she_status() to know if the "Secure Boot" feature offered by HSM firmware is activated or not by checking the bit value at the position COMM_SHE_STATUS_OFFSET_SECURE_BOOT:



- ▶ If that bit is set, then the feature is activated.
- ▶ If that bit is cleared, the feature is not yet activated. The integrator must call the function `eb_hsm_secure_boot()` with the start address (of the bootloader software) to be verified and its length.

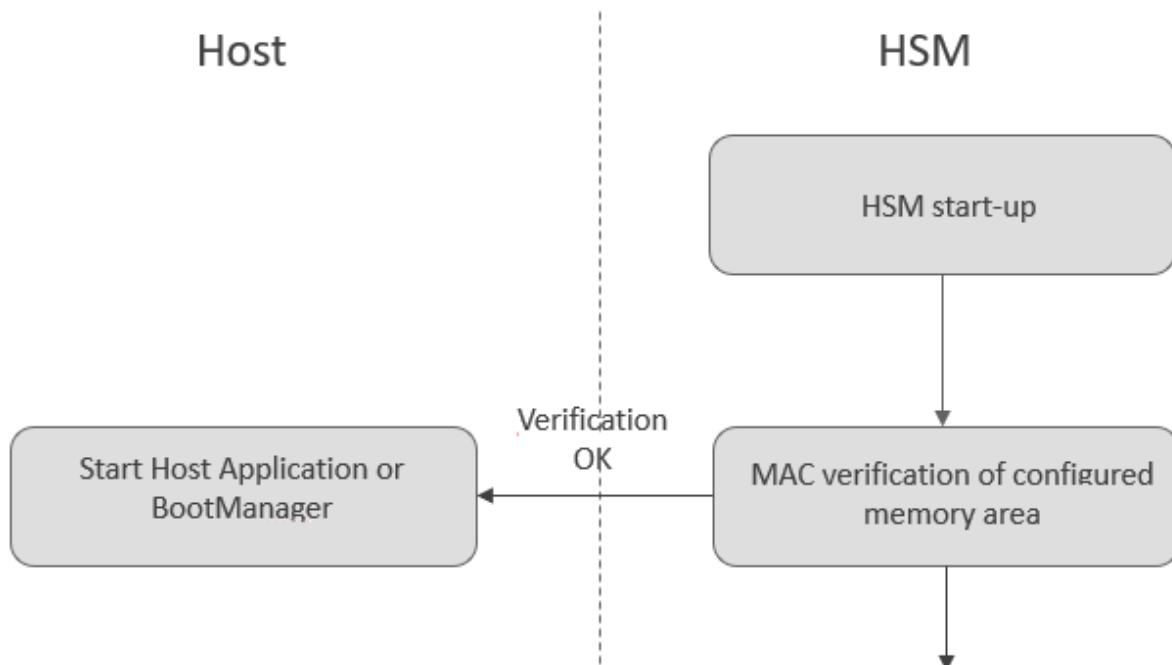


Figure 3.21. HSM firmware verifies the bootloader software

Note: some register(s) of the microcontroller can be configured in such a way if the checksum verification fails, HSM firmware does not start the bootloader software.

3.2.14.5. Application Software Memory Block Information

The bootloader software needs to know some information of the memory areas occupied by the application software in order to realize the "Secure Boot with HSM" feature:

Id:	OSC-INTMAN-BOOTCBK-0256
Version:	1
Description:	If the "Secure Boot with HSM" feature is enabled, the integrator needs to fill the following fields under PROG/Blocks/ */SecureBoot, if the project does not sup-



port "Blocks" configuration, then the integrator needs to fill the following fields under PROG/Segments/ */SecureBoot.

- "Verified in Secure Boot": The checksum of this memory block/segment will be verified at startup of the bootloader software.
- "Blocker for Software execution": In case the checksum controlled at startup is different from the expected one the Application won't be executed.
- "Start Address for the Secure Boot Verification": Start address of the area on which the checksum is computed.
- "Length of the Block area for the Secure Boot Verification": Length of the area on which the checksum is computed.

NOTE: The BLock/Segment that shall be verified by HSM, should not have gaps/empty areas/non-filled areas within the memory area on which verification shall be performed by HSM.

If the memory area information (start address and length) put here is different from which is used to setup the HSM firmware, the CheckMemory routine control will fail. This is not applicable if Crypto stack ASR version 4.3 and above is used for performing the HSM based Application Secure Checksum verification from Bootloader.

Note: the number of memory blocks which can be managed by the HSM firmware is limited. Please refer to "EB zenture HSM Firmware User Manual" for more information.

3.2.15. Bootloader Updater Feature

The bootloader updater feature allows to reprogram the bootloader as if updating an application. When the bootloader updater feature is active, the Bootloader Updater can be transferred like any other block and is capable of erasing the current bootloader and flashing a new bootloader during the download sequence.

The bootloader updater is executed once its transfer and verification succeed, during the execution of bootloader updater, continuous NRC78 will be sent with a constant configurable period.

The new bootloader image present in bootloader updater is capable of updating the application and calibration.

The bootloader updater feature supports a single step download and can update bootloader updater, new bootloader and new application in the same sequence.

3.2.15.1. Bootloader update sequence

Updating the bootloader can be done easily when bootloader updater feature is activated.

The procedure consists of downloading first, the block of bootloader updater including the new bootloader image, followed by the other blocks like application and calibration as we can see in the following figure :

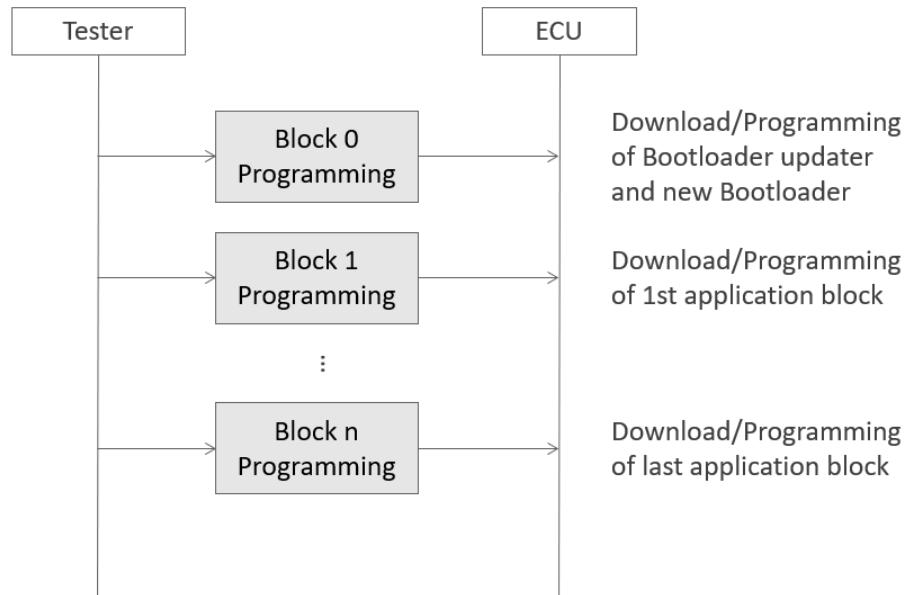
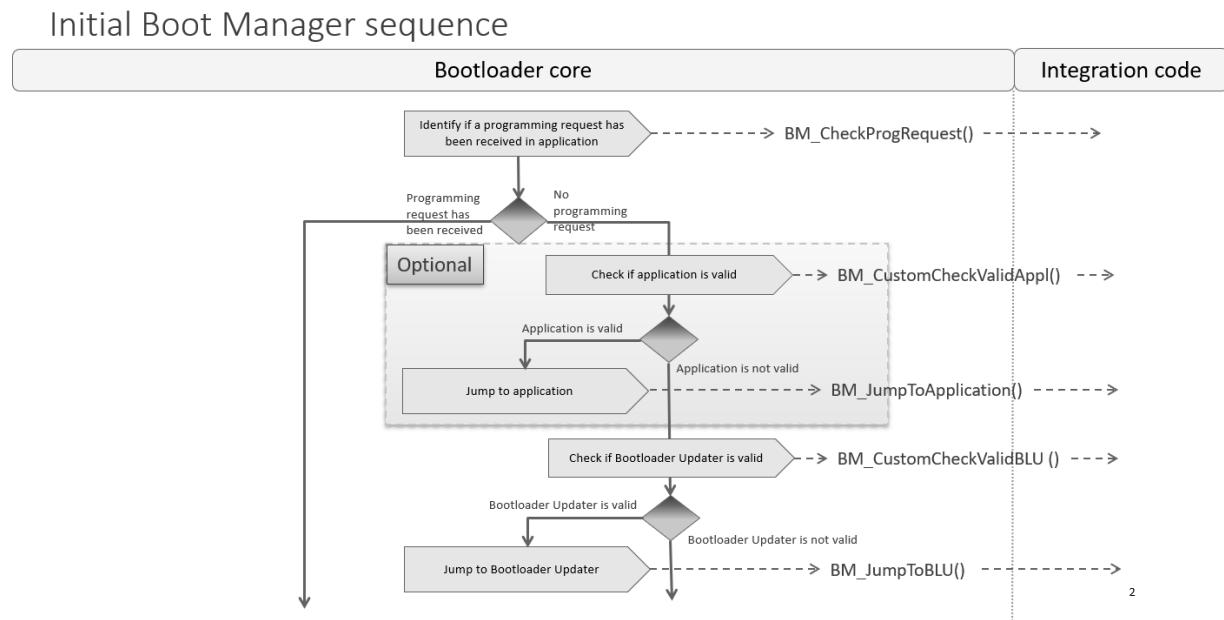


Figure 3.22. Bootloader update sequence

3.2.15.2. Bootloader Updater process

On the reset, a resident Boot Manager (BM) configured as **Initial Boot Manager (IBM)**, executes the Bootloader or the Bootloader updater software by checking the validity flags as described in the scheme :

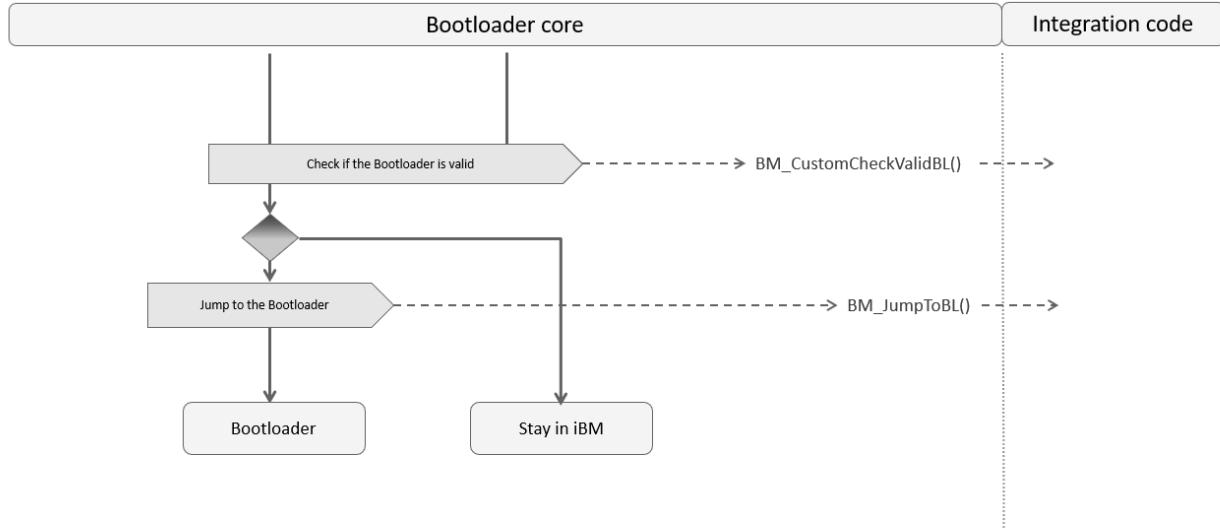


© Elektrobit (EB) 2016 | Confidential

Figure 3.23. Initial Boot Manager Sequence



Initial Boot Manager sequence



© Elektrobit (EB) 2016 | Confidential

3

Figure 3.24. Initial Boot Manager Sequence

Once the old bootloader started (Bootloader Updater is not yet valid), a download sequence can be started with downloading the Bootloader Updater block first, the update of Bootloader is done in several intermediate steps giving the impression to the user that he is just downloading a block :

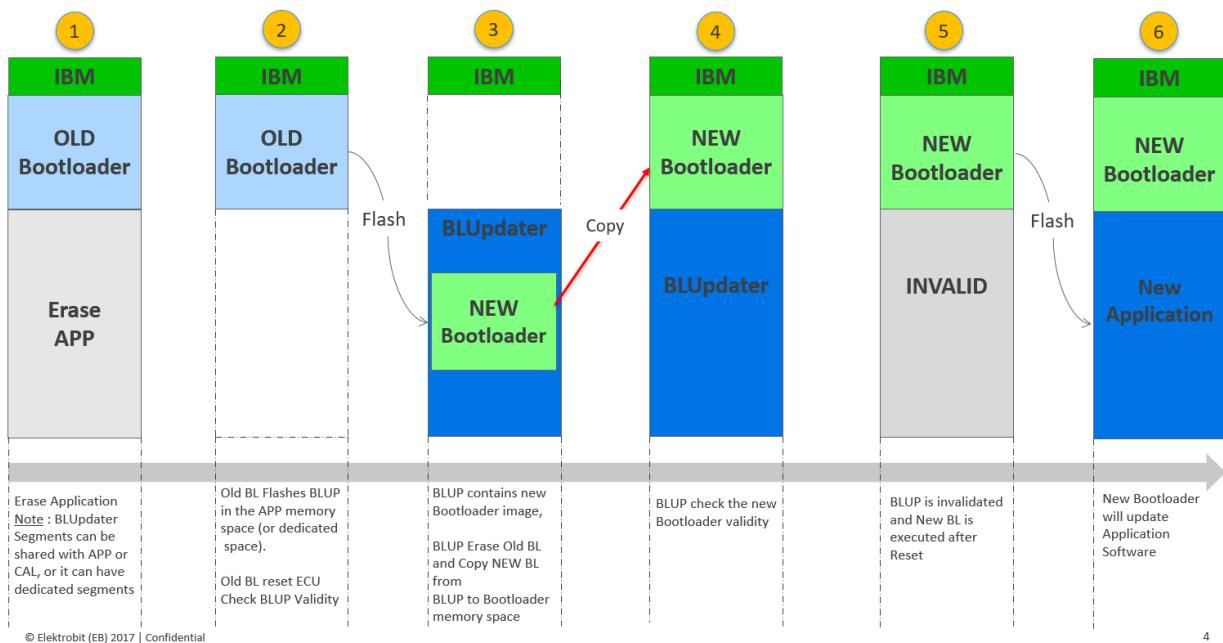


Figure 3.25. Bootloader Update Process

- 1. The old bootloader erase the application (or calibration, or dedicated BLUpdater partition) memory space which will host the Bootloader Updater.



- ▶ 2. The old bootloader Flashes Bootloader Updater Program in the application memory space (or calibration, or dedicated BLUpdater partition) erased before, validate the Bootloader Updater block and reset.
- ▶ 3. After restart IBM executes the Bootloader Updater as it's valid, the Bootloader Updater contains the complete new Bootloader image, it erases the old bootloader and copy the new bootloader to Bootloader memory space
- ▶ 4. The Bootloader Updater checks the new Bootloader validity and set the flag
- ▶ 5. The Bootloader Updater is invalidated and reset is performed
- ▶ 6. After restart, the Bootloader Updater is no longer valid, IBM executes the New Bootloader. Then, the New Bootloader answer the pending request of the Bootloader Updater block verification, and update Application Software and others blocks

3.2.15.3. Activation of the feature

The following steps describe the feature activation.

Id:	OSC-INTMAN-BLU-0001
Version:	1
Description:	<p>If downloading of the Bootloader Updater is required for the project, the feature can be activated automatically when one of these partitions type is configured in segments list :</p> <ul style="list-style-type: none"> ▶ PROG_BLU_APP_PARTITION : The corresponding Application segment memory area can be used to host BLUpdater. ▶ PROG_BLU_CAL_PARTITION : The corresponding Calibration segment memory area can be used to host BLUpdater. ▶ PROG_BLU_PARTITION : The corresponding segment memory area is dedicated to host only the BLUpdater.

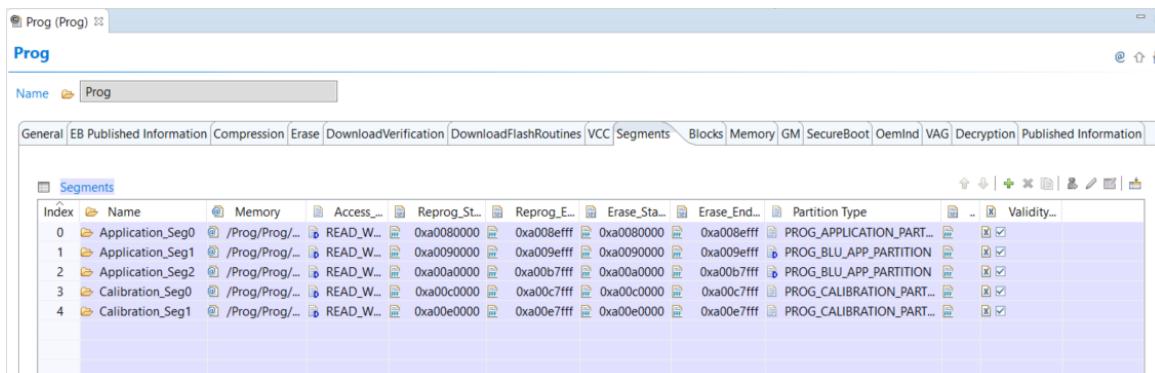


Figure 3.26. Activation of the feature Bootloader Updater



3.2.15.4. Configuration of the feature

The following steps describe the feature configuration.

Id:	OSC-INTMAN-BLU-0002
Version:	1
Description:	<p>To Configure the list of segments which will host the BLUpdater :</p> <ul style="list-style-type: none"> ▶ You can reuse the already configured segments for Application or Calibration by setting the partition type to one which can hosts the BLUpdater. ▶ PROG_BLU_APP_PARTITION : segment shared between Application and BLUpdater. ▶ PROG_BLU_CAL_PARTITION : segment shared between Calibration and BLUpdater. ▶ If BLUpdater needs to be in separate memory area, add PROG_BLU_PARTITION segments type, which will be dedicated for BLUpdater.

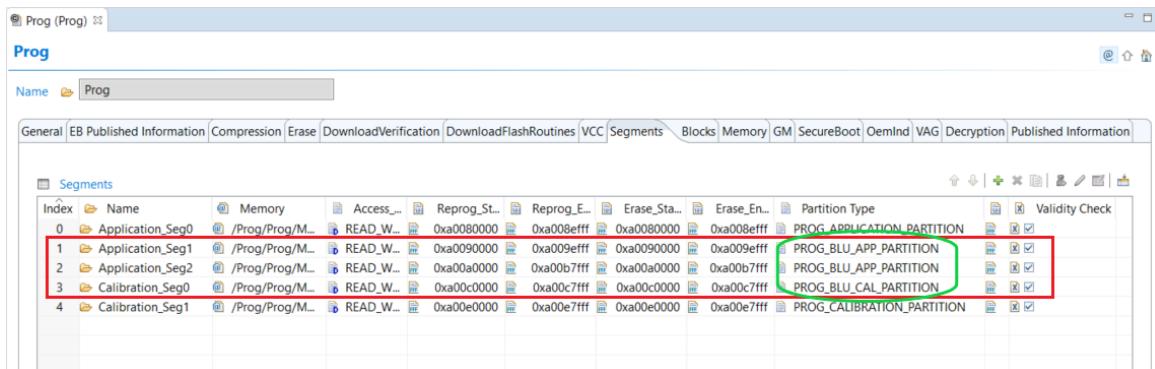


Figure 3.27. Adding Bootloader Updater segments

Id:	OSC-INTMAN-BLU-0003
Version:	1
Description:	If validity check is enabled, and a dedicated BLUpdater segments are configured, then validity check of these segments must be unchecked.



Index	Name	Memory	Access...	Reprog_St...	Reprog_E...	Erase_Sta...	Erase_En...	Partition Type	Validity Check
0	Application_Seg0	/Prog/Prog/M...	READ_W...	0xa0030000	0xa004ffff	0xa0030000	0xa004ffff	PROG_APPLICATION_PARTITION	<input type="checkbox"/> <input checked="" type="checkbox"/>
1	Application_Seg1	/Prog/Prog/M...	READ_W...	0xa0080000	0xa009ffff	0xa0080000	0xa009ffff	PROG_BLU_APP_PARTITION	<input type="checkbox"/> <input checked="" type="checkbox"/>
2	Calibration_Seg0	/Prog/Prog/M...	READ_W...	0xa0200000	0xa024ffff	0xa0200000	0xa024ffff	PROG_BLU_CAL_PARTITION	<input type="checkbox"/> <input checked="" type="checkbox"/>
3	Calibration_Seg1	/Prog/Prog/M...	READ_W...	0xa0260000	0xa027ffff	0xa0260000	0xa027ffff	PROG_CALIBRATION_PARTITION	<input type="checkbox"/> <input checked="" type="checkbox"/>
4	BLUpdater_Seg0	/Prog/Prog/M...	READ_W...	0xa00aa0000	0xa00bbfff	0xa00aa0000	0xa00bbfff	PROG_BLU_PARTITION	<input type="checkbox"/> <input checked="" type="checkbox"/>

Figure 3.28. Adding Dedicated Bootloader Updater segments

3.2.15.5. Integration of the feature

3.2.15.5.1. Initial Boot Manager Software

The following steps shall be done to build the Initial Boot Manager Software :

- ▶ The integrator shall ensure that the Boot Manager (BM) variant is set to Initial Boot Manager (IBM) in [BM Configuration](#).
- ▶ The integrator shall ensure that the [Initial Boot Manager APIs](#) are implemented.

3.2.15.5.2. Bootloader Updater Software

The following steps shall be done to build the Bootloader Updater Software :

- ▶ The integrator shall ensure that the [Bootloader Updater configuration](#) is done as described.
- ▶ The integrator shall ensure that the [BLUpdater APIs](#) are implemented.
- ▶ The integrator shall ensure to define a pattern inside the BLUpdater software to serve as an identification for the bootloader, thus it can check with [PROG_CustomIsBLUPatternPresent](#) the presence of this pattern.
- ▶ The integrator shall ensure that the New Bootloader is incorporated inside the Bootloader Updater software at ranges of addresses defined in [Bootloader Updater configuration](#).

3.2.15.5.3. Bootloader Software

The following steps shall be done to build the Bootloader Software when the Bootloader Updater is used:

- ▶ The integrator shall ensure that the [Bootloader Updater configuration](#) is done as described.



- ▶ The integrator shall ensure that the [Bootloader Updater APIs](#) are implemented.
- ▶ The integrator shall ensure to define a pattern inside the BLUpdater software to serve as an identification for the bootloader, thus it can check with [PROG_CustomIsBLUPatternPresent](#) the presence of this pattern.
- ▶ The integrator shall ensure that the New Bootloader is incorporated inside the Bootloader Updater software at ranges of addresses defined in [Bootloader Updater configuration](#).

3.2.15.6. Bootloader Updater Mandatory API to be called by the system

These API have to be called by the system scheduler at some point to allow the correct behaviour of the bootloader

3.2.15.6.1. Initial Boot Manager Mandatory API

The API [BM_InitialBootStartUp](#) is the entry point of the Initial Boot Manager and shall be the first API to be called by the scheduler

Id:	OSC-INTMAN-BLU-0004
Version:	1
Description:	The integrator shall ensure that BM_InitialBootStartUp is the very first API called once Initial Boot Manager startup is done.

3.2.15.6.2. Bootloader Updater Mandatory API

The API [BLU_Manage](#) shall be cyclically called with the BLU_MANAGE_PERIOD period configured in [Bootloader Updater configuration](#)

Id:	OSC-INTMAN-BLU-0005
Version:	1
Description:	The integrator shall ensure that BLU_Manage is called cyclically with the correct period configured in EB tresos Studio.

The API [BLU_Start](#) is the scheduler of the Bootloader Updater. It shall be called as fast as possible to reduce all Bootloader Updater treatment time (e.g. in background while loop)

Id:	OSC-INTMAN-BLU-0006
Version:	1



Description:	The integrator shall ensure that BLU_Start is called as fast as possible to reduce all Bootloader Updater treatment time.
--------------	---

3.2.16. Bootloader Updater Feature with Secure Boot

When the Secure Boot is activated, the bootloader updater allows to update the secure boot once the bootloader is updated.

The bootloader updater is executed and follow its regular sequence until the verification, if this last one succeed, the bootloader updater proceed to the secure boot update.

To accomplish the secure boot update step, the bootloader updater need a new, or the actual BOOT_MAC_-KEY.

3.2.16.1. Activation of the feature

Following steps describe the activation of the feature.

Id:	OSC-INTMAN-BLU-0007
Version:	1
Description:	If secure bootloader is required for the project the following shall be performed : <ul style="list-style-type: none">▶ Activate the feature in BLUpdater module by checking the box "Enable Secure Boot" in the Security tab.

3.2.16.2. Configuration of the feature

The following steps describe the feature configuration.

Id:	OSC-INTMAN-BLU-0008
Version:	1
Description:	To configure the Bootloader checksum range to be updated : <ul style="list-style-type: none">▶ Set the new Bootloader checksum range with :<ul style="list-style-type: none">▶ Bootloader checksum start address : Start address from where the new bootloader checksum will be calculated▶ Bootloader checksum range length : The length of the new bootloader data on which the checksum will be calculated.



Id:	OSC-INTMAN-BLU-0009
Version:	1
Description:	Ensure to configure the BLUpdater software in the right index as described in Block and Segments Chapter

Id:	OSC-INTMAN-BLU-0010
Version:	1
Description:	<p>The BLUpdater integrity is always checked by the Initial Boot Manager (IBM) at startup, it's not verified in Boot Manager (BM), for this reason, each block or segment dedicated for BLUpdater must be set to not verified in "Verified in Secure Boot" field in Secure Boot Information</p> <ul style="list-style-type: none"> ▶ If a Block is defined for BLUpdater: The BLUpdater block must be set to not verified ▶ If not : Only segments with partition type PROG_BLU_PARTITION if they exist that need to be set to not verified

3.2.16.3. Integration of the feature

3.2.16.3.1. Bootloader Updater Software

The following steps shall be done to enable the update of the secure boot in the Bootloader Updater Software :

- ▶ The integrator shall ensure that the [BLUpdater API](#) is implemented.

3.2.17. Mandatory API to be called by the system

These API have to be called by the system scheduler at some point to allow the correct behaviour of the bootloader

The API [BM_Startup](#) is the entry point of the BootManager and shall be the first API to be called by the scheduler

Id:	OSC-INTMAN-BOOTLOADER-0040
Version:	1
Description:	The integrator shall ensure that BM_Startup is the very first API called once startup is done.



The API [BM_Manage](#) shall be cyclically called with the period configured in EB Tresos Studio, if the BM_-TIMEOUT_CHECK in BM configuration is set to ON

Id:	OSC-INTMAN-BOOTLOADER-0041
Version:	1
Description:	The integrator shall ensure that BM_Manage is called cyclically with the correct period configured in EB tresos Studio, i.e. BM_CYCLE_CALL = 1 if Bootloader CAN and BM_CYCLE_CALL = 5 if Bootloader FlexRay (recommended values).

The APIs [EB_Manage/BIPduR_Manage](#) is the main scheduler of the bootloader software. It calls every other layer Manage fonction. It shall be cyclically called to the period configured in EB tresos Studio.

Depending of your project, EB tresos Bootloader is provided with EB or BIPduR plugin, integration code shall only call the Api of the provided plugin.

Id:	OSC-INTMAN-BOOTLOADER-0042
Version:	1
Description:	The integrator shall ensure that EB_Manage/BIPduR_Manage is called cyclically with the correct period configured in EB tresos Studio, i.e. MANAGE_PERIOD = 1 if Bootloader CAN and MANAGE_PERIOD = 5 if Bootloader FlexRay (recommended values).
Provides coverage to: (Id-Version Variants)	SwAD-ARCH-0110-1

The API [PROG_BckdManage](#) is the scheduler of the bootloader state machine. It shall be called as fast as possible to reduce all bootloader treatment time (e.g. in background while loop)

Id:	OSC-INTMAN-BOOTLOADER-0043
Version:	1
Description:	The integrator shall ensure that PROG_BckdManage is called as fast as possible to reduce all bootloader treatment time.

3.2.18. General Bootloader Performance

Id:	OSC-INTMAN-BOOTLOADER-0081
Version:	1
Description:	If Autosar Flexray stack is used, the integrator shall ensure that Flexray main-functions (i.e. FrIf_MainFunction, FrTp_MainFunction and FrSM_MainFunction) are called every 5ms.



Provides coverage to: (Id-Version Variants)	SwAD-ARCH-0109-1	
Id:	OSC-INTMAN-BOOTLOADER-0082	
Version:	1	
Description:	If Autosar cryptographic libraries are used, the integrator shall ensure that Csm_MainFunction is called continuously in background task.	
Provides coverage to: (Id-Version Variants)	SwAD-ARCH-0109-1	

Note: Integrator shall ensure that FrTp_MainFunction can not be interrupted by FrIf_JobListExec processing that is called under interruption.

3.2.19. Exception handling

Id:	OSC-INTMAN-BOOTLOADER-0090
Version:	1
Description:	The integrator shall ensure the exception handling and defines/implements the action(s) to perform in case an exception happens.

3.2.20. Decryption Feature

Gives the possibility to decrypt encrypted data received by bootloader during transfer data request.

It uses the symmetric AES decryption algorithm in CBC mode with PKCS7 padding provided by CSM library .

Note : If decompression is activated, the decryption is always performed prior to decompression.

3.2.20.1. Activation of the feature

The following steps describe the feature activation.

Id:	OSC-INTMAN-SYMDECRYPT-0001
Version:	1
Description:	If decryption is required for the project then activate the feature in PROG by : ► Checking "Enable_Decryption" box alone in order to use the custom Encryption Algorithm



► If Integrator want to use the AES decryption algorithm, then Check also "Use Symetric algorithm for decryption" box.

3.2.21. Glossary

You can find in the following table a definition of the different naming used with the Bootloader documentation

Term	Definition
Application	Runnable code of ECU software that can be updated by a Bootloader
AUTOSAR	Automotive Open System Architecture A consortium of OEMs, Tier1's and semiconductor vendors that work on standardization of an automotive software architecture.
Bootloader	Permanent software located in Flash memory allowing updating application or calibration software of an ECU. It provides communication with an external Tester Also called: Flashloader.
Calibration	Configuration code of an ECU software that can be updated by a Bootloader
CRY	Cryptographic primitives implementation (used by the CSM) This is an AUTOSAR module
CSM	Cryptographic Service Manager This is an AUTOSAR module
Download verification	Operation realized after a download to verify that the data present in memory match the expected downloaded data. This is usually done using a CRC, Checksum, Signature or Hash. Also called: Check programming dependencies, check memory, message digest
Flash sector	Smallest amount of flash memory that can be erased in one pass. Size depends on flash technology used.
Flash page	Smallest amount of flash memory that can be programmed in one pass. Size depends on flash technology used.
Segment	Continuous address range within a logical block. Static segment: static address and size present in configuration defining the memory section that can be erased or programmed.



Term	Definition
	Dynamic segment: dynamic address and size present in RAM defining the memory section that have been programmed.
HSM	Hardware Security Module
Logical block	Smallest amount of flash memory that can be individually reprogrammed. Size depends on technology (flash sector, flash page) and user settings.
Memory Erased check	<p>Verification if memory is already erased, before trying to erase it. It allow sparing time, if the memory is already erased.</p> <p>Also called: blank check.</p>
PDU	<p>Protocol Data Unit</p> <p>Any piece of information exchanged between two or more communicating entities</p>
PduR	<p>PDU Router</p> <p>This is an AUTOSAR module</p>
SecOC	<p>Secure OnBoard Communication</p> <p>This is an AUTOSAR module</p>
SHE	<p>Secure Hardware Extension</p> <p>A hardware security extension specified by the HIS consortium.</p>
Software integrity	<p>Feature ensuring the software integrity before executing it.</p> <p>Also called: Software authentication.</p>
Software acceptance check	<p>Feature verifying the validity of the received application/calibration software to be updated</p> <p>Also called: Application/Calibration signature check.</p>
Software coherency check	<p>Feature verifying the coherency of the application/calibrations</p> <p>Also called: Application coherency check, consistency check, application validity, Check Programming dependencies.</p>
Streaming	Allow writing data in memory on reception of every consecutive frame of a TransferData request (No wait of the reception of the full request before starting the write). This improve the download performance.

Table 3.2. Bootloader definitions

3.3. Annex: list of requirements

Find below the complete list of userguide requirements. User shall check each requirement for integration.

Id	Description	CheckList
OSC-INTMAN-BOOT-LOADER-0071	The integrator shall ensure that Max_Bytes_in_TD field is filled with a value multiple of Min_Value_To_Flash field value + 2.	
OSC-INTMAN-BOOT-LOADER-0124	The integrator shall ensure that if Reset after S3 Timeout in Programming Session is required or not The Trigger Reset after S3 Timeout in Programming Session checkbox is ticked/enabled	
OSC-INTMAN-BOOT-LOADER-0125	The integrator shall ensure that if Reset after S3 Timeout in Extended Session is required or not The Trigger Reset after S3 Timeout in Extended Session checkbox is ticked/enabled	
OSC-INTMAN-BOOT-LOADER-0123	The integrator shall ensure that if Dual Memory Bank feature shall be used: The Dual Memory Bank Used checkbox is ticked/enabled	
OSC-INTMAN-BOOTCBK-0114	The integrator shall implement in PROG_CustomIsFirstProgramming callback software providing information if Flash was never programmed before and that erase shall be skipped. The callback should return PROG_TRUE if the Flash was never programmed and PROG_FALSE otherwise.	
OSC-INTMAN-BOOTCBK-0159	The integrator shall implement in PROG_CustomDownloadNotification callback software by updating the information that Flash was already programmed before. This indicates that, for any other further	



Id	Description	CheckList
	programming, the Flash memory should be erased.	
OSC-INTMAN-OEMIND-BOOT-LOADER-0122	The integrator shall ensure to set the correct value for the Prog_-RC_CrcOffset eg.: set this value as 0 if the CRC position in the UDS request will be right after the Routine Identifier in the RoutineControl Checkmemory service .	
OSC-INTMAN-OEMIND-BOOT-LOADER-0020	The integrator shall ensure that Transmit_Response_Before_Reset field is set.	
OSC-INTMAN-OEMIND-BOOT-LOADER-0022	The integrator shall ensure that Transmit_Nrc78_On_SecurityAccess field is set if the Security Access seed/key generation process is long (more than P2 time).	
OSC-INTMAN-CRYP-TOASR43-BOOTLOADER-0001	The integrator shall ensure to enable Allow2Cancel_OngoingJobs to cancel the previously started or interrupted or any ongoing jobs, needed only when there are no concurrent jobs configured to run in parallel with this job operation in the Bootloader.(for eg.: during the simultaneous Signature verification and Decryption of the downloaded data, job cancellation of this job shall be disabled). Note:This requirement is applicable only for the Crypto ASR 4.3 integration.	
OSC-INTMAN-CRYP-TOASR43-BOOTLOADER-0002	(Optional)The integrator shall ensure to set the Block Slicing- BS_size parameter value > 0 to enable the feature for the corresponding job. This feature shall perform the slicing of large amount of data (to be processed by crypto stack) into smaller chunks so that the time for	



Id	Description	CheckList
	<p>the asynchronous processing per call at the Crypto driver is reduced.</p> <p>Note: This requirement is applicable only for the Crypto ASR 4.3 integration. It is recommended to call PROG_CsmManage() with high periodicity if this feature is supported. Disclaimer: Setting here a very small value shall significantly increase the overall crypto operation time</p>	
OSC-INTMAN-CRYP-TOASR43-BOOTLOADER-0003	<p>The integrator shall ensure to enable Allow2SetCryptoKey to set the key for any Csm job that needs a key to be passed to the Crypto stack from the Bootlaoder, this is needed especially when the key is not stored in the Crypto Driver or the Hardware Security Module prior to the start of the Crypto operation. Note:This requirement is applicable only for the Crypto ASR 4.-3 integration.</p>	
OSC-INTMAN-CRYP-TOASR43-BOOTLOADER-0004	<p>(Optional)The integrator shall ensure to enable Allow2CustomCsmStartPreprocess the Custom preprocessing feature only when project specific or the algorithm specific operations shall be performed before the start of the current crypto operation. For eg.: implementing the DER encoding of the plain key for performing RSASSA_PSS signature verification. Note:This requirement is applicable only for the Crypto ASR 4.-3 integration.</p>	
OSC-INTMAN-CRYP-TOASR43-BOOTLOADER-0005	<p>If enabled to set the key, the integrator shall ensure to configure the Crypto driver Key element para-</p>	



Id	Description	CheckList
	<p>meter-CryptoKeyElementSize and access type for the corresponding Crypto key. Note: This requirement is applicable only for the Crypto ASR 4.3 integration. Hint: normally the size must include total key size including the all the subelements of the Key. For more info refer to the integration guide of the integrated Crypto driver.</p>	
OSC-INTMAN-CRYP-TOASR43-BOOTLOADER-0006	<p>The integrator shall ensure to enable AllowStreamStart configuration parameter when the Start operation for the Crypto operation cannot be completed with in a single Asynchronous main function call (eg.: for the Signature verification using EDDSA algo.). Note: This requirement is applicable only for the Crypto ASR 4.3 integration. This feature can be enabled only if the Crypto driver supports the StreamStart mode of operation.</p>	
OSC-INTMAN-SIGNATUREVERIFICATION-BOOT-LOADER-0001	<p>The integrator shall ensure to select the appropriate configured Csm Job for the ProgCsmSignatureVerifyConfigId parameter under the SignatureVerification section.</p>	
OSC-INTMAN-SIGNATUREVERIFICATION-BOOT-LOADER-0002	<p>The integrator shall ensure to enable SignFinSendFRP to send response pending before the Crypto operation with the job mode as Finish is executed, this shall be enabled only when the Crypto driver blocks the other Bootlaoder tasks execution for longer duration than the tasks' periodicity while executing the Finish job operation.</p> <p>Note:This requirement is applica-</p>	



Id	Description	CheckList
	ble only for the Crypto ASR 4.3 integration.	
OSC-INTMAN-SIGNATUREVERIFICATION-BOOTLOADER-0003	<p>If enabled to set the key, the integrator shall ensure to configure the Csm asymmetric key size-CsmAsymPublicKeyMaxLength.</p> <p>Note:This requirement is applicable only for the Crypto ASR 4.3 integration.</p>	
OSC-INTMAN-SIGNATUREVERIFICATION-BOOTLOADER-0004	<p>The integrator shall ensure to configure the Csm asymmetric key size(CsmSignatureVerifyMaxKeySize).</p> <p>Note:This requirement is applicable only for the integration of the Crypto stack version below v4.3.</p>	
OSC-INTMAN-HASHVERIFICATION-BOOTLOADER-0001	<p>The integrator shall ensure to select the appropriate configured Csm Job for the ProgCsmHash-ConfigId parameter under the HashVerification section.</p>	
OSC-INTMAN-HASHVERIFICATION-BOOTLOADER-0002	<p>The integrator shall ensure to enable HashFinSendFRP to send response pending before the Crypto operation with the job mode as Finish is executed. This shall be enabled only when the Crypto driver blocks the other Bootlaoder tasks execution for longer duration than the tasks' periodicity while executing the Finish mode job operation.</p> <p>Note:This requirement is applicable only for the Crypto ASR 4.3 integration.</p>	
OSC-INTMAN-HASHVERIFICATION-BOOTLOADER-0003	<p>The integrator shall ensure to configure the Hash digest length(CsmHashResultLength) in the Csm primitives configuration.</p> <p>Note:This requirement is applica-</p>	



Id	Description	CheckList
	ble only for the Crypto ASR 4.3 integration.	
OSC-INTMAN-SECURECHECK-SUMGEN-BOOTLOADER-0001	The integrator shall ensure to select the appropriate configured Csm Job for the ProgCsmSecureConfigId parameter in the PROG module.	
OSC-INTMAN-SECURECHECK-SUMGEN-BOOTLOADER-0002	If enabled to set the key (eg.: for AES CMAC operation), the integrator shall ensure to configure the Csm symmetric key size-CsmSymKeyMaxLength. Note:This requirement is applicable only for the Crypto ASR 4.3 integration.	
OSC-INTMAN-SECURECHECK-SUMGEN-BOOTLOADER-0003	The integrator shall ensure to configure the Csm symmetric key size(CsmSymKeyExtractMaxKeySize). This requirement is applicable only for the integration of crypto stack version below 4.3.	
OSC-INTMAN-DECRYPTION-BOOTLOADER-0003	If Use Symetric algorithm for decryption is set to true, the Bootloader will be able to receive encrypted data and decrypt them before writing to Flash.	
OSC-INTMAN-DECRYPTION-BOOTLOADER-0004	ProgCsmDecryptionConfigId id shall be used to point to the Csm service of the decryption algorithm. The current implementation supports only the CbcPkcs7Decrypt Cry Primitive.	
OSC-INTMAN-DECRYPTION-BOOTLOADER-0005	If enabled to set the key, the integrator shall ensure to configure the Csm symmetric key size-CsmSymKeyMaxLength. Note: The IV and the Decryption key element reference shall be positioned in the CryptoKeyElementRef table of the crypto driver such that	



Id	Description	CheckList
	<p>the IV keyelementref shall be the placed at the first position or row and Cipher keyelementref shall be the placed at the second position or row. When the Signature verification on fly is enabled, ensure to configure Signature verification and the Decryption on separate Csm Queues (also separate driver objects), to allow concurrent jobs execution. This requirement is applicable only for the Crypto ASR 4.-3 integration.</p>	
OSC-INTMAN-BOOT-LOADER-0070	<p>The integrator shall configure all memory used for the project. Only 1 memory of each type (FLASH, FLASH_EXT, RAM) can be configured.</p>	
OSC-INTMAN-BOOT-LOADER-0077	<p>For every memory, the integrator shall configure the memory type: Internal Flash (FLASH), external Flash (FLASH_EXT) or RAM memory (RAM).</p>	
OSC-INTMAN-BOOT-LOADER-0073	<p>For every memory, the integrator shall configure the Memory Mode parameter depending if the Flash driver support synchronous or asynchronous interface call. Synchronous means that when the Flash Api is called, it will returns only when the request operation is performed. Asynchronous means that when the Flash Api is called, it returns before performing the requested operation, Prog module will later call periodically a GetJobStatus Api to be informed when the operation is finished. In case synchronous mode is used Prog module will request the eras-</p>	



Id	Description	CheckList
	ing of Flash sector per sector to avoid a too long block time is Flash call.	
OSC-INTMAN-BOOT-LOADER-0074	For every memory, the integrator shall ensure that Minimum value to write field is set to the minimum size that shall be write for the memory (Flash page size).	
OSC-INTMAN-BOOT-LOADER-0075	For every memory, the integrator shall configure the address offset to be used when accessing the memory. It's used to convert the logical address get from the diagnostic request to the physical address of the memory.	
OSC-INTMAN-BOOT-LOADER-0076	For every memory, the integrator shall configure the erase state value of the memory (0x00 or 0xFF depending of the memory architecture).	
OSC-INTMAN-BOOT-LOADER-0072	The integrator shall ensure that the configuration of the memory areas manipulated by the flashloader is consistent against protected areas (hardware protection key, any other sections that shall not be changed)	
OSC-INTMAN-BOOT-LOADER-0078	The integrator shall ensure to map every segment to the correct memory.	
OSC-INTMAN-OEMIND-BOOT-LOADER-0120	The integrator shall ensure that at least one Application segment is configured as follow: Memory_Type set to FLASH Access_Type set to READ_WRITE Reprog_Start_Address is the start of Application area Reprog_End_Address is the end of Application area Erase_Start_Address is	



Id	Description	CheckList
	<p>equal to Reprog_Start_Address Erase_End_Address is equal to Reprog_End_Address Partition Type is equal to PROG_APPLI- CATION_PARTITION NOTE: Ad- ditional application or calibration segment can be configured.</p>	
OSC-INTMAN-OEMIND-BOOT- LOADER-0121	<p>The integrator shall perform the configuration of all logical blocks that shall be supported by the Bootloader (at least one), to download the Application and Calibration in Blocks, the following information need to be configured</p> <p>BlockId: Identifier of the logical block, it is automatically generated in the order of appearance in Tresos starting from 0. Only one Block for Bootloader update can be configured and it shall be the first of the list (BlockId 0). It shall match the one that will be provided in Erase Routine and RequestDownload request. First segment: reference to the first segment (defined in Segment container) of the logical block If a Block for Bootloader Update has been configured, it shall reference a segment of type Bootloader Updater Number of segments: number of segments in the logical block Block programming counter max: The maximum number of times a logical block can be programmed. If set to 0, no limit will applied. NOTE: The segment list shall be ordered according to their belonging to the logical blocks. (e.g: if a block reference segment 0 as its first segment and is configured with 4 segments, the</p>	



Id	Description	CheckList
	segments 0 to 3 are considered as belonging to this block).	
OSC-INTMAN-OEMIND-BOOT-LOADER-0001	<p>The BLUpdater software index in Block and Segments tables is depending on the Addressing Modes in request download configuration</p> <p>In case of Logical Block : The index of the BLUpdater block in the blocks table must have index 0</p> <p>In other case : The BLUpdater software must be in downloaded in one segment with index 0 in the segments table</p>	
OSC-INTMAN-OEMIND-BOOT-LOADER-0127	In case of Download by logical block is used, The block shall have only one segment	
OSC-INTMAN-OEMIND-BOOT-LOADER-0100	<p>The integrator shall configure Erasing Mode parameter according to the expected erasing request.</p> <p>The following values are possible:</p> <p>All: No information are provided in Erase request, on reception of the erase request, all the memory segments will be erased. Address:</p> <p>The Erase request contains the address range to be erased, only 1 segment can be erased with this request. Note that in this case the checksum will be computed on a single segment.</p> <p>LogicalBlock: The Erase request contains the logical block to be erased.</p>	
OSC-INTMAN-OEMIND-BOOT-LOADER-0101	The integrator shall configure the Erase request ALFI Enable parameter to indicate if the Erase request contains the UDS ALFI field.	
OSC-INTMAN-OEMIND-BOOT-LOADER-0102	The integrator shall configure the Application validity parameter to indicate if Elektrobit algorithm or cus-	



Id	Description	CheckList
	Customer specific algorithm shall be used. If customer specific, the integrator shall implement the PROG_InvalidateSection and PROG_CustomSetCrcCompareSuccess callbacks to manage the application validity status.	
OSC-INTMAN-OEMIND-BOOT-LOADER-0103	The integrator shall configure the CRC algorithm parameter to indicate which checksum shall be used to verify the software download.	
OSC-INTMAN-OEMIND-BOOT-LOADER-0104	The integrator shall configure the FingerPrint Enable parameter to indicate if a fingerprint shall be managed for the software download.	
OSC-INTMAN-OEMIND-BOOT-LOADER-0105	If FingerPrint is enable, the integrator shall configure the Size_Of_FingerPrint parameter to indicate the expected FingerPrint size.	
OSC-INTMAN-BOOT-LOADER-2011	The integrator shall configure the verification buffer size in parameter Verification Buffer size. This buffer is used to temporary locate the data read from memory in order to compute the CRC/Hash/Sig-nature on them. Note that more the buffer is big, more the blocking time for verification computation will be high.	
OSC-INTMAN-OEMIND-VERIF-BOOTLOADER-0001	The integrator shall configure (parameter Maximum RequestDownload Per Block) the maximum number of RequestDownload request that can be received, by the Boot-loader, for a single logical block. This value will be used to size the RAM structure storing the down-loaded memory area that shall be	



Id	Description	CheckList
	used to perform the signature/CRC verification.	
OSC-INTMAN-OEMIND-FLASHROUTINES-BOOT-LOADER-0126	The integrator shall ensure that Compressed Flash Driver feature is activated.	
OSC-INTMAN-OEMIND-BOOT-LOADER-0010	If the feature is enabled integrator shall ensure that BM_TIME-OUT_CHECK field is set.	
OSC-INTMAN-OEMIND-BOOT-LOADER-0011	If the feature is enabled integrator shall ensure that BM_SOURCE_ADDRESS_CHECK field is set.	
OSC-INTMAN-OEMIND-BOOT-LOADER-0012	If the feature is enabled integrator shall ensure that BM_TIME-OUT_VALUE is configured	
OSC-INTMAN-OEMIND-BOOT-LOADER-0013	The integrator shall ensure to select the Boot Manager to generate in Boot_Manager_Variant field Initial Boot Manager : Checks if BootManager is integrated in a Initial Boot Manager software. Boot Manager : Checks if Boot Manager is integrated in a Bootloader software.	
OSC-INTMAN-OEMIND-BOOT-LOADER-0014	If Initial Manager is selected, the integrator shall ensure that Initial_Check_Application field is set if the application validy check need to be done in the Initial Boot Manager.	
OSC-INTMAN-BOOT-LOADER-0200	The integrator shall configure the period for sending NRC78 to manage the sending of NRC78 after a definite interval of time.	
OSC-INTMAN-BOOT-LOADER-0201	For current bootloader, the integrator shall configure the start address, end address of the bootloader.	



Id	Description	CheckList
OSC-INTMAN-BOOT-LOADER-0202	For new bootloader, the integrator shall configure the start address, end address of the bootloader.	
OSC-INTMAN-BOOT-LOADER-0203	For every memory, the integrator shall configure the memory type: Internal Flash (FLASH), external Flash (FLASH_EXT) or RAM memory (RAM).	
OSC-INTMAN-OEMIND-BOOT-LOADER-0070	The integrator shall ensure that the parameter Security_Access_Seed_Length is set to 4	
OSC-INTMAN-OEMIND-BOOT-LOADER-0071	The integrator shall ensure that the parameter Security_Access_Key_Length is set to 4	
OSC-INTMAN-OEMIND-BOOT-LOADER-0072	If the Static Seed (response with precedent seed in the case of successive GetSeed requests without a respective key received) is required for the project, the integrator shall ensure that Enable_Static_Seed field is set.	
OSC-INTMAN-BOOT-LOADER-0104	The integrator shall ensure that Standard field is set to ISO	
OSC-INTMAN-BOOT-LOADER-0105	The integrator shall ensure that UDS_MANAGE_PERIOD field is set to the call period of the UDS_Manage Api	
OSC-INTMAN-BOOT-LOADER-0100	The integrator shall ensure that SecurityCheck field is set and that the SecurityFunction is set to PROG_GetSecurityLevel	
OSC-INTMAN-BOOT-LOADER-0106	The integrator shall ensure that RC_NRC_IMPLEMENTATION field is set to 0x31	
OSC-INTMAN-BOOT-LOADER-0107	The integrator shall ensure that DID_NRC_IMPLEMENTATION field is set to 0x31	



Id	Description	CheckList
OSC-INTMAN-BOOT-LOADER-0101	The integrator shall ensure that RESPONSE_PENDING field is set	
OSC-INTMAN-BOOT-LOADER-0102	The integrator shall ensure that SPREC_IN_RESPONSE field is set	
OSC-INTMAN-BOOT-LOADER-0103	The integrator shall ensure that RELOAD_TSTOPDIAG field is set	
OSC-INTMAN-BOOT-LOADER-0108	If for integration reason the NRC78 response pending shall be send before the end of the configured P2/P2star time, the integrator can configure the P2/P2star adjust parameter. In this case the UDS module will trigg the NRC78 transmission at a time equal to (P2/P2star - P2/P2star adjust)	
OSC-INTMAN-BOOT-LOADER-0126	The integrator shall ensure that the max allowed response length for every DID in the RDBI service request RDBI_MAX_RESPONSE_LENGTH must be less than the configured reception buffer length RxPhysicalBufferSize in BIPduR	
OSC-INTMAN-BOOT-LOADER-0127	The integrator shall ensure that UDS_TIMEOUT_CHECK is set to true if BM_TIMEOUT_CHECK feature is enabled	
OSC-INTMAN-BOOT-LOADER-0120	The integrator shall configure every of following session: DEFAULT, PROGRAMMING, EXTENDED, SUPPLIER, OTHER_01, OTHER_02, OTHER_03, OTHER_04	
OSC-INTMAN-BLPDUR-BOOT-LOADER-0001	The integrator shall configure the TxPDU (TxPdu Reference) to be used for response transmission. It shall reference a valid ECUC PDU.	



Id	Description	CheckList
OSC-INTMAN-BLPDUR-BOOT-LOADER-0002	The integrator shall configure the TxPDU confirmation identifier (Tx-Pdu Identifier) to be used by PduR module to confirm the transmission. The TxPdu Identifiers shall be unique and consecutive.	
OSC-INTMAN-BLPDUR-BOOT-LOADER-0003	The integrator shall configure the Tester Address associated to this connection.	
OSC-INTMAN-BLPDUR-BOOT-LOADER-0004	The integrator shall indicate if this connection is associated to a LIN communication (Lin Connection).	
OSC-INTMAN-BLPDUR-BOOT-LOADER-0005	The integrator shall indicate if this connection shall re-use a functional RxPduld already defined in another connection (Share Functional Id).	
OSC-INTMAN-BLPDUR-BOOT-LOADER-0006	If Share Functional Id is enabled, the integrator shall configured the shared Pdu Reference.	
OSC-INTMAN-BLPDUR-BOOT-LOADER-0007	The integrator shall configure all the RxPDU to be used for request reception. It shall reference a valid ECUC PDU, specify if the Pdu is functional or physical and the associated Pdu Identifier to be used by PduR module. The RxPdu Identifiers shall be unique and consecutive.	
OSC-INTMAN-BLPDUR-BOOT-LOADER-0008	A connection shall contain only one functional RxPdu or one reference functional RxPdu.	
OSC-INTMAN-OEMIND-CRY-BOOTLOADER-0001	The integrator shall configure in CRY module a CryRsaSsaPssVerify configuration with the following configuration: CryRsaSsaPssVerifyUseTimeSlices (Use time slicing for RSASSA-PSS signature verification) set to true	



Id	Description	CheckList
	<p>CryRsaSsaPssVerifyNumberOfTimeSlices (Number of RsaSsaPss time slices) set to 10</p> <p>CryRsaSsaPssVerifyUseCbk (Use configured callback function which returns maximum number of time slices) set to false</p> <p>CryRsaSsaPssVerifyImmediateRestartEnabled (Enable the cancelation of an ongoing calculation regardless of the configuration ID) set to true</p> <p>CryRsaSsaPssVerifyHashConfigRef (Hash configuration) set to CsmHashConfig_0</p> <p>CryRsaSsaPssVerifyKeyLength (Key Length) set to 256</p> <p>CryRsaSsaPssVerifySaltLength (Salt Length) set to 0</p> <p>CryRsaSsaPssVerifyB64Encoded (Base64 Encoded) set to false</p> <p>CryRsaSsaPssVerifyUseBarrett (Barrett reduction) set to false</p> <p>CryRsaSsaPssVerifySupportRestart (Enable the cancelation of ongoing requests) set to true</p>	
OSC-INTMAN-OEMIND-CRY-BOOTLOADER-0002	<p>The integrator shall configure in CRY module the following general parameters:</p> <ul style="list-style-type: none"> CrySHAOneAndTwowImplementation (Implementation variant) set to CRY_SHAONE_ANDTWO_INTERRUPTABLE CryInterruptableLN (Interruptable LN operations) set to true 	
OSC-INTMAN-OEMIND-CRY-BOOTLOADER-0003	<p>The integrator shall configure in CRY module a CrySHA2 configuration with the following configuration:</p> <ul style="list-style-type: none"> CrySHA2ImmediateRestartEnabled (Enable the cancelation of an ongoing calculation regardless of the configuration 	



Id	Description	CheckList
	ID) set to true CrySHA2Type (Prime) set to CRY_SHA_256 CrySHA2IterationsPerMain (Number of iterations per MainFunction) set to 1 CrySHA2SupportRestart (Enable the cancelation of ongoing requests) set to true	
OSC-INTMAN-OEMIND-CRY-BOOTLOADER-0004	The integrator shall configure in CSM module a CsmSignatureVerify configuration with the following configuration: CsmCallbackSignatureVerify set to PROG_CsmNotification CsmSignatureVerifyMaxKeySize set to 524 CsmSignatureVerifyInitConfiguration set to CryRsaSsaPssVerifyConfig_0 (configuration done in OSC-INTMAN-OEMIND-CRY-BOOT-LOADER-0001) CsmSignatureVerifyEnableRteInterface (Enable Rte Interface) set to false CsmSignatureVerifyEnableRestart (Enable the cancelation of ongoing requests) set to true CsmSignatureVerifyUsePriorities (Csm priorities handling) set to true	
OSC-INTMAN-OEMIND-CRY-BOOTLOADER-0005	The integrator shall configure a CsmHash configuration (CsmHashConfig_0) with the following configuration: CsmCallbackHash set to Cry_RsaSsaPssVerifyCallback CsmHashInitConfiguration set to CrySHA2Config_0 (configuration done in OSC-INTMAN-OEMIND-CRY-BOOT-LOADER-0001) CsmHashPrimitiveName set to SHA2 CsmHashEnableRteInterface (Enable Rte Interface) set to false CsmHashEnableRestart	



Id	Description	CheckList
	(Enable the cancelation of on-going requests) set to true CsmHashUsePriorities (Csm priorities handling) set to true	
OSC-INTMAN-BOOT-LOADER-0210	In the Memory Configuration tab, the integrator shall configure all memory used for the project. Only 1 memory of each type (FLASH, FLASH_EXT, RAM, CUSTOM) can be configured.	
OSC-INTMAN-BOOT-LOADER-0211	For every memory, the integrator shall configure the memory type: Internal Flash (FLASH), external Flash (FLASH_EXT), RAM memory (RAM) or Custom memory (CUSTOM).	
OSC-INTMAN-BOOT-LOADER-0212	For every memory, the integrator shall ensure that Minimum value to write field is set to the minimum size that shall be written for the corresponding memory (Flash page size).	
OSC-INTMAN-BOOT-LOADER-0213	For every memory, the integrator shall configure the address offset to be used when accessing the corresponding memory. It's used to convert the logical address get from the diagnostic request to the physical address of the memory.	
OSC-INTMAN-BOOT-LOADER-0214	For every memory, the integrator shall configure the erase state value of the memory (0x00 or 0xFF depending of the memory architecture).	
OSC-INTMAN-BOOT-LOADER-0215	For every memory, the integrator shall configure the corresponding memory start address and entire length.	



Id	Description	CheckList
OSC-INTMAN-BOOT-LOADER-0216	In the Flash Sectors tab, the integrator shall configure all the flash sectors (can be grouped into FlashBanks) with their corresponding start address, length and protection information (Programmable sector or not).	
OSC-INTMAN-CBK-0001	The integrator shall ensure that the callback execution doesn't block the processing of Bootloader Main-Function/Manage cyclic task (e.g BIPduR_Manage)	
OSC-INTMAN-BOOT-LOADER-0001	The integrator shall configure and fill all callbacks out of scope of the main bootloader purpose.	
OSC-INTMAN-BOOT-LOADER-0030	<p>The integrator shall ensure that the following callbacks are implemented:</p> <ul style="list-style-type: none"> UDS_CustomPositiveAnswerInd(): This callback is called to give possibility to the user to execute an action before the positive answer transmission. APP_GetUdsDataBufferInd(): This callback is called when UDS reponse is available but not yet transmitted. The buffer can be updated if necessary. APP_UdsSessionStatusInd(): Notification for diagnostic session transition. APP_TpRxInd(): This callback is called when a message reception is completed, successfully or not. APP_TpTxConf(): This callback is called when a message transmission is completed, successfully or not. UDS_P2AboutToExpireInd(): Notification just before the P2/P2_STAR timeout 	
OSC-INTMAN-BOOT-HW-CBK-0010	The integrator shall ensure that the following callbacks are compiled	



Id	Description	CheckList
	and implemented: BoardSetSleepState() BoardPerformSwReset() BoardIsSwReset() BoardEnableInterrupts() BoardDisableInterrupts() BoardGetRandom()	
OSC-INTMAN-BOOT-HW-CBK-0011	The integrator shall ensure that the API BoardDisableInterrupts is implemented and allow disabling the interruption.	
OSC-INTMAN-BOOT-HW-CBK-0012	The integrator shall ensure that the API BoardEnableInterrupts is implemented and allow enabling the interruption. Note: For Bootloader software that does not use the interruption this API shall stay empty.	
OSC-INTMAN-BOOT-HW-CBK-0016	The integrator shall ensure that the API BoardIsSwReset is implemented and provide the cause of the previous reset.	
OSC-INTMAN-BOOT-HW-CBK-0013	The integrator shall ensure that the API BoardSetSleepState is implemented and allow setting the ECU in sleep state.	
OSC-INTMAN-BOOT-HW-CBK-0014	The integrator shall ensure that the API BoardPerformSwReset is implemented and allows performing a software reset.	
OSC-INTMAN-BOOT-HW-CBK-0015	The integrator shall ensure that the API BoardGetRandom is implemented and return a 32bit pseudo random value.	
OSC-INTMAN-BOOT-LOADER-0020	The integrator shall ensure that the API BM_HardwareInit is implemented and if necessary contains some specific hardware initialization.	



Id	Description	CheckList
OSC-INTMAN-BOOT-LOADER-0021	<p>The integrator shall ensure that the API BM_SoftwareInit is implemented and if necessary contains some specific software initialization.</p>	
OSC-INTMAN-BOOTCBK-0250	<p>This callback must be implemented by the integrator if the feature Secure Boot with HSM is enabled. If EB HSM firmware is used, the integrator shall do the following steps:</p> <p>The integrator shall call the API eb_hsm_mem_block_verify() to start the CMAC value verification asynchronously on HSM channel 0 if the job is not yet started. The integrator shall call the API eb_hsm_poll_channel() to get the status of HSM channel 0 every time this callback is entered. The integrator shall call the API eb_hsm_get_result_channel() to get the job result of HSM channel 0 when the job is finished. The integrator shall return BM_HSM_JOB_OK only if the CMAC verification job is finished without any error. The integrator shall return BM_HSM_JOB_FAILED if the CMAC verification job cannot be started or is finished with some error. The integrator shall return BM_HSM_JOB_PENDING if the CMAC verification job is started but not yet finished.</p> <p>If a third-party HSM firmware is used, the integrator shall call the right API to do the CMAC check of the given memory block asynchronously and report the result.</p>	
OSC-INTMAN-BOOTCBK-0251	<p>The integrator shall implement in BM_CustomGetMacKey callback to provide the MAC key and</p>	



Id	Description	CheckList
	keylength. If the SHE module is present, the SHE key format shall be used.	
OSC-INTMAN-BOOT-LOADER-0022	<p>The integrator shall implement in BM_CheckProgRequest callback software allowing getting information from application if a programming request has been received (e.g: read a flag from noinit RAM shared between Bootloader and Application). Return TRUE if a programming request has been received, return FALSE if no programming request has been received.</p>	
OSC-INTMAN-BOOT-LOADER-0023	<p>The integrator shall implement in BM_CustomCheckValidAppl callback software performing a check of the full software (application and calibration) to identify if application software is in a state where it can be started (valid and coherent). This that e.g (integration dependent) been done by checking that validity flag of every block/segment that are part the application software and check that different block/segment are all coherent with e.g version check. Return TRUE if the application is valid, return FALSE if the application is not valid.</p>	
OSC-INTMAN-BOOT-LOADER-0024	<p>The integrator shall in BM_CustomCheckValidBLU callback software allowing getting information to identify if the Bootloader Updater software is in a state where it can be started (e.g: read a flag from non-volatile memory shared between Pre-Boot Manager and</p>	



Id	Description	CheckList
	Bootloader Updater). Return TRUE if the Bootloader Updater is valid, return FALSE if the Bootloader Updater is not valid.	
OSC-INTMAN-BOOT-LOADER-0025	The integrator shall implement in BM_CustomCheckValidBL callback software allowing getting information to identify if the Bootloader/New Bootloader software is in a state where it can be started (e.g: read a flag from non-volatile memory shared between Pre-Boot Manager and Bootloader Updater). Return TRUE if the Bootloader is valid, return FALSE if the Bootloader is not valid.	
OSC-INTMAN-BOOT-LOADER-0026	The integrator shall implement in PROG_JumpToApplication callback software allowing jumping to application start address.	
OSC-INTMAN-BOOT-LOADER-0027	The integrator shall implement in BM_JumpToBLU callback software allowing jumping to bootloader updater start address.	
OSC-INTMAN-BOOT-LOADER-0028	The integrator shall implement in BM_JumpToBL callback software allowing jumping to bootloader start address.	
OSC-INTMAN-BOOTCBK-0238	The integrator shall implement in BM_CustomDualBankInit callback the configuration of the hardware for the use of dual memory banks. Also, this callback shall initialize all the needed data for the dual memory banks usage.	
OSC-INTMAN-BOOTCBK-0179	The integrator shall implement in BM_CustomIsNormalStartup callback software identifying if a normal or abnormal startup happened.	



Id	Description	CheckList
	Criteria are project/hardware specific (e.g power on reset can be considered as abnormal startup)	
OSC-INTMAN-BOOTSACBK-0001	In SA_CustomCalculateKey callback, the integrator shall implement the key computation based on the provided random value. Computation shall be done using the algorithm required for the project.	
OSC-INTMAN-BOOTSACBK-0002	In SA_CustomStoreAsRetryCnt callback, the integrator shall implement the storage in non-volatile memory of the provided counter value.	
OSC-INTMAN-BOOTSACBK-0003	In SA_CustomRestoreAsRetryCnt callback, the integrator shall implement the get from non-volatile memory of the counter value. It shall be ensure that if value has never been written, the return value is 0.	
OSC-INTMAN-BOOTCBK-0100	The integrator shall implement in PROG_CheckProgRequest callback software allowing getting information from application if a programming request has been received (e.g: read a flag from noinit RAM shared between Bootloader and Application). PROG_BOOT_REPROG value shall be returned if a programming request has been received. PROG_BOOT_NO_REPROG shall be returned if no programming request has been received	
OSC-INTMAN-BOOTCBK-0101	The integrator shall implement in PROG_JumpToApplication call-	



Id	Description	CheckList
	back software allowing jumping to application start address.	
OSC-INTMAN-BOOTCBK-0104	<p>The integrator shall implement in PROG_isValidApplication callback software performing a check of the full software (application and calibration) to identify if application software is in a state where it can be started (valid and coherent). This that e.g (integration dependent) been done by checking that validity flag of every block/segment that are part the application software and check that different block/segment are all coherent with e.g version check. Return TRUE if application is valid, return FALSE if application is not valid.</p>	
OSC-INTMAN-BOOTCBK-0107	<p>The integrator shall implement in PROG_InvalidateSection_BlockID callback software performing operation that can be required by integration software before erasing and invalidating the logical block that will be erased. PROG_E_OK shall be returned if erasing is allowed, PROG_E_NOT_OK in other case.</p>	
OSC-INTMAN-BOOTCBK-0108	<p>The integrator shall implement in PROG_SwitchApplicationModelInd callback software performing operation that can be required before jumping to application.</p>	
OSC-INTMAN-BOOTCBK-0113	<p>The integrator shall implement in PROG_GetSuppressBitFromAppli callback software getting from application information if the suppressPositiveResponse bit was set in the received request (e.g: read</p>	



Id	Description	CheckList
	a flag from noinit RAM shared between Bootloader and Application)	
OSC-INTMAN-BOOTCBK-0116	<p>The integrator shall implement in PROG_CustomWriteFingerprint callback software checking the validity of FingerPrint data and performing the writting in non-volatile memory of the Fingerprint data (pubRamBuffer points on the dataIdentifier field of the WriteDataByIdentifier, allowing integrator identifying the fingerprint using the DID identifier value) request Asynchronous management can be implemented, in this PROG_E_BUSY value is returned and further call to PROG_CustomGetWriteFingerprintStatus will allow Bootloader to get writting status.</p>	
OSC-INTMAN-BOOTCBK-0117	<p>The integrator shall implement in PROG_CustomGetWriteFingerprintStatus callback software providing status of the fingerprint writing</p>	
OSC-INTMAN-BOOTCBK-0142	<p>The integrator shall implement in PROG_CustomGetAsymPublicKey callback the fetching of the asymetrical public key.</p>	
OSC-INTMAN-BOOTCBK-0242	<p>If required, the integrator shall check if the production key is written or not. If it is written, the production key shall be returned, otherwise return the development key.</p>	
OSC-INTMAN-BOOTCBK-0143	<p>The integrator shall implement in PROG_CustomGetSymStaticKey callback the fetching of the symetrical static key.</p>	
OSC-INTMAN-BOOTCBK-0161	<p>The integrator shall implement PROG_CustomDecryptData call-</p>	



Id	Description	CheckList
	back copying decrypted data at the same location than the encrypted one.	
OSC-INTMAN-BOOTCBK-0162	The integrator shall implement in PROG_CustomMemoryAccess-Notification callback the procedure desired like an subsystem update.	
OSC-INTMAN-BOOTCBK-0170	<p>The integrator shall implement in PROG_CustomMemoryErase call-back the desired erase routine. The callback should return one of the following macros PROG_E_OK, PROG_E_NOT_OK or PROG_E_BUSY, according to its memory access status. It is recommended to implement it asynchronously using the callback PROG_CustomMemGetJobStatus if it is a slow operation.</p>	
OSC-INTMAN-BOOTCBK-0171	<p>The integrator shall implement in PROG_CustomMemoryWrite call-back the desired write routine. The callback should return one of the following macros PROG_E_OK, PROG_E_NOT_OK or PROG_E_BUSY, according to its memory access status. It is recommended to implement it asynchronously using the callback PROG_CustomMemGetJobStatus, if it is a slow operation.</p>	
OSC-INTMAN-BOOTCBK-0172	<p>The integrator shall implement in PROG_CustomMemoryRead call-back the desired read routine. The callback should return one of the following macros PROG_E_OK, PROG_E_NOT_OK or PROG_E_BUSY, according to its memory access status. It is recommended to implement it asynchronously us-</p>	



Id	Description	CheckList
	ing the callback PROG_CustomMemGetJobStatus, if it is a slow operation.	
OSC-INTMAN-BOOTCBK-0173	The integrator shall implement in PROG_CustomMemGetJobStatus callback the routine to get the memory job status. After a custom memory access this callback shall be called periodically until getting a status different from PROG_E_BUSY.	
OSC-INTMAN-BOOTCBK-0174	The integrator shall implement in PROG_CustomGetNextSectorAddr callback the routine to get the next sector memory address.	
OSC-INTMAN-BOOTCBK-0239	The integrator shall implement in PROG_CustomCalcInactiveBankWriteAddr callback the calculation of the erase or write address in the inactive bank based on the offset between banks and the current active memory bank.	
OSC-INTMAN-BOOTCBK-0240	The integrator shall implement in PROG_CustomCalcInactiveBankReadAddr callback the calculation of the read address in the inactive bank based on the offset between banks and the current active memory bank.	
OSC-INTMAN-BOOTCBK-0273	The integrator shall implement in PROG_CustomCalcInactiveBankReadAddr callback UDS processing of Routine request 0xDF00 and functions for Bank swap if download and verification of application is successful in inactive memory bank	
OSC-INTMAN-BOOTCBK-0274	In PROG_CustomBankConfirmRollback , the integrator shall impl-	



Id	Description	CheckList
	ment UDS processing of the Routine request and Confirmation of Bank swap after download and verification of application is successful in inactive memory bank or Rollback of Swap. No further swap possible until a new Valid application is download. And this shall be checked at each swap request	
OSC-INTMAN-BOOTCBK-0136	The integrator shall retrieve from the non-volatile memory the value of the programming counter. PROG_CustomGetProgCounter has the following parameter: The block Id for which the information is fetched	
OSC-INTMAN-BOOTCBK-0137	The integrator shall increment the value of the programming counter in the non-volatile memory. PROG_CustomIncrementProgCounter has the following parameter: The block Id for which the information is stored	
OSC-INTMAN-BOOTCBK-0231	The integrator shall implement in PROG_CustomSetBLUDownloadInProgress callback software updating the flag that indicates the Bootloader Updater Program download is in progress and the bootloader is being updated	
OSC-INTMAN-BOOTCBK-0225	The integrator shall implement in PROG_CustomIsBLUDownloadInProgress callback software to get the flag that indicates if the Bootloader Updater Program download is in progress and the bootloader is being updated	
OSC-INTMAN-BOOTCBK-0228	The integrator shall implement in PROG_CustomIsBLUPattern-	



Id	Description	CheckList
	<p>Present callback software a read on the flashed data at the area of the Bootloader Updater pattern and check if it matches the known Bootloader Updater pattern The address and size of the pattern can be decided in integration, the address is depending on the Addressing Modes in request download configuration In case Logical Block : The pattern need to be inside the logical block, the address of pattern can be fixed or relative to the logical block memory area In other case : The pattern need to be inside the the last segment of the Bootloader Updater, the address of pattern can be fixed or relative to the last segment memory area the Bootloader Updater This function take as parameters : ubLogicalBlockId : the current logical block Id, used for Logical Block case ubLogicalSegmentId : the current segment Id, used for other case This function shall return : PROG_TRUE if the Bootloader Updater pattern is present PROG_FALSE if the Bootloader Updater pattern is absent Specific solution : In case Logical Block : If the BLUpdater block id is fixed, check whether the current block corresponds to the BLUpdater block or not can be done only with comparing the logical block id, no need to define the pattern. In other case : If the BLUpdater last segment is fixed, check whether the current segment corresponds to the last one of BLUpdater or not can be done</p>	



Id	Description	CheckList
	only with comparing the segment id, no need to define the pattern.	
OSC-INTMAN-BOOTCBK-0229	The integrator shall implement in PROG_CustomSetDownloadVerificationSuccess callback software updating the Bootloader Updater validity status. This function shall return : PROG_E_OK if no error occurs. PROG_E_NOT_OK in any error occurs.	
OSC-INTMAN-BOOTCBK-0261	The integrator shall implement in PROG_CustomDecryptGetInitVector callback the fetching of the IV used for the decryption.	
OSC-INTMAN-BOOTCBK-0262	The integrator shall implement in PROG_CustomGetSymDecryptionKey callback the fetching of the symetrical decryption key.	
OSC-INTMAN-BOOTCBK-0275	This is an optional callback implementation, the integrator shall implement in PROG_CustomCsmStrtPreproc callback the project specific or the algorithm specific operations. For eg.: implementing the DER encoding of the plain key needed for the RSASSA_PSS signature verification algo.	
OSC-INTMAN-BOOTCBK-0145	The integrator shall implement in callback BIPduR_GetRxPduld software getting the Rx Pdu Id to be used to respond to the request received before the reset.	
OSC-INTMAN-BOOTCBK-0146	The integrator shall implement in callback BIPduR_StoreRxPduld the storage of the Rx Pdu Id to be used by ECU in order to send a response after a reset.	
OSC-INTMAN-BOOTCBK-0168	The Integrator will add the initialization of all the modules in the	



Id	Description	CheckList
	communication stack and it is up to the integrator to define which Communication protocol stack to be initialized based on OEM and their requirement (e.g. CAN or FR or Eth or Lin). The integrator shall ensure that the PDUID of the BIPdur connection is initialized at the start-up if it is not a software reset.	
OSC-INTMAN-BOOTCBK-0169	The Integrator will add disabling the communication or switch the state machine to NO communication mode code and it is up to the integrator to define which Communication protocol stack to be disabled based on OEM and their requirement (e.g. CAN or FR or Eth or Lin)	
OSC-INTMAN-BOOTCBK-0243	The integrator shall implement in BLU_CustomSetValidityNewBootloader callback the marker to set up the validity flag when the new bootloader image is successfully updated.	
OSC-INTMAN-BOOTCBK-0244	The integrator shall implement in BLU_CustomInvalidateBLUpdatorStatus callback the marker to set up the invalidity flag to invalidate the bootloader updater.	
OSC-INTMAN-BOOTCBK-0245	The integrator shall implement in BLU_CustomGetPduID callback to obtain the PduID from the bootloader.	
OSC-INTMAN-BOOTCBK-0246	The integrator shall implement in BLU_CustomTriggerWatchdog callback to trigger the watchdog timer.	
OSC-INTMAN-BOOTCBK-0247	The integrator shall implement in BLU_CustomTriggerWatchdog	



Id	Description	CheckList
	callback to initialize all modules of communication stack.	
OSC-INTMAN-BOOTCBK-0278	<p>The integrator shall implement in BLU_CustomBootChecksumUpdate callback the sequence to update the secure boot of the new boot : Retrieve the current BOOT_MAC_KEY for authorization purpose. The integrator shall call the API eb_hsm_load_key() to load the BOOT_MAC_KEY, a new one can be provided, or the current can be reused. Generate the checksum of the new boot with following steps:</p> <p>The integrator shall call the API eb_hsm_secure_boot() to start the job of updating the CMAC value of a given bootloader range asynchronously by using HSM channel 0 if the job is not yet started.</p> <p>The integrator shall call the API eb_hsm_poll_channel() to get the status of HSM channel 0 every time this callback is entered. The integrator shall call the API eb_hsm_get_result_channel() to get the job result of HSM channel 0 when the job is finished. The integrator shall return BLU_E_OK only if the CMAC value of the given bootloader range is updated without any error. The integrator shall return BLU_E_NOT_OK if the job cannot be started or is finished with some error. The integrator shall return BLU_E_BUSY if the job is started but not yet finished.</p>	
OSC-INTMAN-BOOTCBK-0138	If the feature multiple identifier is configured to External Notification the callback BIPduR_Get-	



Id	Description	CheckList
	GroupIdVal shall be filled to return the group ID to be used	
OSC-INTMAN-BOOTCBK-0285	The integrator shall implement in ReProgMemM_FlashInit callback software calling the initialization APIs of the flash drivers (internal and/or external).	
OSC-INTMAN-BOOTCBK-0286	The integrator shall implement in ReProgMemM_DualBank_Init callback software initializing the DualBank functionality based on the target.	
OSC-INTMAN-BOOTCBK-0288	The integrator shall implement in ReProgMemM_CustomGetActiveBank callback software that will get the information of the current active bank based on the target.	
OSC-INTMAN-BOOTCBK-0287	The integrator shall implement in ReProgMemM_CustomGetInactiveBankAddrOffset callback software that will get the information of the inactive bank address offset based on the target.	
OSC-INTMAN-BOOTCBK-0289	The integrator shall implement in ReProgMemM_FIsDriver_JobStatus callback software that will get the information on the current status of the flash driver. In case of asc_MemAcc module integration, the integrator shall get the driver current status from the MemAcc_-GetJobResult API.	
OSC-INTMAN-BOOTCBK-0290	The integrator shall implement in ReProgMemM_FIsExtDriver_JobStatus callback software that will get the information on the current status of the external flash driver. In case of asc_MemAcc module integration, the integrator shall get	



Id	Description	CheckList
	the driver current status from the MemAcc_GetJobResult API.	
OSC-INTMAN-BOOTCBK-0291	The integrator shall implement in ReProgMemM_FIsDriver_Erase callback software that will call the flash driver erase routine. If the DualBank feature is not used and the erase operation happens on the same bank, the flash driver erase routine shall be copied and executed in RAM memory. In case of asc_MemAcc module integration, the integrator shall call the MemAcc_Erase API.	
OSC-INTMAN-BOOTCBK-0292	The integrator shall implement in ReProgMemM_FIsExtDriver_Erase callback software that will call the external flash driver erase routine. In case of asc_MemAcc module integration, the integrator shall call the MemAcc_Erase API.	
OSC-INTMAN-BOOTCBK-0293	The integrator shall implement in ReProgMemM_FIsDriver_Write callback software that will call the flash driver write routine. If the DualBank feature is not used and the write operation happens on the same bank, the flash driver write routine shall be copied and executed in RAM memory. In case of asc_MemAcc module integration, the integrator shall call the MemAcc_Write API.	
OSC-INTMAN-BOOTCBK-0294	The integrator shall implement in ReProgMemM_FIsExtDriver_Write callback software that will call the external flash driver write routine. In case of asc_MemAcc module integration, the integrator shall call the MemAcc_Write API.	



Id	Description	CheckList
OSC-INTMAN-BOOTCBK-0295	The integrator shall implement in ReProgMemM_FlsDriver_Read callback software that will call the flash driver read routine. In case of asc_MemAcc module integration, the integrator shall call the MemAcc_Read API.	
OSC-INTMAN-BOOTCBK-0296	The integrator shall implement in ReProgMemM_FlsExtDriver_Read callback software that will call the external flash driver read routine. In case of asc_MemAcc module integration, the integrator shall call the MemAcc_Write API.	
OSC-INTMAN-BOOTCBK-0297	The integrator shall implement in ReProgMemM_CustomGetOffset callback software that will return the offset to be added in the request operation address.	
OSC-INTMAN-BOOTCBK-0298	The integrator shall implement in ReProgMemM_CustomGetPhysicalToLogicalAddress callback software that will convert, depending on the target, the physical address contained in the memory operation request into a logical address to be processed by the flash drivers.	
OSC-INTMAN-FLSDOWN-0170	If downloading of the flash routines is required for the project the following shall be performed: Activate the feature in PROG module by setting Download FFlash driver field . Note: Please note that smaller the decompression slice is, bigger is the time needed for the overall decompression process.	
OSC-INTMAN-FLSDOWN-0171	If the rejection of any new attempt of the flash routines after a failed attempt is required for the project	



Id	Description	CheckList
	the following shall be performed: Activate the feature in PROG module by setting Download FFlash driver field . Note: Please note that smaller the decompression slice is, bigger is the time needed for the overall decompression process.	
OSC-INTMAN-FLSDOWN-0172	If not already existing,add a new Ram memory in the Memory tab.	
OSC-INTMAN-FLSDOWN-0173	Add a new (and unique) segment for the flash driver routines, and put the addresses where these routines are mapped.	
OSC-INTMAN-FLSDOWN-0174	Set the partition type of the segment to PROG_FLASH_ROUTINES_PARTITION.	
OSC-INTMAN-FLSDOWN-0175	Map the segment to the RAM memory create previously.	
OSC-INTMAN-FLSDOWN-0176	Deactivate the validity check for this segment, or the generation will fail.	
OSC-INTMAN-FLSDOWN-0177	Add a new block for the flash driver routines and set the First segment field to the created segment in the previous step.	
OSC-INTMAN-FLSDOWN-0178	Add a linker file section in RAM to map all the code from MemMap section FLASH_FLS_START_SEC_CODE.	
OSC-INTMAN-FLSCOMP-0000	If decompression of the flash routines is required for the project the following shall be performed: Activate the feature in PROG module by setting Decompress Flash driver field.	
OSC-INTMAN-FLSCOMP-0001	Add a linker file section in RAM to map all the code from MemMap section FLASH_FLS_START_-	



Id	Description	CheckList
	SEC_CODE. The flash driver area in ROM section shall be located in a contiguous space area. The flash driver area in RAM section shall be located in a contiguous space area.	
OSC-INTMAN-FLSCOMP-0002	In the linker file, add the 4 following symbols: __FLASH_DRIVER_ROM_START_ADDRESS: must be mapped to the beginning of the flash driver area in ROM section __FLASH_DRIVER_ROM_END_ADDRESS: must be mapped to the end of the flash driver area in ROM section __FLASH_DRIVER_RAM_START_ADDRESS: must be mapped to the beginning of the flash driver area in RAM section __FLASH_DRIVER_RAM_END_ADDRESS: must be mapped to the end of the flash driver area in RAM section	
OSC-INTMAN-FLSCOMP-0003	Size of the compressed flash routines in ROM shall be provided in the last 4 bytes of the flash driver area in ROM section.	
OSC-INTMAN-DUALBANK-0001	If dual memory bank feature is required, the integrator shall program the EB tresos Bootloader in both active and inactive memory banks.	
OSC-INTMAN-DUALBANK-0002	If dual memory bank feature is required, the integrator shall implement and manage the swap request (UDS service configuration and handler).	
OSC-INTMAN-DUALBANK-0003	If dual memory bank feature is required, the integrator shall assure that the memory banks configurations are identical in order to have	



Id	Description	CheckList
	similar performances of the software on both banks.	
OSC-INTMAN-DUALBANK-0004	If dual memory bank feature is required, the integrator shall implement and manage the Commit or Rollback Software (UDS service configuration and handler).	
OSC-INTMAN-BOOTCBK-0265	If secure bootloader is required for the project the following shall be performed: Activate the feature in BM module by setting Authenticated / Secure Boot field to Secure.	
OSC-INTMAN-BOOTCBK-0266	If secure bootloader is required for the project the following shall be performed: Flash the bootloader without symbols. Flash the binary file containing the checksum [1] Then load the bootloader symbols [1] The binary file containing the checksum of the bootloader is calculated as follows : Calculate the hash/Mac of the blocksize of the bootloader that needs to be authenticated Write that value inside a binary file to be flashed at a specific address outside the bootloader area Note: The hash should be calculated on a contiguous range of the bootloader without a gap. In the linker File, set a rom area at that specific address for the variable m_aubBootloaderChecksum that will have the checksum value. In the Memory mapping file, define the section of the variable.	
OSC-INTMAN-BOOTCBK-0267	To select the correct algorithm to compute the Secure Boot checksum, following steps shall be performed: Set BMCsmChecksumConfigId in BM module to the cor-	



Id	Description	CheckList
	rect configuration Set ProgCsmSecureConfigId in PROG module to the correct configuration In the case Mac algorithm is selected the configuration in PROG and BM plugin shall refer to a Mac configuration, otherwise an error will be mentioned	
OSC-INTMAN-BOOTCBK-0268	If Checksum computation for Secure Boot shall be done when the Secure Boot feature is deactivated, following steps shall be performed: Authenticated / Secure Boot is set to OFF in BM module. Checksum computation is activated in PROG module.	
OSC-INTMAN-BOOTCBK-0269	If Bootloader checksum verification is not required at startup, the following steps shall be performed: Bootloader verified/ Bootloader not verified is set to OFF in BM module.	
OSC-INTMAN-BOOTCBK-0270	If secure bootloader is required for the project the following information shall be set: Verified in Secure Boot: The checksum will be verified at startup. Blocker for Software execution: In case the checksum controlled at startup is different from the expected one the Application won't be executed. Start Address for the Secure Boot Verification: Start address of the area on which the checksum is computed. Length of the Block area for the Secure Boot Verification: Length of the area on which the checksum is computed. Please note: The Block/Segment that shall be verified by HSM, should not have	



Id	Description	CheckList
	gaps/empty areas/non-filled areas within the memory area on which verification shall be performed by HSM.	
OSC-INTMAN-BOOTCBK-0257	To have a functional EB HSM Proxy, the integrator shall implementing the following functions in eb_hsm_integ.c in the bootloader project: eb_hsm_integ_GetCounterValue: get the tick value of a free-running timer. eb_hsm_integ_GetElapsedValue: get the number of ticks between the current tick value and a previous tick value. eb_hsm_integ_TICKS2MS: convert ticks to milliseconds. If Crypto ASR 4.3 higher version is used, add HSM Crypto Driver initialization before the call to BM_Startup API	
OSC-INTMAN-BOOTCBK-0255	Do the following steps in order to enable the feature Secure Boot with HSM feature: Activate the feature in BM module by setting Authenticated / Secure Boot field to Secure. Check the box Use HSM.	
OSC-INTMAN-BOOTCBK-0258	During the startup process of the bootloader software, the integrator can call the function eb_hsm_get_she_status() to know if the Secure Boot feature offered by HSM firmware is activated or not by checking the bit value at the position COMM_SHE_STATUS_OF_FSET_SECURE_BOOT: If that bit is set, then the feature is activated. If that bit is cleared, the feature is not yet activated. The integrator must call the function eb_hsm_secure_boot() with the start address	



Id	Description	CheckList
	(of the bootloader software) to be verified and its length.	
OSC-INTMAN-BOOTCBK-0256	<p>If the Secure Boot with HSM feature is enabled, the integrator needs to fill the following fields under PROG/Blocks/ */SecureBoot, if the project does not support Blocks configuration, then the integrator needs to fill the following fields under PROG/Segments/ */Secure-Boot. Verified in Secure Boot: The checksum of this memory block/segment will be verified at startup of the bootloader software. Blocker for Software execution: In case the checksum controlled at startup is different from the expected one the Application won't be executed. Start Address for the Secure Boot Verification: Start address of the area on which the checksum is computed. Length of the Block area for the Secure Boot Verification: Length of the area on which the checksum is computed. NOTE: The Block/Segment that shall be verified by HSM, should not have gaps/empty areas/non-filled areas within the memory area on which verification shall be performed by HSM.</p>	
OSC-INTMAN-BLU-0001	<p>If downloading of the Bootloader Updater is required for the project, the feature can be activated automatically when one of these partitions type is configured in segments list : PROG_BLU_APP_PARTITION : The corresponding Application segment memory area can be used to host BLUpdater.</p>	



Id	Description	CheckList
	<p>PROG_BLU_CAL_PARTITION : The corresponding Calibration segment memory area can be used to host BLUpdater. PROG_BLU_PARTITION : The corresponding segment memory area is dedicated to host only the BLUpdater.</p>	
OSC-INTMAN-BLU-0002	<p>To Configure the list of segments which will host the BLUpdater : You can reuse the already configured segments for Application or Calibration by setting the partition type to one which can hosts the BLUpdater. PROG_BLU_APP_PARTITION : segment shared between Application and BLUpdater. PROG_BLU_CAL_PARTITION : segment shared between Calibration and BLUpdater. If BLUpdater needs to be in separate memory area, add PROG_BLU_PARTITION segments type, which will be dedicated for BLUpdater.</p>	
OSC-INTMAN-BLU-0003	<p>If validity check is enabled, and a dedicated BLUpdater segments are configured, then validity check of these segments must be unchecked.</p>	
OSC-INTMAN-BLU-0004	<p>The integrator shall ensure that BM_InitialBootStartUp is the very first API called once Initial Boot Manager startup is done.</p>	
OSC-INTMAN-BLU-0005	<p>The integrator shall ensure that BLU_Manage is called cyclically with the correct period configured in EB tresos Studio.</p>	
OSC-INTMAN-BLU-0006	<p>The integrator shall ensure that BLU_Start is called as fast as pos-</p>	



Id	Description	CheckList
	sible to reduce all Bootloader Updater treatment time.	
OSC-INTMAN-BLU-0007	If secure bootloader is required for the project the following shall be performed : Activate the feature in BLUpdater module by checking the box Enable Secure Boot in the Security tab.	
OSC-INTMAN-BLU-0008	To configure the Bootloader checksum range to be updated : Set the new Bootloader checksum range with : Bootloader checksum start address : Start address from where the new bootloader checksum will be calculated Bootloader checksum range length : The length of the new bootloader data on which the checksum will be calculated.	
OSC-INTMAN-BLU-0009	Ensure to configure the BLUpdater software in the right index as described in Block and Segments Chapter	
OSC-INTMAN-BLU-0010	The BLUpdater integrity is always checked by the Initial Boot Manager (IBM) at startup, it's not verified in Boot Manager (BM), for this reason, each block or segment dedicated for BLUpdater must be set to not verified in Verified in Secure Boot field in Secure Boot Information If a Block is defined for BLUpdater: The BLUpdater block must be set to not verified If not : Only segments with partition type PROG_BLU_PARTITION if they exist that need to be set to not verified	



Id	Description	CheckList
OSC-INTMAN-BOOT-LOADER-0040	The integrator shall ensure that BM_Startup is the very first API called once startup is done.	
OSC-INTMAN-BOOT-LOADER-0041	The integrator shall ensure that BM_Manage is called cyclically with the correct period configured in EB tresos Studio, i.e. BM_CYCLES_CALL = 1 if Bootloader CAN and BM_CYCLE_CALL = 5 if Bootloader FlexRay (recommended values).	
OSC-INTMAN-BOOT-LOADER-0042	The integrator shall ensure that EB_Manage/BIPduR_Manage is called cyclically with the correct period configured in EB tresos Studio, i.e. MANAGE_PERIOD = 1 if Bootloader CAN and MANAGE_PERIOD = 5 if Bootloader FlexRay (recommended values).	
OSC-INTMAN-BOOT-LOADER-0043	The integrator shall ensure that PROG_BckdManage is called as fast as possible to reduce all bootloader treatment time.	
OSC-INTMAN-BOOT-LOADER-0081	If Autosar Flexray stack is used, the integrator shall ensure that Flexray mainfunctions (i.e. FrIf_MainFunction, FrTp_MainFunction and FrSM_MainFunction) are called every 5ms.	
OSC-INTMAN-BOOT-LOADER-0082	If Autosar cryptographic libraries are used, the integrator shall ensure that Csm_MainFunction is called continuously in background task.	
OSC-INTMAN-BOOT-LOADER-0090	The integrator shall ensure the exception handling and defines/implements the action(s) to perform in case an exception happens.	



Id	Description	CheckList
OSC-INTMAN-SYMDE-CRYPT-0001	<p>If decryption is required for the project then activate the feature in PROG by : Checking Enable_Decryption box alone in order to use the custom Encryption Algorithm If Integrator want to use the AES decryption algorithm, then Check also Use Symetric algorithm for decryption box.</p>	

Table 3.3. List of requirements



4. BL for Essentials module references

4.1. Overview

This chapter provides module references for the BL for Essentials product modules. These include a detailed description of all configuration parameters. Furthermore this chapter lists the application programming interface with all data types, constants and functions.

The content of the sections is sorted alphabetically according the EB tresos AutoCore Generic module names.

For further information on the functional behavior of these modules, refer to the chapter BL for Essentials user's guide.

4.2. APP

4.2.1. Configuration parameters

Containers included		
Container name	Multiplicity	Description
General	1..1	This container describes the general properties of the node.
Bootloader_Options	1..1	This container contains the options for bootloader mode.
CommonPublishedInformation	1..1	Label: Common Published Information Common container, aggregated by all modules. It contains published information about vendor and versions.
PublishedInformation	1..1	Label: EB Published Information Additional published parameters not covered by CommonPublishedInformation container.

4.2.1.1. General



4.2.1.2. Bootloader_Options

4.2.1.3. CommonPublishedInformation

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

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

Parameter Name	ArMinorVersion
Label	AUTOSAR Minor Version
Description	Minor version number of AUTOSAR specification on which the appropriate implementation is based on.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0



Configuration class	PublishedInformation:	
Origin	Elektrobit Automotive GmbH	

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

Parameter Name	SwMajorVersion
Label	Software Major Version
Description	Major version number of the vendor specific implementation of the module.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	3
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	SwMinorVersion
Label	Software Minor Version
Description	Minor version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	8
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	SwPatchVersion
Label	Software Patch Version



Description	Patch level version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	ModuleId
Label	Numeric Module ID
Description	Module ID of this module from Module List
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	VendorId
Label	Vendor ID
Description	Vendor ID of the dedicated implementation of this module according to the AU-TOSAR vendor list
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	1
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	Release
Label	Release Information
Multiplicity	1..1
Type	STRING_LABEL
Default value	
Configuration class	PublishedInformation:



Origin	Elektrobit Automotive GmbH
--------	----------------------------

4.2.1.4. PublishedInformation

Parameters included	
Parameter name	Multiplicity
PbcfgMSupport	1..1

Parameter Name	PbcfgMSupport
Label	PbcfgM support
Description	Specifies whether or not the APP can use the PbcfgM module for post-build support.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

4.2.2. Application programming interface (API)

4.2.2.1. Functions

4.2.2.1.1. APP_CalculateCrc

Purpose	Implement CRC calculation.	
Synopsis	u16 APP_CalculateCrc (tFlashAddress uFlashAddress , u32 ulLen) ;	
Parameters (in)	uFlashAddress	flash data address
	ulLen	flash data length
Return Value	CRC value	



Description	This function allows to calculate CRC from an custom algorithm Only available for EB BOOTLOADER
--------------------	--

4.2.2.1.2. APP_Init

Purpose	Initialize layer.
Synopsis	<code>void APP_Init (void);</code>
Description	This function initializes all layers of the complete STACK. EB_Init is called in this API. It shall be called only once at ECU startup.

4.2.2.1.3. APP_Manage

Purpose	Regular tick of the layer.
Synopsis	<code>void APP_Manage (void);</code>
Description	Ensure cyclic tasks of the layer.

4.2.2.1.4. APP_ReadBootFlag

Purpose	Get Boot flag from NVM.	
Synopsis	<code>tFlashData APP_ReadBootFlag (void);</code>	
Return Value	Boot flag from NVM	
	DOWNLOAD_REQUESTED	Download is requested from application
	GO_IN_BOOT	Request to switch to Bootloader
	GO_IN_APP	Request to switch to Application
	STAY_IN_BOOT	Request to stay in Bootloader
Description	<p>This function allows to get Boot flag. It is called at start up to either:</p> <ul style="list-style-type: none">▶ switch to Bootloader▶ switch to Application▶ download new binary <p>Only available for PSA BOOTLOADER</p>	



4.2.2.1.5. APP_ReprogReqManage

Purpose	Handle switching from Appli to BOOT.
Synopsis	<code>void APP_ReprogReqManage (void);</code>
Description	<p>This function is called in APP_Manage (scheduler). If switching from Appli to Boot is allowed, UDS response requested in Appli is now sent.</p> <p>Only available for EB, RSA_CAN_HS and PSA BOOTLOADER</p>

4.2.2.1.6. APP_WriteBootFlag

Purpose	Write Boot flag into NVM.	
Synopsis	<code>tFlashStatus APP_WriteBootFlag (tFlashData aubData);</code>	
Parameters (in)	aubData	Boot flag value (DOWNLOAD_REQUESTED/GO_IN_BOOT/GO_IN_APP)
Return Value	Flash status	
	FLASH_NO_ERROR	no error in flash
	FLASH_ACCESS_ERROR	protection error in flash (access refused)
Description	<p>This function allows to set Boot flag:</p> <ul style="list-style-type: none"> ▶ at the end of downloading sequence (switch from Boot to Appli). ▶ in application if a downloading is requested. <p>Only available for PSA BOOTLOADER</p>	

4.2.2.1.7. PROG_WriteBootFlag

Purpose	Write Boot flag into NVM.	
Synopsis	<code>void PROG_WriteBootFlag (u32 ulBootram);</code>	
Parameters (in)	ulBootram	boot flag value (DOWNLOAD_REQUESTED/GO_IN_BOOT/GO_IN_APP)
Description	<p>This function is called by PROG to write a new value of boot flag in Memory. The save in memory shall be done by the customer.</p> <p>Only available for RSA BOOTLOADER</p>	

4.2.3. Integration notes

4.2.3.1. Exclusive areas

Exclusive areas information is not available for this module.

4.2.3.2. Production errors

Production errors information is not available for this module.

4.2.3.3. Memory mapping

General information about memory mapping is provided in the EB tresos AutoCore Generic documentation. Refer to the section [Memory mapping and compiler abstraction](#) in the [Integration notes](#) section for details.

Memory mapping information is not available for this module.

4.2.3.4. Integration requirements

WARNING**Integration requirements list is not exhaustive**

The following list of integration requirements helps you to integrate your product. However, this list is not exhaustive. You also require information from the user guide, release notes, and EB tresos AutoCore known issues to successfully integrate your product.

Integration requirements are not listed for the APP module.

4.3. BLUpdater

4.3.1. Configuration parameters

Containers included		
Container name	Multiplicity	Description



Containers included		
General	1..1	This container contains the general boot manager configuration element.
Bootloader_Configuration	1..1	<p>This container contains the description of the Memory Segment of Old and New Bootloader</p> <p>The memory addresses concerned to the respective blocks are described in this section.</p>
Memory	0..n	This container contains the description of the memories used by the bootloader.
Security	1..1	
PduConnection	0..n	Configuration of all the PduConnection
CommonPublishedInformation	1..1	<p>Label: Common Published Information</p> <p>Common container, aggregated by all modules. It contains published information about vendor and versions.</p>
PublishedInformation	1..1	<p>Label: EB Published Information</p> <p>Additional published parameters not covered by CommonPublishedInformation container.</p>

4.3.1.1. General

Parameters included	
Parameter name	Multiplicity
BLU_Communication_Support	1..1
Can_Protocol_Supported	1..1
FlexRay_Protocol_Supported	1..1
Eth_Protocol_Supported	1..1
Lin_Protocol_Supported	1..1
NRC78_Sending_Period	1..1
BLU_MANAGE_PERIOD	1..1

Parameter Name	BLU_Communication_Support
Description	This entry allows to specify if the Communication shall be supported
Multiplicity	1..1
Type	BOOLEAN
Default value	true



Origin	EB
---------------	----

Parameter Name	Can_Protocol_Supported
Description	This entry allows to specify if the CAN network shall be supported
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	FlexRay_Protocol_Supported
Description	This entry allows to specify if the FlexRay network shall be supported
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Eth_Protocol_Supported
Description	This entry allows to specify if the Ethernet network shall be supported
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Lin_Protocol_Supported
Description	This entry allows to specify if the LIN network shall be supported
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	NRC78_Sending_Period
Label	NRC78 Sending Period
Description	Specify the NRC78 Sending Period value in ms. After this timeout NRC78 will be triggered Default value: 4000 (4 Seconds)



	<p>It shall be multiple of BLU_MANAGE_PERIOD.</p> <p>There will be an error in Error Log when NRC78 Sending Period is not a multiple of BLU_MANAGE_PERIOD</p>
Multiplicity	1..1
Type	INTEGER
Default value	4000
Origin	EB

Parameter Name	BLU_MANAGE_PERIOD
Label	Manage Period
Description	Specifies the period of the BLU manage task in ms.
Multiplicity	1..1
Type	INTEGER
Default value	10
Range	<p>>=1</p> <p><=25</p>
Origin	EB

4.3.1.2. Bootloader_Configuration

Containers included		
Container name	Multiplicity	Description
Current_Bootloader	1..1	
New_Bootloader	1..1	

4.3.1.3. Current_Bootloader

Parameters included	
Parameter name	Multiplicity
Memory	1..1
Current_Bootloader_Start_Address	1..1
Current_Bootloader_End_Address	1..1

Parameter Name	Memory
Description	Reference to the memory which contains the segment. <ul style="list-style-type: none"> ▶ FLASH ▶ FLASH_EXT ▶ EEPROM ▶ RAM ▶ SCRATCHPAD
Multiplicity	1..1
Type	REFERENCE
Origin	EB

Parameter Name	Current_Bootloader_Start_Address
Description	Start address of Current Bootloader Coded on 32 bits
Multiplicity	1..1
Type	INTEGER
Origin	EB

Parameter Name	Current_Bootloader_End_Address
Description	End address of Current Bootloader Coded on 32 bits
Multiplicity	1..1
Type	INTEGER
Origin	EB

4.3.1.4. New_Bootloader

Parameters included	
Parameter name	Multiplicity
Memory	1..1
New_Bootloader_Start_Address	1..1
New_Bootloader_End_Address	1..1



Parameter Name	Memory
Description	Reference to the memory which contains the segment. <ul style="list-style-type: none"> ▶ FLASH ▶ FLASH_EXT ▶ EEPROM ▶ RAM ▶ SCRATCHPAD
Multiplicity	1..1
Type	REFERENCE
Origin	EB

Parameter Name	New_Bootloader_Start_Address
Description	Start address of New Bootloader Image which is present inside the Bootloader Updater. Coded on 32 bits
Multiplicity	1..1
Type	INTEGER
Origin	EB

Parameter Name	New_Bootloader_End_Address
Description	End address of New Bootloader Image Coded on 32 bits
Multiplicity	1..1
Type	INTEGER
Origin	EB

4.3.1.5. Memory

Parameters included	
Parameter name	Multiplicity
Memory_Type	1..1
Memory_Mode	1..1
Min_Value_To_Write	1..1



Parameters included

Addr_Offset	1..1
Erase_Value	1..1

Parameter Name	Memory_Type
Label	Memory Type
Description	<p>Type of the memory.</p> <ul style="list-style-type: none"> ▶ FLASH ▶ FLASH_EXT ▶ RAM ▶ CUSTOM <p>NOTE: Only one memory of each type can be defined!</p>
Multiplicity	1..1
Type	ENUMERATION
Range	<p>FLASH</p> <hr/> <p>FLASH_EXT</p> <hr/> <p>RAM</p> <hr/> <p>CUSTOM</p>
Origin	EB

Parameter Name	Memory_Mode
Label	Memory Mode
Description	<p>This entry allows to specify if the memory is access synchronously or asynchronously.</p> <p>Please select between :</p> <ul style="list-style-type: none"> ▶ asynchronous ▶ synchronous
Multiplicity	1..1
Type	STRING
Default value	synchronous
Range	<p>asynchronous</p> <hr/> <p>synchronous</p>



Origin	EB
---------------	----

Parameter Name	Min_Value_To_Write
Label	Minimum value to write
Description	Define the minimum size the memory driver could write at a time. Coded on 32 bits.
Multiplicity	1..1
Type	INTEGER
Default value	8
Range	>=8
Origin	EB

Parameter Name	Addr_Offset
Label	Address Offset
Description	Define the start address offset for the memory. In the case of the EXTERNAL FLASH and CUSTOM this offset is SUBSTRACTED from the received address. In the case of the RAM and INTERNAL FLASH this offset is ADDED to the received address. Coded on 32 bits.
Multiplicity	1..1
Type	INTEGER
Default value	0
Origin	EB

Parameter Name	Erase_Value
Label	Erase Value
Description	Define the value to set for each byte when the memory is erased Example: 0xFFu will affect the value 0xFF to every byte of the memory when erasure. Range: [0x00 ; 0xFF]. Coded on 8 bits.
Multiplicity	1..1



Type	INTEGER
Default value	255
Range	≥ 0 ≤ 255
Origin	EB

4.3.1.6. Security

Containers included		
Container name	Multiplicity	Description
SecureBoot	1..1	This container contains all configurations for Secure bootloader features. Configuration can be done only if Secure features are activated.

4.3.1.7. SecureBoot

Parameters included	
Parameter name	Multiplicity
SECURE_BOOT	1..1
BootCksStartAddress	1..1
BootCksRangeLength	1..1

Parameter Name	SECURE_BOOT
Label	Enable Secure Boot
Description	<p>Enable or disable the Secure Boot features. When Secure Boot feature is enabled, new Bootloader checksum will be generated once the new bootloader updated successfully.</p> <ul style="list-style-type: none"> ▶ If this box is checked, Secure boot feature is enabled. ▶ Otherwise, Secure boot feature is disabled
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB



Parameter Name	BootCksStartAddress
Label	Bootloader checksum start address
Description	This value indicated from which address the bootloader checksum shall be computed. This value is 4 bytes long
Multiplicity	1..1
Type	INTEGER
Default value	00000000
Range	<=4294967295
Origin	EB

Parameter Name	BootCksRangeLength
Label	Bootloader checksum range length
Description	This value indicated the data length on which the Bootloader checksum shall be computed. It can be the full Bootloader software length or only a part of the bootloader software. This value is 4 bytes long
Multiplicity	1..1
Type	INTEGER
Default value	00000000
Range	<=4294967295
Origin	EB

4.3.1.8. PduConnection

Containers included		
Container name	Multiplicity	Description
RxPdu	0..n	Configuration of all the RxPdus references

Parameters included	
Parameter name	Multiplicity
TxPduRef	1..1
TxConfPduld	1..1



Parameters included

TesterAddress	1..1
LinConnection	1..1
ShareFunctionalId	1..1
SharedPduReference	1..1

Parameter Name	TxPduRef
Label	TxPdu Reference
Description	<p>Reference to the Pdu in the EcucPduCollection configured for this Transmission Channel.</p> <p>Through this reference, BLUpdater can resolve the PduId used for Transmission in the PduR_BlUpdaterTransmit() API and defined by the PduR.</p> <p>Through this reference, PduRouter can resolve the PduId used for Transmission by in the following APIs:</p> <p>and defined by the BlUpdater in TxConfirmationPduId.</p>
Multiplicity	1..1
Type	REFERENCE
Origin	AUTOSAR_ECUC

Parameter Name	TxConfPduld
Label	TxPdu Identifier
Description	<p>This entry allows to configure the TxConfirmationPduld that shall be used by PduR to transmit Tx confirmation</p> <p>In case of Lin connection this field is used to define the MsgIdx of the configured LTP message</p>
Multiplicity	1..1
Type	INTEGER
Default value	0
Origin	EB

Parameter Name	TesterAddress
Label	Tester Address
Description	This defines the Tester Address used in source/target address of the current connection
Multiplicity	1..1



Type	INTEGER
Default value	0
Origin	EB

Parameter Name	LinConnection
Label	Lin Connection
Description	Defines if the connection is to managed Lin Pdu If so the generation of this connection will be done differently
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	ShareFunctionalId
Label	Share Functional Id
Description	Defines if this connection shall reuse an existing functional Id from another connection
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	SharedPduReference
Label	Shared Pdu Reference
Description	Reference to the shared Pdu
Multiplicity	1..1
Type	REFERENCE
Origin	EB

4.3.1.9. RxPdu

Parameters included	
Parameter name	Multiplicity



Parameters included

Ref	1..1
Type	1..1
Id	1..1

Parameter Name	Ref
Label	RxPdu Reference
Description	Reference to <code>Pdu</code> from the <code>EcucPduCollection</code> configured for this Reception Channel. Through this reference, the <code>PduRouter</code> can resolve the <code>PduId</code> used for Reception by <code>BLUpdator</code> in the following APIs:
Multiplicity	1..1
Type	REFERENCE
Origin	AUTOSAR_ECUC

Parameter Name	Type
Description	This entry allows to specify which kind of RxPdu is used. Please select between : ▶ PHYSICAL ▶ FUNCTIONAL
Multiplicity	1..1
Type	STRING
Default value	PHYSICAL
Range	PHYSICAL FUNCTIONAL
Origin	EB

Parameter Name	Id
Description	This entry allows to configure the RxPduld that shall be used by BLUpdator to transmit diagnostic response In case of Lin connection this field is used to define the MsgIdx of the configured LTP message
Multiplicity	1..1
Type	INTEGER

Default value	0
Origin	EB

4.3.1.10. CommonPublishedInformation

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

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

Parameter Name	ArMinorVersion
Label	AUTOSAR Minor Version
Description	Minor version number of AUTOSAR specification on which the appropriate implementation is based on.
Multiplicity	1..1
Type	INTEGER_LABEL



Default value	0
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

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

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

Parameter Name	SwMinorVersion
Label	Software Minor Version
Description	Minor version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	3
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	SwPatchVersion
-----------------------	-----------------------



Label	Software Patch Version
Description	Patch level version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	ModuleId
Label	Numeric Module ID
Description	Module ID of this module from Module List
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	VendorId
Label	Vendor ID
Description	Vendor ID of the dedicated implementation of this module according to the AU-TOSAR vendor list
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	1
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	Release
Label	Release Information
Multiplicity	1..1
Type	STRING_LABEL
Default value	
Configuration class	PublishedInformation:

Origin	Elektrobit Automotive GmbH
--------	----------------------------

4.3.1.11. PublishedInformation

Parameters included	
Parameter name	Multiplicity
PbcfgMSupport	1..1

Parameter Name	PbcfgMSupport
Label	PbcfgM support
Description	Specifies whether or not the BLUpdater can use the PbcfgM module for post-build support.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

4.3.2. Application programming interface (API)

4.3.2.1. Type definitions

4.3.2.1.1. tBLUBoolean

Purpose	
Type	u8

4.3.2.1.2. tBLUOperation

Purpose	
---------	--



Type	u8
------	----

4.3.2.1.3. tBLUStatus

Purpose	
Type	u8

4.3.2.1.4. tCfgSegmentType

Purpose	
Type	struct
Members	u32 ulStartAddress
	u32 ulEndAddress

4.3.2.2. Macro constants

4.3.2.2.1. BLU_COMPARE

Purpose	
Value	0x02U

4.3.2.2.2. BLU_ERASE

Purpose	
Value	0x00U

4.3.2.2.3. BLU_E_BUSY

Purpose	
Value	0x02U



4.3.2.2.4. BLU_E_NOT_OK

Purpose	
Value	0x01U

4.3.2.2.5. BLU_E_OK

Purpose	
Value	0x00U

4.3.2.2.6. BLU_FALSE

Purpose	
Value	0U

4.3.2.2.7. BLU_FINISH

Purpose	
Value	0x04U

4.3.2.2.8. BLU_SECUREBOOT_UPDATE

Purpose	
Value	0x03U

4.3.2.2.9. BLU_TRUE

Purpose	
Value	1U

4.3.2.2.10. BLU_WRITE

Purpose	
---------	--



Value	0x01U
--------------	-------

4.3.2.3. Functions

4.3.2.3.1. BLU_CustomBootChecksumUpdate

Purpose	Callback to update the checksum of the Bootloader.	
Synopsis	<code>tBLUStatus BLU_CustomBootChecksumUpdate (u32 ulBootStartAddress , u32 ulBootSize);</code>	
Parameters (in)	ulBootStartAddress	: Start of the Boot to be verified by the SecureBoot
	ulBootSize	: Size of the Boot to be verified by the SecureBoot
Return Value	BLU_E_OK	Treatment finish successfully
	BLU_E_NOT_OK	Error happened during treatment
BLU_E_BUSY	Treatment in progress	

4.3.2.3.2. BLU_CustomGetPduID

Purpose	Callback to get Pdu ID Get the Rx Pdu Identifier on which the response after reset shall be sent.	
Synopsis	<code>void BLU_CustomGetPduID (u16 * pubRxPduId);</code>	
Parameters (out)	pubRxPduId	Rx Pdu Identifier pointer

4.3.2.3.3. BLU_CustomSetValidityBLUpdater

Purpose	Callback to set validity marker pf the Bootloader Updater.	
Synopsis	<code>tBLUBoolean BLU_CustomSetValidityBLUpdater (tBLUBoolean eBLUValidity);</code>	
Parameters (in)	eBLUValidity	status to write
	Result application invalidation	
	BLU_TRUE	Treatment finish successfully



	BLU_FALSE	Error happened during treatment
Description	The validity of the Bootloader Updater shall be saved here	

4.3.2.3.4. BLU_CustomSetValidityNewBootloader

Purpose	Callback to set validity marker for New Bootloader.	
Synopsis	<code>tBLUBoolean BLU_CustomSetValidityNewBootloader (tBLUBoolean eNewBootValidity);</code>	
Parameters (in)	eNewBootValidity	status to write
Return Value	Result application validation	
	BLU_TRUE	finish successfully
	BLU_FALSE	Error happened during treatment (Error-Code shall be filled in this case)
Description	The validity of the New Bootloader shall be saved here	

4.3.2.3.5. BLU_CustomTriggerWatchdog

Purpose	Callback to trigger the Watchdog Watchdog can be triggered directly by using this call-back.
Synopsis	<code>void BLU_CustomTriggerWatchdog (void);</code>

4.3.2.3.6. BLU_Custom_Com_Init

Purpose	Initialization of modules of communication stack.
Synopsis	<code>void BLU_Custom_Com_Init (void);</code>

4.3.2.3.7. BLU_Init

Purpose	API that allows to initialize all parameters before calling any other functions.
Synopsis	<code>void BLU_Init (void);</code>
Description	This API is used to initialize all parameters for Bootloader Updater



4.3.2.3.8. BLU_Manage

Purpose	API that allows to achieve internal cyclic feature.
Synopsis	<code>void BLU_Manage (void);</code>
Description	This API is used to schedule internal cyclic feature

4.3.2.3.9. BLU_Start

Purpose	API that executes all different activities of BLUpdater.
Synopsis	<code>void BLU_Start (void);</code>
Description	This API is called for calling all actions of erase, write and compare for BLUpdater, trigger different operations in flash and provides the current status of flash

4.3.2.3.10. BLUpdater_PduRIfRxIndication

Purpose	API that will be used for reception indication of message frame.	
Synopsis	<code>void BLUpdater_PduRIfRxIndication (PduIdType RxPduId , PduInfoType * PduInfoPtr);</code>	
Parameters (in)	RxPduId	Identification of the received I-PDU.
	PduInfoPtr	Pointer to the buffer (SduDataPtr) and its length (SduLength) containing the data to be copied by PDU Router module

4.3.2.3.11. BLUpdater_PduRIfTxConfirmation

Purpose	API that will be used for transmission confirmation of message frame.	
Synopsis	<code>void BLUpdater_PduRIfTxConfirmation (PduIdType TxPduId);</code>	
Parameters (in)	TxPduId	Identification of the transmitted I-PDU.

4.3.3. Integration notes

4.3.3.1. Exclusive areas

Exclusive areas information is not available for this module.



4.3.3.2. Production errors

Production errors information is not available for this module.

4.3.3.3. Memory mapping

General information about memory mapping is provided in the EB tresos AutoCore Generic documentation. Refer to the section [Memory mapping and compiler abstraction](#) in the [Integration notes](#) section for details.

Memory mapping information is not available for this module.

4.3.3.4. Integration requirements

WARNING**Integration requirements list is not exhaustive**

The following list of integration requirements helps you to integrate your product. However, this list is not exhaustive. You also require information from the user guide, release notes, and EB tresos AutoCore known issues to successfully integrate your product.

Integration requirements are not listed for the BLUpdater module.

4.4. BM

4.4.1. Configuration parameters

Containers included		
Container name	Multiplicity	Description
General	1..1	This container contains the general boot manager configuration element.
Security	1..1	
CommonPublishedInformation	1..1	Label: Common Published Information



Containers included

		Common container, aggregated by all modules. It contains published information about vendor and versions.
PublishedInformation	1..1	Label: EB Published Information Additional published parameters not covered by Common-PublishedInformation container.

4.4.1.1. General

Parameters included

Parameter name	Multiplicity
Boot_Manager_Variant	1..1
Test_Application	1..1
Initial_Boot_Check_Application	1..1
BM_TIMEOUT_CHECK	1..1
BM_TIMEOUT_VALUE	1..1
BM_CALL_CYCLE	1..1
BM_FR_CYCLE_LENGTH	1..1
BM_SOURCE_ADDRESS_CHECK	1..1

Parameter Name

Boot_Manager_Variant

Label

Boot Manager Variant

Description

It allows to select between Initial Boot Manager and Boot Manager :

- ▶ Initial Boot Manager : Checks if jump is needed towards Bootloader Updater, Bootloader or Application. Use this option if a Bootloader Updater is needed.
- ▶ Boot Manager : Checks if jump is needed towards Bootloader or Application.

Multiplicity

1..1

Type

STRING

Default value

Boot Manager

Range

Initial Boot Manager

Boot Manager

Origin

EB



Parameter Name	Test_Application
Label	Support_TestApplication
Description	Tick this to enable the test Application execution from BM.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Initial_Boot_Check_Application
Label	Check Application Validity
Description	Enable or disable the Application Validity Check Initial Boot Manager
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	BM_TIMEOUT_CHECK
Description	This entry defines if a delay has to be waited before checking the application Validity. Note: <ul style="list-style-type: none">▶ In case of CAN network: it allows to start a new reprog session even if the application is valid, by receiving a DSC02 during this delay.▶ In case of FLEXRAY network: it is forced to true.
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	BM_TIMEOUT_VALUE
Description	Define the delay value to wait before checking application validity Note: <ul style="list-style-type: none">▶ In case of CAN network: it corresponds to the max time (in ms) waiting for DSC02 request.▶ In case of FLEXRAY network: it corresponds to the max time (in FR cycle) waiting for FR synchronization or a valid NetworkStatus.



Multiplicity	1..1
Type	INTEGER
Default value	20
Origin	EB

Parameter Name	BM_CALL_CYCLE
Description	<p>Define the periodicity of the call to BM_Manage.</p> <p>Note:</p> <ul style="list-style-type: none"> ▶ In case of CAN network: at each BM_manage a check is done to verify if the DSC02 have been received. This value shall be a multiple of BM_TIMEOUT_VALUE. ▶ In case of FLEXRAY network: a check is done to test the NetworkStatus (if the ECU is synchronized).
Multiplicity	1..1
Type	INTEGER
Default value	1
Origin	EB

Parameter Name	BM_FR_CYCLE_LENGTH
Description	<p>Define the FlexRay cycle length.</p> <p>Note: has to be configured with the same value (in ms) than the FrIfGdCycle used parameter.</p>
Multiplicity	1..1
Type	INTEGER
Default value	5
Origin	EB

Parameter Name	BM_SOURCE_ADDRESS_CHECK
Description	<p>Enable or disable the management of diagnostic source filtering on a single address</p> <p>Note:</p> <ul style="list-style-type: none"> ▶ If no programming is requested by application, all tester requests shall be accepted.
Multiplicity	1..1
Type	BOOLEAN



Default value	false
Origin	EB

4.4.1.2. Security

Containers included		
Container name	Multiplicity	Description
BMCsmReferences	1..1	Contains references to Csm configuration. If HSM is used, this container will be disabled.
SecureBoot	1..1	This container contains all configurations for Authenticated/Secure bootloader features. Configuration can be done only if Authenticated or Secure features are activated.

4.4.1.3. BMCsmReferences

Parameters included	
Parameter name	Multiplicity
BMCsmChecksumConfigId	1..1
BS_size	1..1
Cancel_OngoingJobs	1..1
SetCryptoKey	1..1

Parameter Name	BMCsmChecksumConfigId	
Label	BMCsmChecksumConfigId	
Description	Reference a <i>CsmHash</i> or a <i>CsmMacVerify</i> Dependencies: ▶ Reference shall be valid	
Multiplicity	1..1	
Type	CHOICE-REFERENCE	
Configuration class	VariantPreCompile:	VariantPreCompile
Origin	Elektrobit Automotive GmbH	

Parameter Name	BS_size
-----------------------	----------------



Label	Data size for Block Slicing
Description	<p>This feature shall allow the Integrator to decide the size of the data to be processed in a single Asynchronous Crypto main function call.</p> <p>it allows to reduce the time blocked by the CryptoMainfunction in a single main function call.</p> <p>This feature can be enabled by entering any value other than 0 in this field. Setting it to 0 shall allow the Bootlaoder to pass the entire available data for processing in a single CryptoMainfunction call.</p> <p>Note:</p> <ol style="list-style-type: none"> 1. This field is only enabled for the integration of ASR Crypto stack version 4.3 or higher. 2. Enabling this feature (by setting a very small size) shall increase the time for verification significantly, integrator shall decide the value to be set. 3. It is recommended to set this value to a maximum value that is sufficient to not to cause timeouts in the system.
Multiplicity	1..1
Type	INTEGER
Default value	0
Origin	EB

Parameter Name	Cancel_OngoingJobs
Label	Allow cancellation of any ongoing Crypto job
Description	<p>If enabled, any ongoing or active Crypto job operations shall be cancelled at the start of this job operation.</p> <p>This option is applicable only for the integration of Crypto 4.3 or higher versions, for older version this feature is realized using 'Enable the cancelation of ongoing requests' in the Csm/Cry modules.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	SetCryptoKey
Label	Allow to fetch the stored Key



Description	This will allow the integrator to pass the desired key and its elements at the start of the Crypto operation. This option is applicable only for the integration of Crypto 4.3 or higher versions.
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

4.4.1.4. SecureBoot

Parameters included	
Parameter name	Multiplicity
SECURE_AUTHENTICATED_BOOT	1..1
BMSecureBootWithHsm	1..1
BOOT_VERIFICATION	1..1
CHECKSUM_LENGTH	1..1
BootCksStartAddress	1..1
BootCksRangeLength	1..1

Parameter Name	SECURE_AUTHENTICATED_BOOT
Label	Authenticated / Secure Boot
Description	Enable or disable the Authenticated or Secure Boot features. When the Authenticated Boot feature is enabled, Application and Bootloader checksum will be verified before any software execution by comparison to the previous one. When Secure Boot feature is enabled, Application and Bootloader checksum will be computed and verified before any software execution. These two features cannot be enabled at the same time. <ul style="list-style-type: none"> ▶ OFF : Neither Authenticated Boot nor Secure Boot feature is enabled. ▶ Authenticated : Authenticated Boot feature is enabled. ▶ Secure : Secure boot feature is enabled.
Multiplicity	1..1
Type	ENUMERATION
Default value	OFF
Range	OFF



	Authenticated
	Secure
Origin	EB

Parameter Name	BMSecureBootWithHsm
Label	Use HSM (Hardware Security Module)
Description	<p>Use HSM to implement Secure Boot feature.</p> <p>Note:</p> <ul style="list-style-type: none"> ▶ If this box is checked, CSM is not used for implementing Secure Boot feature. ▶ HSM does not support Authenticated Boot feature.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	BOOT_VERIFICATION
Label	Bootloader verified/ Bootloader not verified
Description	<p>Enable or disable the Bootloader verification at startup when Secure Boot or Authenticated Boot feature are enabled. If a HSM is used the Bootloader verification shall be disabled as ensured by the HSM.</p> <ul style="list-style-type: none"> ▶ ON : Bootloader is verified at startup. ▶ OFF : Bootloader is not verified at startup. <p>When HSM is used, this feature will no longer be handled by the bootloader software itself.</p>
Multiplicity	1..1
Type	ENUMERATION
Default value	OFF
Range	OFF ON
Origin	EB

Parameter Name	CHECKSUM_LENGTH
Label	Checksum length



Description	Size of the checksum for Authenticated / Secure Boot feature in Bytes
Multiplicity	1..1
Type	INTEGER
Default value	512
Origin	EB

Parameter Name	BootCksStartAddress
Label	Bootloader checksum start address
Description	This value indicated from which address the bootloader checksum shall be computed. This value is 4 bytes long
Multiplicity	1..1
Type	INTEGER
Default value	00000000
Range	<=4294967295
Origin	EB

Parameter Name	BootCksRangeLength
Label	Bootloader checksum range length
Description	This value indicated the data length on which the Bootloader checksum shall be computed. It can be the full Bootloader software length or only a part of the bootloader software. This value is 4 bytes long
Multiplicity	1..1
Type	INTEGER
Default value	00000000
Range	<=4294967295
Origin	EB

4.4.1.5. CommonPublishedInformation

Parameters included	
Parameter name	Multiplicity

Parameters included

ArMajorVersion	1..1
ArMinorVersion	1..1
ArPatchVersion	1..1
SwMajorVersion	1..1
SwMinorVersion	1..1
SwPatchVersion	1..1
ModuleId	1..1
VendorId	1..1
Release	1..1

Parameter Name

[ArMajorVersion](#)

Label

AUTOSAR Major Version

Description

Major version number of AUTOSAR specification on which the appropriate implementation is based on.

Multiplicity

1..1

Type

INTEGER_LABEL

Default value

0

Configuration class

[PublishedInformation:](#)

Origin

Elektrobit Automotive GmbH

Parameter Name

[ArMinorVersion](#)

Label

AUTOSAR Minor Version

Description

Minor version number of AUTOSAR specification on which the appropriate implementation is based on.

Multiplicity

1..1

Type

INTEGER_LABEL

Default value

0

Configuration class

[PublishedInformation:](#)

Origin

Elektrobit Automotive GmbH

Parameter Name

[ArPatchVersion](#)

Label

AUTOSAR Patch Version

Description

Patch level version number of AUTOSAR specification on which the appropriate implementation is based on.



Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

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

Parameter Name	SwMinorVersion
Label	Software Minor Version
Description	Minor version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	13
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	SwPatchVersion
Label	Software Patch Version
Description	Patch level version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	ModuleId
Label	Numeric Module ID
Description	Module ID of this module from Module List
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	VendorId
Label	Vendor ID
Description	Vendor ID of the dedicated implementation of this module according to the AU-TOSAR vendor list
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	1
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

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

4.4.1.6. PublishedInformation

Parameters included	
Parameter name	Multiplicity
PbcfgMSupport	1..1
Parameter Name	PbcfgMSupport



Label	PbcfgM support
Description	Specifies whether or not the BM can use the PbcfgM module for post-build support.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

4.4.2. Application programming interface (API)

4.4.2.1. Functions

4.4.2.1.1. BM_CheckProgRequest

Purpose	API that check if a programming request has been received by the application.	
Synopsis	<code>tBMBoolean BM_CheckProgRequest (void);</code>	
Return Value	Result of check	
	TRUE	Reprogramming request has been received
	FALSE	No reprogramming request received
Description	<p>Callback is called: at Initial Boot startup to know if a programming request has been received in Application</p> <p>Callback shall implement: get information from application if a programming request has been received (e.g: read a flag from noinit RAM shared between Bootloader and Application)</p>	

4.4.2.1.2. BM_CustomAllowTestApplExe

Purpose	Notify the Integration-code to check and allow Test-Application execution.
----------------	--



Synopsis	<code>void BM_CustomAllowTestApplExe (void);</code>
Description	This function is called at the startup to notify the Integration code to verify and execute the Test-Application if allowed to be executed. If the execution of the Test-Application is not allowed, callback shall return the control to BM.

4.4.2.1.3. BM_CustomBckOperation

Purpose	Allow customer operation during long operation.
Synopsis	<code>void BM_CustomBckOperation (void);</code>
Description	This function is called during long operation (e.g. cryptography) allowing customer performing actions that cannot be stopped during a long time (e.g watchdog triggering).

4.4.2.1.4. BM_CustomCheckValidAppl

Purpose	Callback checking if the application is valid or not.	
Synopsis	<code>tBMBoolean BM_CustomCheckValidAppl (void);</code>	
Return Value	Result of check	
	TRUE	Application is valid
	FALSE	Application is not valid or not present
Description	<p>Callback is called: at Initial Boot startup.</p> <p>Callback shall implement: It shall verify that the application with all its dependencies are correctly flashed and return the result of the check</p>	

4.4.2.1.5. BM_CustomCheckValidBL

Purpose	Callback checking if the Bootloader is valid or not.	
Synopsis	<code>tBMBoolean BM_CustomCheckValidBL (void);</code>	
Return Value	Result of check	
	TRUE	Bootloader is valid
	FALSE	Bootloader is not valid or not present
Description	Callback is called: at Initial Boot startup.	



	Callback shall implement: It shall verify that the Bootloader is correctly flashed and return the result of the check
--	---

4.4.2.1.6. BM_CustomCheckValidBLU

Purpose	Callback checking if the Bootloader Updater Programm is valid or not.	
Synopsis	<code>tBMBoolen BM_CustomCheckValidBLU (void);</code>	
Return Value	Result of check	
	TRUE	BLU Updater Programm is valid
	FALSE	BLU is not valid or not present
Description	<p>Callback is called: at Initial Boot startup.</p> <p>Callback shall implement: It shall verify that the Bootloader Updater Programm is correctly flashed and return the result of the check</p>	

4.4.2.1.7. BM_CustomDualBankInit

Purpose	Initialize the Dual Memory Bank.
Synopsis	<code>void BM_CustomDualBankInit (void);</code>
Description	This function is called when Bootloader starts to prepare the usage of the dual memory banks.

4.4.2.1.8. BM_CustomGetComputedApplicationChecksum

Purpose	Get the last computed application checksum.	
Synopsis	<code>void BM_CustomGetComputedApplicationChecksum (u8 * pubChecksum , u16 uwBlockIdentifier);</code>	
Parameters (in)	<code>uwBlockIdentifier</code>	Block Identifier
Parameters (out)	<code>pubChecksum</code>	address where the computed checksum shall be copied
Description	This function is called at Bootloader startup to verify the application checksum before allowing the start of application. This checksum shall have been computed by application on configured application data range.	



4.4.2.1.9. BM_CustomGetExpectedApplicationChecksum

Purpose	Get the expected application checksum.	
Synopsis	<code>void BM_CustomGetExpectedApplicationChecksum (u8 * pubChecksum , u16 uwBlockIdentifier);</code>	
Parameters (in)	uwBlockIdentifier	Block Identifier
Parameters (out)	pubChecksum	address where the checksum shall be copied
Description	This function is called at Bootloader startup to verify the application checksum before allowing the start of application. It gets from non-volatile memory the checksum that had been computed during the last application download.	

4.4.2.1.10. BM_CustomGetMacKey

Purpose	Get the mac key used in Mac verification (should be the same key used in the generation).	
Synopsis	<code>void BM_CustomGetMacKey (const u8 ** paubKeyData , u32 * pulKeyLength);</code>	
Description	This function is called at the beginning of the Mac verification to get the key yo be used in the process	

4.4.2.1.11. BM_CustomHsmVerifyMac

Purpose	Verify the Mac with HSM This function is called to trigger a Mac verification for corresponding block.	
Synopsis	<code>tBMHsmJobResult BM_CustomHsmVerifyMac (u16 uwBlockId);</code>	
Parameters (in)	uwBlockId	Block Identifier
Return Value	Result of check	
	BM_HSM_JOB_OK	Mac verification succeed
	BM_HSM_JOB_FAILED	Mac verification failed
	BM_HSM_JOB_PENDING	Mac verification is pending

4.4.2.1.12. BM_CustomIsNormalStartup

Purpose	Request if ECU has started normally or not.
----------------	---



Synopsis	tBMBoolen BM_CustomIsNormalStartup (void);
Return Value	
Description	This function is called at startup to know if normal or abnormal startup has been done. In case of abnormal startup ECU stay in Bootloader mode during a configured time before jumping to application (if valid). This time window allow the tester to send a programming session request to force the Boot mode

4.4.2.1.13. BM_CustomSetInvalidAppli

Purpose	Notification that Application is invalid.	
Synopsis	void BM_CustomSetInvalidAppli (u16 uwBlockIdentifier);	
Parameters (in)	uwBlockIdentifier	Identifier of invalid Block
Description	This function is called when Application checksum verification failed and that Application cannot be started. This information shall be store in non-volatile memory.	

4.4.2.1.14. BM_CustomSetInvalidBlu

Purpose	Notification that Application is invalid.	
Synopsis	void BM_CustomSetInvalidBlu (void);	
Description	This function is called when BLUpdater checksum verification failed and that BLUpdater cannot be started. This information shall be store in non-volatile memory.	

4.4.2.1.15. BM_CustomSetInvalidBoot

Purpose	Notification that Bootloader is invalid.	
Synopsis	void BM_CustomSetInvalidBoot (void);	
Description	This function is called when Bootloader checksum verification failed and that Bootloader cannot be started. By this callback application shall be informed that Bootloader cannot be executed anymore.	

4.4.2.1.16. BM_DisableECCCheck

Purpose	Callback that shall disable ECC if needed Callback is called: When Bootloader perform read access on Flash that can be unprogrammed and can cause ECC error.
----------------	--



Synopsis	<code>void BM_DisableECCCheck (void);</code>
Description	Callback shall implement: If needed, disabling of ECC check (Hardware specific)

4.4.2.1.17. BM_EnableECCCheck

Purpose	Callback that shall enable ECC if needed. Callback is called: After Bootloader has performed a read access on Flash that can be unprogrammed and can cause ECC error.
Synopsis	<code>void BM_EnableECCCheck (void);</code>
Description	Callback shall implement: If needed, enabling of ECC check (Hardware specific)

4.4.2.1.18. BM_GetTesterAddress

Purpose	Get the diagnostic tester source address.	
Synopsis	<code>void BM_GetTesterAddress (u8 * ubTesterAddress);</code>	
Parameters (in,out)	ubTesterAddress	pointer on tester address
Description	This function is called when programming is requested by application (eBootFromAppli = BM_TRUE) and BM_SOURCE_ADDRESS_CHECK is set. This is a callback that get the tester address	

4.4.2.1.19. BM_HardwareInit

Purpose	Hardware initialization.
Synopsis	<code>void BM_HardwareInit (void);</code>
Description	This function is called at the very beginning of the Boot manager. It allows the application to do the minimum hardware initialization before the Boot manager start to check the application validity.

4.4.2.1.20. BM_InitialBootStartUp

Purpose	Initial Boot Manager Startup.
Synopsis	<code>void BM_InitialBootStartUp (void);</code>
Description	Called at the startup of the ECU to check if we start the Application, BLUpdater or the Bootloader



This function handles:

- ▶ The check of application validity flag
- ▶ The check of BLUpdater validity flag
- ▶ The check of Bootloader validity flag

4.4.2.1.21. **BM_JumpToApplication**

Purpose	Callback performing jump to application software.
Synopsis	<code>void BM_JumpToApplication (void);</code>
Description	Callback is called: at Initial Boot startup if application is valid and shall be executed Callback shall implement: jump to application start address

4.4.2.1.22. **BM_JumpToBL**

Purpose	Callback performing jump to Bootloader.
Synopsis	<code>void BM_JumpToBL (void);</code>
Description	Callback is called: at Initial Boot startup if Bootloader is valid and shall be executed Callback shall implement: jump to Bootloader start address

4.4.2.1.23. **BM_JumpToBLU**

Purpose	Callback performing jump to Bootloader Updater Programm.
Synopsis	<code>void BM_JumpToBLU (void);</code>
Description	Callback is called: at Initial Boot startup if Bootloader Updater Programm is valid and shall be executed Callback shall implement: jump to BLU start address

4.4.2.1.24. **BM_SoftwareInit**

Purpose	Software RAM initialization.
Synopsis	<code>void BM_SoftwareInit (void);</code>



Description	This function is called at the very beginning of the Boot manager. It allows the application to do the minimum software initialization before the Boot manager start to check the application validity.
--------------------	---

4.4.3. Integration notes

4.4.3.1. Exclusive areas

Exclusive areas information is not available for this module.

4.4.3.2. Production errors

Production errors information is not available for this module.

4.4.3.3. Memory mapping

General information about memory mapping is provided in the EB tresos AutoCore Generic documentation. Refer to the section [Memory mapping and compiler abstraction](#) in the [Integration notes](#) section for details.

Memory mapping information is not available for this module.

4.4.3.4. Integration requirements

WARNING**Integration requirements list is not exhaustive**

The following list of integration requirements helps you to integrate your product. However, this list is not exhaustive. You also require information from the user guide, release notes, and EB tresos AutoCore known issues to successfully integrate your product.

Integration requirements are not listed for the BM module.

4.5. BIPduR



4.5.1. Configuration parameters

Containers included		
Container name	Multiplicity	Description
General	1..1	This container describes the general properties of the node.
SleepManagement	1..1	Label: Sleep Management
MultipleIdentifier	1..1	
PduConnection	0..n	Configuration of all the PduConnection
IDGroup	0..4	Configuration of all the connection ID Group
CommonPublishedInformation	1..1	Label: Common Published Information Common container, aggregated by all modules. It contains published information about vendor and versions.
PublishedInformation	1..1	Label: EB Published Information Additional published parameters not covered by CommonPublishedInformation container.

4.5.1.1. General

Parameters included	
Parameter name	Multiplicity
MANAGE_PERIOD	1..1
Can_Protocol_Supported	1..1
FlexRay_Protocol_Supported	1..1
Eth_Protocol_Supported	1..1
Lin_Protocol_Supported	1..1
QueuedManagement	1..1
MultipleRxBuffer	1..1
RxPhysicalBufferSize	1..1
RxFunctionalBufferSize	1..1
RxBufferNum	1..1
Enable_DownloadData_Streaming	1..1

Parameter Name	MANAGE_PERIOD
Description	This entry allows to configure the period of the cyclic BIPduR task.



Multiplicity	1..1
Type	INTEGER
Range	>=1
Origin	EB

Parameter Name	Can_Protocol_Supported
Description	This entry allows to specify if the CAN network shall be supported
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	FlexRay_Protocol_Supported
Description	This entry allows to specify if the FlexRay network shall be supported
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Eth_Protocol_Supported
Description	This entry allows to specify if the Ethernet network shall be supported
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Lin_Protocol_Supported
Description	This entry allows to specify if the LIN network shall be supported
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	QueuedManagement
-----------------------	-------------------------



Label	Queued Management
Description	<p>Specify if the Queued management feature is enable.</p> <p>This feature allows the bootloader to be able to receive a second physical request before finishing processing the response to the first one.</p> <p>The Bootloader shall store the second request in a FIFO queue for later processing.</p> <p>Queued management is used to reduce download time and latency caused by the gateways.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	MultipleRxBuffer
Label	Multiple Receive Buffer
Description	<p>Specify if the multiple receive buffer feature is enable</p> <p>This feature allows supporting in parallel data reception and data flash writing.</p> <p>Multiple receive buffers is used to improve global downloading time</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	RxPhysicalBufferSize
Label	Rx Physical Buffer Size
Description	<p>Define the size of the Rx physical buffer in bytes</p> <p>If the multiple buffer is enabled, all physical buffers will have this same size</p>
Multiplicity	1..1
Type	INTEGER
Default value	4095
Origin	EB

Parameter Name	RxFUNCTIONALBufferSize
-----------------------	-------------------------------



Label	Rx Functional Buffer Size
Description	Define the size of the Rx functional buffer in bytes
Multiplicity	1..1
Type	INTEGER
Default value	8
Origin	EB

Parameter Name	RxBUFFERNum
Label	Rx Buffer Number
Description	Define the number of Rx buffer usable in Reception when multiple receive buffer feature or queued management feature is activated Maximum 4 buffers are allowed
Multiplicity	1..1
Type	INTEGER
Default value	2
Range	<=4 >=2
Origin	EB

Parameter Name	Enable_DownloadData_Streaming
Label	DownloadData Streaming
Description	Tick this option to support Streaming of the data received in the UDS Transfer-Data Request. This feature shall improve the download performance.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

4.5.1.2. SleepManagement

Parameters included	
Parameter name	Multiplicity



Parameters included

Grouping_NM_Message	1..1
PduConnectionGroupRef	1..1
IDGroupPDURef	1..1
RxPduld	1..1

Parameter Name	Grouping_NM_Message
Label	NM Message will belong to a Pdu connection group
Description	This entry allows to specify if the Pdu shall belong to a group and must be filtered if its group is not selected
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	PduConnectionGroupRef
Label	Pdu connection group reference
Description	Reference to the Pdu Connection where the NM message shall belong to
Multiplicity	1..1
Type	REFERENCE
Origin	AUTOSAR_ECUC

Parameter Name	IDGroupPDURef
Label	ID group PDU reference
Description	Reference to the PDU that will contain the information on which Connection Group shall be selected This PDU shall also be referenced in the CanIf module of the communication stack
Multiplicity	1..1
Type	REFERENCE
Origin	AUTOSAR_ECUC

Parameter Name	RxPduld
Description	This entry allows to configure the RxPduld that shall be used by BIPduR to transmit diagnostic response



	In case of NM Message this field is used to define the MsgIdx of the configured NM Message
Multiplicity	1..1
Type	INTEGER
Default value	0
Origin	EB

4.5.1.3. MultipleIdentifier

Parameters included	
Parameter name	Multiplicity
MultipleIdentifierGroup	1..1
MultipleIdentifierSelectTimeout	1..1
IDGroupPDURef	1..1
IDGroupPDUId	1..1
IDGroupByteNum	1..1
RxPduld	1..1

Parameter Name	MultipleIdentifierGroup
Label	Multiple Idenfier Group
Description	<p>Specify if the Multiple Identifier Group feature is enable</p> <p>This feature allows supporting several connection group and enable only one of them at initialization</p> <p>The connection can be selected from different source</p> <ul style="list-style-type: none"> ▶ OFF: The feature is disabled ▶ CAN_NOTIFICATION: A CAN frame reception will notify which Identifier Group shall be selected ▶ EXTERNAL_NOTIFICATION: A callback function will be called at startup to retrieve the Identifier Group
Multiplicity	1..1
Type	STRING
Default value	OFF
Range	OFF



	CAN_NOTIFICATION
	EXTERNAL_NOTIFICATION
Origin	EB

Parameter Name	MultipleIdentifierSelectTimeout
Label	Multiple Idenfier Timeout
Description	<p>Specify The timeout (in ms) to select a default connection group</p> <p>In case no connection has been chosen after this timeout the default connection group will be selected</p>
Multiplicity	1..1
Type	INTEGER
Default value	500
Origin	EB

Parameter Name	IDGroupPDURef
Label	ID group PDU reference
Description	<p>Reference to the PDU that will contain the information on which Connection Group shall be selected</p> <p>This PDU shall as well be reference in the CanIf module of the communication stack</p>
Multiplicity	1..1
Type	REFERENCE
Origin	AUTOSAR_ECUC

Parameter Name	IDGroupPDUId
Label	ID group PDU Id
Description	Defines the Pdu ID that will be used by PduR
Multiplicity	1..1
Type	INTEGER
Default value	0
Origin	EB

Parameter Name	IDGroupByteNum
Label	ID group Byte Number



Description	Defines the index of the byte that contain information regarding the group ID within the data received The values allowed are between 0 (LSB) and 7 (MSB) of the received data
Multiplicity	1..1
Type	INTEGER
Default value	0
Range	>=0 <=7
Origin	EB

Parameter Name	RxPduld
Description	This entry allows to configure the RxPduld that shall be used by BiPduR to transmit diagnostic response In case of Lin connection this field is used to define the MsgIdx of the configured LTP message
Multiplicity	1..1
Type	INTEGER
Default value	0
Origin	EB

4.5.1.4. PduConnection

Containers included		
Container name	Multiplicity	Description
RxPdu	0..n	Configuration of all the RxPdus references

Parameters included	
Parameter name	Multiplicity
TxPduRef	1..1
TxConfPduld	1..1
TesterAddress	1..1
LinConnection	1..1
ShareFunctionalId	1..1



Parameters included

SharedPduReference	1..1
------------------------------------	------

Parameter Name	TxPduRef
Label	TxPdu Reference
Description	<p>Reference to the <code>Pdu</code> in the <code>EcucPduCollection</code> configured for this Transmission Channel.</p> <p>Through this reference, <code>BlPduR</code> can resolve the <code>PduId</code> used for Transmission in the <code>PduR_BlPduRTransmit()</code> API and defined by the <code>PduR</code>.</p> <p>Through this reference, <code>PduRouter</code> can resolve the <code>PduId</code> used for Transmission by in the following APIs:</p> <ul style="list-style-type: none"> ▶ <code>BlPduR_CopyTxData()</code> ▶ <code>BlPduR_TpTxConfirmation()</code> <p>and defined by the <code>BlPduR</code> in <code>TxConfirmationPduId</code>.</p>
Multiplicity	1..1
Type	REFERENCE
Origin	AUTOSAR_ECUC

Parameter Name	TxConfPduld
Label	TxPdu Identifier
Description	<p>This entry allows to configure the <code>TxConfirmationPduld</code> that shall be used by <code>PduR</code> to transmit Tx confirmation</p> <p>In case of Lin connection this field is used to define the <code>MsgIdx</code> of the configured LTP message</p>
Multiplicity	1..1
Type	INTEGER
Default value	0
Origin	EB

Parameter Name	TesterAddress
Label	Tester Address
Description	This defines the Tester Address used in source/target address of the current connection
Multiplicity	1..1



Type	INTEGER
Default value	0
Origin	EB

Parameter Name	LinConnection
Label	Lin Connection
Description	Defines if the connection is to managed Lin Pdu If so the generation of this connection will be done differently
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	ShareFunctionalId
Label	Share Functional Id
Description	Defines if this connection shall reuse an existing functional Id from another connection
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	SharedPduReference
Label	Shared Pdu Reference
Description	Reference to the shared Pdu
Multiplicity	1..1
Type	REFERENCE
Origin	EB

4.5.1.5. RxPdu

Parameters included	
Parameter name	Multiplicity



Parameters included

Ref	1..1
Type	1..1
Id	1..1

Parameter Name	Ref
Label	RxPdu Reference
Description	<p>Reference to <code>Pdu</code> from the <code>EcucPduCollection</code> configured for this Reception Channel.</p> <p>Through this reference, the <code>PduRouter</code> can resolve the <code>PduId</code> used for Reception by <code>B1PduR</code> in the following APIs:</p> <ul style="list-style-type: none"> ▶ <code>B1PduR_StartOfReception()</code> ▶ <code>B1PduR_CopyRxData()</code> ▶ <code>B1PduR_TpRxIndication()</code>
Multiplicity	1..1
Type	REFERENCE
Origin	AUTOSAR_ECUC

Parameter Name	Type
Description	<p>This entry allows to specify which kind of RxPdu is used.</p> <p>Please select between :</p> <ul style="list-style-type: none"> ▶ PHYSICAL ▶ FUNCTIONAL
Multiplicity	1..1
Type	STRING
Default value	PHYSICAL
Range	PHYSICAL FUNCTIONAL
Origin	EB

Parameter Name	Id
Description	This entry allows to configure the RxPduld that shall be used by BIPduR to transmit diagnostic response



	In case of Lin connection this field is used to define the MsgIdx of the configured LTP message
Multiplicity	1..1
Type	INTEGER
Default value	0
Origin	EB

4.5.1.6. IDGroup

Containers included		
Container name	Multiplicity	Description
ConnectionReflist	0..n	

Parameters included	
Parameter name	Multiplicity
Default	1..1
ArchitectureId	1..1

Parameter Name	Default
Label	Default ID group
Description	Specify if this IDGroup is the default one Only one ID Group can be selected in the configuration
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	ArchitectureId
Label	Architecture Id
Description	This defines the Architecture Identification for this group This value will be used to find the associated ID to choose link to the received ID in the architecture frame
Multiplicity	1..1

Type	INTEGER
Default value	4
Range	>=0 <=255
Origin	EB

4.5.1.7. ConnectionReflist

Parameters included	
Parameter name	Multiplicity
ConnectionRef	1..1

Parameter Name	ConnectionRef
Label	Connection reference
Description	Reference to the <code>connection</code> that will be part of the used connection group
Multiplicity	1..1
Type	REFERENCE
Origin	AUTOSAR_ECUC

4.5.1.8. CommonPublishedInformation

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



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

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

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

Parameter Name	SwMajorVersion
Label	Software Major Version
Description	Major version number of the vendor specific implementation of the module.
Multiplicity	1..1



Type	INTEGER_LABEL
Default value	0
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	SwMinorVersion
Label	Software Minor Version
Description	Minor version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	23
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	SwPatchVersion
Label	Software Patch Version
Description	Patch level version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	ModuleId
Label	Numeric Module ID
Description	Module ID of this module from Module List
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH



Parameter Name	VendorId
Label	Vendor ID
Description	Vendor ID of the dedicated implementation of this module according to the AU-TOSAR vendor list
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	1
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

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

4.5.1.9. PublishedInformation

Parameters included	
Parameter name	Multiplicity
PbcfgMSupport	1..1

Parameter Name	PbcfgMSupport
Label	PbcfgM support
Description	Specifies whether or not the BIPduR can use the PbcfgM module for post-build support.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH



4.5.2. Application programming interface (API)

4.5.2.1. Functions

4.5.2.1.1. BIPduR_AllSlots

Purpose	Activate the FlexRay mode AllSlots.
Synopsis	<code>void BIPduR_AllSlots (void);</code>
Description	If FlexRay is used, this Api call FlexRay state manager to request him the AllSlots mode

4.5.2.1.2. BIPduR_CopyRxData

Purpose	This service is called by the receiving Tp module, through PduR, requesting a TP-buffer.	
Synopsis	<code>BufReq_ReturnType BIPduR_CopyRxData (PduIdType RxPduId , const PduInfoType * PduInfoPtr , PduLengthType * BufferSizePtr);</code>	
Parameters (in)	RxPduId	Identification of the received I-PDU.
	PduInfoPtr	Pointer to the buffer (SduDataPtr) and its length (SduLength) containing the data to be copied by PDU Router module in case of gateway or upper layer module in case of reception.
Parameters (out)	BufferSizePtr	Available receive buffer after data has been copied.
Return Value	Result of buffer request	
	BUFREQ_OK	Buffer request accomplished successfully.
	BUFREQ_E_BUSY	Temporarily no buffer available. It's up the requestor to retry request for a certain time.
	BUFREQ_E_NOT_OK	Buffer request not successful. Buffer cannot be accessed.



4.5.2.1.3. BIPduR_CopyTxData

Purpose	This service is called by the sending Tp module, through PduR, requesting a TP-buffer.	
Synopsis	<pre>BufReq_ReturnType B1PduR_CopyTxData (PduIdType TxPduId , PduInfoType * PduInfoPtr , RetryInfoType * RetryInfoPtr , Pdu- LengthType * AvailableDataPtr);</pre>	
Parameters (in)	TxPduId	Identification of the transmitted I-PDU.
	RetryInfoPtr	This parameter is used to retransmit data because problems during the last service call. If the I-PDU is transmitted from a local module (e.g. DCM) the PDU router module will just forward the parameter value without check. If the I-PDU is gatewayed from another bus, the PDU Router module will make the following interpretation: If the transmitted TP I-PDU does not support the retry feature a NULL_PTR is provided. It indicates that the copied transmit data can be removed from the buffer after it has been copied. If the retry feature is used by the Tx I-PDU, RetryInfoPtr must point to a valid RetryInfoType element. If TpDataState indicates TP_-CONF_PENDING, the previously copied data must remain in the TP buffer to be available for error recovery. TP_DATA-CONF indicates that all data that have been copied so far are confirmed and can be removed from the TP buffer. Data copied by this API call are excluded and will be confirmed later. TP_DATA_RETRY indicates that this API call shall copy already copied data in order to recover from an error. In this case AvailableDataPtr specifies the offset of the first byte to be copied by the API call.
	RTabLoTpConfigIdx	Index referring to the configuration of the LoTp-Module.
	PduInfoPtr	Provides destination buffer and the number of bytes to copy. In case of gateway



		the PDU Router module will copy, otherwise the source upper layer module will copy the data. If not enough transmit data is available, no data is copied. The transport protocol module will retry. A copy size of 0 can be used to indicate state changes in the retry parameter.
	AvailableDataPtr	Indicates the remaining number of bytes that are available in the PduR Tx buffer. AvailableDataPtr can be used by TP modules that support dynamic payload lengths (e.g. Iso FrTp) to determine the size of the following CFs.
Return Value		Result of buffer request
	BUFREQ_OK	Data has been copied to the transmit buffer completely as requested.
	BUFREQ_E_BUSY	Request could not be fulfilled, because the required amount of Tx data is not available. TP layer might retry later on. No data has been copied.
	BUFREQ_E_NOT_OK	Data has not been copied. Request failed.

4.5.2.1.4. BIPduR_Custom_Com_Deactivate

Purpose	Deactivate the Communication.
Synopsis	<code>void BIPduR_Custom_Com_Deactivate (void);</code>

4.5.2.1.5. BIPduR_Custom_Com_Init

Purpose	Initialization of modules of communication stack and initialization of the stored PDUID of the active connection.
Synopsis	<code>void BIPduR_Custom_Com_Init (void);</code>

4.5.2.1.6. BIPduR_GetGroupIdVal

Purpose	This service is at system initialization by BIPduR module to retrieve Group Id use in the ECU.
----------------	--



Synopsis	<code>u8 B1PduR_GetGroupIdVal (void);</code>
Return Value	value of the Group ID that shall be used in by the bootloader to run
Description	The Group ID shall be retrieve depending on the system architecture (e.g. from NVM, from a specific I/O,..) It shall then be returned.

4.5.2.1.7. BIPduR_GetNextBuffer

Purpose	This service is called by the Prog module, to get the next buffer information to treat.	
Synopsis	<code>void B1PduR_GetNextBuffer (PduLengthType * pullen , u8 ** paubData);</code>	
Parameters (out)	pullen	Length of the data store in the provided buffer
	paubData	Address of the buffer to treat (Null pointer if no more buffer)

4.5.2.1.8. BIPduR_GetRxPduld

Purpose	Get the Rx Pdu Identifier on which the response after reset shall be sent.	
Synopsis	<code>void B1PduR_GetRxPduId (u8 * pubRxPduId);</code>	
Parameters (out)	pubRxPduId	Rx Pdu Identifier pointer

4.5.2.1.9. BIPduR_GetTpParameter

Purpose	This API gets the value of a TP parameter (STmin or BS).	
Synopsis	<code>u16 B1PduR_GetTpParameter (tTpParameterId ubParameterId);</code>	
Parameters (in)	ubParameterId	parameter ID to get (BLPDUR_TP_STMIN or BLPDUR_TP_BS)
Return Value	u16 value of the requested TP parameter	

4.5.2.1.10. BIPduR_GroupIdFrameFilter

Purpose	This service is Called by Can module to filter receive frames.	
Synopsis	<code>boolean B1PduR_GroupIdFrameFilter (Can_HwHandleType Hrh , Can_IdType CanId , uint8 CanDlc , const uint8 * CanSduPtr);</code>	



Parameters (in)	Hrh	Hardward object number
	CanId	Can ID of the received frame
	CanDlc	Length of the received frame
	CanSduPtr	Pointer to the data of the received frame
Return Value		
Description	This service is used to know if the given CanId shall be accepted or rejected depending on the active group ID	

4.5.2.1.11. BIPduR_Init

Purpose	Initialize all layers.
Synopsis	<code>void BIPduR_Init (void);</code>
Description	This function call all the subInit function BIPduR_InitX It shall be called only once at ECU start-up.

4.5.2.1.12. BIPduR_Init1

Purpose	Initialize the communication stack.
Synopsis	<code>void BIPduR_Init1 (void);</code>
Description	This function initializes the communication stack (CAN or FR or Ethernet) by calling there *_Init function.

4.5.2.1.13. BIPduR_Init2

Purpose	Initialize all specific bootloader layers.
Synopsis	<code>void BIPduR_Init2 (void);</code>
Description	This function initializes the Bootloader specific layer (PROG, SA and ReProgMemM or Flash) by calling there *_Init function.

4.5.2.1.14. BIPduR_IsNetworkSynchronized

Purpose	Check if FlexRay network is synchronized.
Synopsis	<code>u8 BIPduR_IsNetworkSynchronized (u8 * frCycle);</code>
Return Value	synchronization status



	BLPDUR_TRUE	Network is synchronized
	BLPDUR_FALSE	Network is not synchronized
Description	If FlexRay is used, this Api check if the Flexray network is synchronized and return the status and the FlexRay cycle.	

4.5.2.1.15. BIPduR_IsTcpConnectionReestablished

Purpose	Calls SoAd API that indicates an incoming TCP connection on a server socket.	
Synopsis	<code>tBLPduRBoolean BIPduR_IsTcpConnectionReestablished (void);</code>	
Return Value	connection status	
	BLPDUR_TRUE	SoAd accepts the established connection
	BLPDUR_FALSE	SoAd refuses the established connection, Tcplp stack shall close the connection.
Description	This service is used to know if the given CanId shall be accepted or rejected depending on the active group ID	

4.5.2.1.16. BIPduR_LockBuffer

Purpose	This service is called by the Prog module, to lock the buffer receive for treatment.	
Synopsis	<code>void BIPduR_LockBuffer (u8 * pubIsLastBuffer);</code>	
Parameters (out)	pubIsLastBuffer	Information if there is still some buffer available to lock

4.5.2.1.17. BIPduR_Manage

Purpose	Periodical task of all layers.
Synopsis	<code>void BIPduR_Manage (void);</code>
Description	Modules periodic functions are called in this function (ex: COM_Manage) BLPDUR_MANAGE_PERIOD is configured in Tresos BIPduR plugin. The reception of segmented frames are handled in this function.

4.5.2.1.18. BIPduR_RetrieveConnectionInfo

Purpose	Retrieve the active connection information.
----------------	---



Synopsis	<code>void BlPduR_RetrieveConnectionInfo (void);</code>
-----------------	---

4.5.2.1.19. BlPduR_RxIndication

Purpose	This service is Called by CanIf through PduR to notify a Pdu reception.	
Synopsis	<code>void BlPduR_RxIndication (PduIdType RxPduId , PduInfoType * PduInfoPtr);</code>	
Parameters (in)	RxPduId	Pdu Number received
	PduInfoPtr	Pointer to the Pdu Information

4.5.2.1.20. BlPduR_SaveTesterAddress

Purpose	This service is called by the PROG module to save tester address to be used later in tester filtering.	
Synopsis	<code>void BlPduR_SaveTesterAddress (void);</code>	

4.5.2.1.21. BlPduR_SendMsgData

Purpose	This service is used to trigger transmission .	
Synopsis	<code>tBlPduRStatus BlPduR_SendMsgData (PduIdType PduId , PduLengthType ulLen , u8 * paubData);</code>	
Parameters (in)	PduId	Identification of the PDU
	ulLen	Message length
	paubData	Address of the buffer to treat
Return Value		

4.5.2.1.22. BlPduR_Send_TPFrame

Purpose	Send A TP frame using the current Tester Address.	
Synopsis	<code>void BlPduR_Send_TPFrame (PduLengthType ulLen , u8 * paubData);</code>	
Parameters (in)	ulLen	Message length
	paubUdsData	message data pointer
Description	Send A TP frame using the current Tester Address	



4.5.2.1.23. BIPduR_SetTesterAddress

Purpose	Store the tester address in global variable.
Synopsis	<code>void BIPduR_SetTesterAddress (u8 ubTesterAddress);</code>
Description	Provide to BIPduR the tester address that shall be accepted in reception.

4.5.2.1.24. BIPduR_SimulateRxRequest

Purpose	Simulate the reception of a frame.	
Synopsis	<code>void BIPduR_SimulateRxRequest (PduLengthType ulLen , u8 * paubUdsData , u8 ubWithResp);</code>	
Parameters (in)	ulLen	Message length
	paubUdsData	message data pointer
	ubWithResp	indicate if a response will be performed (TRUE/FALSE)
Description	Simulate the reception of a frame	

4.5.2.1.25. BIPduR_StartOfReception

Purpose	This service is called by the receiving Tp module, through PduR, requesting a TP-buffer.	
Synopsis	<code>BufReq_ReturnType BIPduR_StartOfReception (PduIdType RxPduId , PduLengthType TpSduLength , PduLengthType * BufferSizePtr);</code>	
Parameters (in)	RxPduId	Identification of the received I-PDU.
	TpSduLength	Total length of the PDU to be received.
Parameters (out)	BufferSizePtr Available receive buffer in the receiving module. This parameter will be used to compute the Block Size (BS) in the transport protocol module.	
Return Value	Result of buffer request	
	BUFREQ_OK	Connection has been accepted. BufferSizePtr indicates the available receive buffer.
	BUFREQ_E_BUSY	Currently no buffer of the requested size is available. BufferSizePtr remains unchanged. Connection has been rejected.



	BUFREQ_E_OVFL	No Buffer of the required length can be provided.
	BUFREQ_E_NOT_OK	Connection has been rejected. BufferSizePtr remains unchanged.

4.5.2.1.26. **BIPduR_StoreConnectionInfo**

Purpose	Store the active connection information.
Synopsis	<code>void BIPduR_StoreConnectionInfo (void);</code>

4.5.2.1.27. **BIPduR_StoreRxPduld**

Purpose	Store the Rx Pdu Identifier on which the request shall be responded after reset has been received.	
Synopsis	<code>void BIPduR_StoreRxPduld (u8 ubRxPduld);</code>	
Parameters (in)	pubRxPduld	Rx Pdu Identifier

4.5.2.1.28. **BIPduR_TpChangeParameter**

Purpose	This API changes the value of a TP parameter (STmin or BS).	
Synopsis	<code>tBIPduRStatus BIPduR_TpChangeParameter (tTpParameterId ubParameterId , u16 uwParameterValue);</code>	
Parameters (in)	ubParameterId	parameter ID to change (BLPDUR_TP_STMIN or BLPDUR_TP_BS)
	uwParameterValue	new value to set
Return Value	BIPduRStatus variable change status	
	BLPDUR_E_OK	Parameter is changed
	BLPDUR_E_NOT_OK	Parameter isn't changed

4.5.2.1.29. **BIPduR_TpRxIndication**

Purpose	This service is called by the Tp module, through PduR, after an I-PDU has been received successfully or when an error occurred. It is also used to confirm cancellation of an I-PDU.
----------------	--



Synopsis	<code>void B1PduR_TpRxIndication (PduIdType RxPduId , NotifResultType Result);</code>	
Parameters (in)	RxPduId	Identification of the received I-PDU.
	Result	Result of the reception.

4.5.2.1.30. B1PduR_TpTxConfirmation

Purpose	This service is called by the Tp module, through PduR, after the I-PDU has been transmitted on its network, the result will reveal if the transmission was successful or not.	
Synopsis	<code>void B1PduR_TpTxConfirmation (PduIdType TxPduId , NotifResultType Result);</code>	
Parameters (in)	TxPduId	Identification of the transmitted I-PDU.
	Result	Result of the transmission of the I-PDU.

4.5.2.1.31. B1PduR_UnlockBuffer

Purpose	This service is called by the Prog module, to unlock one or all buffer used.	
Synopsis	<code>void B1PduR_UnlockBuffer (u8 ubBufferType);</code>	
Parameters (in)	ubBufferType	Buffer to be unlock (one or all)

4.5.2.1.32. LIN_ComLossInd

Purpose	This API.	
Synopsis	<code>void LIN_ComLossInd (void);</code>	

4.5.2.1.33. LIN_StatusInd

Purpose	This API.	
Synopsis	<code>void LIN_StatusInd (tLinFrameIdx uFrameIdx , tLinFrameStatus eStatus);</code>	
Parameters (in)	uFrameIdx	transmitted or received frame
	eStatus	LIN message status



4.5.2.1.34. LIN_WakeUpInd

Purpose	This API.
Synopsis	<code>void LIN_WakeUpInd (void);</code>

4.5.2.1.35. LTP_RxInd

Purpose	This API.	
Synopsis	<code>void LTP_RxInd (u8 ebStatus);</code>	
Parameters (in)	ebStatus	LTP message status

4.5.2.1.36. LTP_TxConf

Purpose	This API gives transmission indication depending on the LTP module message status.	
Synopsis	<code>void LTP_TxConf (u8 ebStatus);</code>	
Parameters (in)	ebStatus	LTP message status

4.5.3. Integration notes

4.5.3.1. Exclusive areas

Exclusive areas information is not available for this module.

4.5.3.2. Production errors

Production errors information is not available for this module.

4.5.3.3. Memory mapping

General information about memory mapping is provided in the EB tresos AutoCore Generic documentation. Refer to the section [Memory mapping and compiler abstraction](#) in the [Integration notes](#) section for details.

Memory mapping information is not available for this module.



4.5.3.4. Integration requirements

WARNING



Integration requirements list is not exhaustive

The following list of integration requirements helps you to integrate your product. However, this list is not exhaustive. You also require information from the user guide, release notes, and EB tresos AutoCore known issues to successfully integrate your product.

Integration requirements are not listed for the BIPduR module.

4.6. Prog

4.6.1. Configuration parameters

Containers included		
Container name	Multiplicity	Description
CommonPublishedInformation	1..1	Label: Common Published Information Common container, aggregated by all modules. It contains published information about vendor and versions.
PublishedInformation	1..1	Label: EB Published Information Additional published parameters not covered by CommonPublishedInformation container.
General	1..1	This container contains the general proprieties of the node.
DownloadVerification	1..1	This container contains the configuration element for download verification
DownloadFlashRoutines	1..1	This container contains the configuration element for downloading Flash routines
SBLVerificationBlockTable	1..1	This container contains the location data of the verification block table of the secondary bootloader
SBLVerificationStructure	1..1	This container contains the location data of the verification structure of the secondary bootloader
Startup	1..1	This container contains the general proprieties of the node
CompleteAndCompatible-Block	1..1	



Containers included		
Segments	0..n	<p>This container contains the description of the Segments of available memory to reprogram and the kind of memory used.</p> <p>PLEASE NOTE THAT:</p> <ul style="list-style-type: none"> - ONLY ONE SEGMENT SHALL BE OF DEFINED FOR UPDATER PARTITION - UNIQUE SEGMENT FOR UPDATER PARTITION SHALL REFER TO BLOCK DEFINED WITH INDEX 0
Blocks	0..250	<p>This container contains the description of the blocks used to request the erasing.</p> <p>PLEASE NOTE THAT BLOCK CONTAINING BOOTLOADER SEGMENT SHALL HAVE INDEX 0.</p>
Memory	0..n	<p>This container contains the description of the memories used by the bootloader.</p>
GM	1..1	<p>This container contains the GM specific configuration</p>
NRC78_Transmission	1..1	<p>Label: NRC78_Transmission Contains configuration for the NRC78 transmission before SecurityAccess.</p>
Security	1..1	
OemInd	1..1	<p>This container contains the OemInd specific configuration</p>
VAG	1..1	<p>This container contains the Volkswagen specific configuration</p>
PSA	1..1	<p>This container contains the PSA specific configuration</p>
Decryption	1..1	<p>Label: Decryption This container contains the Decryption specific configuration.</p>

4.6.1.1. CommonPublishedInformation

Parameters included	
Parameter name	Multiplicity
ArMajorVersion	1..1
ArMinorVersion	1..1
ArPatchVersion	1..1



Parameters included

SwMajorVersion	1..1
SwMinorVersion	1..1
SwPatchVersion	1..1
ModuleId	1..1
VendorId	1..1
Release	1..1

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

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

Parameter Name	ArPatchVersion
Label	AUTOSAR Patch Version
Description	Patch level version number of AUTOSAR specification on which the appropriate implementation is based on.
Multiplicity	1..1
Type	INTEGER_LABEL



Default value	0
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	SwMajorVersion
Label	Software Major Version
Description	Major version number of the vendor specific implementation of the module.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	2
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	SwMinorVersion
Label	Software Minor Version
Description	Minor version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	42
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	SwPatchVersion
Label	Software Patch Version
Description	Patch level version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	ModuleId
-----------------------	-----------------



Label	Numeric Module ID
Description	Module ID of this module from Module List
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	VendorId
Label	Vendor ID
Description	Vendor ID of the dedicated implementation of this module according to the AUTOSAR vendor list
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	1
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

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

4.6.1.2. PublishedInformation

Parameters included	
Parameter name	Multiplicity
PbcfgMSupport	1..1
Parameter Name	PbcfgMSupport
Label	PbcfgM support



Description	Specifies whether or not the Prog can use the PbcfgM module for post-build support.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

4.6.1.3. General

Containers included		
Container name	Multiplicity	Description
ProgCalReferences	1..1	Label: ProgCalReferences Contains references to Cal configuration identifiers.

Parameters included	
Parameter name	Multiplicity
MANAGE_PERIOD	1..1
NO_SECURITYLEVEL_RESET_ON_SESSIONCHANGE	1..1
Enable_Compression	1..1
Compression_Algorithm	1..1
Decomp_Out_Buffer_size	1..1
Enable_Decompression_Slicing	1..1
Decomp_Slice_size	1..1
Data_Size_In_RD	1..1
Dsc_Prog_Response	1..1
Prog_RC_CrcOffset	1..1
Expected_Crc_Location	1..1
Network_Management	1..1
Auto_Control	1..1
Use_CSM_ASR430_DemoWrapper	1..1
Tunable_Parameters	1..1
Dual_Memory_Bank_Used	1..1



Parameters included

Transmit_Nrc78_Before_EraseCheck	1..1
Transmit_Nrc78_On_Erase	1..1
PreliminaryErasing	1..1
MaxBlockID	1..1
Number_Of_Sector_To_Erase_Before_Sending_NRC78	1..1
Erase_Check	1..1
Transmit_Response_Before_Reset	1..1
Check_Programming_PreConditions	1..1
ResetAfterS3TimeoutInProgrammingSession	1..1
ResetAfterS3TimeoutInExtendedSession	1..1
ResetAfterDsc01InDefaultSession	1..1
Sleep_Management_Type	1..1
Sleep_Management_NM_Message	1..1
Sleep_Timeout	1..1
Max_Bytes_in_TD	1..1
FAR_POINTER_Definition	1..1
ResumableReprog	1..1
BypassSACompareKey	1..1

Parameter Name	MANAGE_PERIOD
Description	Specifies the period of the PROG manage task in ms. This period must be multiple of EB periodical value in EB module configuration. Range:[1ms ; 25ms]
Multiplicity	1..1
Type	INTEGER
Default value	10
Range	>=1 <=25
Origin	EB

Parameter Name	NO_SECURITYLEVEL_RESET_ON_SESSIONCHANGE
Label	No Security Level reset on Session change



Description	<p>Enabling this parameter allows the Bootloader:</p> <ol style="list-style-type: none"> 1. To have the security level unchanged(no security level reset) across the transitions between Non-Default sessions. 2. No need to unlock the same security level in the Bootloader, if the unlocking of Security (for the same Security level) is already done from the Application just before starting/jumping to the Bootloader. <p>Note: To have the second functionality enabled, the SecurityLevel variable shall be mapped to the shared Non volatile memory of Application and Bootloader.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Enable_Compression
Description	Specify if the compression is enable.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Compression_Algorithm
Label	Compression algorithm Id
Description	Compression algorithm id that shall be supported from dataFormatIdentifier field of RequestDownload service
Multiplicity	1..1
Type	INTEGER
Default value	1
Range	<=15
Origin	EB

Parameter Name	Decomp_Out_Buffer_size
Description	Size of the Output decompression buffer.
Multiplicity	1..1
Type	INTEGER



Default value	4000
Range	>=1000
	<=50000
Origin	EB

Parameter Name	Enable_Decompression_Slicing
Description	Specify if the slicing decompression is enable.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Decomp_Slice_size
Description	Size of the decompression input buffer slice.
Multiplicity	1..1
Type	INTEGER
Default value	512
Range	>=1
	<=1000
Origin	EB

Parameter Name	Data_Size_In_RD
Description	This entry allows to specify if the data size passed in the RequestDownload service means the compressed or decompressed data size. Please select between : ▶ compressed ▶ decompressed
Multiplicity	1..1
Type	STRING
Default value	compressed
Range	compressed
	decompressed
Origin	EB



Parameter Name	Dsc_Prog_Response
Label	Diagnostic Reprogramming response
Description	<p>Specify if the DSC 02 response shall be sent by the bootloader if the application receive a reprogramming request</p> <ul style="list-style-type: none"> ▶ Case tick: The response will be sent by the bootloader ▶ Case untick: The response will not be sent by the bootloader
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Prog_RC_CrcOffset
Label	PROG RoutineControl CheckMemory CRC OFFSET
Description	the CRC Offset is applicable for RoutineControl CheckMemory and the configured CRC Offset will be added to the PROG_MSG_BYTE_INDEX. For offset "0" CRC will be added right after the RoutineControl CheckMemory(i.e 31 01 "CheckMemoryRID" "CRC").
Multiplicity	1..1
Type	INTEGER
Default value	0
Range	>=0 <=4294967295
Origin	EB

Parameter Name	Expected_Crc_Location
Label	CRC Location: Request/Application
Description	<p>Specify if the expected CRC should be get by calling a callback or if it is passed in the request.</p> <ul style="list-style-type: none"> ▶ Case Application: The CRC is get by calling PROG_CustomGetExpected-Crc callback ▶ Case Request: The CRC is passed in the CheckMemory routine request.
Multiplicity	1..1
Type	ENUMERATION
Default value	Request



Range	Request Application
Origin	EB

Parameter Name	Network_Management
Description	<p>Specify if the network management shall be supported or not.</p> <p>This feature shall only be activated for VCC or Ford Bootloader on FlexRay.</p> <ul style="list-style-type: none"> ▶ Case tick: Network management supported ▶ Case untick: Network management not supported
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Auto_Control
Description	<p>Specify if auto-control shall be done or not at the end of application download.</p> <ul style="list-style-type: none"> ▶ Case tick: Auto-Control shall be done ▶ Case untick: Auto-Control shall not be done
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Use_CSM_ASR430_DemoWrapper
Label	Use ASR 4.3 Demo-CSM Wrapper
Description	<p>To have the compatibility of BL modules with Cryto ASR 4.3.x the CSM wrapper in the demo is introduced.</p> <p>Tick this option if you wish to make use of CSM wrapper to integrate ASR version 4.3.x Crypto modules.</p> <p>Note: This parameter is deprecated in the current Prog module version and integration of the Csm ASR 4.3 wrapper is no longer needed as the product supports the integration of Csm stack ASR 4.3. For having the backward compatibility with the previous versions of the BL modules, this parameter is retained and shall be removed in the future releases.</p>



Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Tunable_Parameters
Description	<p>This option allows to set the Segment configuration of the layer (in PROG_Cfg.-c) in RAM.</p> <p>An API called PROG_ParametersInit is also generated to initialize the parameters from the ROM.</p> <p>This allows the customer to change dynamically the segment address values after the initialization.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Dual_Memory_Bank_Used
Label	Dual Memory Bank Used
Description	This option allows the usage of dual memory bank feature on the supported hardware. Please reffer to the user manual for details.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Transmit_Nrc78_Before_EraseCheck
Label	NRC78 transmission before software invalidation
Description	<p>Specify if an NRC78 response shall be systematically sent receiving the Erase request and before the routine processing (before software invalidation).</p> <ul style="list-style-type: none"> ▶ Case untick: NRC78 will not be sent immediatly after a valid Erase request.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB



Parameter Name	Transmit_Nrc78_On_Erase
Label	NRC78 transmission after software invalidation
Description	<p>Specify if an NRC78 response shall be sent before starting the memory erasing and after the software invalidation.</p> <ul style="list-style-type: none"> ▶ Case untick: NRC78 will not be sent before the start of the memory erasing.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	PreliminaryErasing
Label	Preliminary Erasing Enable
Description	Enable the Preliminary Erasing in EraseMemory request (Available only if erasing mode is by LogicalBlock)
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	MaxBlockID
Label	Max Block ID
Description	The identifier used for the Preliminary Erasing can be defined in 1 or 2 bytes (e.g. 0xFF or 0xFFFF).
Multiplicity	1..1
Type	ENUMERATION
Default value	FF
Range	FF FFFF
Origin	EB

Parameter Name	Number_Of_Sector_To_Erase_Before_Sending_NRC78
Label	Number of sector to erase before sending NRC78
Description	<p>Defines the number of sectors to erase before sending NRC78 to the tester</p> <ul style="list-style-type: none"> ▶ If this value is set to "0" then the feature erase by sector is deactivated and the whole erase is performed



	<ul style="list-style-type: none"> ▶ If this value is greater than "0" then the erase by sector is enabled. In this case the NRC78 is sent each time the number of erased sectors reaches this value. It allows specifying a minimum time between two NRC78
Multiplicity	1..1
Type	INTEGER
Default value	0
Range	<p>>=0</p> <p><=255</p>
Origin	EB

Parameter Name	Erase_Check
Description	<p>Specify if software shall check if memory is already erased before doing an erase.</p> <ul style="list-style-type: none"> ▶ Disabled: No check will be done and memory will be always erase on request. ▶ First Programming Check: Memory will not be erased if it has never been programmed. ▶ Memory Block Erased Check: If logical block is used for erasing, the memory block is not erased if the corresponding flag is set. One flag per memory block is used to know if the memory block is already erased.
Multiplicity	1..1
Type	ENUMERATION
Default value	Disabled
Range	<p>Disabled</p> <p>First Programming Check</p> <p>Memory Block Erased Check</p>
Origin	EB

Parameter Name	Transmit_Response_Before_Reset
Description	<p>Specify if response shall be sent before resetting software.</p> <p>Case untick: No response transmitted before performing the reset</p> <p>Case tick: Response is transmitted before performing the reset</p>
Multiplicity	1..1
Type	BOOLEAN



Default value	false
Origin	EB

Parameter Name	Check_Programming_PreConditions
Description	<p>Specify if the programming pre-conditions should be checked.</p> <p>If activated, Prog module will call callback PROG_CustCheck-ProgPrecond on reception of pre-conditions check routine and DiagnosticSessionControl(ProgrammingSession) request.</p> <p>Project specific condition check shall be implemented in this callback.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	ResetAfterS3TimeoutInProgrammingSession
Label	Trigger Reset after S3 Timeout in Programming Session
Description	<p>Specify if software reset is required after S3 timeout coming from the Programming session.</p> <ul style="list-style-type: none"> ▶ Case tick: Trigger reset after S3 timeout coming from the Programming session. ▶ Case untick: No reset triggered after S3 timeout coming from the Programming session.
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	ResetAfterS3TimeoutInExtendedSession
Label	Trigger Reset after S3 Timeout in Extended Session
Description	<p>Specify if software reset is required after S3 timeout coming from the Extended session.</p> <ul style="list-style-type: none"> ▶ Case tick: Trigger reset after S3 timeout coming from the Extended session. ▶ Case untick: No reset triggered after S3 timeout coming from the Extended session.
Multiplicity	1..1



Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	ResetAfterDsc01InDefaultSession
Label	Trigger Reset While Switching from default to Default Session
Description	<p>Specify if ECU reset shall be triggered receiving a DefaultSession request while the ECU is already in Default Session.</p> <ul style="list-style-type: none"> ▶ Case tick: Trigger reset from any session to default session. ▶ Case untick: No reset triggered if current session is already Default.
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	Sleep_Management_Type
Description	<p>Specify the sleep management type that shall be use</p> <ul style="list-style-type: none"> ▶ Off: No sleep management managed by the bootloader ▶ Timeout: The bootloader will go into sleep mode on after a timeout without bus communication and wake up with network communication. This is also valid in case of NM message reception (when enabled) ▶ I/O: The bootloader will go into sleep mode on activation of an external I/O or switch (not supported yet)
Multiplicity	1..1
Type	ENUMERATION
Default value	Timeout
Range	Off Timeout
Origin	EB

Parameter Name	Sleep_Management_NM_Message
Label	Sleep Management for NM Message
Description	Specify if Sleep Management feature shall be active or not active for NM Message.



	<ul style="list-style-type: none"> ▶ Case tick: Sleep Management feature shall be active for NM Message. ▶ Case untick: Sleep Management feature shall not be active for NM Message.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Sleep_Timeout
Description	<p>Specify the sleep timeout value in ms.</p> <p>This timeout is started when the ECU enter in default session</p> <p>After this timeout expired the ECU will go in sleep mode</p> <p>This feature is only be supported when Sleep_Management type is "Timeout".</p> <p>Default value: 15000 (15 seconds)</p>
Multiplicity	1..1
Type	INTEGER
Default value	15000
Origin	EB

Parameter Name	Max_Bytes_in_TD
Description	<p>Define the maxblock length for a TransferData If Osek stack is used, make sure this value was lower than buffer message value in TP module.</p> <p>This value shall be lower than "Rx Physical Buffer Size" value. Please check BIPduR .</p>
Multiplicity	1..1
Type	INTEGER
Default value	4095
Origin	EB

Parameter Name	FAR_POINTER_Definition
Description	<p>Define the syntax for far pointer.</p> <ul style="list-style-type: none"> ▶ Example1: __far will done Data = (* (volatile __far *)<POINTER>)



	► Example2: (empty) will done Data = (* (volatile *)<POINTER>)
Multiplicity	1..1
Type	STRING
Default value	
Origin	EB

Parameter Name	ResumableReprog
Label	Resumable reprogramming
Description	<p>Specify if the resumable reprogramming feature shall be used.</p> <p>If activated, Prog module will store information allowing to resume an interrupted reprogramming.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	BypassSACCompareKey
Label	Bypass the SA compare key
Description	<p>Specify if SA compare key (sub-function sendKey) is bypassed or not</p> <p>Enabled: The SA sub-function sendKey is bypassed</p> <p>Disabled: The SA sub-function sendKey is performed</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

4.6.1.4. ProgCalReferences

Parameters included	
Parameter name	Multiplicity
ProgCalDecompressConfigId	1..1
Parameter Name	ProgCalDecompressConfigId



Label	ProgCalDecompressConfigId	
Description	Reference a <i>Cal/Decompress</i> Dependencies: ► Reference shall be valid	
Multiplicity	1..1	
Type	CHOICE-REFERENCE	
Configuration class	PreCompile:	VariantPreCompile
Origin	Elektrobit Automotive GmbH	

4.6.1.5. DownloadVerification

Parameters included	
Parameter name	Multiplicity
FCA_Reprogramming	1..1
VerificationOnTheFly	1..1
Verification_Buffer_size	1..1
Checksum_Algo	1..1
SignatureVerificationOnFlashData	1..1
SignatureVerificationWithAddrLen	1..1
SignatureVerificationWithPhyAddr	1..1
AdditionalCRCComputation	1..1
CVN_Verification	1..1
MaxNumberOfRDPerBlock	1..1
Allow2MaxSuccessiveCheckMemoryRequests	1..1

Parameter Name	FCA_Reprogramming
Label	FCA reprogramming type
Description	Specify which reprogramming type shall be used. Standard : use CRC16 flashed with data at address given by CRC Address field. Authenticated : use a header containing a hash.
Multiplicity	1..1
Type	ENUMERATION



Default value	Authenticated
Range	Standard
	Authenticated
Origin	EB

Parameter Name	VerificationOnTheFly
Label	Verification on the fly
Description	Specify if the download verification shall be performed after data download or in parallel (on the fly) of the download. Feature availability is OEM dependent.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Verification_Buffer_size
Label	Verification Buffer size
Description	Buffer size used during the Verification calculation. Increase the value if you need to accelerate the calculation.
Multiplicity	1..1
Type	INTEGER
Default value	32
Origin	EB

Parameter Name	Checksum_Algo
Label	CRC algorithm: Signature / CRC32 Ethernet / CRC16
Description	Specify which checksum algorithm shall be used <ul style="list-style-type: none"> ▶ Case Signature: A cryptographic signature verification will be done (only possible if cryptographic libraries are used). An additional CRC32 computation can be done by enabling "Additional CRC computation" parameter ▶ Case CRC32 Ethernet: Polynomial 0x04C11DB7 / Init value 0xFFFFFFFF / ReflectIn TRUE / ReflectOut TRUE / XOR on Output 0xFFFFFFFF
Multiplicity	1..1
Type	ENUMERATION



Default value	CRC32 Ethernet
Range	Signature
	CRC32 Ethernet
	CRC16
Origin	EB

Parameter Name	SignatureVerificationOnFlashData
Label	Signature Verification on Flashed data
Description	If enabled: the signature will be verified by reading the Flash memory after the data have been written. If compression is used signature is so verified on de-compressed data. If disabled: the signature will be verified on received data before writing to Flash memory. If compression is used signature is so verified on compressed data.
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	SignatureVerificationWithAddrLen
Label	Signature Verification with segment address/length
Description	If enabled: the signature will be computed including the address and length of the segment. If disabled: the signature will be computed including only the programmed data.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	SignatureVerificationWithPhyAddr
Label	Signature Verification with physical address
Description	If enabled: When address shall be included in signature computation, the physical address will be used (i.e in case of external Flash, the configured offset will apply). If disabled: When address shall be included in signature computation, the logical address (received address) will be used.



Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	AdditionalCRCComputation
Label	Additional CRC Computation
Description	In case signature computation is performed, if enabled an additional CRC32 computation will be performed after signature verification on the reception of the CheckMemory routine. Feature availability is OEM dependent.
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	CVN_Verification
Label	CVN Verification
Description	If enabled a calibration verification number will be performed on the reception of the verify partial software checksum routine.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	MaxNumberOfRDPerBlock
Label	Maximum RequestDownload Per Block
Description	Define the maximum number of RequestDownload that Bootloader shall support for a single logical block. This shall not exceed 255 It will impact the RAM memory consumption to store the downloaded memory area that shall be used for checksum computation.
Multiplicity	1..1
Type	INTEGER
Default value	10
Origin	EB



Parameter Name	Allow2MaxSuccessiveCheckMemoryRequests
Label	Allow 2 maximum successive Check Memory Requests
Description	If enabled: Only a second successive Check Memory Requests can be allowed by the Bootloader if the result of the first one was not sucessful. If disabled: No successive Check Memory Requests will be allowed by the Bootloader.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

4.6.1.6. DownloadFlashRoutines

Parameters included	
Parameter name	Multiplicity
Download_Flash_Routines	1..1
Decompress_Flash_Routines	1..1
Reject_RD_After_Corrupt_Flash_Routines	1..1

Parameter Name	Download_Flash_Routines
Label	Download FFlash driver
Description	Specify if the flash routines will be downloaded to RAM via tester tool. If activated, a RAM segment shall be configured.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Decompress_Flash_Routines
Label	Decompress Flash driver
Description	Specify if the driver flash routines are compressed in the bootloader and need to be decompressed to RAM on security access unlock.
Multiplicity	1..1



Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Reject_RD_After_Corrupt_Flash_Routines
Label	Reject second attempt after failure
Description	if activated, after a failed check memory, a second attempt for downloading flash routines will be rejected.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

4.6.1.7. SBLVerificationBlockTable

Parameters included	
Parameter name	Multiplicity
Verification_Block_Table_Start_Address	1..1
Verification_Block_Table_Length	1..1

Parameter Name	Verification_Block_Table_Start_Address
Description	Start address of the verification block table of the secondary bootloader
Multiplicity	1..1
Type	INTEGER
Default value	0x7000DC00
Origin	EB

Parameter Name	Verification_Block_Table_Length
Description	Length of the verification block table of the secondary bootloader before any processing (i.e. padding)
Multiplicity	1..1
Type	INTEGER
Default value	0x0000002C
Origin	EB



4.6.1.8. SBLVerificationStructure

Parameters included	
Parameter name	Multiplicity
Verification_Structure_Start_Address	1..1
Verification_Structure_Length	1..1

Parameter Name	Verification_Structure_Start_Address
Description	Start address of the verification structure of the secondary bootloader
Multiplicity	1..1
Type	INTEGER
Default value	0x7000DC00
Origin	EB

Parameter Name	Verification_Structure_Length
Description	Length of the verification structure of the secondary bootloader before any processing (i.e. padding)
Multiplicity	1..1
Type	INTEGER
Default value	0x0000002C
Origin	EB

4.6.1.9. Startup

Parameters included	
Parameter name	Multiplicity
PROG_Signature_High	1..1
PROG_Signature_Low	1..1
PROG_Signature_Clear	1..1

Parameter Name	PROG_Signature_High
Description	MSB value set by the application to know if the reset is done from application. the default value is recommended by VCC
Multiplicity	1..1
Type	INTEGER



Default value	0x50726f67
Origin	EB

Parameter Name	PROG_Signature_Low
Description	LSB value set by the application to know if the reset is done from application. the default value is recommended by VCC
Multiplicity	1..1
Type	INTEGER
Default value	0x5369676e
Origin	EB

Parameter Name	PROG_Signature_Clear
Description	LSB value set by the application to know if the reset is done from application. the default value is recommended by VCC
Multiplicity	1..1
Type	INTEGER
Default value	0x00000000
Origin	EB

4.6.1.10. CompleteAndCompatibleBlock

Parameters included	
Parameter name	Multiplicity
Block_Start_Addr	1..1
Start_Complete_Compatible_Signature_High	1..1
Start_Complete_Compatible_Signature_Low	1..1
End_Complete_Compatible_Signature_High	1..1
End_Complete_Compatible_Signature_Low	1..1
CompleteCompatibleFunction_Timeout	1..1

Parameter Name	Block_Start_Addr
Description	First address of the Complete and compatible block.
Multiplicity	1..1
Type	INTEGER



Origin	EB
---------------	----

Parameter Name	Start_Complete_Compatible_Signature_High
Description	MSB value set by the application at the start of the complete and compatible block the default value is recommended by VCC
Multiplicity	1..1
Type	INTEGER
Default value	0x53746172
Origin	EB

Parameter Name	Start_Complete_Compatible_Signature_Low
Description	LSB value set by the application at the start of the complete and compatible block the default value is recommended by VCC
Multiplicity	1..1
Type	INTEGER
Default value	0x74536967
Origin	EB

Parameter Name	End_Complete_Compatible_Signature_High
Description	MSB Value set by the application at the start of the complete and compatible block the default value is recommended by VCC
Multiplicity	1..1
Type	INTEGER
Default value	0x456e6453
Origin	EB

Parameter Name	End_Complete_Compatible_Signature_Low
Description	LSB Value set by the application at the start of the complete and compatible block the default value is recommended by VCC
Multiplicity	1..1
Type	INTEGER



Default value	0x69676e61
Origin	EB

Parameter Name	CompleteCompatibleFunction_Timeout
Description	Timeout value of the completecompatibleFunction 15ms by default
Multiplicity	1..1
Type	INTEGER
Default value	15
Origin	EB

4.6.1.11. Segments

Containers included		
Container name	Multiplicity	Description
SecureBoot	1..1	This container contains all configurations for Secure/Authen-ticated bootloader features for the Implementation 30 Boot-loader, and for the other bootloader variants please check under (Blocks)-tab. Configuration can be done only if Secure Boot feature or Authenticated Boot feature is activated under BM.

Parameters included	
Parameter name	Multiplicity
Memory	1..1
Access_Type	1..1
Reprog_Start_Address	1..1
Reprog_End_Address	1..1
Erase_Start_Address	1..1
Erase_End_Address	1..1
Partition_Type	1..1
Protected_Partition_ID	1..1
HSM_PartitionID	1..1
HSM_RAM_Buffer	1..1



Parameters included

PROG_HSM_Timeout	1..1
ValidityCheck	1..1
SignatureVerification	1..1

Parameter Name	Memory
Description	Reference to the memory which contains the segment. <ul style="list-style-type: none"> ▶ FLASH ▶ FLASH_EXT ▶ EEPROM ▶ RAM ▶ SCRATCHPAD
Multiplicity	1..1
Type	REFERENCE
Origin	EB

Parameter Name	Access_Type
Description	Define Authorized memory access types to this segment <ul style="list-style-type: none"> ▶ READ: Only allow Read memory access ▶ WRITE: Only allow Write and Erase memory access ▶ READ_WRITE: Allow Read, Write and Erase memory access
Multiplicity	1..1
Type	ENUMERATION
Default value	READ_WRITE
Range	READ WRITE READ_WRITE
Origin	EB

Parameter Name	Reprog_Start_Address
Description	Start address of the segment in the memory. Range: [0x00000000 ; 0xFFFFFFFF] Coded on 32 bits



Multiplicity	1..1
Type	INTEGER
Origin	EB

Parameter Name	Reprog_End_Address
Description	<p>End address of the segment in the memory.</p> <p>Range: [0x00000000 ; 0xFFFFFFFF]</p> <p>Coded on 32 bits</p>
Multiplicity	1..1
Type	INTEGER
Origin	EB

Parameter Name	Erase_Start_Address
Description	<p>Erasing start address of the segment in the memory.</p> <p>Range: [0x00000000 ; 0xFFFFFFFF]</p> <p>Coded on 32 bits</p>
Multiplicity	1..1
Type	INTEGER
Origin	EB

Parameter Name	Erase_End_Address
Description	<p>Erasing End address of the segment in the memory.</p> <p>Range: [0x00000000 ; 0xFFFFFFFF]</p> <p>Coded on 32 bits</p>
Multiplicity	1..1
Type	INTEGER
Origin	EB

Parameter Name	Partition_Type
Label	Partition Type
Description	<p>Define the partition type</p> <ul style="list-style-type: none"> ▶ Application partition ▶ Calibration partition



	<ul style="list-style-type: none"> ▶ BLUpdater/Application partition ▶ BLUpdater/Calibration partition ▶ BLUpdater partition ▶ HSM partition ▶ Protected calibration partition - will not be erased ▶ Bootloader partition ▶ Flash Routines partition (here will be the erase/write routines of the flash driver, this partition shall be configured to RAM) ▶ Software Structure partition
Multiplicity	1..1
Type	STRING
Default value	PROG_APPLICATION_PARTITION
Range	PROG_FLASH_ROUTINES_PARTITION PROG_APPLICATION_PARTITION PROG_CALIBRATION_PARTITION PROG_BLU_APP_PARTITION PROG_BLU_CAL_PARTITION PROG_BLU_PARTITION PROG_PROT_CALIBRATION_PARTITION PROG_ESS_PARTITION PROG_HSM_PARTITION
Origin	EB

Parameter Name	Protected_Partition_ID
Label	Protected Partition ID
Description	Define the partition ID for protected segment
Multiplicity	1..1
Type	INTEGER
Range	>=0x2
Origin	EB

Parameter Name	HSM_PartitionID
Label	HSM Partition ID



Description	Define the BootLoader HSM Partition ID The Partition ID needs to be defined
Multiplicity	1..1
Type	INTEGER
Default value	8
Range	>=0x2
Origin	EB

Parameter Name	HSM_RAM_Buffer
Label	HSM Buffer Size
Description	Define the BootLoader HSM Partition RAM buffer size
Multiplicity	1..1
Type	INTEGER
Default value	4000
Origin	EB

Parameter Name	PROG_HSM_Timeout
Label	HSM TD Response Timeout
Description	Define the BootLoader HSM TD response Timeout in ms
Multiplicity	1..1
Type	INTEGER
Default value	10
Origin	EB

Parameter Name	ValidityCheck
Label	Validity Check
Description	Check or not the status of this segment for the application validity check
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	SignatureVerification
Label	Verify Signature



Description	Enable this to perform Signature verification for this segment.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

4.6.1.12. SecureBoot

Parameters included	
Parameter name	Multiplicity
Verified_For_Secure_Boot	1..1
Blocker_for_Software_execution	1..1
Start_Address_Secure_Boot_Verification	1..1
Length_Secure_Boot_Verification	1..1

Parameter Name	Verified_For_Secure_Boot
Label	Verified in Secure Boot
Description	Specify if the Block shall be verified when Secure Boot is activated.
Multiplicity	1..1
Type	ENUMERATION
Default value	Block won't be verified
Range	Block will be verified Block won't be verified
Origin	EB

Parameter Name	Blocker_for_Software_execution
Label	Blocker for Software execution
Description	Specify if the Block will prevent the corresponding software to be executed in case of Secure Boot verification failure.
Multiplicity	1..1
Type	ENUMERATION
Default value	Won't block software execution
Range	Will block software execution



	Won't block software execution
Origin	EB

Parameter Name	Start_Address_Secure_Boot_Verification
Label	Start Address for the Secure Boot Verification
Description	Specify the Start Address of the area on which the Secure Boot verification will be done.
Multiplicity	1..1
Type	INTEGER
Default value	0
Range	<=4294967295
Origin	EB

Parameter Name	Length_Secure_Boot_Verification
Label	Length of the Block area for the Secure Boot Verification
Description	Specify the length of the area on which the Secure Boot verification will be done.
Multiplicity	1..1
Type	INTEGER
Default value	0
Range	<=4294967295
Origin	EB

4.6.1.13. Blocks

Containers included		
Container name	Multiplicity	Description
SecureBoot	1..1	This container contains all configurations for Secure/Authenticated bootloader features, for all the supported Bootloaders other than Implementation 30, and for the Implementation 30 bootloader please check under (Segments)-tab. Configuration can be done only if Secure Boot feature or Authenticated Boot feature is activated under BM.

Parameters included	
Parameter name	Multiplicity



Parameters included

First_Segment	1..1
Segment_Number	1..1
Block_Programming_Counter_Max	1..1
Block_Identifier	1..1
Crc_Address	1..1
Downgrade_Protected	1..1

Parameter Name	First_Segment
Description	Reference to the first segment of the block
Multiplicity	1..1
Type	REFERENCE
Origin	EB

Parameter Name	Segment_Number
Description	<p>Number of Segment in the block.</p> <p>These represent the number of segment in the block based on the first configured segment</p> <p>(segments of the same block shall be consecutive in segments configuration)</p>
Multiplicity	1..1
Type	INTEGER
Origin	EB

Parameter Name	Block_Programming_Counter_Max
Description	<p>Maximum number of allowed to program the block.</p> <p>This represent the maximum number of times allowed to program the block</p> <p>(If the counter is set to 0 then there will be no limit for programming)</p>
Multiplicity	1..1
Type	INTEGER
Origin	EB

Parameter Name	Block_Identifier
Description	Identifier of the block.



Multiplicity	1..1
Type	INTEGER
Origin	EB

Parameter Name	Crc_Address
Label	CRC Address
Description	The address of the CRC in the memory block.
Multiplicity	1..1
Type	INTEGER
Origin	EB

Parameter Name	Downgrade_Protected
Label	Downgrade Protected
Description	<p>Specify if the logical block needs to be downgrade protected.</p> <ul style="list-style-type: none"> ▶ Case tick: Downgrade protection applies on the corresponding block ▶ Case untick: Downgrade protection does not apply on the corresponding block
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

4.6.1.14. SecureBoot

Parameters included	
Parameter name	Multiplicity
<u>Verified_For_Secure_Boot</u>	1..1
<u>Blocker_for_Software_execution</u>	1..1
<u>Start_Address_Secure_Boot_Verification</u>	1..1
<u>Length_Secure_Boot_Verification</u>	1..1

Parameter Name	Verified_For_Secure_Boot
Label	Verified in Secure Boot



Description	Specify if the Block shall be verified when Secure Boot is activated.
Multiplicity	1..1
Type	ENUMERATION
Default value	Block won't be verified
Range	Block will be verified
	Block won't be verified
Origin	EB

Parameter Name	Blocker_for_Software_execution
Label	Blocker for Software execution
Description	Specify if the Block will prevent the corresponding software to be executed in case of Secure Boot verification failure.
Multiplicity	1..1
Type	ENUMERATION
Default value	Won't block software execution
Range	Will block software execution
	Won't block software execution
Origin	EB

Parameter Name	Start_Address_Secure_Boot_Verification
Label	Start Address for the Secure Boot Verification
Description	Specify the Start Address of the area on which the Secure Boot verification will be done.
Multiplicity	1..1
Type	INTEGER
Default value	0
Range	<=4294967295
Origin	EB

Parameter Name	Length_Secure_Boot_Verification
Label	Length of the Block area for the Secure Boot Verification
Description	Specify the length of the area on which the Secure Boot verification will be done.
Multiplicity	1..1
Type	INTEGER

Default value	0
Range	<=4294967295
Origin	EB

4.6.1.15. Memory

Parameters included	
Parameter name	Multiplicity
<u>Memory_Type</u>	1..1
<u>Memory_Mode</u>	1..1
<u>Min_Value_To_Write</u>	1..1
<u>Addr_Offset</u>	1..1
<u>Erase_Value</u>	1..1

Parameter Name	Memory_Type
Label	Memory Type
Description	<p>Type of the memory.</p> <ul style="list-style-type: none"> ▶ FLASH ▶ FLASH_EXT ▶ RAM ▶ CUSTOM <p>NOTE: Only one memory of each type can be defined!</p>
Multiplicity	1..1
Type	ENUMERATION
Range	<p>FLASH</p> <hr/> <p>FLASH_EXT</p> <hr/> <p>RAM</p> <hr/> <p>CUSTOM</p>
Origin	EB

Parameter Name	Memory_Mode
Label	Memory Mode



Description	This entry allows to specify if the memory is access synchronously or asynchronously. Please select between : ▶ asynchronous ▶ synchronous
Multiplicity	1..1
Type	STRING
Default value	synchronous
Range	asynchronous synchronous
Origin	EB

Parameter Name	Min_Value_To_Write
Label	Minimum value to write
Description	Define the minimum size the memory driver could write at a time. Range: [0x00000000 ; 0xFFFFFFFF]. Coded on 32 bits.
Multiplicity	1..1
Type	INTEGER
Default value	8
Range	>=8
Origin	EB

Parameter Name	Addr_Offset
Label	Address Offset
Description	Define the start address offset for the memory. In the case of the EXTERNAL FLASH and CUSTOM this offset is SUBSTRACTED from the received address. In the case of the RAM and INTERNAL FLASH this offset is ADDED to the received address. Range: [0x00000000 ; 0xFFFFFFFF]. Coded on 32 bits.



Multiplicity	1..1
Type	INTEGER
Default value	0
Origin	EB

Parameter Name	Erase_Value
Label	Erase Value
Description	<p>Define the value to set for each byte when the memory is erased</p> <p>Example: 0xFFu will affect the value 0xFF to every byte of the memory when erasure.</p> <p>Range: [0x00 ; 0xFF].</p> <p>Coded on 8 bits.</p>
Multiplicity	1..1
Type	INTEGER
Default value	255
Range	<p>>=0</p> <p><=255</p>
Origin	EB

4.6.1.16. GM

Containers included		
Container name	Multiplicity	Description
ProgCsmReferences	1..1	<p>Label: ProgCsmReferences</p> <p>Contains references to Csm configuration identifiers.</p>

Parameters included	
Parameter name	Multiplicity
PEC_Enable	1..1
MAX_PARTITION	1..1
MAX_REGION_ALLOWED	1..1
Security_Access_Seed_Length	1..1
ECU_ADDR	1..1



Parameters included

PSI_Programmed	1..1
PSI_Revoked	1..1
BCID	1..1
Eculd_Source	1..1
ECU_ID	1..1
Subject_Name	1..1
ECU_Name	1..1
BOOT_Part_Number	1..1
BOOT_DLS	1..1

Parameter Name **PEC_Enable**

Label	PEC Enable
Description	Define if the PEC is enable
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name **MAX_PARTITION**

Label	Maximum Partition
Description	Define the maximum partition supported by the ECU
Multiplicity	1..1
Type	INTEGER
Default value	2
Origin	EB

Parameter Name **MAX_REGION_ALLOWED**

Label	Maximum Number of regions
Description	Define the maximum number of region that is allowed in the software
Multiplicity	1..1
Type	INTEGER
Default value	2
Range	>=2



	<255
Origin	EB

Parameter Name	Security_Access_Seed_Length
Label	Length of the security seed
Description	Define the length of the seed to be able to return the correct size
Multiplicity	1..1
Type	INTEGER
Default value	31
Origin	EB

Parameter Name	ECU_ADDR
Label	Ecu Address
Description	Define the ECU address that will identify the ECU to then reprogramming tool
Multiplicity	1..1
Type	INTEGER
Default value	0xEB
Range	>=0 <255
Origin	EB

Parameter Name	PSI_Programmed
Label	Programmed PSI Value
Description	Define PSI value when it is programmed The value shall be enter as hexadecimal value without the 0x
Multiplicity	1..1
Type	STRING
Default value	A555A5AA5AAA5A55
Origin	EB

Parameter Name	PSI_Revoked
Label	Revoked PSI Value
Description	Define PSI value when it is revoked The value shall be enter as hexadecimal value without the 0x



Multiplicity	1..1
Type	STRING
Default value	505245564F4B4544
Origin	EB

Parameter Name	BCID
Description	This value shall contains the Bootloader Compatibility Identifier (BCID) This value is two bytes long
Multiplicity	1..1
Type	INTEGER
Default value	60395
Range	<=65535
Origin	EB

Parameter Name	Eculd_Source
Label	ECU ID Source
Description	This entry allows to specify if the ECU ID is coming from Tresos Configuration or from User Callback. If the User Callback option is used, the integrator shall implement the callback that will provide the ECU Id. Please select between : ▶ Tresos Configuration ▶ User Callback
Multiplicity	1..1
Type	ENUMERATION
Default value	Tresos_Configuration
Range	Tresos_Configuration User_Callback
Origin	EB

Parameter Name	ECU_ID
Label	ECU ID
Description	Define the ECU ID The value shall be enter as hexadecimal value without the 0x



	the parameter shall be 16 bytes long
Multiplicity	1..1
Type	STRING
Default value	00000000000000000000000000000000
Origin	EB

Parameter Name	Subject_Name
Label	ECU Subject Name
Description	<p>Define the ECU subject name</p> <p>The value shall be enter as hexadecimal value without the 0x</p> <p>the parameter shall be 16 bytes long</p>
Multiplicity	1..1
Type	STRING
Default value	00112233445566778899001122334455
Origin	EB

Parameter Name	ECU_Name
Label	ECU Name
Description	<p>Define the ECU name</p> <p>The name shall be set as a string with exactly 8 characters (space can be used)</p>
Multiplicity	1..1
Type	STRING
Default value	GM BOOT
Origin	EB

Parameter Name	BOOT_Part_Number
Label	BootLoader Part Number
Description	<p>Define the BootLoader Part Number</p> <p>The value shall be entered as hexadecimal value without the 0x</p> <p>the parameter shall be 4 bytes long</p>
Multiplicity	1..1
Type	STRING
Default value	00112233



Origin	EB
---------------	----

Parameter Name	BOOT_DLS
Label	BootLoader DLS
Description	<p>Define the BootLoader DLS</p> <p>The name shall be set as a string with exactly 2 characters (space cannot be used)</p>
Multiplicity	1..1
Type	STRING
Default value	AB
Origin	EB

4.6.1.17. ProgCsmReferences

Containers included		
Container name	Multiplicity	Description
SignatureVerification	1..1	
HashVerification	1..1	

4.6.1.18. SignatureVerification

Parameters included	
Parameter name	Multiplicity
ProgCsmSignatureVerifyConfigId	1..1
BS_size	1..1
Allow2Cancel_OngoingJobs	1..1
Allow2SetCryptoKey	1..1
Allow2CustomCsmStartPreprocess	1..1
AllowStreamStart	1..1
SignFinSendFRP	1..1

Parameter Name	ProgCsmSignatureVerifyConfigId
Label	ProgCsmSignatureVerifyConfigId



Description	Reference a <i>CsmSignatureVerify</i> Dependencies: ▶ Reference shall be valid
Multiplicity	1..1
Type	CHOICE-REFERENCE
Configuration class	VariantPreCompile: VariantPreCompile
Origin	Elektrobit Automotive GmbH

Parameter Name	BS_size
Label	Data size for Block Slicing
Description	<p>This feature shall allow the Integrator to decide the size of the data to be processed in a single Asynchronous Crypto main function call.</p> <p>it allows to reduce the time blocked by the CryptoMainfunction in a single main function call.</p> <p>This feature can be enabled by entering any value other than 0(in bytes) in this field. Setting it to 0 shall allow the Bootlaoder to pass the entire available data for processing in a single CryptoMainfunction call.</p> <p>Note:</p> <ol style="list-style-type: none"> 1. This field is only enabled for the integration of ASR Crypto stack version 4.3 or higher. 2. Enabling this feature (by setting a very small size) shall increase the time for verification significantly, integrator shall decide the value to be set. 3. It is recommended to set this value to a maximum value that is sufficient to not to cause timeouts in the system. 4. If this feature is enabled together with verification on fly, the minimum value of Verification_Buffer_size and BS_size shall be effectively selected as the processing data size.
Multiplicity	1..1
Type	INTEGER
Default value	0
Origin	EB

Parameter Name	Allow2Cancel_OngoingJobs
-----------------------	---------------------------------



Label	Allow cancellation of any ongoing Crypto job
Description	<p>If enabled, any ongoing or active Crypto job operations shall be cancelled at the start of this job operation.</p> <p>This option is applicable only for the integration of Crypto ASR 4.3 or higher versions, for older version this feature is realized using 'Enable the cancelation of ongoing requests' in the Csm/Cry modules.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	Allow2SetCryptoKey
Label	Allow to fetch the stored Key
Description	<p>This will allow the integrator to pass the desired key and its elements at the start of the Crypto operation.</p> <p>This option is applicable only for the integration of Crypto ASR 4.3 or higher versions.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	Allow2CustomCsmStartPreprocess
Label	Custom CSM start pre-processing
Description	<p>This will allow the integrator to perform the encoding (eg. Base64, DER etc.) on the Key stored at the host side at the start of the Crypto operation</p> <p>This option is applicable only for the integration of Crypto ASR 4.3 or higher versions.</p> <p>Note: The key encoding shall be performed only if the Crypto driver expects the key to be encoded in an expected format.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB



Parameter Name	AllowStreamStart
Label	Enable StreamStart mode
Description	<p>This parameter shall be enabled if the Signature Start operation takes more than one cycle of CryptoMainFunction execution.</p> <p>Note:</p> <ol style="list-style-type: none"> 1. This option is applicable only for the integration of Crypto ASR 4.3 or higher versions. 2. This shall be enabled only if the crypto engine supports the StreamStart feature.
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	SignFinSendFRP
Label	Send NRC78 for Signature Finish
Description	<p>This parameter if enabled shall provide a force response Pending (NRC78) at the beginning of the Signature Finish operation.</p> <p>This parameter is introduced to avoid causing P2 timeouts at client side when the Crypto driver main function call for the Crypto Finish operation blocks the other Boot functions longer than the modules manage period.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

4.6.1.19. HashVerification

Parameters included	
Parameter name	Multiplicity
ProgCsmHashConfigId	1..1
BS_size	1..1
Allow2Cancel_OngoingJobs	1..1



Parameters included

HashFinSendFRP	1..1
--------------------------------	------

Parameter Name	ProgCsmHashConfigId	
Label	ProgCsmHashConfigId	
Description	Reference a <i>CsmHash</i> Dependencies: ▶ Reference shall be valid	
Multiplicity	1..1	
Type	CHOICE-REFERENCE	
Configuration class	VariantPreCompile:	VariantPreCompile
Origin	Elektrobit Automotive GmbH	

Parameter Name	BS_size
Label	Data size for Block Slicing
Description	This feature shall allow the Integrator to decide the size of the data to be processed in a single Asynchronous Crypto main function call. It allows to reduce the time blocked by the CryptoMainfunction in a single main function call. This feature can be enabled by entering any value other than 0(in bytes) in this field. Setting it to 0 shall allow the Bootlaoder to pass the entire available data for processing in a single CryptoMainfunction call. Note: 1. This field is only enabled for the integration of ASR Crypto stack version 4.3 or higher. 2. Enabling this feature (by setting a very small size) shall increase the time for verification significantly. The integrator shall decide the value to be set. 3. It is recommended to set this value to a maximum value that is sufficient to not to cause timeouts in the system. 4. If this feature is enabled together with verification on fly, the minimum value of Verification_Buffer_size and BS_size shall be effectively selected as the processing data size.
Multiplicity	1..1



Type	INTEGER
Default value	0
Origin	EB

Parameter Name	Allow2Cancel_OngoingJobs
Label	Allow cancellation of any ongoing Crypto job
Description	<p>If enabled, any ongoing or active Crypto job operations shall be cancelled at the start of this job operation.</p> <p>This option is applicable only for the integration of Crypto ASR 4.3 or higher versions, for older version this feature is realized using 'Enable the cancelation of ongoing requests' in the Csm/Cry modules.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	HashFinSendFRP
Label	Send NRC78 for Hash Finish
Description	<p>This parameter if enabled shall provide a force response Pending (NRC78) at the beginning of the Hash Finish operation.</p> <p>This parameter is introduced to avoid causing P2 timeouts at client side when the Crypto driver main function call for the Crypto Finish operation blocks the other Boot functions longer than the modules manage period.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

4.6.1.20. NRC78_Transmission

Parameters included	
Parameter name	Multiplicity
Transmit_Nrc78_On_SecurityAccess	1..1
Parameter Name	Transmit_Nrc78_On_SecurityAccess



Description	Specify if an NRC78 response shall be sent before starting the Security Access. ► Case untick: NRC78 will not be sent
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

4.6.1.21. Security

Containers included		
Container name	Multiplicity	Description
ProgCsmReferences	1..1	Label: ProgCsmReferences Contains references to Csm configuration identifiers.

Parameters included	
Parameter name	Multiplicity
Secure_Checksum_computation	1..1
CHECKSUM_LENGTH	1..1
ProgCsmSecureConfigId	1..1
Submit_signature_request	1..1
BS_size	1..1
Allow2Cancel_OngoingJobs	1..1
Allow2SetCryptoKey	1..1
Allow2CustomCsmStartPreprocess	1..1

Parameter Name	Secure_Checksum_computation
Label	Checksum computation
Description	It allows the computation and writing of the checksum (Hash, MAC...) of logical blocks of Application used when Secure Boot or Authenticated Boot is activated. The verification of the checksum computed is activated when the parameter SECURE_AUTHENTICATED_BOOT is set either to Secure or to Authenticated in BM plugin.
Multiplicity	1..1
Type	BOOLEAN

Default value	false
Origin	EB

Parameter Name	CHECKSUM_LENGTH
Label	Checksum length
Description	Size of the checksum for Authenticated / Secure Boot feature in Bytes
Multiplicity	1..1
Type	INTEGER
Default value	16
Origin	EB

Parameter Name	ProgCsmSecureConfigId	
Label	ProgCsmSecureConfigId	
Description	Reference a <i>CsmHash</i> or <i>CsmMacGenerate</i> for the Secure Boot checksum generation Dependencies: ▶ Reference shall be valid This field is disabled if HSM is used.	
Multiplicity	1..1	
Type	CHOICE-REFERENCE	
Configuration class	PreCompile:	VariantPreCompile
Origin	Elektrobit Automotive GmbH	

Parameter Name	Submit_signature_request	
Label	Submit Signature Request	
Description	It allows to select between two type of Submit Signature : ▶ With Address : The Routine control contains two parameters, the first identifies which software component the signature refers to by nominally being the start address of the hash table for the object in the ECU memory, the second parameter is the signature. ▶ Without Address : The Routine control contains only one parameter which is the signature.	
Multiplicity	1..1	



Type	STRING
Default value	With Address
Range	With Address
	Without Address
Origin	EB

Parameter Name	BS_size
Label	Data size for Block Slicing
Description	<p>This feature shall allow the Integrator to decide the size of the data to be processed in a single Asynchronous Crypto main function call.</p> <p>it allows to reduce the time blocked by the CryptoMainfunction in a single main function call.</p> <p>This feature can be enabled by entering any value other than 0(in bytes) in this field. Setting it to 0 shall allow the Bootlaoder to pass the entire available data for processing in a single CryptoMainfunction call.</p> <p>Note:</p> <ol style="list-style-type: none"> 1. This field is only enabled for the integration of ASR Crypto stack version 4.3 or higher. 2. Enabling this feature (by setting a very small size) shall increase the time for verification significantly, integrator shall decide the value to be set. 3. It is recommended to set this value to a maximum value that is sufficient to not to cause timeouts in the system.
Multiplicity	1..1
Type	INTEGER
Default value	0
Origin	EB

Parameter Name	Allow2Cancel_OngoingJobs
Label	Allow cancellation of any ongoing Crypto job
Description	<p>If enabled, any ongoing or active Crypto job operations shall be cancelled at the start of this job operation.</p> <p>This option is applicable only for the integration of Crypto ASR 4.3 or higher versions, for older version this feature is realized using 'Enable the cancelation of ongoing requests' in the Csm/Cry modules.</p>



Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	Allow2SetCryptoKey
Label	Allow to fetch the stored Key
Description	<p>This will allow the integrator to pass the desired key and its elements at the start of the Crypto operation.</p> <p>This option is applicable only for the integration of Crypto ASR 4.3 or higher versions.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	Allow2CustomCsmStartPreprocess
Label	Custom CSM start pre-processing
Description	<p>This will allow the integrator to perform the encoding (eg. Base64, DER etc.) on the Key stored at the host side at the start of the Crypto operation</p> <p>This option is applicable only for the integration of Crypto ASR 4.3 or higher versions.</p> <p>Note: The key encoding shall be performed only if the Crypto driver expects the key to be encoded in an expected format.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

4.6.1.22. ProgCsmReferences

Containers included		
Container name	Multiplicity	Description



Containers included

SignatureVerification	1..1	
HashVerification	1..1	

4.6.1.23. SignatureVerification

Parameters included

Parameter name	Multiplicity
ProgCsmSignatureVerifyConfigId	1..1
BS_size	1..1
Allow2Cancel_OngoingJobs	1..1
Allow2SetCryptoKey	1..1
Allow2CustomCsmStartPreprocess	1..1
AllowStreamStart	1..1
SignFinSendFRP	1..1

Parameter Name	ProgCsmSignatureVerifyConfigId	
Label	ProgCsmSignatureVerifyConfigId	
Description	<p>Reference a <i>CsmSignatureVerificationID</i></p> <p>Specify which csm config Id shall be used for Signature Verification.</p> <p>When the signature verification is used during the Download procedure(Erase, RequestDownload, CheckMemory).</p> <p>An appropriate csm signature verification configuration should be referenced.</p> <p>Dependencies:</p> <ul style="list-style-type: none"> ▶ Cryptographic libraries shall be present 	
Multiplicity	1..1	
Type	CHOICE-REFERENCE	
Configuration class	PreCompile:	VariantPreCompile
Origin	Elektrobit Automotive GmbH	

Parameter Name	BS_size
-----------------------	----------------



Label	Data size for Block Slicing
Description	<p>This feature shall allow the Integrator to decide the size of the data to be processed in a single Asynchronous Crypto main function call.</p> <p>it allows to reduce the time blocked by the CryptoMainfunction in a single main function call.</p> <p>This feature can be enabled by entering any value other than 0(in bytes) in this field. Setting it to 0 shall allow the Bootlaoder to pass the entire available data for processing in a single CryptoMainfunction call.</p> <p>Note:</p> <ol style="list-style-type: none"> 1. This field is only enabled for the integration of ASR Crypto stack version 4.3 or higher. 2. Enabling this feature (by setting a very small size) shall increase the time for verification significantly, integrator shall decide the value to be set. 3. It is recommended to set this value to a maximum value that is sufficient to not to cause timeouts in the system. 4. If this feature is enabled together with verification on fly, the minimum value of Verification_Buffer_size and BS_size shall be effectively selected as the processing data size.
Multiplicity	1..1
Type	INTEGER
Default value	0
Origin	EB

Parameter Name	Allow2Cancel_OngoingJobs
Label	Allow cancellation of any ongoing Crypto job
Description	<p>If enabled, any ongoing or active Crypto job operations shall be cancelled at the start of this job operation.</p> <p>This option is applicable only for the integration of Crypto ASR 4.3 or higher versions, for older version this feature is realized using 'Enable the cancelation of ongoing requests' in the Csm/Cry modules.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB



Parameter Name	Allow2SetCryptoKey
Label	Allow to fetch the stored Key
Description	<p>This will allow the integrator to pass the desired key and its elements at the start of the Crypto operation.</p> <p>This option is applicable only for the integration of Crypto ASR 4.3 or higher versions.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	Allow2CustomCsmStartPreprocess
Label	Custom CSM start pre-processing
Description	<p>This will allow the integrator to perform the custom operations such as key encoding (eg. Base64, DER etc.) on the Key stored at the host side at the start of the Crypto operation</p> <p>This option is applicable only for the integration of Crypto ASR 4.3 or higher versions.</p> <p>Note: The key encoding shall be performed only if the Crypto driver expects the key to be encoded in an expected format.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	AllowStreamStart
Label	Enable StreamStart mode
Description	<p>This parameter shall be enabled if the Signature Start operation takes more than one cycle of CryptoMainFunction execution.</p> <p>Note:</p> <ol style="list-style-type: none"> 1. This option is applicable only for the integration of Crypto ASR 4.3 or higher versions. 2. This shall be enabled only if the crypto engine supports the StreamStart feature.



Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	SignFinSendFRP
Label	Send NRC78 for Signature Finish
Description	<p>This parameter if enabled shall provide a force response Pending (NRC78) at the beginning of the Signature Finish operation.</p> <p>This parameter is introduced to avoid causing P2 timeouts at client side when the Crypto driver main function call for the Crypto Finish operation blocks the other Boot functions longer than the modules manage period.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

4.6.1.24. HashVerification

Parameters included	
Parameter name	Multiplicity
ProgCsmHashConfigId	1..1
BS_size	1..1
Allow2Cancel_OngoingJobs	1..1
HashFinSendFRP	1..1

Parameter Name	ProgCsmHashConfigId
Label	ProgCsmHashConfigId
Description	<p>Reference a <i>CsmHashVerificationID</i></p> <p>Specify which csm config Id shall be used for Hash Verification.</p> <p>When the hash verification is used during the Download procedure(Erase, RequestDownload, CheckMemory).</p> <p>An appropriate csm hash verification configuration should be referenced.</p>



	Dependencies: ▶ Cryptographic libraries shall be present
Multiplicity	1..1
Type	CHOICE-REFERENCE
Configuration class	PreCompile: VariantPreCompile
Origin	Elektrobit Automotive GmbH

Parameter Name	BS_size
Label	Data size for Block Slicing
Description	<p>This feature shall allow the Integrator to decide the size of the data to be processed in a single Asynchronous Crypto main function call.</p> <p>It allows to reduce the time blocked by the CryptoMainfunction in a single main function call.</p> <p>This feature can be enabled by entering any value other than 0(in bytes) in this field. Setting it to 0 shall allow the Bootlaoder to pass the entire available data for processing in a single CryptoMainfunction call.</p> <p>Note:</p> <ol style="list-style-type: none"> 1. This field is only enabled for the integration of ASR Crypto stack version 4.3 or higher. 2. Enabling this feature (by setting a very small size) shall increase the time for verification significantly. The integrator shall decide the value to be set. 3. It is recommended to set this value to a maximum value that is sufficient to not to cause timeouts in the system. 4. If this feature is enabled together with verification on fly, the minimum value of Verification_Buffer_size and BS_size shall be effectively selected as the processing data size.
Multiplicity	1..1
Type	INTEGER
Default value	0
Origin	EB

Parameter Name	Allow2Cancel_OngoingJobs
Label	Allow cancellation of any ongoing Crypto job



Description	If enabled, any ongoing or active Crypto job operations shall be cancelled at the start of this job operation. This option is applicable only for the integration of Crypto ASR 4.3 or higher versions, for older version this feature is realized using 'Enable the cancelation of ongoing requests' in the Csm/Cry modules.
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	HashFinSendFRP
Label	Send NRC78 for Hash Finish
Description	This parameter if enabled shall provide a force response Pending (NRC78) at the beginning of the Hash Finish operation. This parameter is introduced to avoid causing P2 timeouts at client side when the Crypto driver main function call for the Crypto Finish operation blocks the other Boot functions longer than the modules manage period.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

4.6.1.25. OemInd

Parameters included	
Parameter name	Multiplicity
<u>Erase_Mode</u>	1..1
<u>EraseALFI_Enable</u>	1..1
<u>Request_Download_Address_Mode</u>	1..1
<u>CoherencyCheck_Enable</u>	1..1
<u>Application_Validity_Algo</u>	1..1
<u>Checksum_Algo</u>	1..1
<u>Size_Of_FingerPrint</u>	1..1
Parameter Name	Erase_Mode



Label	Erasing mode
Description	<p>Specify how the erasing shall be performed.</p> <ul style="list-style-type: none"> ▶ Case All: No information are provided in EraseMemory routine and all the configured segments will be erased on reception of the routine. ▶ Case Address: Erasing will be performed on the configured segment matching the address provided in EraseMemory routine ▶ Case LogicalBlock: Erasing will be performed on the configured segments associated (by configuration) to the logical block Id provided in EraseMemory routine
Multiplicity	1..1
Type	ENUMERATION
Default value	Address
Range	<ul style="list-style-type: none"> All Address LogicalBlock
Origin	EB

Parameter Name	EraseALFI_Enable
Label	Erase request ALFI Enable
Description	Enable the ALFI UDS field in EraseMemory request (Available only if erasing mode is by address)
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Request_Download_Address_Mode
Label	Request Download Addressing Mode
Description	<p>Specify the content of the request download start address</p> <ul style="list-style-type: none"> ▶ Case Download by logical block: Only block ID will be sent in the request and the block cannot have more than one segment ▶ Case Download by logical block and segment: block ID and segment ID will be sent in the request and the block can have multiple segments
Multiplicity	1..1



Type	ENUMERATION
Default value	Download by Segment Address
Range	Download by Segment Address
	Download by logical block
	Download by logical block and segment
Origin	EB

Parameter Name	CoherencyCheck_Enable
Label	Coherency check Enable
Description	Enable the coherency check (check programming dependencies).
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Application_Voidality_Algo
Label	Application validity check and fingerprint: EB specific / Customer specific
Description	<p>Specify if the Elektrobit algorithm shall be used for application validity management and fingerprint management or a customer specific one.</p> <ul style="list-style-type: none"> ▶ Case EB: standard algorithm will be used ▶ Case Custom: Customer shall complete the PROG_IsValidApplication/PROG_CustomSetDownloadVerificationSuccess/PROG_InvalidateSection/PROG_EntryWriteFingerprint and managed the application status.
Multiplicity	1..1
Type	ENUMERATION
Default value	EB
Range	EB Custom
Origin	EB

Parameter Name	Checksum_Algo
Label	CRC algorithm: NO CRC / Signature / Hash256 / Hash512 / CRC16 CCITT/ CRC32 Ethernet / CRC32 International Standard 32-Bit CRC
Description	Specify which checksum algorithm shall be used



	<ul style="list-style-type: none"> ▶ Case NO CRC: No checksum needed. ▶ Case Signature: A cryptographic signature verification will be done (only possible if cryptographic libraries are used) ▶ Case Hash256: A cryptographic Hash verification will be done with SHA-256(only possible if cryptographic libraries are used) ▶ Case Hash512: A cryptographic Hash verification will be done SHA-512(only possible if cryptographic libraries are used) ▶ Case CRC16 CCITT : Polynomial 0x1021 / Init value 0xFFFF / ReflectIn FALSE / ReflectOut FALSE / No XOR on Output ▶ Case CRC32 Ethernet: Polynomial 0x04C11DB7 / Init value 0xFFFFFFFF / ReflectIn TRUE / ReflectOut TRUE / XOR on Output 0xFFFFFFFF ▶ Case CRC32 International Standard 32-Bit CRC: Polynomial 0xEDB88320 / Init value 0xFFFFFFFF / ReflectIn FALSE / ReflectOut FALSE / XOR on Output 0xFFFFFFFF
Multiplicity	1..1
Type	ENUMERATION
Default value	CRC32 Ethernet
Range	NO CRC Signature Hash256 Hash512 CRC16 CRC32 Ethernet CRC32 InternationalStandard
Origin	EB

Parameter Name	Size_Of_FingerPrint
Description	Define the size of the FingerPrint in Bytes. The size can't be negative.
Multiplicity	1..1
Type	INTEGER
Default value	16
Range	>=0
Origin	EB



4.6.1.26. VAG

Parameters included	
Parameter name	Multiplicity
Request_Download_Address_Mode	1..1
Downgrade_Protection	1..1
BID_Length	1..1
PFT_Block_Id	1..1

Parameter Name	Request_Download_Address_Mode
Label	Request Download Addressing Mode
Description	<p>Specify the content of the request download start address</p> <ul style="list-style-type: none"> ▶ Case Download by logical block: Only block ID will be sent in the request and the block cannot have more than one segment ▶ Case Download by logical block and segment: block ID and segment ID will be sent in the request and the block can have multiple segments
Multiplicity	1..1
Type	ENUMERATION
Default value	Download by logical block and segment
Range	<p>Download by logical block</p> <hr/> <p>Download by logical block and segment</p>
Origin	EB

Parameter Name	Downgrade_Protection
Label	Downgrade Protection
Description	<p>Enable the downgrade protection feature</p> <ul style="list-style-type: none"> ▶ Case tick : Downgrade protection feature is enabled and the configuration of the PFT block shall be performed ▶ Case untick: Downgrade protection feature is disabled
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB
Parameter Name	BID_Length

Label	Block Identifier Length		
Description	<p>Length of the block identifier contained in the pseudo flash driver 1 or 2 bytes</p> <ul style="list-style-type: none"> ▶ Case 1 byte : BIDLEN is 1 byte ▶ Case 2 bytes : BIDLEN is 2 bytes 		
Multiplicity	1..1		
Type	ENUMERATION		
Default value	1		
Range	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="padding: 2px;">1</td> </tr> <tr> <td style="padding: 2px;">2</td> </tr> </table>	1	2
1			
2			
Origin	EB		

Parameter Name	PFT_Block_Id
Label	PFT Block Identifier
Description	<p>Reference to the <i>pseudo flash driver block identifier</i></p> <p>Dependencies:</p> <ul style="list-style-type: none"> ▶ Reference shall be valid
Multiplicity	1..1
Type	CHOICE-REFERENCE
Origin	Elektrobit Automotive GmbH

4.6.1.27. PSA

Parameters included	
Parameter name	Multiplicity
Memory_Base_Address	1..1

Parameter Name	Memory_Base_Address
Label	Memory Base Address
Description	<p>Provides the Most Significant Byte (MSB) of addresses</p> <p>Applicable only for 32 bits ECU address</p> <p>If ECU address is different from 32 bits, set this value 0x00</p> <p>Range [0x00; 0xFF]</p>



Multiplicity	1..1
Type	INTEGER
Default value	0xA0
Range	>=0
	<=255
Origin	EB

4.6.1.28. Decryption

Parameters included	
Parameter name	Multiplicity
Enable_Decryption	1..1
Decryption_Algorithm	1..1
Enable_Csm_Decryption	1..1
ProgCsmDecryptionConfigId	1..1
Allow2Cancel_OngoingJobs	1..1
Allow2SetCryptoKey	1..1
Allow2SetIV	1..1
Allow2CustomCsmStartPreprocess	1..1

Parameter Name	Enable_Decryption
Description	If enabled, a callback will be called on data reception allowing integration code to perform data decryption.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Decryption_Algorithm
Label	Decryption algorithm Id
Description	Decryption algorithm id that shall be supported from dataFormatIdentifier field of RequestDownload service
Multiplicity	1..1
Type	INTEGER



Default value	1
Range	<=15
Origin	EB

Parameter Name	Enable_Csm_Decryption
Label	Use Symetric algorithm for decryption
Description	If enabled, it allows to use a symetric algorithm provided by CSM library. Current implementation is limited to CBC decryption with PKCS7 Padding. Cry Primitive name : CbcPkcs7Decrypt. If disabled, the user is free to implement its own decryption algorithm within callback function.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	ProgCsmDecryptionConfigId	
Label	ProgCsmDecryptionConfigId	
Description	Reference a <i>CsmSymDecryptConfig</i> Dependencies: ▶ Reference shall be valid	
Multiplicity	1..1	
Type	CHOICE-REFERENCE	
Configuration class	PreCompile:	VariantPreCompile
Origin	Elektrobit Automotive GmbH	

Parameter Name	Allow2Cancel_OngoingJobs	
Label	Allow cancellation of all ongoing Crypto jobs	
Description	If enabled, any ongoing or active Crypto job operations shall be cancelled at the start of this job operation. This option is applicable only for the integration of Crypto ASR 4.3 or higher versions, for older version this feature is realized using 'Enable the cancelation of ongoing requests' in the Csm/Cry modules.	
Multiplicity	1..1	



Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	Allow2SetCryptoKey
Label	Allow to set the stored Key
Description	<p>This will allow the integrator to pass the desired key and its elements at the start of the Crypto operation.</p> <p>This option is applicable only for the integration of Crypto 4.3 or higher versions.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	Allow2SetIV
Label	Allow to set the IV
Description	<p>This will allow the integrator to pass the desired IV value needed for the decryption (this is AES algo dependant).</p> <p>This option is applicable only for the integration of Crypto 4.3 or higher versions.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	Allow2CustomCsmStartPreprocess
Label	Custom CSM start pre-processing
Description	<p>This will allow the integrator to perform Csm job specific preprocessing operations such as performing encoding (eg. Base64, DER etc.) on the Key stored at the host side just before the start of the Crypto operation</p> <p>This option is applicable only for the integration of Crypto 4.3 or higher versions.</p> <p>Note: The key encoding shall be performed only if the Crypto driver expects the key to be encoded in an expected format.</p>
Multiplicity	1..1
Type	BOOLEAN



Default value	false
Origin	EB

4.6.2. Application programming interface (API)

4.6.2.1. Type definitions

4.6.2.1.1. ptAPPL_START_ADDR

Purpose	
Type	void(*) (void)

4.6.2.1.2. ptCompleteCompatibleCallOut

Purpose	
Type	tProgCompleteStatus (*) (void)

4.6.2.1.3. ptSBL_StartUp_Code

Purpose	
Type	void(*) (u32 ulInfoSBL, u8 ubRxPduId)

4.6.2.1.4. tCryptoData

Purpose	
Type	struct
Members	u8 aubCryptoBuffer
	u32 ulNextWritePosition
	u32 ulCryptoBufferSize
	u32 ulEncryptDataSize
	u32 ulTotDecryptedWrittenSize



	u16 uwRemainNotWrittenLength	
	tProgEncryptOperation ubOperationInProgress	

4.6.2.1.5. tDataBlockType

Purpose	
Type	struct
Members	u32 ulStartAddress
	u32 ulLength
	u8 aubDigest

4.6.2.1.6. tDataBufferType

Purpose	
Type	u8

4.6.2.1.7. tDataLength

Purpose	
Type	u32

4.6.2.1.8. tMultipleBuffReprogInfo

Purpose	
Type	struct
Members	u32 ulBufferedSizeOfData
	u8 eResponsePending
	u8 eBufferProcessing

4.6.2.1.9. tOperationType

Purpose	
---------	--



Type	u8
------	----

4.6.2.1.10. tPageBuffer

Purpose	
Type	struct
Members	u8 aubData
	PduLengthType ulOldDataLength
	PduLengthType ulNewDataLength

4.6.2.1.11. tProgAccessType

Purpose	
Type	u8

4.6.2.1.12. tProgAddressType

Purpose	
Type	u32

4.6.2.1.13. tProgBoolean

Purpose	
Type	u8

4.6.2.1.14. tProgCompTimeoutStatus

Purpose	
Type	u8

4.6.2.1.15. tProgCompleteStatus

Purpose	
---------	--



Type	u32
------	-----

4.6.2.1.16. tProgCsmJobConf

Purpose	
Type	struct
Members	u32 ulBSsize u32 ulCsmJobId tProgCsmNotification pfuProgCsmNotification u32 ulCsmKeyID u32 ulCryptoElementID u32 ulCryptoKeyLength tProgBoolean eAllowJobCancellation tProgBoolean eAllowKeySet tProgBoolean eAllowCustCsmPreProc

4.6.2.1.17. tProgCsmNotification

Purpose	
Type	void(*) (u8 eCsmStatus)

4.6.2.1.18. tProgDownloadType

Purpose	
Type	u8

4.6.2.1.19. tProgEncryptOperation

Purpose	
Type	u8



4.6.2.1.20. tProgMemIdx

Purpose	
Type	u8

4.6.2.1.21. tProgMemMode

Purpose	
Type	u8

4.6.2.1.22. tProgMemType

Purpose	
Type	u8

4.6.2.1.23. tProgPECError

Purpose	
Type	u16

4.6.2.1.24. tProgPartitionType

Purpose	
Type	u8

4.6.2.1.25. tProgPsiValue

Purpose	
Type	u8

4.6.2.1.26. tProgRequestRoutineData

Purpose	
Type	struct



Members	u8 aubRoutineData	
	u16 uwRoutineLength	

4.6.2.1.27. tProgRequestRoutineResult

Purpose	
Type	u8

4.6.2.1.28. tProgResCause

Purpose	
Type	u8

4.6.2.1.29. tProgRoutineResultInfo

Purpose	
Type	struct
Members	u16 uwRoutineId
	u8 ubRoutineIndexList

4.6.2.1.30. tProgSigBypass

Purpose	
Type	u8

4.6.2.1.31. tProgStatus

Purpose	
Type	u8

4.6.2.1.32. tProgVerifAlgo

Purpose	
----------------	--



Type	u8
------	----

4.6.2.1.33. tProgVerificationInfo

Purpose	
Type	struct
Members	u32 ulAdd u32 ulCnt u32 ulVal u16 uwVal tProgVerifAlgo ubAlgo u8 ubLogicalBlockId

4.6.2.1.34. tProgrammingFlags

Purpose	
Type	struct
Members	u8 ubSeedSent u8 ubEcuUnlock u8 ubTunBlank u8 ubGlobalBlank u8 ubProgInit u8 ubAuthorizeLog u8 ubWriteOk u8 ubRDOK u8 ubRDLogOk u8 ubRDKeyAppliOk u8 ubLCBaudOk u8 ubLCProgOk

4.6.2.1.35. tRDParam

Purpose	
---------	--



Type	struct
Members	u32 ulStartAddress u32 ulMemorySize u16 uwBlockIdentifier u8 ubSegmentId u8 ubDataFormatId

4.6.2.1.36. tRegionotype

Purpose	
Type	struct
Members	u32 ulAddress u32 ulSize

4.6.2.1.37. tReprogInfo

Purpose	
Type	struct
Members	u32 ulMemorySizeExpected u32 ulReceivedSizeOfData u32 ulTotalDecompDataWrite u16 uwExpectedModuleId u8 ubBlockSequenceCounter u8 ubRDReceived u8 ubCompRequired

4.6.2.1.38. tSegmentType

Purpose	
Type	struct
Members	u32 ulStartAddress u32 ulSize tRegionotype astRegion



	u8 * pubRegionPtr	
	u16 uwModuleId	
	u8 ubRegionNbr	
	u8 ubLogicalSegmentId	
	u8 ubLogicalBlockId	
	u8 ubSegmentId	
	u8 ubSegmentNbr	

4.6.2.1.39. tSignatureHandlingstatus

Purpose	
Type	u8

4.6.2.1.40. tVDSlayout

Purpose		
Type	struct	
Members	u32 ulBlock_StartAddress	
	u32 ulBlock_Length	
	u8 aubBlockDigest	

4.6.2.1.41. tWriteInfo

Purpose		
Type	struct	
Members	u8 * pubDecompData	
	u32 ulAddressToWrite	
	u32 ulDataToBeWritten	
	u32 ulWrittenData	
	u32 ulWriteLength	
	u32 ulInDecompDataLength	
	u32 ulNewAddressToWrite	
	u8 ubTDReceived	



4.6.2.1.42. t_PROG_fctptr

Purpose	
Type	void * (*) (void)

4.6.2.1.43. t_secondary_bootloader_interface

Purpose	
Type	struct
Members	u32 software_version
	u8 referenceString
	void * ptr_function
	u32 uSBLValidityFlagAddr

4.6.2.1.44. tpulGetAddress

Purpose	
Type	void * (*) (u8, u32)

4.6.2.1.45. tpulVerifySectionCrc

Purpose	
Type	u8 (*) (void)

4.6.2.1.46. tpulinvalidateSection

Purpose	
Type	u8 (*) (u32)

4.6.2.1.47. tpulisValidApplication

Purpose	
Type	u8 (*) (void)



4.6.2.1.48. tpulskipPage

Purpose	
Type	u8 (*) (u32 *)

4.6.2.2. Macro constants

4.6.2.2.1. CRY_

Purpose	
Value	[!"\$AlgoType"!]_WORD_SIZE [!WS "9"!]32U

4.6.2.2.2. PROG_1ST_PROGRAMMING_ERASE_CHECK

Purpose	
Value	0x01U

4.6.2.2.3. PROG_APPLICATION_PARTITION

Purpose	
Value	0x01U

4.6.2.2.4. PROG_AUTHENTICATED_REPROG

Purpose	
Value	2U

4.6.2.2.5. PROG_BLOCK_ERASE_CHECK

Purpose	
Value	0x02U



4.6.2.2.6. PROG_BLU_APP_PARTITION

Purpose	
Value	0x11U

4.6.2.2.7. PROG_BLU_CAL_PARTITION

Purpose	
Value	0x12U

4.6.2.2.8. PROG_BLU_PARTITION

Purpose	
Value	0x10U

4.6.2.2.9. PROG_BLU_PARTITION_MASK

Purpose	
Value	0x10U

4.6.2.2.10. PROG_BOOTLOADER_PARTITION

Purpose	
Value	0x05U

4.6.2.2.11. PROG_BOOT_DLS_SIZE

Purpose	
Value	2U

4.6.2.2.12. PROG_BOOT_MAX_PROT_PARTITIONS

Purpose	
---------	--



Value	[!"num:decoint(\$NBR_PROT_CAL)"!]U
-------	------------------------------------

4.6.2.2.13. PROG_BOOT_MODULE_ID_SIZE

Purpose	
Value	1U

4.6.2.2.14. PROG_BOOT_NB_MODULE_SIZE

Purpose	
Value	1U

4.6.2.2.15. PROG_BOOT_NUMBER_OF_MODULES

Purpose	
Value	0x01U

4.6.2.2.16. PROG_BOOT_PART_NUMBER_SIZE

Purpose	
Value	4U

4.6.2.2.17. PROG_BOOT_PRIMARY_MICRO_ID

Purpose	
Value	0x47U

4.6.2.2.18. PROG_BOOT_PROT_CALIB_NUMBER_SIZE

Purpose	
Value	1U



4.6.2.2.19. PROG_BOOT_PROT_CALIB_PARTITION_ID_SIZE

Purpose	
Value	1U

4.6.2.2.20. PROG_CALIBRATION_PARTITION

Purpose	
Value	0x02U

4.6.2.2.21. PROG_CIPHERED_MIN_DATA_SIZE

Purpose	
Value	256

4.6.2.2.22. PROG_COMPLETECOMPATIBLE_END

Purpose	
Value	0x02U

4.6.2.2.23. PROG_COMPLETECOMPATIBLE_ERROR

Purpose	
Value	0x03U

4.6.2.2.24. PROG_COMPLETECOMPATIBLE_START

Purpose	
Value	0x01U

4.6.2.2.25. PROG_CRCDONE

Purpose	
---------	--



Value	0U
--------------	----

4.6.2.2.26. PROG_CRYPTOFINISHDONE

Purpose	
Value	3U

4.6.2.2.27. PROG_CRYPTOSTARTUPDATEINPROGRESS

Purpose	
Value	1U

4.6.2.2.28. PROG_CRYPTOUPDATEDONE

Purpose	
Value	2U

4.6.2.2.29. PROG_DECRYPTION_IN_PROGRESS

Purpose	
Value	0xAAU

4.6.2.2.30. PROG_DECRYPT_BUFFER_SIZE

Purpose	
Value	4112U

4.6.2.2.31. PROG_DIGEST_LENGTH

Purpose	
Value	32U



4.6.2.2.32. PROG_DISABLED_ERASE_CHECK

Purpose	
Value	0x00U

4.6.2.2.33. PROG_DOWNLOAD_BY_ADDR

Purpose	
Value	0x01U

4.6.2.2.34. PROG_DOWNLOAD_BY_LOGICAL_BLOCK

Purpose	
Value	0x02U

4.6.2.2.35. PROG_DOWNLOAD_BY_LOGICAL_BLOCK_SEGMENT

Purpose	
Value	0x03U

4.6.2.2.36. PROG_ECU_ID_SIZE

Purpose	
Value	16U

4.6.2.2.37. PROG_ECU_NAME_SIZE

Purpose	
Value	8U

4.6.2.2.38. PROG_ERASEDONE

Purpose	
---------	--



Value	5U
--------------	----

4.6.2.2.39. PROG_ERASESTART

Purpose	
Value	4U

4.6.2.2.40. PROG_ERR_APP_NBID

Purpose	
Value	0x0016U

4.6.2.2.41. PROG_ERR_BCID

Purpose	
Value	0x0010U

4.6.2.2.42. PROG_ERR_CAL_REGION

Purpose	
Value	0x001AU

4.6.2.2.43. PROG_ERR_CCID

Purpose	
Value	0x0011U

4.6.2.2.44. PROG_ERR_CERT

Purpose	
Value	0x0019U



4.6.2.2.45. PROG_ERR_COMPRESSION

Purpose	
Value	0x000BU

4.6.2.2.46. PROG_ERR_DATA_TYPE

Purpose	
Value	0x000AU

4.6.2.2.47. PROG_ERR_ECU_ID

Purpose	
Value	0x0013U

4.6.2.2.48. PROG_ERR_ECU_NAME

Purpose	
Value	0x0012U

4.6.2.2.49. PROG_ERR_ERASE_CAL

Purpose	
Value	0x0007U

4.6.2.2.50. PROG_ERR_ERASE_SW

Purpose	
Value	0x0004U

4.6.2.2.51. PROG_ERR_FLASH_WRITE

Purpose	
----------------	--



Value	0x000EU
--------------	---------

4.6.2.2.52. PROG_ERR_GET_APP_INFO

Purpose	
Value	0x0005U

4.6.2.2.53. PROG_ERR_GET_CAL_INFO

Purpose	
Value	0x0008U

4.6.2.2.54. PROG_ERR_KEY_NBID

Purpose	
Value	0x0018U

4.6.2.2.55. PROG_ERR_LENGTH_EXCEEDED

Purpose	
Value	0x000CU

4.6.2.2.56. PROG_ERR_MD

Purpose	
Value	0x001BU

4.6.2.2.57. PROG_ERR_MODULE_ID

Purpose	
Value	0x000FU



4.6.2.2.58. PROG_ERR_MORE_DATA_EXPECTED

Purpose	
Value	0x000DU

4.6.2.2.59. PROG_ERR_MSG_OUT_OF_SEQUENCE

Purpose	
Value	0x001DU

4.6.2.2.60. PROG_ERR_PARTITION_ID

Purpose	
Value	0x0001U

4.6.2.2.61. PROG_ERR_PER_DATA_TX_NOT_ALLOW

Purpose	
Value	0x0037U

4.6.2.2.62. PROG_ERR_PROTECTEDCAL_NOT_DEFINED

Purpose	
Value	0x0035U

4.6.2.2.63. PROG_ERR_REVOKE_CAL

Purpose	
Value	0x0006U

4.6.2.2.64. PROG_ERR_REVOKE_SW

Purpose	
---------	--



Value	0x0003U
--------------	---------

4.6.2.2.65. PROG_ERR_ROOT_SIGNATURE

Purpose	
Value	0x001CU

4.6.2.2.66. PROG_ERR_SBA_CERT

Purpose	
Value	0x08U

4.6.2.2.67. PROG_ERR_SBA_ECU_ID

Purpose	
Value	0x02U

4.6.2.2.68. PROG_ERR_SBA_ECU_NAME

Purpose	
Value	0x01U

4.6.2.2.69. PROG_ERR_SBA_SIGNATURE

Purpose	
Value	0x04U

4.6.2.2.70. PROG_ERR_SIGNATURE

Purpose	
Value	0x0015U



4.6.2.2.71. PROG_ERR SUBJECT NAME

Purpose	
Value	0x0017U

4.6.2.2.72. PROG_ERR_SW_NOT_PRESENT

Purpose	
Value	0x0002U

4.6.2.2.73. PROG_ERR_SW_REGION

Purpose	
Value	0x0014U

4.6.2.2.74. PROG_ERR_UNDEFINED

Purpose	
Value	0x0020U

4.6.2.2.75. PROG_ERR_UPDATE_PSI

Purpose	
Value	0x0009U

4.6.2.2.76. PROG_ESS_PARTITION

Purpose	
Value	0x07U

4.6.2.2.77. PROG_E_BUSY

Purpose	
---------	--



Value	0x02U
--------------	-------

4.6.2.2.78. PROG_E_BYPASS

Purpose	
Value	0x04U

4.6.2.2.79. PROG_E_CHECK_FAILED

Purpose	
Value	0x03U

4.6.2.2.80. PROG_E_COHCHK_CORRECT

Purpose	
Value	0x00U

4.6.2.2.81. PROG_E_COHCHK_INCORRECT

Purpose	
Value	0x01U

4.6.2.2.82. PROG_E_COHCHK_INCORRECT_OTHER

Purpose	
Value	0x04U

4.6.2.2.83. PROG_E_COHCHK_INCORRECT_SW_HW

Purpose	
Value	0x02U



4.6.2.2.84. PROG_E_COHCHK_INCORRECT_SW_SW

Purpose	
Value	0x03U

4.6.2.2.85. PROG_E_COHPRECHK_CORRECT

Purpose	
Value	0x00U

4.6.2.2.86. PROG_E_COHPRECHK_INCORRECT_HW_SW

Purpose	
Value	0x02U

4.6.2.2.87. PROG_E_COHPRECHK_INCORRECT_SW_SW

Purpose	
Value	0x03U

4.6.2.2.88. PROG_E_COHPRECHK_INTERNAL_ERROR

Purpose	
Value	0x01U

4.6.2.2.89. PROG_E_ERASED

Purpose	
Value	0x01U

4.6.2.2.90. PROG_E_NOT_BUSY

Purpose	
---------	--



Value	0x80U
--------------	-------

4.6.2.2.91. PROG_E_NOT_ERASED

Purpose	
Value	0x00U

4.6.2.2.92. PROG_E_NOT_OK

Purpose	
Value	0x01U

4.6.2.2.93. PROG_E_OK

Purpose	
Value	0x00U

4.6.2.2.94. PROG_E_RFS_DRIVER_FAIL

Purpose	
Value	0x05U

4.6.2.2.95. PROG_E_RFS_VERSION_FAIL

Purpose	
Value	0x06U

4.6.2.2.96. PROG_E_TXCONF_OK

Purpose	
Value	0x40U



4.6.2.2.97. PROG_FALSE

Purpose	
Value	0U

4.6.2.2.98. PROG_FLASH_ROUTINES_PARTITION

Purpose	
Value	0x06U

4.6.2.2.99. PROG_HSM_PARTITION

Purpose	
Value	0x08U

4.6.2.2.100. PROG_LOGICAL_BLOCK_VALUE_INIT

Purpose	
Value	[!"\$BLOCK_IDENTIFIER_VALUE"]

4.6.2.2.101. PROG_MAX_LENGTH_CHECKMEMORY_ANSWER

Purpose	
Value	6U

4.6.2.2.102. PROG_MAX_PARTITION

Purpose	
Value	[!"num:decoint(GM/MAX_PARTITION)"]U

4.6.2.2.103. PROG_MAX_RD_PER_BLOCK

Purpose	
---------	--



Value	[!"num:decioint(DownloadVerification/MaxNumberOfRDPerBlock)"!]U
--------------	---

4.6.2.2.104. PROG_MAX_REGION_ALLOWED

Purpose	
Value	[!"num:decioint(GM/MAX_REGION_ALLOWED)"!]U

4.6.2.2.105. PROG_MEMORY_ASYNCROUS

Purpose	
Value	0x02U

4.6.2.2.106. PROG_MEMORY_NB

Purpose	
Value	[!"\$MEMORY_MAX"!]U

4.6.2.2.107. PROG_MEMORY_NOTUSED

Purpose	
Value	0x03U

4.6.2.2.108. PROG_MEMORY_SYNCHRONOUS

Purpose	
Value	0x01U

4.6.2.2.109. PROG_MEM_ACCESS_TYPE_NONE

Purpose	
Value	0x00U



4.6.2.2.110. PROG_MEM_ACCESS_TYPE_READ

Purpose	
Value	0x01U

4.6.2.2.111. PROG_MEM_ACCESS_TYPE_READ_WRITE

Purpose	
Value	0x03U

4.6.2.2.112. PROG_MEM_ACCESS_TYPE_WRITE

Purpose	
Value	0x02U

4.6.2.2.113. PROG_MEM_OPERATION_TYPE_ERASE

Purpose	
Value	0x01U

4.6.2.2.114. PROG_MEM_OPERATION_TYPE_READ

Purpose	
Value	0x03U

4.6.2.2.115. PROG_MEM_OPERATION_TYPE_WRITE

Purpose	
Value	0x02U

4.6.2.2.116. PROG_MEM_TYPE_CUSTOM

Purpose	
---------	--



Value	0x05U
--------------	-------

4.6.2.2.117. PROG_MEM_TYPE_EEPROM

Purpose	
Value	0x01U

4.6.2.2.118. PROG_MEM_TYPE_FLASH

Purpose	
Value	0x00U

4.6.2.2.119. PROG_MEM_TYPE_FLASH_EXT

Purpose	
Value	0x04U

4.6.2.2.120. PROG_MEM_TYPE_INIT

Purpose	
Value	0xFFU

4.6.2.2.121. PROG_MEM_TYPE_RAM

Purpose	
Value	0x02U

4.6.2.2.122. PROG_MEM_TYPE_SCRATCHPAD

Purpose	
Value	0x03U



4.6.2.2.123. PROG_MIN_VAL_TO_WRITE

Purpose	
Value	[!"\$MIN_VAL_TO_WRITE_FOR_ALL_MEMORYIES"!]U

4.6.2.2.124. PROG_NO_OPERATION_IN_PROGRESS

Purpose	
Value	0x00U

4.6.2.2.125. PROG_NO_REPROG_TYPE

Purpose	
Value	0U

4.6.2.2.126. PROG_PEC_NO_ERROR

Purpose	
Value	0x0000U

4.6.2.2.127. PROG_PROT_CALIBRATION_PARTITION

Purpose	
Value	0x03U

4.6.2.2.128. PROG_PSI_INVALID

Purpose	
Value	0x02U

4.6.2.2.129. PROG_PSI_PROGRAMMED

Purpose	
---------	--



Value	0x00U
--------------	-------

4.6.2.2.130. PROG_PSI_REVOKED

Purpose	
Value	0x01U

4.6.2.2.131. PROG_RC_WITH_RI_LENGTH

Purpose	
Value	4U

4.6.2.2.132. PROG_REQUEST_ROUTINE_RESULT_FAIL

Purpose	
Value	0x03

4.6.2.2.133. PROG_REQUEST_ROUTINE_RESULT_INIT

Purpose	
Value	0x00

4.6.2.2.134. PROG_REQUEST_ROUTINE_RESULT_PENDING

Purpose	
Value	0x01

4.6.2.2.135. PROG_REQUEST_ROUTINE_RESULT_SUCCESS

Purpose	
Value	0x02



4.6.2.2.136. PROG_RESET_CAUSE_BLU

Purpose	
Value	0x05U

4.6.2.2.137. PROG_RESET_CAUSE_DSC01

Purpose	
Value	0x01U

4.6.2.2.138. PROG_RESET_CAUSE_DSC02

Purpose	
Value	0x02U

4.6.2.2.139. PROG_RESET_CAUSE_ER

Purpose	
Value	0x00U

4.6.2.2.140. PROG_RESET_CAUSE_S3_TIMEOUT

Purpose	
Value	0x03U

4.6.2.2.141. PROG_RESET_CAUSE_SBLACTIVEFAIL

Purpose	
Value	0x04U

4.6.2.2.142. PROG_SBA_OK

Purpose	
---------	--



Value	0x80U
--------------	-------

4.6.2.2.143. PROG_SBA_PARTITION

Purpose	
Value	0x04U

4.6.2.2.144. PROG_SEGMENT_NB

Purpose	
Value	[!"\$SEGMENT_MAX"!]U

4.6.2.2.145. PROG_STANDARD_REPROG

Purpose	
Value	1U

4.6.2.2.146. PROG SUBJECT NAME SIZE

Purpose	
Value	16U

4.6.2.2.147. PROG_TRUE

Purpose	
Value	1U

4.6.2.2.148. PROG_VDS_SIZE

Purpose	
Value	(PROG_ERASE_ADDR_LEN + PROG_ERASE_SIZE_LEN + PROG_DIGEST_LENGTH)



4.6.2.2.149. PROG_VERIFY_CRC

Purpose	
Value	0x00U

4.6.2.2.150. PROG_VERIFY_HASH

Purpose	
Value	0x02U

4.6.2.2.151. PROG_VERIFY_SIGNATURE

Purpose	
Value	0x01U

4.6.2.3. Objects

4.6.2.3.1. m_astProgCsmJobConf

Purpose	
Type	const tProgCsmJobConf

4.6.2.3.2. m_aubBlockDownloadStatus

Purpose	This Variable is set after a successful or unsuccessful download verification in PROG_CustomSetDownloadVerificationSuccess and variable value shall be checked for validity status of the logical block during CheckValidApplication treatment. PROG_TRUE Treatment finish successfully PROG_FALSE Error happened during treatment.
Type	tProgBoolean

4.6.2.3.3. m_ubCheckMemoryStatus

Purpose	
Type	u8



4.6.2.3.4. m_ubFailedCheckMemoryCount

Purpose	
Type	u8

4.6.2.3.5. m_ubSimulateProgSessionWithResponse

Purpose	
Type	u8

4.6.2.3.6. m_ubValidateMemoryStatus

Purpose	
Type	u8

4.6.2.3.7. m_ulRsDelayTimer

Purpose	
Type	u32

4.6.2.3.8. tFirstCheckMemoryAnswerInfo

Purpose	
Type	

4.6.2.3.9. ubDiagStatus

Purpose	
Type	u8

4.6.2.3.10. ulLength

Purpose	
---------	--



Type	PduLengthType
------	---------------

4.6.2.4. Functions

4.6.2.4.1. PROG_ACT_Check_Impl80

Purpose	API called on the do of the ACT check state.
Synopsis	<code>void PROG_ACT_Check_Impl80 (void);</code>

4.6.2.4.2. PROG_ActiveSBL

Purpose	UDS callback for ActiveSBL.	
Synopsis	<code>tUdsStatus PROG_ActiveSBL (PduLengthType * pulLen , u8 * aubUdsData);</code>	
Parameters (in,out)	pulLen	pointer on data length
	aubUdsData	pointer on data
Return Value	UDS status	
	UDS_ACK	positive response
	UDS_NRC_xxx	negative response
	UDS_NRC_31	Input pointer parameters NULL value
Description	This function handles the ActiveSBL UDS request. It shall be configured in UDS Tresos Studio plugin for the request RC (0x31 0x01 0x03 0x01) with this exact name and with Callback_Origin set to OTHER.	

4.6.2.4.3. PROG_ActiveSBL_Check

Purpose	Check if SBL is valid and compatible.	
Synopsis	<code>tProgStatus PROG_ActiveSBL_Check (void);</code>	
Return Value	Result of check	
	PROG_E_OK	SBL is valid and compatible with PBL
	PROG_E_NOT_OK	SBL is not valid and/or incompatible with PBL



4.6.2.4.4. PROG_AnswerSuccessiveCheckMemoryRequests

Purpose	Called in CHECK_MEMORY_FINISH state.
Synopsis	<code>void PROG_AnswerSuccessiveCheckMemoryRequests (void);</code>

4.6.2.4.5. PROG_AutoControl

Purpose	UDS callback for AutoControl.	
Synopsis	<code>tUdsStatus PROG_AutoControl (PduLengthType * pulLen , u8 * aubUdsData);</code>	
Parameters (in,out)	pulLen	pointer on data length
	aubUdsData	pointer on data
Return Value	UDS status	
	UDS_ACK	positive response
	UDS_NRC_xxx	negative response
	UDS_NRC_31	Input pointer parameters NULL value
Description	This function handles the AutoControl UDS request. It shall be configured in UDS Tre-sos Studio plugin for the request RC of autocontrol of the application with this exact name and with Callback_Origin set to OTHER.	

4.6.2.4.6. PROG_AutoControl_Process

Purpose	manage asynchronous autocontrol
Synopsis	<code>void PROG_AutoControl_Process (void);</code>

4.6.2.4.7. PROG_BckdManage

Purpose	Manage function to be called as fast as possible to perform background actions.
Synopsis	<code>void PROG_BckdManage (void);</code>

4.6.2.4.8. PROG_CRC

Purpose	
Synopsis	<code>tProgStatus PROG_CRC (void);</code>



Return Value	
--------------	--

4.6.2.4.9. PROG_CRC_Compare

Purpose	Compare the expected and the calculated CRCs.	
Synopsis	<code>tProgStatus PROG_CRC_Compare (void);</code>	
Return Value	Result of treatment	
	PROG_E_OK	Compare OK
	PROG_E_BUSY	Compare in progress
	PROG_E_NOT_OK	Compare finished on error

4.6.2.4.10. PROG_CalcCrc16

Purpose	Called to calculate CRC.	
Synopsis	<code>void PROG_CalcCrc16 (const u8 * aubCrcData , u32 ulReadLength , u16 * uwCrcValue);</code>	
Parameters (in)	aubCrcData	Data to add in the CRC calculation
	ulReadLength	Data length to add in the CRC calculation
Parameters (out)	uwCrcValue	Pointer to variable where to set the CRC result

4.6.2.4.11. PROG_CalcCrc32

Purpose	Called to calculate CRC.	
Synopsis	<code>void PROG_CalcCrc32 (const u8 * aubCrcData , u32 ulReadLength , u32 * ulCrcValue);</code>	
Parameters (in)	aubCrcData	Data to add in the CRC calculation
	ulReadLength	Data length to add in the CRC calculation
Parameters (out)	ulCrcValue	Pointer to variable where to set the CRC result

4.6.2.4.12. PROG_CheckDecompHeaderStatus

Purpose	
---------	--



Synopsis	<code>void PROG_CheckDecompHeaderStatus (void);</code>
-----------------	--

4.6.2.4.13. PROG_CheckMemory

Purpose	UDS callback for CheckMemory.	
Synopsis	<code>tUdsStatus PROG_CheckMemory (PduLengthType * pulLen , u8 * aubUdsData);</code>	
Parameters (in,out)	pulLen	pointer on data length
	aubUdsData	pointer on data
Return Value	UDS status	
	UDS_ACK	positive response
	UDS_NRC_XXX	negative response
	UDS_NRC_31	Input pointer parameters NULL value
Description	This function handles the CheckMemory UDS request. It shall be configured in UDS Tresos Studio plugin for the request XXXXXX with this exact name and with Callback_Origin set to OTHER.	

4.6.2.4.14. PROG_CheckPartialSegmentListCrc

Purpose	This API allow to perform the partial software CRC computation over all programmed segment.
Synopsis	<code>void PROG_CheckPartialSegmentListCrc (void);</code>

4.6.2.4.15. PROG_CheckPartialSwCvnStatus

Purpose	This API allow to perform the partial software CVN check.	
Synopsis	<code>tProgStatus PROG_CheckPartialSwCvnStatus (tProgStatus eProgStatus);</code>	
Return Value		

4.6.2.4.16. PROG_CheckProgRequest

Purpose	API that check if a programming request has been received by the application.
----------------	---



Synopsis	u8 PROG_CheckProgRequest (void);	
Return Value	Result of check	
	PROG_BOOT_REPROG	Reprogramming request has been received
	PROG_BOOT_NO_REPROG	No reprogramming request received
Description	<p>Callback is called: at Bootloader startup to know if a programming request has been received in Application</p> <p>Callback shall implement: get information from application if a programming request has been received (e.g: read a flag from noinit RAM shared between Bootloader and Application)</p>	

4.6.2.4.17. PROG_CheckProgrammingCounter

Purpose	API for checking the programming counter.	
Synopsis	tProgStatus PROG_CheckProgrammingCounter (u8 ubBlockId);	
Parameters (in)	ubBlockId	BlockID
Return Value		

4.6.2.4.18. PROG_CheckProgrammingDependencies

Purpose	UDS callback for CheckProgrammingDependencies.	
Synopsis	tUdsStatus PROG_CheckProgrammingDependencies (PduLengthType * pulLen , u8 * aubUdsData);	
Parameters (in,out)	pulLen	pointer on data length
	aubUdsData	pointer on data
Return Value	UDS status UDS_ACK positive response UDS_NRC_XXX negative response UDS_NRC_31 Input pointer parameters NULL value	
Description	This function handles the CheckProgrammingDependencies UDS request. It shall be configured in UDS Tresos Studio plugin for the request RC CheckProgrammingDependencies (0x31 0x01 0xFF 0x01) with this exact name and with Callback_Origin set to OTHER.	



4.6.2.4.19. PROG_CheckProgrammingPreCondition

Purpose	Function providing the programming pre-conditions check status.
Synopsis	tProgStatus PROG_CheckProgrammingPreCondition (void);
Return Value	Programming Pre-conditions check Status

4.6.2.4.20. PROG_CheckProgrammingRequest

Purpose	Check programming request.
Synopsis	tProgBoolean PROG_CheckProgrammingRequest (void);
Return Value	Programming Request Check Status

4.6.2.4.21. PROG_CheckSegmentListCrc

Purpose	API for CRC calculation on all downloaded segments.
Synopsis	void PROG_CheckSegmentListCrc (void);

4.6.2.4.22. PROG_CheckValidAppl

Purpose	
Synopsis	tProgBoolean PROG_CheckValidAppl (void);
Return Value	

4.6.2.4.23. PROG_CheckValidateApplicationFailed

Purpose	Send response upon failure of CheckValidateApplication.
Synopsis	void PROG_CheckValidateApplicationFailed (void);
Description	This function is called when Check Valid Application is received and in the event of a failure operation completion of CheckValidateApplication

4.6.2.4.24. PROG_CheckValidateApplicationFinish

Purpose	Send response upon successful completion of Compatibility check.
----------------	--



Synopsis	<code>void PROG_CheckValidateApplicationFinish (void);</code>
Description	This function is called when Check Valid Application is received and in the event of a successful operation completion of Compatibility check

4.6.2.4.25. PROG_Check_Prg_Dep_Check

Purpose	Called to verify if CheckProgDependencies routine can be executed.	
Synopsis	<code>tProgStatus PROG_Check_Prg_Dep_Check (void);</code>	
Return Value	Result of check	
	<code>PROG_E_OK</code>	check is allowed
	<code>PROG_E_NOT_OK</code>	check is not allowed

4.6.2.4.26. PROG_CloseExtendedSession

Purpose	request to close the extended session	
Synopsis	<code>void PROG_CloseExtendedSession (tUdsChangeReason eUdsChangeReason);</code>	
Parameters (in)	<code>eUdsChangeReason</code>	reason why the close is requested
Description	This function is called to request the close of the extended session	

4.6.2.4.27. PROG_CloseProgrammingSession

Purpose	request to close the programming session	
Synopsis	<code>void PROG_CloseProgrammingSession (tUdsChangeReason eUdsChangeReason);</code>	
Parameters (in)	<code>eUdsChangeReason</code>	reason why the close is requested
Description	This function is called to request the close of the programming session	

4.6.2.4.28. PROG_CommunicationControl

Purpose	UDS callback for CommunicationControl.
----------------	--



Synopsis	<code>tUdsStatus PROG_CommunicationControl (PduLengthType * pulLen , u8 * aubUdsData);</code>	
Parameters (in,out)	pulLen	pointer on data length
	aubUdsData	pointer on data
Return Value	UDS status	
	UDS_ACK	positive response
	UDS_NRC_xxx	negative response
	UDS_NRC_31	Input pointer parameters NULL value
Description	This function handles the CommunicationControl UDS request. It shall be configured in UDS Tresos Studio plugin for the request CC (0x23) with this exact name and with Callback_Origin set to OTHER.	

4.6.2.4.29. PROG_ComputeBlockHash

Purpose	Called to calculate the hash of the logical block requested.
Synopsis	<code>tProgStatus PROG_ComputeBlockHash (void);</code>
Return Value	

4.6.2.4.30. PROG_ComputeMessageDigest

Purpose	
Synopsis	<code>void PROG_ComputeMessageDigest (void);</code>

4.6.2.4.31. PROG_ControlDTCSetting

Purpose	UDS callback for ControlDTCSetting.	
Synopsis	<code>tUdsStatus PROG_ControlDTCSetting (PduLengthType * pulLen , u8 * aubUdsData);</code>	
Parameters (in,out)	pulLen	pointer on data length
	aubUdsData	pointer on data
Return Value	UDS status	
	UDS_ACK	positive response
	UDS_NRC_xxx	negative response



	UDS_NRC_31	Input pointer parameters NULL value
Description	This function handles the ControlDTCSetting UDS request. It shall be configured in UDS Tresos Studio plugin for the request CDTs (0x14) with this exact name and with Callback_Origin set to OTHER.	

4.6.2.4.32. PROG_CopySBATicket

Purpose	Copy the SBA ticket to the provided RAM buffer.	
Synopsis	<code>tProgStatus PROG_CopySBATicket (u8 * pubRamBuffer);</code>	
Parameters (out)	pubRamBuffer	pointer to a RAM buffer where to copy the SBA ticket
Return Value		Result
	PROG_E_OK	Copy ok
	PROG_E_NOT_OK	Copy failed
Description	<p>Callback is called: On Bootloader startup during SBA check</p> <p>Callback shall implement: the reading from non volatile memory of the SBA ticket (822 bytes long, starting with the data type)</p>	

4.6.2.4.33. PROG_CsmCancelActiveJobs

Purpose	Cancels all the Csm jobs before start of a new Crypto ASR 4.3 job.	
Synopsis	<code>Csm_ReturnType PROG_CsmCancelActiveJobs (void);</code>	
Return Value		
Description	This function is called to cancel all the Csm jobs before the start of a new crypto operation.	

4.6.2.4.34. PROG_CsmCheckResult

Purpose	Evaluates the Csm status returned to the Prog module.
Synopsis	<code>void PROG_CsmCheckResult (Csm_ReturnType eCsmStatus);</code>
Description	This function is called to cancel all the Csm jobs before the start of a new crypto operation.



4.6.2.4.35. PROG_CsmManage

Purpose	Manages the asynchronous crypto operation.
Synopsis	<code>void PROG_CsmManage (void);</code>
Description	This function is called to manage the asynchronous crypto operation for the Crypto ASR v4.3 stack integration.

4.6.2.4.36. PROG_CsmSetPreConditions

Purpose	Set the preconditions before start of a new Crypto ASR 4.3 job.
Synopsis	<code>Csm_ReturnType PROG_CsmSetPreConditions (const u8 * ubKeyPtr , u8 ubProgCsmJobTabId);</code>
Return Value	
Description	This function is called in the Prog module before the start of the crypto operation.

4.6.2.4.37. PROG_CustCheckProgPrecond

Purpose	Check if all the programming pre-conditions are met.		
Synopsis	<code>tProgStatus PROG_CustCheckProgPrecond (tUdsStatus * ubDiagStatus);</code>		
Return Value	state		
	PROG_E_OK	All the programming pre-conditions are met	
	PROG_E_NOT_OK	At least one programming pre-condition is NOT met	
Description	<p>Callback is called: On Programming precondition check</p> <p>Callback shall implement: programming precondition check. Inform Bootloader if the ECU is in a state where programming can be performed. If condition are not correct programming will be rejected by Bootloader and negative response will be sent on the network.</p>		

4.6.2.4.38. PROG_CustCheckProgPrecondList

Purpose	Check if all the programming pre-conditions are met.
----------------	--



Synopsis	<code>void PROG_CustCheckProgPrecondList (u8 * pubProgrammingConditionNumber , u8 * paubConditionList);</code>	
Parameters (out)	pubProgrammingConditionNumber	Number of failed Programming Conditions that shall be returned in the response
	paubConditionList	List of failed conditions
Description	<p>Callback is called: On Programming precondition check</p> <p>Callback shall implement: programming precondition check. Inform Bootloader if the ECU is in a state where programming can be performed. If condition are not correct programming will be rejected by Bootloader and negative response will be sent on the network.</p>	

4.6.2.4.39. PROG_CustomCalcInactiveBankReadAddr

Purpose	Calculate the read address on inactive memory bank.	
Synopsis	<code>u32 PROG_CustomCalcInactiveBankReadAddr (u32 ulAddr);</code>	
Parameters (in)	ulAddr	Address on active memory bank
Return Value	Calculated address on inactive bank	

4.6.2.4.40. PROG_CustomCalcInactiveBankWriteAddr

Purpose	Calculate the write address on inactive memory bank.	
Synopsis	<code>u32 PROG_CustomCalcInactiveBankWriteAddr (u32 ulAddr);</code>	
Parameters (in)	ulAddr	Write address on active memory bank
Return Value	Calculated address on inactive memory bank	

4.6.2.4.41. PROG_CustomCheckCertificateVerification

Purpose	Get the result of the certificate verification.	
Synopsis	<code>tProgStatus PROG_CustomCheckCertificateVerification (void);</code>	
Return Value	eProgStatus success of the certificate verification	
	PROG_E_OK	Certificate verification passed
	PROG_E_NOT_OK	Certificate verification failed
Description	Callback is called: On checkMemory request	



Callback shall implement: the certificate verification or its result

4.6.2.4.42. PROG_CustomCheckCompatibilityId

Purpose	Callback used to check the CompatibilityId.	
Synopsis	<code>tProgStatus PROG_CustomCheckCompatibilityId (u8 * pubCompatibilityIdAddress , u8 ubLen);</code>	
Parameters (in)	pubCompatibilityIdAddress	CompatibilityId address
	ubLen	CompatibilityId length
Return Value	return status	
	PROG_E_OK	
	PROG_E_NOT_OK	

4.6.2.4.43. PROG_CustomCheckRollbackId

Purpose	Callback used to check the RollbackId.	
Synopsis	<code>tProgStatus PROG_CustomCheckRollbackId (u8 * pubRollbackIdAddress , u8 ubLen);</code>	
Parameters (in)	pubRollbackIdAddress	RollbackId address
	ubLen	RollbackId length
Return Value	return status	
	PROG_E_OK	
	PROG_E_NOT_OK	

4.6.2.4.44. PROG_CustomCheckSigningInfo

Purpose	Callback used to check the SigningInfo (SigningName and SigningKeyIdentifier).	
Synopsis	<code>tProgStatus PROG_CustomCheckSigningInfo (u8 * pubSigningInfoAddress , u8 ubLen);</code>	
Parameters (in)	pubSigningInfoAddress	SigningInfo address
	ubLen	SigningInfo length
Return Value	return status	
	PROG_E_OK	



	PROG_E_NOT_OK	
--	---------------	--

4.6.2.4.45. PROG_CustomCheckTargetName

Purpose	Callback used to check the TargetName.	
Synopsis	<code>tProgStatus PROG_CustomCheckTargetName (u8 * pubTargetNameAddress , u8 ubLen);</code>	
Parameters (in)	pubTargetNameAddress	TargetName address
	ubLen	TargetName length
Return Value	return status	
	PROG_E_OK	
	PROG_E_NOT_OK	

4.6.2.4.46. PROG_CustomCheckUuid

Purpose	Callback used to check the Uuid.	
Synopsis	<code>tProgStatus PROG_CustomCheckUuid (u8 * pubUuidAddress , u8 ubLen);</code>	
Parameters (in)	pubUuidAddress	Uuid address
	ubLen	Uuid length
Return Value	return status	
	PROG_E_OK	
	PROG_E_NOT_OK	

4.6.2.4.47. PROG_CustomCheckZIAvailableSpace

Purpose	Checks the available space for writing ZI.	
Synopsis	<code>tProgStatus PROG_CustomCheckZIAvailableSpace (void);</code>	
Return Value	Result of the check	
	PROG_E_OK	enough space is available for writing ZI
	PROG_E_NOT_OK	space not sufficient for writing ZI
Description	<p>Callback is called: during transfer data of log zone writing</p> <p>Callback shall implement: verification if enough space is available for writing log zone (ZI)</p>	



4.6.2.4.48. PROG_CustomChecksumCalc

Purpose	Get result of checksum calculation.	
Synopsis	<code>tProgStatus PROG_CustomChecksumCalc (u16 * puwCalculatedCks);</code>	
Parameters (out)	puwCalculatedCks	pointer to calculated checksum
Return Value	state	
	PROG_E_OK	Calculation finished successfully
	PROG_E_BUSY	Calculation in progress
	PROG_E_NOT_OK	Calculation finished on error
Description	<p>Callback is called: To get result of a checksum calculation, after PROG_CustomStartChecksumCalc/PROG_CustomUpdateChecksumCalc calls</p> <p>Callback shall implement: Provide result of checksum calculation. Checksum calculation is customer specific (Checksum, CRC16,...)</p>	

4.6.2.4.49. PROG_CustomClearPartProgSegList

Purpose	This API allows the integrator to clear information stored in the non-volatile memory regarding the partial software download for a specific block. The callback is called at every block erase.	
Synopsis	<code>void PROG_CustomClearPartProgSegList (u8 ubBlockId);</code>	
Parameters (in)	ubBlockId	block Id for which the information is stored

4.6.2.4.50. PROG_CustomCoherencyCheck

Purpose	This API is called to do the coherency check treatment.	
Synopsis	<code>tProgStatus PROG_CustomCoherencyCheck (tProgCohChkResult * eCohChkResult);</code>	
Parameters (out)	eCohChkResult	The result of the coherency check. Can be: 0-correct, 1-incorrect, 2-incorrect error SW-HW, 3-incorrect error SW-SW, 4-incorrect other error
Return Value	Coherency Check result	
	PROG_E_OK	when the check has finished



	PROG_E_BUSY	if the check is on going
Description	<p>Callback is called: on the coherency check request reception</p> <p>Callback shall implement: the algorithm performing the coherency check of the previously programmed blocks (E.g.: checking of blocks versions compatibility).</p>	

4.6.2.4.51. PROG_CustomCompatibilityCheck

Purpose	This Callback is called to do the Compatibility check treatment.	
Synopsis	<code>tProgStatus PROG_CustomCompatibilityCheck (void);</code>	
Return Value	state	
	PROG_E_OK	Treatment finish successfully
	PROG_E_NOT_OK	Error happened during treatment
	PROG_E_BUSY	if the check is on going
Description	<p>Callback is called: on reception of Check Valid Application request after successful verification of downloaded software to perform the compatibility Checks</p> <p>Callback shall implement: The algorithm performing the compatibility check of the previously programmed blocks (E.g.: checking of blocks versions compatibility) and set the application validity Status. (checks of All blocks previously programmed are valid, SW-HW mismatch, SW-SW mismatch, other error)</p>	

4.6.2.4.52. PROG_CustomComputeCoherencyPreCheck

Purpose	This API is called to perform the Pre Check Coherency on data passed in parameter.	
Synopsis	<code>tProgCohPreChkResult PROG_CustomComputeCoherencyPreCheck (u8 * paubUdsData , PduLengthType pulLen , tUdsStatus * eUdsStatus);</code>	
Parameters (in)	paubUdsData	Pointer to the data on which Pre Check shall be performed
	pulLen	Number of bytes of the data
Parameters (out)	eUdsStatus	The result of the pre check coherency. Can be: UDS_ACK, UDS_NRC_XX (XX can take the values defined in ISO 14429 - Annex A - Negative Response codes)
Return Value	Coherency Pre Check Status Record	
	0-No	Failure



	1-Internal	Error
	2-Error	HW-SW
	3-Error	SW-SW
Description	<p>Callback is called: on the reception of the Pre Check Programming Dependencies Routine</p> <p>Callback shall implement the algorithm performing the Pre Check Programming Dependencies.</p>	

4.6.2.4.53. PROG_CustomCsmStrtPreproc

Purpose	This Optional Callback used for the Custom Csm Implementation, called before the Csm Start mode of operation.	
Synopsis	<pre>Csm_ReturnType PROG_CustomCsmStrtPreproc (const u8 ** ubKeyPtr , u32 ulCsmJobId);</pre>	
Parameters (in)	ulCsmJobId	is the Csm Job ID for which the key pointer could be updated or read
Parameters (in,out)	ubKeyPtr	is the pointer to the address of the key buffer
Return Value	state	
	E_OK	Treatment finished successfully
	E_NOT_OK	Error happened during treatment
Description	<p>Callback is called: Before the start of Csm operation, after all successful job cancel operations(if cancellation is enabled)</p> <p>Callback shall implement: the Custom operation such as Key DER encoding or set IV or just the notification about the start of the crypto operation or any custom operation needed for the Integration.</p>	

4.6.2.4.54. PROG_CustomCvnVerification

Purpose	Callback for CVN check.	
Synopsis	<code>tProgStatus PROG_CustomCvnVerification (u8 ubLogicalBlockId , const u8 * paubExpectedCvn);</code>	
Parameters (in)	ubLogicalBlockId	Block identifier value
	paubExpectedCvn	CVN value



Return Value	state
	PROG_E_OK
	PROG_E_BUSY
	PROG_E_NOT_OK
Description	<p>Callback is called: on reception of verify partial sw</p> <p>Callback shall implement: the check of CVN value</p>

4.6.2.4.55. PROG_CustomCvnVerificationStatus

Purpose	Callback for CVN check.		
Synopsis	<code>tProgStatus PROG_CustomCvnVerificationStatus (void);</code>		
Return Value	state		
	PROG_E_OK	Treatment finish successfully	
	PROG_E_BUSY	Treatment is in progress	
	PROG_E_NOT_OK	Error happened during treatment	
Description	<p>Callback is called: on reception of verify partial sw</p> <p>Callback shall implement: the return of CVN status check updated in the verification callback</p>		

4.6.2.4.56. PROG_CustomDecryptData

Purpose	Callback that shall request data decryption before writing them to memory Callback is called: receiving a TransferData before the decompression (if activated).	
Synopsis	<code>tProgStatus PROG_CustomDecryptData (u8 ubEncryptionMethod , u8 * pubData , PduLengthType DataLength);</code>	
Parameters (in)	ubEncryptionMethod	Encrypting method indicator (from RequestDownload dataFormatIdentifier field)
	DataLength	received data length
Parameters (in,out)	pubData	received data pointer (points to the encrypted data and callback implementation shall copy decrypted data at the same location than the encrypted one.)
Return Value	Result of the decryption	



	PROG_E_OK	Decryption finish successfully
	PROG_E_NOT_OK	Error happened during decryption
Description	Callback shall implement: If needed, it shall decrypt the received data according to the EncryptingMethod.	

4.6.2.4.57. PROG_CustomDecryptGetInitVector

Purpose	Get initialization vector and its length for decryption.	
Synopsis	<code>void PROG_CustomDecryptGetInitVector (const u8 ** pubInitVect, u32 * pulInitVectLength);</code>	
Parameters (out)	pubInitVect	pointer to IV array location
	pulInitVectLength	to IV length location
Description	Callback shall implement: Retrieve the IV and its length for decryption	

4.6.2.4.58. PROG_CustomDownloadNotification

Purpose	Notification of a download event.	
Synopsis	<code>tProgStatus PROG_CustomDownloadNotification (u32 ulStartAddress , u32 ulMemorySize);</code>	
Parameters (in)	ulStartAddress	received start address value
	ulMemorySize	received memory size value
Return Value	eProgStatus success of the operation	
	PROG_E_OK	
	PROG_E_NOT_OK	
Description	<p>Callback is called: On reception of RequestDownload routine</p> <p>Callback shall implement: provide information that the Flash Memory was programmed and should be erased before a new writing and customer specific implementation on download event</p>	

4.6.2.4.59. PROG_CustomGetAsymPublicKey

Purpose	Get the public key modulus and exponent when using asymmetric cryptography. Used in SA_InitCrypto.
----------------	--



Synopsis	<code>void PROG_CustomGetAsymPublicKey (const u8 ** paubPublicModulus , u32 * pulPublicExponent);</code>	
Parameters (out)	paubPublicModulus	Pointer to asymmetric cryptography public key modulus array
	pulPublicExponent	Pointer to asymmetric cryptography public key exponent

4.6.2.4.60. PROG_CustomGetComputedBootloaderChecksum

Purpose	This API is called to get the computed Bootloader checksum stored in non-volatile memory.	
Synopsis	<code>void PROG_CustomGetComputedBootloaderChecksum (u8 * pubComputedChecksum);</code>	
Parameters (out)	pubComputedChecksum	Pointer where to copy the checksum.
Description	<p>Callback is called: Before starting Bootloader to verify the computed checksum</p> <p>Callback shall implement: get from non-volatile memory the Bootloader checksum and copy it to the provided pointer</p>	

4.6.2.4.61. PROG_CustomGetDownloadedSegmentSize

Purpose	Retrieve the size of the latest downloaded segments.	
Synopsis	<code>void PROG_CustomGetDownloadedSegmentSize (u32 * pulSegSize , u8 ubSegmentId);</code>	
Parameters (in)	ubSegmentId	segment ID in prog plugin configuration
Parameters (out)	pulSegSize	address to the buffer to fill
Description	<p>Callback is called: upon receiving a request download for application and/or calibration</p> <p>Callback shall implement: retrieving of size of the downloaded segments</p>	

4.6.2.4.62. PROG_CustomGetECUStatus

Purpose	Callback getting the ECU status value.
Synopsis	<code>void PROG_CustomGetECUStatus (u8 * pubEcuStatus);</code>



Parameters (in,out)	pubEcuStatus	Current ECU status
Description	Callback is called: at bootloader start up Callback shall implement: Gets the value of the ECU Status	

4.6.2.4.63. PROG_CustomGetEcuid

Purpose	API to be called in order to get the ECU Id from a custom location.	
Synopsis	<code>void PROG_CustomGetEcuid (u8 * paubEcuid);</code>	
Parameters (out)	paubEcuid	Pointer to ECU Id
Description	This API is called to get the ECU ID during the TransferData. The implementation of this API shall permit to give the Ecuid to the bootloader from a custom location (Flash, RAM, custom configuration, etc.).	

4.6.2.4.64. PROG_CustomGetEraseStatus

Purpose	Get the erase status of the memory block.							
Synopsis	<code>tProgEraseStatus PROG_CustomGetEraseStatus (u8 ubBlockId);</code>							
Parameters (in)	ubBlockId	ID of the Memory block to be erased						
Return Value	<table border="1"> <tr> <td>state</td> <td></td> </tr> <tr> <td>PROG_E_NOT_ERASED</td> <td></td> </tr> <tr> <td>PROG_E_ERASED</td> <td></td> </tr> </table>		state		PROG_E_NOT_ERASED		PROG_E_ERASED	
state								
PROG_E_NOT_ERASED								
PROG_E_ERASED								
Description	<p>Callback is called: On reception of Erase routine</p> <p>Callback shall implement: provide information if logical block is already erase and that erase shall be skipped</p>							

4.6.2.4.65. PROG_CustomGetExpectedCrc

Purpose	This API is called in order to get the expected CRC corresponding to the required logical block, if this one is not found in the request.
Synopsis	<code>void PROG_CustomGetExpectedCrc (u8 ubLogicalBlockId , u32 * pulExpectedCrc);</code>



Parameters (in)	ubLogicalBlockId	The logical block on which corresponding CRC is required
Parameters (out)	pulExpectedCrc	The expected CRC
Description	Callback is called: During CRC verification Callback shall implement: extract from downloaded software the expected CRC value	

4.6.2.4.66. PROG_CustomGetMacKey

Purpose	
Synopsis	<code>void PROG_CustomGetMacKey (const u8 ** paubKeyData , u32 * pulKeyLength);</code>

4.6.2.4.67. PROG_CustomGetNextSectorAddr

Purpose	Get next sector start address.	
Synopsis	<code>tProgAddressType PROG_CustomGetNextSectorAddr (tProgAddressType uMemAddress);</code>	
Parameters (in)	uMemAddress	Memory address of reference sector
Return Value	Start address of the next sector.	
Description	Callback is called: After an erase operation to set the beginning of the next sector address Callback shall implement: Operation to get the next sector address	

4.6.2.4.68. PROG_CustomGetPartProgSegList

Purpose	This API allows the integrator to store information regarding the partial software download for a specific blockId. The callback is called after each successful CheckMemory routine processing.	
Synopsis	<code>void PROG_CustomGetPartProgSegList (u8 ubBlockId , u32 * aulSegProgSize , u8 * ubSegNbr);</code>	
Parameters (in)	ubBlockId	block Id for which the information is stored
Parameters (out)	aulSegProgSize	pointer to an array containing the size of programmed data for each segment in the block



	ubSegNbr	number of the segments programmed
--	----------	-----------------------------------

4.6.2.4.69. PROG_CustomGetProgCounter

Purpose	API to get the stored value of the programming counter.	
Synopsis	<code>u16 PROG_CustomGetProgCounter (u8 ubBlockId);</code>	
Parameters (in)	ubBlockId	BlockID
Return Value	programming counter on 16 bits	
Description	<p>Callback is called: Before erasing the block</p> <p>Callback shall implement: return the current value of the programming counter</p>	

4.6.2.4.70. PROG_CustomGetResetCause

Purpose	Restore the reset cause and the need of response.	
Synopsis	<code>void PROG_CustomGetResetCause (tProgResCause * pubResetCause , tProgBoolean * pubSendResp);</code>	
Parameters (out)	pubResetCause	pointer to the reset cause
	pubSendResp	Provide information if positive response shall be sent depending of the value of the suppressPositiveResponse bit from the request TRUE: response shall be sent / FALSE: no response shall be sent
Description	<p>Callback is called: At Bootloader startup to get the UDS request that has caused the reset.</p> <p>Callback shall implement: provide the cause of the reset (UDS request) that has been set by application or Bootloader (by call to PROG_CustomStoreResetCause)</p>	

4.6.2.4.71. PROG_CustomGetResumeAddress

Purpose	Get resume address.	
Synopsis	<code>u32 PROG_CustomGetResumeAddress (u8 ubBlockId);</code>	
Parameters (in)	ubBlockId	Index of the logical block



Return Value	
Description	Callback is called: On reception of ReadDataByIdentifier for DID "Reprogramming Resume Information"

4.6.2.4.72. PROG_CustomGetSBLStartAddress

Purpose	Callback performing the read of the SBL software start address.
Synopsis	u32 PROG_CustomGetSBLStartAddress (void);
Return Value	SBL Startup Address
Description	Callback is called: While jumping to SBL to read the SBL startup address to execute it Callback shall implement: jump to SBL start address

4.6.2.4.73. PROG_CustomGetSegmentList

Purpose	Retrieve segment list stored in memory.	
Synopsis	void PROG_CustomGetSegmentList (tSegmentListType * pstSegList);	
Parameters (in,out)	pstSegList	pointer on structure where to copy the data
Description	Callback is called: This function is called in case reprogramming shall be resume in order to get all data that have been previously written for the logical block Callback shall implement: Copy from memory of data that have been previously stored with PROG_CustomStoreSegmentList	

4.6.2.4.74. PROG_CustomGetSymDecryptionKey

Purpose	Get symetrical decryption key and its length.	
Synopsis	void PROG_CustomGetSymDecryptionKey (const u8 ** pubKey , u32 * pulKeyLength);	
Parameters (out)	pubKey	pointer to key array location
	pulKeyLength	pointer to key length location
Description	Callback shall implement: Retrieve the symetrical decryption key and its length	



4.6.2.4.75. PROG_CustomGetTpBsValue

Purpose	Get the TP Blocksize value.	
Synopsis	<code>tProgStatus PROG_CustomGetTpBsValue (u8 * ubTpBsValue);</code>	
Parameters (in,out)	ubTpBsValue	Blocksize value
Return Value	return Get TP value status	
	PROG_E_OK	
	PROG_E_NOT_OK	
Description	<p>Callback is called: On bootloader startup or on demand of reprogramming, in order to get the TP Blocksize value used in application</p> <p>Callback shall implement: The get of the TP Blocksize value used in application. If no valid value is available PROG_E_NOT_OK shall be returned.</p>	

4.6.2.4.76. PROG_CustomGetTpStminValue

Purpose	Get the STmin value.	
Synopsis	<code>tProgStatus PROG_CustomGetTpStminValue (u8 * ubTpStminValue);</code>	
Parameters (in,out)	ubTpStminValue	STmin value
Return Value	Get STmin value status	
	PROG_E_OK	
	PROG_E_NOT_OK	
Description	<p>Callback is called: On bootloader startup or on demand of reprogramming, in order to get the STmin value used in application</p> <p>Callback shall implement: The get of the STmin value used in application. If no valid value is available PROG_E_NOT_OK shall be returned.</p>	

4.6.2.4.77. PROG_CustomGetVerificationParameters

Purpose	Callback to fetch the Signature verification Data start/end address and Signature start address.
Synopsis	<code>void PROG_CustomGetVerificationParameters (u32 * ulStartAddress , u32 * ulEndAddress , u32 * ulSignatureStartAddress);</code>



Parameters (in,out)	ulStartAddress	
	ulEndAddress	
Parameters (out)	ulSignatureStartAddress	Callback shall implement: Updating of the signature verification data start/end and signature start address
Description	Callback is called: during the routine control check programming dependency request	

4.6.2.4.78. PROG_CustomGetWriteJobResult

Purpose	Checks the status of memory writing.	
Synopsis	tProgStatus PROG_CustomGetWriteJobResult (u8 ubLogicalMarker) ;	
Parameters (in)	ubLogicalMarker	logical marker for which the writing status is requested value [0x83 or 0x88]
Return Value	Result of the job	
	PROG_E_OK	the job is finished correctly
	PROG_E_NOT_OK	the job fails
	PROG_E_BUSY	the job is still on-going
Description	Callback is called: during transfer data of application key or the log zone writing Callback shall implement: return the status of memory access after requesting the writing	

4.6.2.4.79. PROG_CustomHsmUpdateBlock

Purpose	ask HSM to update the SecureBoot checksum of a memory block	
Synopsis	tProgStatus PROG_CustomHsmUpdateBlock (u16 uwBlockId , u8 * pubMemoryAddress , u32 ulMemorySize) ;	
Parameters (in)	uwBlockID	
	ulMemorySize	Callback shall implement: call the APIs provided by HSM to update the Secure-Boot checksum of the memory block
Parameters (in,out)	pubMemoryAddress	
Return Value	Result of the job	



	PROG_E_OK	the job is finished correctly
	PROG_E_NOT_OK	the job fails
	PROG_E_BUSY	the job is still on-going
Description	Callback is called: during the routine control check memory, update the MAC value of the given memory block	

4.6.2.4.80. PROG_CustomHsmUpdateFinish

Purpose	notify the user that the SecureBoot checksum of the memory block is updated successfully or not	
Synopsis	<code>void PROG_CustomHsmUpdateFinish (tProgStatus ubJobResult);</code>	
Parameters (in)	ubJobResult	Callback shall implement: keep the job result somewhere if necessary
Description	Callback is called: during the routine control check memory, when the memory updating process is completed.	

4.6.2.4.81. PROG_CustomHsmUpdateInitBlock

Purpose	ask HSM to update the SecureBoot checksum of a memory block	
Synopsis	<code>tProgStatus PROG_CustomHsmUpdateInitBlock (ul6 uwBlockId , u8 * pubMemoryAddress , u32 ulMemorySize);</code>	
Parameters (in)	uwBlockId	
	ulMemorySize	Callback shall implement: call the APIs provided by HSM to Initialize the update operation the SecureBoot checksum of the memory block
Parameters (in,out)	pubMemoryAddress	
Return Value	Result of the job	
	PROG_E_OK	the job is finished correctly
	PROG_E_NOT_OK	the job fails
	PROG_E_BUSY	the job is still on-going
Description	Callback is called: during the routine control check memory, update the MAC value of the given memory block	



4.6.2.4.82. PROG_CustomIncrementProgCounter

Purpose	API to increment the programming counter for the erased logical block.	
Synopsis	<code>tProgStatus PROG_CustomIncrementProgCounter (u8 ubBlockId);</code>	
Parameters (in)	ubBlockId	BlockID
Return Value	Result of incrementation operation	
	PROG_E_OK	incrementation operation finished successfully
	PROG_E_NOT_OK	incrementation operation error happened
Description	<p>This API is called to increment the programming counter of the block. The maximum value shall be limited at 0xFFFF and it shall not overflow.</p> <p>Callback is called: Before erasing the block</p> <p>Callback shall implement: increment the current value of the programming counter</p>	

4.6.2.4.83. PROG_CustomInvalidateBootloaderChecksum

Purpose	This API is called to set the Bootloader checksum as invalid.	
Synopsis	<code>void PROG_CustomInvalidateBootloaderChecksum (void);</code>	
Description	<p>Callback is called: After Bootloader checksum computation, before updating the checksum.</p> <p>Callback shall implement: Set in non-volatile memory the Bootloader checksum validity flag with invalid value</p>	

4.6.2.4.84. PROG_CustomIsBLUDownloadInProgress

Purpose	This API is called to get a flag which indicate if the BLU download is in progress.	
Synopsis	<code>tProgBoolean PROG_CustomIsBLUDownloadInProgress (void);</code>	
Return Value	BLU Download status	
	PROG_TRUE	if BLU Download is in progress
	PROG_FALSE	if BLU Download isn't in progress
Description	<p>Callback is called: during the reset to check if BLU download is in progress, if true resume the BLU download</p> <p>Callback shall implement: read of the flag from a non-volatile memory</p>	



4.6.2.4.85. PROG_CustomIsBLUPatternPresent

Purpose	This API is called to check if the current block/segment is BLUUpdater.	
Synopsis	<code>tProgBoolean PROG_CustomIsBLUPatternPresent (u8 ubLogicalBlockId , u8 ubLogicalSegmentId);</code>	
Parameters (in)	ubLogicalBlockId	The id of the logical block for which the presence of BLU pattern is checked
	ubLogicalSegmentId	The id of the logical segment for which the presence of BLU pattern is checked (ignore if the full logical block is verified)
Return Value	BLU Pattern presence	
	PROG_TRUE	if BLU Pattern is present
	PROG_FALSE	if BLU Pattern is absent
Description	<p>Callback is called: After a successful or unsuccessful block/segment verification of the BLUUpdater</p> <p>Callback shall implement: read on the flashed data the area of the BLU marker and check it if it matches the BLU pattern</p>	

4.6.2.4.86. PROG_CustomIsFirstProgramming

Purpose	Get the status of the Flash memory if it's full erased or not (i.e first download on this ECU).
Synopsis	<code>tProgBoolean PROG_CustomIsFirstProgramming (void);</code>
Return Value	elsFirstProgramming status return by the function (PROG_TRUE / PROG_FALSE)
Description	<p>Callback is called: On reception of Erase routine to skip erasing if memory has never been written</p> <p>Callback shall implement: provide information if this is the first Flash programming</p>

4.6.2.4.87. PROG_CustomIsValidBootloaderChecksum

Purpose	This API is called to know if the stored Bootloader checksum is valid.
Synopsis	<code>tProgBoolean PROG_CustomIsValidBootloaderChecksum (void);</code>



Return Value	Validity status	
	PROG_E_OK	Checksum is valid
	PROG_E_NOT_OK	Checksum is invalid
Description	Callback is called: Before reading the bootloader checksum Callback shall implement: get from non-volatile memory the Bootloader checksum status	

4.6.2.4.88. PROG_CustomMemGetJobStatus

Purpose	Get the status of memory job.	
Synopsis	<code>tProgStatus PROG_CustomMemGetJobStatus (void);</code>	
Return Value	eProgStatus success of the operation(s)	
	PROG_E_OK	
	PROG_E_NOT_OK	
	PROG_E_BUSY	
Description	Callback is called: After each memory access operation Callback shall implement: After PROG_CustomMemoryErase/PROG_CustomMemoryWrite/PROG_CustomMemoryRead returns PROG_E_BUSY, this callback is called periodically until getting a status different from PROG_E_BUSY.	

4.6.2.4.89. PROG_CustomMemoryAccessNotification

Purpose	Notification of memory access to allow customers to place their routines.	
Synopsis	<code>tProgStatus PROG_CustomMemoryAccessNotification (tProgMemType eMemType , tOperationType eOperationType , tProgAddressType uMemAddress , tDataLength ulLength , tDataBufferType PROG_FAR_POINTER paubDataBuffer);</code>	
Parameters (in)	eMemType	Memory type (RAM, Flash and Flash Ext)
	eOperationType	Operation type (Read, write and erase)
	uMemAddress	Start address
	ulLength	Data length
	paubDataBuffer	Data buffer



Return Value	eProgStatus success of the operation(s)	
	PROG_E_OK	
	PROG_E_NOT_OK	
Description	Callback is called: After successful memory data access Callback shall implement: Operation that need to be performed after a memory data access	

4.6.2.4.90. PROG_CustomMemoryErase

Purpose	This API is called to perform an erase operation.	
Synopsis	tProgStatus PROG_CustomMemoryErase (tProgAddressType uMemAddress , tDataLength ulLength);	
Parameters (in)	uMemAddress	Start address
	ulLength	Data length
Return Value	eProgStatus success of the operation(s)	
	PROG_E_OK	
	PROG_E_NOT_OK	
	PROG_E_BUSY	
Description	Callback is called: After request erasing of the custom memory Callback shall implement: Operation to erase the custom memory	

4.6.2.4.91. PROG_CustomMemoryRead

Purpose	This API is called to perform a reading operation.	
Synopsis	tProgStatus PROG_CustomMemoryRead (tProgAddressType uMemAddress , tDataLength ulLength , tDataBufferType PROG_FAR_POINTER paubDataBuffer);	
Parameters (in)	uMemAddress	Start address
	ulLength	Data length
	paubDataBuffer	Data buffer
Return Value	eProgStatus success of the operation(s)	
	PROG_E_OK	



	PROG_E_NOT_OK	
	PROG_E_BUSY	
Description		Callback is called: After request reading data on the custom memory
Callback shall implement: Operation to read data on custom memory		

4.6.2.4.92. PROG_CustomMemoryWrite

Purpose	This API is called to perform a writing operation.	
Synopsis	tProgStatus PROG_CustomMemoryWrite (tProgAddressType uMemAddress , tDataLength ulLength , tDataBufferType PROG_FAR_POINTER paubDataBuffer);	
Parameters (in)	uMemAddress	Start address
	ulLength	Data length
	paubDataBuffer	Data buffer
Return Value	eProgStatus success of the operation(s)	
	PROG_E_OK	
	PROG_E_NOT_OK	
	PROG_E_BUSY	
Description	Callback is called: After request writing data on the custom memory	
	Callback shall implement: Operation to write data on custom memory	

4.6.2.4.93. PROG_CustomReadKeyAppli

Purpose	Allow to read the application key.	
Synopsis	tProgStatus PROG_CustomReadKeyAppli (u16 * aubKeyAppli);	
Parameters (out)	aubKeyAppli	buffer of the application key data to write
Return Value	Result of the job	
	PROG_E_OK	the job is finished correctly
	PROG_E_NOT_OK	the job fails
Description	Callback is called: upon receiving an SA request with sub-function sendKey	
	Callback shall implement: the reading of the application key	



4.6.2.4.94. PROG_CustomSetAppValidity

Purpose	Set the Application Validity.	
Synopsis	<code>void PROG_CustomSetAppValidity (u32 ulAddress , u32 ulEndAddress);</code>	
Parameters (in)	ulAddress	Start address of sector on which the CRC has succeeded
	ulEndAddress	End address of sector on which the CRC has succeeded
Description	<p>Callback is called: After Checksum computation has succeeded</p> <p>Callback shall implement: Update the application validity flag as valid (additional customer coherency check can be required to conclude on application validity)</p>	

4.6.2.4.95. PROG_CustomSetApplicationChecksum

Purpose	This API is called to store the computed Application checksum in non-volatile memory.	
Synopsis	<code>void PROG_CustomSetApplicationChecksum (u8 * pubComputedChecksum , u16 uwBlockIdentifier);</code>	
Parameters (in)	pubComputedChecksum	Pointer to Application checksum to store.
	uwBlockIdentifier	Block Identifier.
Description	<p>Callback is called: Before sending response to CheckMemory request for authenticated block</p> <p>Callback shall implement: store in non-volatile memory the Application checksum (will be compare at next startup with one calculated by application). It shall also update the checksum computed by application.</p>	

4.6.2.4.96. PROG_CustomSetBLUDownloadInProgress

Purpose	This API is called to set a flag which indicate that the BLU donwload is in progress.	
Synopsis	<code>void PROG_CustomSetBLUDownloadInProgress (tProgBoolean ubBLUDownload);</code>	
Parameters (in)	ubBLUDownload	indicating if BLU download is in progress
Description	<p>Callback is called: After a BLU sucessfull download</p> <p>Callback shall implement: set the flag in a non-volatile memory</p>	



4.6.2.4.97. PROG_CustomSetBLUVerificationSuccess

Purpose	Callback used for BLUpdater download, called after comparing the expected CRC and the calculated one.							
Synopsis	<code>tProgStatus PROG_CustomSetBLUVerificationSuccess (tProgBoolean ubCompareSuccess);</code>							
Parameters (in)	ubCompareSuccess	TRUE if the 2 elements of the comparison are identical						
Return Value	<table border="1"> <tr> <td>state</td> <td></td> </tr> <tr> <td><code>PROG_E_OK</code></td><td>Treatment finished successfully</td></tr> <tr> <td><code>PROG_E_NOT_OK</code></td><td>Error happened during treatment</td></tr> </table>		state		<code>PROG_E_OK</code>	Treatment finished successfully	<code>PROG_E_NOT_OK</code>	Error happened during treatment
state								
<code>PROG_E_OK</code>	Treatment finished successfully							
<code>PROG_E_NOT_OK</code>	Error happened during treatment							
Description	<p>Callback is called: After a successful or unsuccessful block/segment verification of the BLUpdater</p> <p>Callback shall implement: update of the logical block validity status</p>							

4.6.2.4.98. PROG_CustomSetBootloaderChecksum

Purpose	This API is called to store the computed Bootloader checksum in non-volatile memory.	
Synopsis	<code>void PROG_CustomSetBootloaderChecksum (u8 * pubComputedChecksum);</code>	
Parameters (in)	pubComputedChecksum	Pointer to Bootloader checksum to store.
Description	<p>Callback is called: After Bootloader checksum computation at Bootloader start</p> <p>Callback shall implement: store in non-volatile memory the Bootloader checksum</p>	

4.6.2.4.99. PROG_CustomSetDownloadVerificationSuccess

Purpose	Callback called after comparing the expected checksum or signature and the calculated one.	
Synopsis	<code>tProgStatus PROG_CustomSetDownloadVerificationSuccess (u8 ubLogicalBlockId , u8 ubLogicalSegmentId , tProgBoolean ubCompareSuccess);</code>	



Parameters (in)	ubLogicalBlockId	The id of the logical block for which the compare was done
	ubLogicalSegmentId	The id of the logical segment for which the compare was done (ignore if the full logical block is verified)
	ubCompareSuccess	TRUE if the 2 elements of the comparison are identical
Return Value	state	
	PROG_E_OK	Treatment finish successfully
	PROG_E_NOT_OK	Error happened during treatment
Description	<p>Callback is called: After a successful or unsuccessful download verification</p> <p>Callback shall implement: update the external global variable m_aubBlockDownloadStatus for the logical block validity status If Dual bank feature is used, this call back can be used to know that a complete, and successful application download happened and therefore allow bank swaps</p>	

4.6.2.4.100. PROG_CustomSetECUStatus

Purpose	Callback setting the ECU status value.	
Synopsis	<code>void PROG_CustomSetECUStatus (u8 ubEcuStatusValue);</code>	
Parameters (in)	ubEcuStatusValue	New ECU status value to be set
Description	<p>Callback is called: at reception of erase memory or end programming</p> <p>Callback shall implement: Update the value of the ECU Status</p>	

4.6.2.4.101. PROG_CustomSetEraseStatus

Purpose	Set the erase status of the memory block.	
Synopsis	<code>tProgStatus PROG_CustomSetEraseStatus (u8 ubBlockId , tProgEraseStatus eEraseStatus);</code>	
Parameters (in)	ubBlockId	ID of the Memory block to be erased
	eEraseStatus	New erase status
Return Value	eProgStatus success of the erase status update	
	PROG_E_OK	



	PROG_E_NOT_OK	
Description	Callback is called: After successful logical block erasing and RequestDownload request reception	
	Callback shall implement: storage of the logical block erase status	

4.6.2.4.102. PROG_CustomSetOpenProgSession

Purpose	Called in order to set the state of ECU in programming session.
Synopsis	<code>void PROG_CustomSetOpenProgSession (void);</code>
Description	Callback is called: during the failure of SBL activation Callback shall implement: customer code that shall be executed to simulate the ECU programming session

4.6.2.4.103. PROG_CustomSetPartProgSegList

Purpose	This API allows the integrator to store information regarding the partial software download for a specific blockId. The callback is called after each successful CheckMemory routine processing.	
Synopsis	<code>void PROG_CustomSetPartProgSegList (u8 ubBlockId , u32 * aulSegProgSize , u8 * ubSegNbr);</code>	
Parameters (in)	ubBlockId	block Id for which the information is stored
	aulSegProgSize	pointer to an array containing the size of programmed data for each segment in the block
	ubSegNbr	number of the segments programmed

4.6.2.4.104. PROG_CustomStartChecksumCalc

Purpose	Initialization of the Custom Checksum calculation.
Synopsis	<code>void PROG_CustomStartChecksumCalc (void);</code>
Description	Callback is called: On start of a checksum calculation Callback shall implement: Initialization of checksum calculation. Checksum calculation is customer specific (Checksum, CRC16,...)



4.6.2.4.105. PROG_CustomStoreDownloadedSegmentSize

Purpose	Stores the size of the latest downloaded segments.	
Synopsis	<code>void PROG_CustomStoreDownloadedSegmentSize (u32 * pulSegSize , u8 ubSegmentId);</code>	
Parameters (in)	pulSegSize	address to the buffer to save in non-volatile memory
	ubSegmentId	segment ID in prog plugin configuration
Description	<p>Callback is called: upon receiving a request transfer exit for application and/or calibration</p> <p>Callback shall implement: storage of the size of the downloaded segments</p>	

4.6.2.4.106. PROG_CustomStoreResetCause

Purpose	Store the reset cause and the need of response.	
Synopsis	<code>void PROG_CustomStoreResetCause (tProgResCause ubResetCause , tProgBoolean ubSendResp);</code>	
Parameters (in)	ubResetCause	the reset cause
	ubSendResp	the need of response according to suppressPositiveResponse bit from the request
Description	<p>Callback is called: Before Bootloader perform a reset to set reset cause (UDS request that has caused the reset).</p> <p>Callback shall implement: storage of the reset cause (UDS request)</p>	

4.6.2.4.107. PROG_CustomStoreResumeAddress

Purpose	Store resume address.	
Synopsis	<code>void PROG_CustomStoreResumeAddress (u8 ubBlockId , u32 ulAddress);</code>	
Parameters (in)	ubBlockId	Index of the logical block
	ulAddress	Address to store
Description	Callback is called: During reprogramming to store resume address	



4.6.2.4.108. PROG_CustomStoreSegmentList

Purpose	Store segment list in memory.	
Synopsis	<pre>void PROG_CustomStoreSegmentList (tSegmentListType * pstSegList);</pre>	
Parameters (in)	pstSegList	address of structure to copy
Description	<p>Callback is called: This function is called on RequestTransferExit to store information about the downloaded segment</p> <p>Callback shall implement: Copy to memory of the segment list</p>	

4.6.2.4.109. PROG_CustomUpdateCertAsymPublicKey

Purpose	Callback performing the update of Software Signing Certificate/Software Signing Public Key for Signature verification.	
Synopsis	<pre>tUdsStatus PROG_CustomUpdateCertAsymPublicKey (u16 * puwLen , u8 * aubUdsData);</pre>	
Parameters (in,out)	puwLen	received length pointer (in : length of request, out : Shall be updated with length of response)
	aubUdsData	received data pointer to Diagnostic buffer (in : Request data, out : Response data)
Return Value	<p>eStatus : Diag response to routine request</p> <p>UDS_ACK - Positive response for routine request</p> <p>UDS_NRC_13 - NRC13 (Incorrect message length) for routine request</p> <p>UDS_NRC_31 - NRC31 (Request out of range) for routine request</p>	
Description	<p>Callback is called: While Software Signing Certificate/Software Signing Public Key is to be updated for Signature verification.</p> <p>Callback shall implement: Call functions for certificate update / Copy public key as below. The 5th byte in the request shall be checked for Type</p> <p>For Type : 01- Software Signing Certificate</p> <p>Length of request : 4bytes of the service + 1 byte for the Type(01) + Certificate Size</p>	



Call functions for certificate update, parsing and verification that shall provide a status (certificate update pass or fail) and also the public Key. If Certificate update passed : copy the retrieved public key to the variable m_aubPublicModulus, update length and status 0x02 for the response If Certificate update failed : update length and status 0x3 for the response.

For Type : 02- Software Signing Public Key

Length of request : 4bytes of the service + 1 byte for the Type(02) + Public Key Size

Copy the public Key to the variable m_aubPublicModulus and the length and respective status shall be updated for the response.

Request of Routine Control : 1. aubUdsData[0]: Shall contain Service ID for Routine Control 2. aubUdsData[1]: Shall contain Sub function for Routine Control 3. aubUdsData[2]: Shall contain Routine Control Identifier (MSB) 4. aubUdsData[3]: Shall contain Routine Control Identifier (LSB) 5. aubUdsData[4]: Shall contain the type parameter for certificate or the public key. 6. aubUdsData[5]: Shall contain first byte certificate data or the public key data : n. aubUdsData[n]: Shall contain last byte certificate data or the public key data

Response of Routine Control (Positive) : 1. aubUdsData[0]: Shall contain Service ID for Routine Control + 0x40 2. aubUdsData[1]: Shall contain Sub function for Routine Control 3. aubUdsData[2]: Shall contain Routine Control Identifier (MSB) 4. aubUdsData[3]: Shall contain Routine Control Identifier (LSB) 5. aubUdsData[4]: Shall contain the byte 0x4A. 6. aubUdsData[5]: Shall contain status of update 0x02 - Update of Certificate/Public Key Accepted/Success or Shall contain status of update 0x03 - Update of Certificate/Public Key Rejected/Failed

4.6.2.4.110. PROG_CustomUpdateChecksumCalc

Purpose	Update (transfer data blocks) for Custom Checksum calculation.	
Synopsis	<code>void PROG_CustomUpdateChecksumCalc (u8 * pubData , u32 ulDataSize) ;</code>	
Parameters (in)	pubData	pointer to the data to compute
	ulDataSize	Length of data to compute
Description	<p>Callback is called: After a PROG_CustomStartChecksumCalc call and when data to be used for checksum calculation have been read from Flash memory</p> <p>Callback shall implement: Checksum calculation. Checksum calculation is customer specific (Checksum, CRC16,...)</p>	



4.6.2.4.111. PROG_CustomUpdateLogSaveMarkingByte

Purpose	Allows to write the log save marking byte.	
Synopsis	<code>tProgStatus PROG_CustomUpdateLogSaveMarkingByte (u8 ubNewLogSaveMarkingByte);</code>	
Parameters (in)	ubNewLogSaveMarkingByte	new value of the log save marking
Return Value	Result of the job	
	PROG_E_OK	the job is finished correctly
	PROG_E_NOT_OK	the job fails
	PROG_E_BUSY	the job is still on-going
Description	<p>Callback is called: if error occurs on writing the log zone (ZI)</p> <p>Callback shall implement: the update of the log zave marking byte into NVM</p>	

4.6.2.4.112. PROG_CustomVDStable_update

Purpose	Callback for the VDS table access.	
Synopsis	<code>tProgStatus PROG_CustomVDStable_update (tVDSLayout * pstVDSLayout , u8 * pubSegmentId , u8 ubVDSaccesstype);</code>	
Parameters (in)	ubVDSaccesstype	is the VDS access type (Read/Update/Clear/fetch ID) specifier
Parameters (in,out)	pstVDSLayout	is the pointer to the VDS buffer
	pubSegmentId	is the pointer to the corresponding Segment ID
Return Value	<p><code>tProgStatus</code> : status of the VDS access</p> <p>PROG_E_OK</p> <p>PROG_E_NOT_OK</p>	
Description	<p>Callback is called: 1. During processing of the UDS Erase or Reset request for clearing/Erasing VDS table 2. During processing of the UDS RTE request for Updating VDS table 3. During processing of the UDS Routine Control for Submit Signature request for Reading VDS table</p> <p>Callback shall be implemented: to store, clear, read and fetch segment Id based on the VDS address of the VDS table.</p>	



4.6.2.4.113. PROG_CustomValidateBootloaderChecksum

Purpose	This API is called to set the Bootloader checksum as valid.
Synopsis	<code>void PROG_CustomValidateBootloaderChecksum (void);</code>
Description	Callback is called: After Bootloader checksum computation, after updating the checksum. Callback shall implement: Set in non-volatile memory the Bootloader checksum validity flag with valid value

4.6.2.4.114. PROG_CustomWriteCRC

Purpose	Callback for CRC storage.	
Synopsis	<code>void PROG_CustomWriteCRC (u32 ulCrcVal);</code>	
Parameters (in)	ulCrcVal	CRC value
Description	Callback is called: After CRC calculation Callback shall implement: storage of the CRC value for further use	

4.6.2.4.115. PROG_CustomWriteKeyAppli

Purpose	Allow to write the application key and perform a byte by byte compare.	
Synopsis	<code>tProgStatus PROG_CustomWriteKeyAppli (u8 * pubData , u8 ubDataLength);</code>	
Parameters (in)	pubData	buffer of the application key data to write
	ubDataLength	length of the application key
Return Value	Result of the job	
	PROG_E_OK	the job is finished correctly
	PROG_E_NOT_OK	the job fails
	PROG_E_BUSY	the job is still on-going
Description	Callback is called: during transfer data of the application key Callback shall implement: the writing of the application key into memory and once it is written perform a comparison byte by byte	



4.6.2.4.116. PROG_CustomWriteProgStatus

Purpose	Callback storing the programming status structure.	
Synopsis	<code>void PROG_CustomWriteProgStatus (u32 ulProgrammingStatus);</code>	
Parameters (in)	ulProgrammingStatus	Programming Status (4 Bytes)
Description	<p>Callback is called: After Programming status update</p> <p>Callback shall implement: Storage of Programming status in RAM. The storage in non-volatile memory shall be done before the ECU is</p>	

4.6.2.4.117. PROG_CustomWriteZI

Purpose	Allow to write the log zone (ZI) and perform a byte by byte compare.							
Synopsis	<code>tProgStatus PROG_CustomWriteZI (u8 * pubData , u8 ubDataLength);</code>							
Parameters (in)	pubData	buffer of the log zone (ZI) data to write						
	ubDataLength	length of the log zone (ZI)						
Return Value	<p>Result of the job</p> <table> <tr> <td>PROG_E_OK</td> <td>the job is finished correctly</td> </tr> <tr> <td>PROG_E_NOT_OK</td> <td>the job fails</td> </tr> <tr> <td>PROG_E_BUSY</td> <td>the job is still on-going</td> </tr> </table>		PROG_E_OK	the job is finished correctly	PROG_E_NOT_OK	the job fails	PROG_E_BUSY	the job is still on-going
PROG_E_OK	the job is finished correctly							
PROG_E_NOT_OK	the job fails							
PROG_E_BUSY	the job is still on-going							
Description	<p>Callback is called: during transfer data of the log zone (ZI)</p> <p>Callback shall implement: the writing of the log zone (ZI) into memory and once it is written perform a comparison byte by byte</p>							

4.6.2.4.118. PROG_DisableECCCheck

Purpose	Callback that shall disable ECC if needed
Synopsis	<code>void PROG_DisableECCCheck (void);</code>
Description	Callback shall implement: If needed, disabling of ECC check Hardware specific)



4.6.2.4.119. PROG_Do_CheckHash

Purpose	
Synopsis	<code>void PROG_Do_CheckHash (void);</code>

4.6.2.4.120. PROG_Do_CheckPrgDependencies

Purpose	Called to calculate the CRC for CheckPrgDependencies routine.
Synopsis	<code>void PROG_Do_CheckPrgDependencies (void);</code>

4.6.2.4.121. PROG_Do_CheckSignature

Purpose	
Synopsis	<code>void PROG_Do_CheckSignature (void);</code>

4.6.2.4.122. PROG_Do_CheckValidateApplication

Purpose	Perform checks for validating application for the requested RC Check Valid Application.
Synopsis	<code>void PROG_Do_CheckValidateApplication (void);</code>
Description	This function is called when Check Valid Application is received, for performing compatibility checks before validating application

4.6.2.4.123. PROG_Do_CoherencyCheck

Purpose	Called to do the Coherency Check treatment.
Synopsis	<code>void PROG_Do_CoherencyCheck (void);</code>

4.6.2.4.124. PROG_Do_CompareKey

Purpose	API called to get seed result.
Synopsis	<code>void PROG_Do_CompareKey (void);</code>



4.6.2.4.125. PROG_Do_DecryptionFinish

Purpose	Called on do of DECRYPT_FINISH state.
Synopsis	<code>void PROG_Do_DecryptionFinish (void);</code>
Description	This function is called to finish the decryption

4.6.2.4.126. PROG_Do_DecryptUpdate

Purpose	Called on do of DECRYPT_UPDATE state.
Synopsis	<code>void PROG_Do_DecryptUpdate (void);</code>
Description	This function is called to launch the writing after decryption has finished

4.6.2.4.127. PROG_Do_GetSeed

Purpose	API called to get seed.
Synopsis	<code>void PROG_Do_GetSeed (void);</code>

4.6.2.4.128. PROG_Do_RTE_Impl80

Purpose	API called on the do of the RTE state.
Synopsis	<code>void PROG_Do_RTE_Impl80 (void);</code>

4.6.2.4.129. PROG_Do_SelfCheck_Impl80

Purpose	API called on the entry of the Self check state.
Synopsis	<code>void PROG_Do_SelfCheck_Impl80 (void);</code>

4.6.2.4.130. PROG_DrvDown_IsFlashRoutinesPresent

Purpose	Returns the value of m_ubFlashRoutinesPresent, that represents the presence of the flash routines in RAM.
----------------	---



Synopsis	<code>tProgBoolean PROG_DrvDown_IsFlashRoutinesPresent (void);</code>	
Return Value	Result of treatment	
	PROG_TRUE	Flash routines are present in RAM
	PROG_FALSE	Flash routines are not present in RAM
Description	This function is called to verify the presence of the flash routines	

4.6.2.4.131. PROG_Dsc01Cbk

Purpose	UDS callback for DiagnosticSessionControlDefault.	
Synopsis	<code>tUdsStatus PROG_Dsc01Cbk (PduLengthType * pulLen , u8 * aubUdsData);</code>	
Parameters (in,out)	pulLen	pointer on data length
	aubUdsData	pointer on data
Return Value	UDS status	
	UDS_ACK	positive response
	UDS_NRC_xxx	negative response
	UDS_NRC_31	Input pointer parameters NULL value
Description	This function handles the DiagnosticSessionControlDefault UDS request. It shall be configured in UDS Tresos Studio plugin for the request DSC (0x10 0x01) with this exact name and with Callback_Origin set to OTHER.	

4.6.2.4.132. PROG_Dsc03Cbk

Purpose	UDS callback for DiagnosticSessionControlExtended.	
Synopsis	<code>tUdsStatus PROG_Dsc03Cbk (PduLengthType * pulLen , u8 * aubUdsData);</code>	
Parameters (in,out)	pulLen	pointer on data length
	aubUdsData	pointer on data
Return Value	UDS status	
	UDS_ACK	positive response
Description	This function handles the DiagnosticSessionControlExtended UDS request. It shall be configured in UDS Tresos Studio plugin for the request DSC (0x10 0x03) with this exact name and with Callback_Origin set to OTHER.	



4.6.2.4.133. PROG_ERASE_Check

Purpose	
Synopsis	tProgStatus PROG_ERASE_Check (void);
Return Value	

4.6.2.4.134. PROG_EcuReset

Purpose	UDS callback for EcuReset.	
Synopsis	tUdsStatus PROG_EcuReset (PduLengthType * pulLen , u8 * aubUdsData);	
Parameters (in,out)	pulLen	pointer on data length
	aubUdsData	pointer on data
Return Value	UDS status	
	UDS_ACK	positive response
	UDS_NRC_xxx	negative response
	UDS_NRC_31	Input pointer parameters NULL value
Description	This function handles the EcuReset UDS request. It shall be configured in UDS Tresos Studio plugin for the request ER (0x11 0x01) with this exact name and with Callback_Origin set to OTHER.	

4.6.2.4.135. PROG_EnableECCCheck

Purpose	Callback that shall enable ECC if needed
Synopsis	void PROG_EnableECCCheck (void);
Description	Callback shall implement: If needed, enabling of ECC check (Hardware specific)

4.6.2.4.136. PROG_Entry_ACT_Check_Impl80

Purpose	API called on the entry of the ACT check state.
----------------	---

**Synopsis**

```
void PROG_Entry_ACT_Check_Impl80 ( void );
```

4.6.2.4.137. PROG_Entry_ActiveSBL

Purpose	Called on entry to ActiveSBL state.
Synopsis	<code>void PROG_Entry_ActiveSBL (void);</code>

4.6.2.4.138. PROG_Entry_Alive

Purpose	Called on entry to Alive state.
Synopsis	<code>void PROG_Entry_Alive (void);</code>

4.6.2.4.139. PROG_Entry_AutoControl

Purpose	Called on entry to AutoControl state.
Synopsis	<code>void PROG_Entry_AutoControl (void);</code>

4.6.2.4.140. PROG_Entry_BLU_Resume

Purpose	Resume BLU Download.
Synopsis	<code>void PROG_Entry_BLU_Resume (void);</code>
Description	This function is called in the state machine to restore the state of bootloader saved before ECU reset in order to resume the BLU download

4.6.2.4.141. PROG_Entry_CheckDependenciesFinish

Purpose	Called to send CheckProgDependencies routine response.
Synopsis	<code>void PROG_Entry_CheckDependenciesFinish (void);</code>

4.6.2.4.142. PROG_Entry_CheckHash

Purpose	
Synopsis	<code>void PROG_Entry_CheckHash (void);</code>



4.6.2.4.143. PROG_Entry_CheckMemory

Purpose	Check memory programming.
Synopsis	<code>void PROG_Entry_CheckMemory (void);</code>

4.6.2.4.144. PROG_Entry_CheckMemoryCompute

Purpose	Check memory programming.
Synopsis	<code>void PROG_Entry_CheckMemoryCompute (void);</code>

4.6.2.4.145. PROG_Entry_CheckMemoryFinish

Purpose	Called on entry to CHECK_MEMORY_FINISH state.
Synopsis	<code>void PROG_Entry_CheckMemoryFinish (void);</code>

4.6.2.4.146. PROG_Entry_CheckMemoryFinish_Impl80

Purpose	API called on the entry of the check memory finish state.
Synopsis	<code>void PROG_Entry_CheckMemoryFinish_Impl80 (void);</code>

4.6.2.4.147. PROG_Entry_ChecksumByRange

Purpose	Init of the Checksum Calculation.
Synopsis	<code>void PROG_Entry_ChecksumByRange (void);</code>

4.6.2.4.148. PROG_Entry_CoherencyPreCheck

Purpose	Called to do the Coherency Pre Check treatment.
Synopsis	<code>void PROG_Entry_CoherencyPreCheck (void);</code>

4.6.2.4.149. PROG_Entry_CompareKey

Purpose	Called on entry to CompareKey state.
Synopsis	<code>void PROG_Entry_CompareKey (void);</code>



4.6.2.4.150. PROG_Entry_CompareKeyCheck

Purpose	Called on entry to CompareKey state.
Synopsis	<code>void PROG_Entry_CompareKeyCheck (void);</code>

4.6.2.4.151. PROG_Entry_DecomphHeader

Purpose	
Synopsis	<code>void PROG_Entry_DecomphHeader (void);</code>

4.6.2.4.152. PROG_Entry_DecryptFinish

Purpose	Called on entry of DECRYPT_FINISH state.
Synopsis	<code>void PROG_Entry_DecryptFinish (void);</code>
Description	This function is called to finish the decryption

4.6.2.4.153. PROG_Entry_DecryptUpdate

Purpose	Called on entry of DECRYPT_UPDATE state.
Synopsis	<code>void PROG_Entry_DecryptUpdate (void);</code>
Description	This function is called to update the decryption of data

4.6.2.4.154. PROG_Entry_DefaultSession

Purpose	Called on entry to DefaultSession state.
Synopsis	<code>void PROG_Entry_DefaultSession (void);</code>

4.6.2.4.155. PROG_Entry_EcuReset

Purpose	Called on entry to EcuReset state.
Synopsis	<code>void PROG_Entry_EcuReset (void);</code>



4.6.2.4.156. PROG_Entry_Erase

Purpose	Called on entry to Erase state.
Synopsis	<code>void PROG_Entry_Erase (void);</code>

4.6.2.4.157. PROG_Entry_EraseCheck

Purpose	Called on entry to EraseCheck state.
Synopsis	<code>void PROG_Entry_EraseCheck (void);</code>

4.6.2.4.158. PROG_Entry_EraseFinish

Purpose	Called on entry to EraseFinish state.
Synopsis	<code>void PROG_Entry_EraseFinish (void);</code>

4.6.2.4.159. PROG_Entry_EraseNRC78

Purpose	Called on entry to EraseNRC78 state.
Synopsis	<code>void PROG_Entry_EraseNRC78 (void);</code>

4.6.2.4.160. PROG_Entry_EraseTransmitNRC78

Purpose	Called on entry to EraseTransmitNRC78 state.
Synopsis	<code>void PROG_Entry_EraseTransmitNRC78 (void);</code>

4.6.2.4.161. PROG_Entry_ExtendedSession

Purpose	Called on entry to ExtendedSession state.
Synopsis	<code>void PROG_Entry_ExtendedSession (void);</code>

4.6.2.4.162. PROG_Entry_GetSeed

Purpose	Called on entry to GetSeed state.
----------------	-----------------------------------



Synopsis	<code>void PROG_Entry_GetSeed (void);</code>
----------	--

4.6.2.4.163. PROG_Entry_GetSeedCheck

Purpose	Called on entry to GetSeed state.
Synopsis	<code>void PROG_Entry_GetSeedCheck (void);</code>

4.6.2.4.164. PROG_Entry_HSMUpdate_TDFinish

Purpose	
Synopsis	<code>void PROG_Entry_HSMUpdate_TDFinish (void);</code>

4.6.2.4.165. PROG_Entry_INIT

Purpose	Called on entry to INIT state.
Synopsis	<code>void PROG_Entry_INIT (void);</code>

4.6.2.4.166. PROG_Entry_LogicalBlockHash

Purpose	Called on Logical Block Hash routine reception.
Synopsis	<code>void PROG_Entry_LogicalBlockHash (void);</code>

4.6.2.4.167. PROG_Entry_PartialVerificationCrc

Purpose	Called on VerifyPartialSoftwareChecksum routine reception.
Synopsis	<code>void PROG_Entry_PartialVerificationCrc (void);</code>

4.6.2.4.168. PROG_Entry_PreInit

Purpose	Called on entry to PreInit state.
Synopsis	<code>void PROG_Entry_PreInit (void);</code>



4.6.2.4.169. PROG_Entry_ProgrammingSession

Purpose	Called on entry to ProgrammingSession state.
Synopsis	<code>void PROG_Entry_ProgrammingSession (void);</code>

4.6.2.4.170. PROG_Entry_RD

Purpose	Called on entry to RD state.
Synopsis	<code>void PROG_Entry_RD (void);</code>

4.6.2.4.171. PROG_Entry_RD_Finish

Purpose	Called on entry to RD Finish state.
Synopsis	<code>void PROG_Entry_RD_Finish (void);</code>

4.6.2.4.172. PROG_Entry_RD_Impl80

Purpose	API called on the entry of the RD state.
Synopsis	<code>void PROG_Entry_RD_Impl80 (void);</code>

4.6.2.4.173. PROG_Entry_RD_Signature

Purpose	Called on entry to RD SIGNATURE state.
Synopsis	<code>void PROG_Entry_RD_Signature (void);</code>

4.6.2.4.174. PROG_Entry_RTE

Purpose	Called on entry to RTE state.
Synopsis	<code>void PROG_Entry_RTE (void);</code>

4.6.2.4.175. PROG_Entry_RTEFailed

Purpose	Called on entry to RTEFailed state.
----------------	-------------------------------------



Synopsis	<code>void PROG_Entry_RTEFailed (void);</code>
----------	--

4.6.2.4.176. PROG_Entry_RTEFinish

Purpose	Called on entry to RTEFinish state.
Synopsis	<code>void PROG_Entry_RTEFinish (void);</code>

4.6.2.4.177. PROG_Entry_RTE_Impl80

Purpose	API called on the entry of the RTE state.
Synopsis	<code>void PROG_Entry_RTE_Impl80 (void);</code>

4.6.2.4.178. PROG_Entry_Reset

Purpose	Called on entry to Reset state.
Synopsis	<code>void PROG_Entry_Reset (void);</code>

4.6.2.4.179. PROG_Entry_ResumeVerification

Purpose	Entry function for Resume verification state.
Synopsis	<code>void PROG_Entry_ResumeVerification (void);</code>

4.6.2.4.180. PROG_Entry_Resume_Finish

Purpose	Entry function for Resume verification state.
Synopsis	<code>void PROG_Entry_Resume_Finish (void);</code>

4.6.2.4.181. PROG_Entry_SblSynch

Purpose	Called on entry to SblSynch state.
Synopsis	<code>void PROG_Entry_SblSynch (void);</code>



4.6.2.4.182. PROG_Entry_SecureChecksumFailed

Purpose	Send response in the case of a failure on secure checksum calculation.
Synopsis	<code>void PROG_Entry_SecureChecksumFailed (void);</code>
Description	This function is called at the end of the Check Memory in the case of a failure on secure checksum calculation

4.6.2.4.183. PROG_Entry_SelfCheck_Impl80

Purpose	API called on the entry of the Self check state.
Synopsis	<code>void PROG_Entry_SelfCheck_Impl80 (void);</code>

4.6.2.4.184. PROG_Entry_SignatureCheck

Purpose	
Synopsis	<code>void PROG_Entry_SignatureCheck (void);</code>

4.6.2.4.185. PROG_Entry_SignatureVerify

Purpose	Perform Signature verification for the requested RC Submit Signature.
Synopsis	<code>void PROG_Entry_SignatureVerify (void);</code>
Description	This function is called when Submit Signature Routine is received, for performing Signature Verification

4.6.2.4.186. PROG_Entry_Sleep

Purpose	Called on entry to Sleep state.
Synopsis	<code>void PROG_Entry_Sleep (void);</code>

4.6.2.4.187. PROG_Entry_Streaming

Purpose	Called on entry to Streaming state.
Synopsis	<code>void PROG_Entry_Streaming (void);</code>



4.6.2.4.188. PROG_Entry_TD

Purpose	Called on entry to TD state.
Synopsis	<code>void PROG_Entry_TD (void);</code>

4.6.2.4.189. PROG_Entry_TD_Failed

Purpose	Called on entry to TD_Failed state.
Synopsis	<code>void PROG_Entry_TD_Failed (void);</code>

4.6.2.4.190. PROG_Entry_TD_Header

Purpose	
Synopsis	<code>void PROG_Entry_TD_Header (void);</code>

4.6.2.4.191. PROG_Entry_TD_Impl80

Purpose	API called on the entry of the TD state.
Synopsis	<code>void PROG_Entry_TD_Impl80 (void);</code>

4.6.2.4.192. PROG_Entry_UpdatePSI

Purpose	Called on entry to UpdatePSI state.
Synopsis	<code>void PROG_Entry_UpdatePSI (void);</code>

4.6.2.4.193. PROG_Entry_ValidateSBASignature

Purpose	
Synopsis	<code>void PROG_Entry_ValidateSBASignature (void);</code>

4.6.2.4.194. PROG_Entry_ValidateSBASignerInfo

Purpose	
Synopsis	<code>void PROG_Entry_ValidateSBASignerInfo (void);</code>



4.6.2.4.195. PROG_Entry_ValidateSignature

Purpose	
Synopsis	<code>void PROG_Entry_ValidateSignature (void);</code>

4.6.2.4.196. PROG_Entry_ValidateSignerInfo

Purpose	
Synopsis	<code>void PROG_Entry_ValidateSignerInfo (void);</code>

4.6.2.4.197. PROG_Entry_Validate_Application

Purpose	called on Entry to RC Check Valid Application
Synopsis	<code>void PROG_Entry_Validate_Application (void);</code>
Description	This function is called at when Check Valid Application Routine is received for integrity checks, essentials SW parts presence

4.6.2.4.198. PROG_Entry_ValidationFailed

Purpose	Send response in the case of a failure on secure checksum calculation/Signature verification.
Synopsis	<code>void PROG_Entry_ValidationFailed (void);</code>
Description	This function is called at when Submit Signature Routine is received and in the event of a failure on secure checksum calculation or Signature Verification

4.6.2.4.199. PROG_Entry_ValidationFinish

Purpose	Send response upon successful completion of secure checksum calculation/Signature verification.
Synopsis	<code>void PROG_Entry_ValidationFinish (void);</code>
Description	This function is called at when Submit Signature Routine is received and in the event of a successful operation completion of secure checksum calculation and/or Signature Verification



4.6.2.4.200. PROG_Entry_Write

Purpose	Called on entry to Write state.
Synopsis	<code>void PROG_Entry_Write (void);</code>

4.6.2.4.201. PROG_Entry_WriteFingerprint

Purpose	Called on entry to WriteFingerprint state.
Synopsis	<code>void PROG_Entry_WriteFingerprint (void);</code>

4.6.2.4.202. PROG_Entry_Write_Impl80

Purpose	API called on the entry of the write state.
Synopsis	<code>void PROG_Entry_Write_Impl80 (void);</code>

4.6.2.4.203. PROG_Erase

Purpose	Called in Erase state.
Synopsis	<code>void PROG_Erase (void);</code>

4.6.2.4.204. PROG_EraseFirstCheck_Impl80

Purpose	API called on the entry of erase state.	
Synopsis	<code>tProgStatus PROG_EraseFirstCheck_Impl80 (void);</code>	
Return Value	Operation status	
	PROG_E_OK	at least calibration and/or application shall be erased
	PROG_E_BYPASS	no erase to perform, areas are blank

4.6.2.4.205. PROG_EraseMemoryRequest

Purpose	UDS callback for EraseMemoryRequest.
----------------	--------------------------------------



Synopsis	<code>tUdsStatus PROG_EraseMemoryRequest (PduLengthType * pulLen , u8 * aubUdsData);</code>	
Parameters (in,out)	pulLen	pointer on data length
	aubUdsData	pointer on data
Return Value	UDS status	
	UDS_ACK	positive response
	UDS_NRC_XXX	negative response
	UDS_NRC_31	Input pointer parameters NULL value
Description	This function handles the EraseMemoryRequest UDS request. It shall be configured in UDS Tresos Studio plugin for the request RC Erase Memory (0x31 0x01 0xFF 0x00) with this exact name and with Callback_Origin set to OTHER.	

4.6.2.4.206. PROG_Erase_Guard

Purpose	Called before going to PROG_Erase state.	
Synopsis	<code>tProgStatus PROG_Erase_Guard (void);</code>	
Return Value	Result of check	
	PROG_E_OK	Erase request accepted
	PROG_E_NOT_OK	Erase request not accepted

4.6.2.4.207. PROG_Exit_CheckMemory

Purpose	Check memory programming.	
Synopsis	<code>void PROG_Exit_CheckMemory (void);</code>	

4.6.2.4.208. PROG_Exit_CheckMemoryFinish

Purpose	Called on exit to CHECK_MEMORY_FINISH state.	
Synopsis	<code>void PROG_Exit_CheckMemoryFinish (void);</code>	

4.6.2.4.209. PROG_Exit_INIT

Purpose	Called on entry to INIT state.	
----------------	--------------------------------	--



Synopsis	<code>void PROG_Exit_INIT (void);</code>
----------	--

4.6.2.4.210. PROG_Exit_PartialVerificationCrc

Purpose	Called on VerifyPartialSoftwareChecksum routine reception.
Synopsis	<code>void PROG_Exit_PartialVerificationCrc (void);</code>

4.6.2.4.211. PROG_Exit_RTE_Decrypt

Purpose	Called on exit to RTE Decrypt state.
Synopsis	<code>void PROG_Exit_RTE_Decrypt (void);</code>

4.6.2.4.212. PROG_Exit_TD_Write

Purpose	Called on exit from TD_Write state.
Synopsis	<code>void PROG_Exit_TD_Write (void);</code>

4.6.2.4.213. PROG_FindBlockIndexInTable

Purpose	Look for block index corresponding to block ID provided in fingerprint.	
Synopsis	<code>tProgStatus PROG_FindBlockIndexInTable (u16 uwBlockIdent , u8 * pubBlockId);</code>	
Parameters (in)	uwBlockIdent	block identifier
Parameters (out)	pubBlockId	index of the matching segment in configuration
Return Value	Result of treatment	
	PROG_E_OK	Index found
	PROG_E_NOT_OK	Index not found

4.6.2.4.214. PROG_FlashPage

Purpose	Called to write one or more flash pages.
---------	--



Synopsis	u8 PROG_FlashPage (u32 ulAddress , u8 ubPages , u32 * pulDataBuffer , u16 uwDataBufferLength);	
Parameters (in)	ulAddress	Address of the first page to write
	ubPages	number of pages to write
	pulDataBuffer	pointer to the buffer of data to write
	uwDataBufferLength	Exact length of data to write
Return Value	Result of the write operation	
	PROG_FLASH_PAGE_E_OK	write is successfull
	PROG_FLASH_PAGE_E_NOT_OK	write is not successfull
Description	This API can be called by the Framework to write one or more specific flash pages.	

4.6.2.4.215. PROG_GetActiveCurrentSession

Purpose	UDS callback for GetActiveCurrentSession.	
Synopsis	tUdsStatus PROG_GetActiveCurrentSession (PduLengthType * pulLen , u8 * aubUdsData);	
Parameters (in,out)	pulLen	pointer on data length
	aubUdsData	pointer on data
Return Value	UDS status	
	UDS_ACK	positive response
	UDS_NRC_xxx	negative response
	UDS_NRC_31	Input pointer parameters NULL value
Description	This function handles the GetActiveCurrentSession UDS request. It shall be configured in UDS Tresos Studio plugin for the request RDBI with this exact name and with Callback_Origin set to OTHER.	

4.6.2.4.216. PROG_GetBlockDowngradeFlagByIndex

Purpose	Get the block downgrade protection flag by using the its index in the configured table.	
Synopsis	tProgBoolean PROG_GetBlockDowngradeFlagByIndex (u8 ubLogicalBlockIndex);	
Parameters (in)	ubLogicalBlockIndex	index of the block identifier in the Prog plugin



Return Value	Block downgrade protection flag value
---------------------	---------------------------------------

4.6.2.4.217. PROG_GetBlockIdByIndex

Purpose	Get the block identifier by using the its index in the configured table.	
Synopsis	<code>u16 PROG_GetBlockIdByIndex (u8 ubLogicalBlockIndex);</code>	
Parameters (in)	ubLogicalBlockIndex	index of the block identifier in the Prog plugin
Return Value	Block identifier	

4.6.2.4.218. PROG_GetComputedBootloaderChecksum

Purpose	API for getting the previously computed Bootloader checksum.	
Synopsis	<code>tProgStatus PROG_GetComputedBootloaderChecksum (u8 * pubComputedChecksum);</code>	
Parameters (out)	pubComputedChecksum	Computed Checksum (if return is PROG_E_OK)
Return Value	Operation status	
	PROG_E_OK	Checksum has been provided
	PROG_E_NOT_OK	Checksum is invalid and cannot be provided

4.6.2.4.219. PROG_GetCurrentDiagApp

Purpose	UDS callback for GetCurrentDiagApp.	
Synopsis	<code>tUdsStatus PROG_GetCurrentDiagApp (PduLengthType * pulLen , u8 * aubUdsData);</code>	
Parameters (in,out)	pulLen	pointer on data length
	aubUdsData	pointer on data
Return Value	UDS status	
	UDS_ACK	positive response
	UDS_NRC_XXX	negative response
	UDS_NRC_31	Input pointer parameters NULL value



Description	This function handles the GetCurrentDiagApp UDS request. It shall be configured in UDS Tresos Studio plugin for the request RDBI with this exact name and with Call-back_Origin set to OTHER.
--------------------	---

4.6.2.4.220. PROG_GetDidF0F3

Purpose	Called by external module to get data of DID F0F3.	
Synopsis	<code>void PROG_GetDidF0F3 (u8 * aubData);</code>	
Parameters (out)	aubData	DID data
Description	This API is called by an external value to get the data of the DID F0F3 (Eculd)	

4.6.2.4.221. PROG_GetDidF0F6

Purpose	Called by external module to get data of DID F0F6.	
Synopsis	<code>void PROG_GetDidF0F6 (u8 * aubData);</code>	
Parameters (out)	aubData	DID data
Description	This API is called by an external value to get the data of the DID F0F6 (Eculd)	

4.6.2.4.222. PROG_GetKeyNBIDValue

Purpose	Callback retrieving the Key NBID value stored in memory.
Synopsis	<code>u16 PROG_GetKeyNBIDValue (void);</code>
Return Value	Key NBID value on 16 bits
Description	Callback is called: during reprogramming process to get the key NBID of the current application stored in NVM. It will be used to know if the new download application can be accepted or not Callback shall implement: reading from non volatile memory of the Key NBID value

4.6.2.4.223. PROG_GetMacKey

Purpose	Get the key used to compute MAC checksum.
----------------	---



Synopsis	<code>void PROG_GetMacKey (Csm_SymKeyType * pstMacKey);</code>	
Parameters (out)	<code>pstMacKey</code>	Pointer to the Key
Description	This function is called to get the key used for MAC computation	

4.6.2.4.224. PROG_GetNBIDValue

Purpose	Callback retrieving the NBID value store in memory.
Synopsis	<code>u16 PROG_GetNBIDValue (void);</code>
Return Value	NBID value on 16 bits
Description	<p>Callback is called: during reprogramming process to get the NBID of the current application stored in NVM.</p> <p>It will be used to know if the new download application can be accepted or not</p> <p>Callback shall implement: reading from non volatile memory of the NBID value</p>

4.6.2.4.225. PROG_GetNetworkStatus

Purpose	Function providing the network status.
Synopsis	<code>u16 PROG_GetNetworkStatus (void);</code>
Return Value	Network Status

4.6.2.4.226. PROG_GetProgCntrLockVal

Purpose	API for getting the programming counter lock value (maximum programming counter) for a specific block.	
Synopsis	<code>tProgStatus PROG_GetProgCntrLockVal (u8 ubBlockId , u16 * puwProgCntrMax);</code>	
Parameters (in)	<code>ubBlockId</code>	BlockID
Parameters (out)	<code>puwProgCntrMax</code>	Programming Counter lock value
Return Value	Operation status	
	<code>PROG_E_OK</code>	Block index has been found
	<code>PROG_E_NOT_OK</code>	Block index has not been found



4.6.2.4.227. PROG_GetPseudoFlashDriverAddress

Purpose	Get the start address of the pseudo flash driver block.
Synopsis	<code>tProgAddressType PROG_GetPseudoFlashDriverAddress (void);</code>
Return Value	Start Address of the pseudo flash driver

4.6.2.4.228. PROG_GetRequestRoutineResult_Impl80

Purpose	API called on processing a routine control.
Synopsis	<code>tUdsStatus PROG_GetRequestRoutineResult_Impl80 (PduLengthType * puwLen , u8 * aubUdsData);</code>
Return Value	UDS status
	UDS_NRC_XXX Negative response code
	UDS_ACK no erase to perform, areas are blank
Description	It allows to get the current routine status

4.6.2.4.229. PROG_GetRequestSeedCounter

Purpose	Get the current Request seed counter value.
Synopsis	<code>u8 PROG_GetRequestSeedCounter (void);</code>
Return Value	value u8 request seed counter to get

4.6.2.4.230. PROG_GetRequestSeedTimer

Purpose	Set the current Request seed counter.
Synopsis	<code>u32 PROG_GetRequestSeedTimer (void);</code>
Return Value	value u32 request seed Timer to get

4.6.2.4.231. PROG_GetRsDelayTimer

Purpose	Get the current Request seed delay timer value.
Synopsis	<code>u32 PROG_GetRsDelayTimer (void);</code>



Return Value	value u32 request seed delay timer to get
--------------	---

4.6.2.4.232. PROG_GetSBIFlagValue

Purpose	Retrieve the SBI flag value stored in memory.
Synopsis	u8 PROG_GetSBIFlagValue (void);
Return Value	SBI flag value on 8 bits
Description	Callback is called: during reprogramming process to update the SBI flag for the SBA ticket stored in Secure RAM. Callback shall implement: Reading in secure volatile memory of the SBI flag value

4.6.2.4.233. PROG_GetSecurityLevel

Purpose	Get the current security level.
Synopsis	u8 PROG_GetSecurityLevel (void);
Return Value	the current security level
	0x00U ECU is locked
	0xOnU ECU is in security level n
Description	This function handles the Security Level feature managed by UDS. It shall be configured in UDS Tresos Studio plugin (if security check is needed).

4.6.2.4.234. PROG_GetSeed_Unlocked

Purpose	Called in GetSeed_Unlocked state.
Synopsis	void PROG_GetSeed_Unlocked (void);

4.6.2.4.235. PROG_GetSuppressBitFromAppli

Purpose	Get the status of the suppress positive response bit from the last reprogramming request in application.
Synopsis	u8 PROG_GetSuppressBitFromAppli (void);



Return Value	Suppression bit value	
	TRUE	The suppress positive response bit was set (response will not be sent)
	FALSE	The suppress positive response bit was not set (response will be sent)
Description	<p>Callback is called: At startup when Bootloader shall send a response to a request that has been received in application</p> <p>Callback shall implement: get from application information if the suppressPositiveResponse bit was set in the received request (e.g: read a flag from noinit RAM shared between Bootloader and Application)</p>	

4.6.2.4.236. PROG_Guard_Erase_Check_EraseFinish

Purpose	Called receiving a Erase request after completion Erase Request.	
Synopsis	<code>tProgStatus PROG_Guard_Erase_Check_EraseFinish (void);</code>	
Return Value	Result of check	
	PROG_E_OK	Erase request accepted
	PROG_E_NOT_OK	Erase request not accepted

4.6.2.4.237. PROG_Guard_RD_Check_RTEFinish

Purpose	Called receiving a RD request after a RTE.	
Synopsis	<code>tProgStatus PROG_Guard_RD_Check_RTEFinish (void);</code>	
Return Value	Result of check	
	PROG_E_OK	RD request accepted
	PROG_E_NOT_OK	RD request not accepted

4.6.2.4.238. PROG_HSMLastDataReceived

Purpose	Request to HSM Processor for Update.	
Synopsis	<code>tProgPECError PROG_HSMLastDataReceived (void);</code>	
Return Value	Result	



	PROG_PEC_NO_ERROR	HSM Processor Received the last Data
	ERROR_CODE	HSM Processor Update Failed
Description	<p>Callback is called: To Inform HSM Processor with Last of Data received</p> <p>Callback shall implement: the reading of status to check if the HSM Processor Received last data</p>	

4.6.2.4.239. PROG_HSMRequestUpdate

Purpose	Request to HSM Processor for Update.	
Synopsis	<code>tProgStatus PROG_HSMRequestUpdate (void);</code>	
Return Value	Result	
	PROG_E_OK	HSM Processor Update Allowed
	PROG_E_NOT_OK	HSM Processor Update Not Allowed
Description	<p>Callback is called: To Send Request to HSM Processor for Update</p> <p>Callback shall implement: the reading of status to check if the HSM Processor allowed the Update</p>	

4.6.2.4.240. PROG_HSMStatusManage

Purpose	
Synopsis	<code>void PROG_HSMStatusManage (void);</code>

4.6.2.4.241. PROG_HSMUpdate_TD

Purpose	
Synopsis	<code>void PROG_HSMUpdate_TD (void);</code>

4.6.2.4.242. PROG_HsmManage

Purpose	Manage HSM related actions.
Synopsis	<code>void PROG_HsmManage (void);</code>



Description	This function is called in the state machine during Check Memory in order to update the MAC value of a memory block.
--------------------	--

4.6.2.4.243. PROG_IMPL30_HSMEEntry

Purpose	HSM related Entry actions.
Synopsis	<code>void PROG_IMPL30_HSMEEntry (void);</code>
Description	This function is called in the state machine during Check Programming Dependency in order to start the MAC for a memory block.

4.6.2.4.244. PROG_IMPL30_SignatureHandling

Purpose	Signature Handler for Impl30.
Synopsis	<code>void PROG_IMPL30_SignatureHandling (tSignatureHandlingstatus m_setCHSstatus);</code>
Description	This function is called in the state machine during Check Programming Dependency in order to perform the Signature Verification for the Programmed Segment.

4.6.2.4.245. PROG_Impl10_CheckDataBlocksResult

Purpose	Called on entry to HASH_CHECK.
Synopsis	<code>void PROG_Impl10_CheckDataBlocksResult (void);</code>

4.6.2.4.246. PROG_Impl10_CheckMemoryAllowed

Purpose	Called on guard of check memory to check whether it is allowed or not.
Synopsis	<code>tProgStatus PROG_Impl10_CheckMemoryAllowed (void);</code>
Return Value	

4.6.2.4.247. PROG_Impl10_CompareDataBlockHash

Purpose	Called on entry to RTE_COMPARE_HASH state.
----------------	--

**Synopsis**

```
void PROG_Impl10_CompareDataBlockHash ( void );
```

4.6.2.4.248. PROG_Impl10_Do_CheckHashOfKey**Purpose**

Called during the write production key state to check if hash is finished.

Synopsis

```
void PROG_Impl10_Do_CheckHashOfKey ( void );
```

4.6.2.4.249. PROG_Impl10_Do_HashMoreUnwrittenData**Purpose**

Called while RTE_COMPARE_HASH is active, it updates the hash calculation according to the length mentioned in VBT.

Synopsis

```
void PROG_Impl10_Do_HashMoreUnwrittenData ( void );
```

4.6.2.4.250. PROG_Impl10_Entry_CheckMemoryFailed**Purpose**

Called on entry to CHECK_MEMORY_FAILED state.

Synopsis

```
void PROG_Impl10_Entry_CheckMemoryFailed ( void );
```

4.6.2.4.251. PROG_Impl10_Entry_CheckReceivedKey**Purpose**

Called upon the reception of write key request.

Synopsis

```
void PROG_Impl10_Entry_CheckReceivedKey ( void );
```

4.6.2.4.252. PROG_Impl10_Entry_SignatureCheck**Purpose**

Called on entry to SIGNATURE_CHECK.

Synopsis

```
void PROG_Impl10_Entry_SignatureCheck ( void );
```

4.6.2.4.253. PROG_Impl10_Entry_WriteKeyFinished**Purpose**

Called on when the hash calculation is finished.

**Synopsis**

```
void PROG_Impl10_Entry_WriteKeyFinished ( void );
```

4.6.2.4.254. PROG_Impl10_FinalizeHash

Purpose	Called on entry to RTE_COMPARE_HASH.
Synopsis	<pre>void PROG_Impl10_FinalizeHash (void);</pre>

4.6.2.4.255. PROG_Impl10_GenerateMac

Purpose	Called on entry to MAC_GENERATION, it generates the mac for each software part.
Synopsis	<pre>void PROG_Impl10_GenerateMac (void);</pre>

4.6.2.4.256. PROG_Impl90_CheckDataBlocksResult

Purpose	Called on entry to HASH_CHECK.
Synopsis	<pre>void PROG_Impl90_CheckDataBlocksResult (void);</pre>

4.6.2.4.257. PROG_Impl90_CompareDataBlockHash

Purpose	Called on entry to RTE_COMPARE_HASH state.
Synopsis	<pre>void PROG_Impl90_CompareDataBlockHash (void);</pre>

4.6.2.4.258. PROG_Impl90_Do_CheckHashOfKey

Purpose	Called during the write production key state to check if hash is finished.
Synopsis	<pre>void PROG_Impl90_Do_CheckHashOfKey (void);</pre>

4.6.2.4.259. PROG_Impl90_Do_HashMoreUnwrittenData

Purpose	Called while RTE_COMPARE_HASH is active, it updates the hash calculation according to the length mentioned in VBT.
----------------	--

**Synopsis**

```
void PROG_Impl90_Do_HashMoreUnwrittenData ( void );
```

4.6.2.4.260. PROG_Impl90_Entry_CheckReceivedKey**Purpose**

Called upon the reception of write key request.

Synopsis

```
void PROG_Impl90_Entry_CheckReceivedKey ( void );
```

4.6.2.4.261. PROG_Impl90_Entry_SignatureCheck**Purpose**

Called on entry to SIGNATURE_CHECK.

Synopsis

```
void PROG_Impl90_Entry_SignatureCheck ( void );
```

4.6.2.4.262. PROG_Impl90_Entry_ValidateFailed**Purpose**

Called on entry to CHECK_MEMORY_FAILED state.

Synopsis

```
void PROG_Impl90_Entry_ValidateFailed ( void );
```

4.6.2.4.263. PROG_Impl90_Entry_ValidateFinish**Purpose**

Called on entry to CHECK_MEMORY_FINISH state.

Synopsis

```
void PROG_Impl90_Entry_ValidateFinish ( void );
```

4.6.2.4.264. PROG_Impl90_Entry_WriteKeyFinished**Purpose**

Called on when the hash calculation is finished.

Synopsis

```
void PROG_Impl90_Entry_WriteKeyFinished ( void );
```

4.6.2.4.265. PROG_Impl90_FinalizeHash**Purpose**

Called on entry to RTE_COMPARE_HASH.

Synopsis

```
void PROG_Impl90_FinalizeHash ( void );
```



4.6.2.4.266. PROG_Init

Purpose	Initialize PROG module.
Synopsis	<code>void PROG_Init (void);</code>

4.6.2.4.267. PROG_InvalidateBlock

Purpose	API that invalidate the logical block.	
Synopsis	<code>tProgStatus PROG_InvalidateBlock (u8 ubBlockId);</code>	
Parameters (in)	ubBlockId	The ID of the block that will be erased
Return Value	Result application invalidation	
	PROG_E_OK	Treatment finish successfully
	PROG_E_NOT_OK	Error happened during treatment
Description	<p>Callback is called: On Erase routine reception</p> <p>Callback shall implement:</p> <ul style="list-style-type: none"> 1- customer code that shall be executed before performing an erasing of a logical block 2- It shall invalidate the logical block that will be erased to make sure no jump to the application will be done if an error occurred and the application is not fully erased or reprogrammed 	

4.6.2.4.268. PROG_InvalidateSection

Purpose	Callback invalidating the application markers.	
Synopsis	<code>tProgStatus PROG_InvalidateSection (tProgAddressType ulStartAddress , u32 ulEraseLength , tUdsStatus * ErrorCode);</code>	
Parameters (in)	ulStartAddress	Erased Start address of the segment that will be erased
	ulEraseLength	requested erase length
Parameters (out)	ErrorCode	UDS error code that shall be return in case of error during API treatment
Return Value	Result application invalidation	
	PROG_E_OK	Treatment finish successfully
	PROG_E_NOT_OK	Error happened during treatment (Error-Code shall be filled in this case)



Description	<p>Callback is called: On Erase routine reception</p> <p>Callback shall implement: 1- customer code that shall be executed before performing an erasing</p> <p>2- It shall invalidate the application or the current erased section to make sure no jump to the application will be done if an error occurred and the application is not fully erased or reprogrammed</p>
--------------------	---

4.6.2.4.269. PROG_InvalidateSection_BlockID

Purpose	API that invalidate the application marker.	
Synopsis	<code>tProgStatus PROG_InvalidateSection_BlockID (u8 ubBlockId);</code>	
Parameters (in)	ubBlockId	The ID of the block that will be erased
Return Value	Result application invalidation	
	PROG_E_OK	Treatment finish successfully
	PROG_E_NOT_OK	Error happened during treatment
Description	<p>Callback is called: On Erase routine reception</p> <p>Callback shall implement: 1- customer code that shall be executed before performing an erasing of a logical block</p> <p>2- It shall invalidate the logical block that will be erased to make sure no jump to the application will be done if an error occurred and the application is not fully erased or reprogrammed</p>	

4.6.2.4.270. PROG_IsValidApplication

Purpose	Callback checking if the application is valid or not.	
Synopsis	<code>tProgBoolean PROG_IsValidApplication (void);</code>	
Return Value	Result of check	
	TRUE	Application is valid
	FALSE	Application is not valid or not present
Description	<p>Callback is called: at startup and on some routine.</p> <p>Callback shall implement: It shall verify that the application with all its dependencies are correctly flashed and return the result of the check</p>	



4.6.2.4.271. PROG_JumpToApplication

Purpose	Callback performing jump to application software.
Synopsis	<code>void PROG_JumpToApplication (void);</code>
Description	Callback is called: at Bootloader startup if application is valid and shall be executed Callback shall implement: jump to application start address

4.6.2.4.272. PROG_JumpToSBL

Purpose	Called to perform the jump to SBL.
Synopsis	<code>void PROG_JumpToSBL (void);</code>

4.6.2.4.273. PROG_LogicalBlockHash

Purpose	UDS callback for LogicalBlockHash.	
Synopsis	<code>tUdsStatus PROG_LogicalBlockHash (PduLengthType * pulLen , u8 * aubUdsData);</code>	
Parameters (in,out)	pulLen	pointer on data length
	aubUdsData	pointer on data
Return Value	UDS status	
	UDS_ACK	positive response
	UDS_NRC_xxx	negative response
	UDS_NRC_31	Input pointer parameters NULL value
Description	This function handles the LogicalBlockHash UDS request. It shall be configured in UDS Tresos Studio plugin for the request RC (0x31 0x01 0xD0 0x03)/(0x31 0x01 0xD0 0x04) with this exact name and with Callback_Origin set to OTHER.	

4.6.2.4.274. PROG_LogicalBlockHashFinish

Purpose	Called after hash calculation success to send a response with the hash.
----------------	---

**Synopsis**

```
void PROG_LogicalBlockHashFinish ( void );
```

4.6.2.4.275. PROG_Manage

Purpose	Manage function to be called periodically.
Synopsis	void PROG_Manage (void);

4.6.2.4.276. PROG_MessageDigestCheck

Purpose	
Synopsis	void PROG_MessageDigestCheck (void);

4.6.2.4.277. PROG_OpenProgrammingSession

Purpose	Reception of a programming session request.
Synopsis	void PROG_OpenProgrammingSession (void);

4.6.2.4.278. PROG_Prelinit

Purpose	Called in Prelinit state.
Synopsis	void PROG_Prelinit (void);

4.6.2.4.279. PROG_RD_Check

Purpose	Called receiving a RD request.
Synopsis	tProgStatus PROG_RD_Check (void);
Return Value	Result of check
	PROG_E_OK
	PROG_E_NOT_OK

4.6.2.4.280. PROG_RTE

Purpose	Called on cyclically in RTE state.
----------------	------------------------------------



Synopsis	<code>void PROG_RTE (void);</code>
-----------------	--------------------------------------

4.6.2.4.281. PROG_RangeChecksumFinish

Purpose	Finishing the checksum calculation.		
Synopsis	<code>tProgStatus PROG_RangeChecksumFinish (void);</code>		
Return Value	state		
	PROG_E_OK	Calculation finished successfully	
	PROG_E_BUSY	Calculation in progress	
	PROG_E_NOT_OK	Calculation finished on error	

4.6.2.4.282. PROG_ReadKeyProgrammedStatus

Purpose	Read the Keys status used for Security Unlock.		
Synopsis	<code>tUdsStatus PROG_ReadKeyProgrammedStatus (PduLengthType * pulLen , u8 * aubUdsData);</code>		
Return Value			
Description	This function is called when the Read DID request is received for reading the Secured Keys Programming Status		

4.6.2.4.283. PROG_RequestDownload

Purpose	UDS callback for RequestSeed.		
Synopsis	<code>tUdsStatus PROG_RequestDownload (PduLengthType * pulLen , u8 * aubUdsData);</code>		
Parameters (in,out)	pulLen	pointer on data length	
	aubUdsData	pointer on data	
Return Value	UDS status		
	UDS_ACK	positive response	
	UDS_NRC_xxx	negative response	
	UDS_NRC_31	Input pointer parameters NULL value	
Description	This function handles the RequestSeed UDS request. It shall be configured in UDS Tresos Studio plugin for the request RD (0x34) with this exact name and with Call-back-Origin set to OTHER.		



4.6.2.4.284. PROG_RequestSeed

Purpose	UDS callback for RequestSeed.	
Synopsis	<code>tUdsStatus PROG_RequestSeed (PduLengthType * pulLen , u8 * aubUdsData);</code>	
Parameters (in,out)	pulLen	pointer on data length
	aubUdsData	pointer on data
Return Value	UDS status	
	UDS_ACK	positive response
	UDS_NRC_xxx	negative response
	UDS_NRC_31	Input pointer parameters NULL value
Description	This function handles the RequestSeed UDS request. It shall be configured in UDS Tresos Studio plugin for the request SA1 (0x27 0x01) with this exact name and with Callback_Origin set to OTHER.	

4.6.2.4.285. PROG_RequestTransferExit

Purpose	UDS callback for RequestTransferExit.	
Synopsis	<code>tUdsStatus PROG_RequestTransferExit (PduLengthType * pulLen , u8 * aubUdsData);</code>	
Parameters (in,out)	pulLen	pointer on data length
	aubUdsData	pointer on data
Return Value	UDS status	
	UDS_ACK	positive response
	UDS_NRC_xxx	negative response
	UDS_NRC_31	Input pointer parameters NULL value
Description	This function handles the RequestTransferExit UDS request. It shall be configured in UDS Tresos Studio plugin for the request RTE (0x37) with this exact name and with Callback_Origin set to OTHER.	

4.6.2.4.286. PROG_ResReprog_CheckSegmentListVerif

Purpose	Provide next segment to verify.
Synopsis	<code>void PROG_ResReprog_CheckSegmentListVerif (void);</code>



Description	This function is called after verification of a segment to get information of the next segment
--------------------	--

4.6.2.4.287. PROG_ReturnsIsReProgRequestFromAppli

Purpose	API that returns information if a programming request has been received by the application.	
Synopsis	<code>tProgBoolean PROG_ReturnsIsReProgRequestFromAppli (void);</code>	
Return Value	Information checked in PROG_CheckProgRequest API	
	TRUE	Reprogramming request has been received in Application
	FALSE	Reprogramming request has been received in Application
Description	<p>Callback is called: at reception of a request in BIPduR module to ignore the request if Rx simulation request is ongoing (programming request has been received by the application)</p> <p>Callback shall implement: returns information from application if a programming request has been received or not (e.g: from PROG_CheckProgRequest API)</p>	

4.6.2.4.288. PROG_SA2_RD_Check

Purpose	
Synopsis	<code>tProgStatus PROG_SA2_RD_Check (void);</code>
Return Value	

4.6.2.4.289. PROG_SBASignatureCheck

Purpose	
Synopsis	<code>void PROG_SBASignatureCheck (void);</code>

4.6.2.4.290. PROG_SBAVerifierInfoCheck

Purpose	
Synopsis	<code>void PROG_SBAVerifierInfoCheck (void);</code>



4.6.2.4.291. PROG_SecurityComputeAppChecksum

Purpose	Start Application checksum computation.
Synopsis	<code>void PROG_SecurityComputeAppChecksum (void);</code>
Description	This function is called before sending response to CheckMemory request or Submit Signature Request for authenticated block to perform Application checksum computation

4.6.2.4.292. PROG_SendKey

Purpose	UDS callback for SendKey.	
Synopsis	<code>tUdsStatus PROG_SendKey (PduLengthType * pulLen , u8 * aubUdsData);</code>	
Parameters (in,out)	pulLen	pointer on data length
	aubUdsData	pointer on data
Return Value	UDS status	
	UDS_ACK	positive response
	UDS_NRC_xxx	negative response
	UDS_NRC_31	Input pointer parameters NULL value
Description	This function handles the SendKey UDS request. It shall be configured in UDS Tresos Studio plugin for the request SA2 (0x27 0x02) with this exact name and with Callback_Origin set to OTHER.	

4.6.2.4.293. PROG_SendNRC78

Purpose	Send the NRC_78 instantaneously to gain time.
Synopsis	<code>void PROG_SendNRC78 (void);</code>

4.6.2.4.294. PROG_Send_NRC

Purpose	Called to send negative response.	
Synopsis	<code>void PROG_Send_NRC (tUdsStatus eUdsStatus);</code>	
Parameters (in)	eUdsStatus	Error code to use in negative response



4.6.2.4.295. PROG_SetCryptoDataSizeDataAddr

Purpose	Fetches the datasize and the datapointer for the current Crypto ASR 4.3 job.
Synopsis	<code>void PROG_SetCryptoDataSizeDataAddr (u8 const * pubData , u32 ulDataSize);</code>
Description	This function is called to fetch the datasize and the datapointer for the current Crypto ASR 4.3 job update mode operation.

4.6.2.4.296. PROG_SetKeyNBIDValue

Purpose	Retrieve the Key NBID value store in memory.	
Synopsis	<code>void PROG_SetKeyNBIDValue (u16 uwKeyNBID);</code>	
Parameters (in)	uwKeyNBID	New value of the Key NBID to be store in memory
Description	<p>Callback is called: during reprogramming process to update the key NBID of the downloaded application stored in NVM.</p> <p>Callback shall implement: Writting in non volatile memory of the Key NBID value</p>	

4.6.2.4.297. PROG_SetNBIDValue

Purpose	Retrieve the NBID value store in memory.	
Synopsis	<code>void PROG_SetNBIDValue (u16 uwNBID);</code>	
Parameters (in)	uwNBID	New value of the NBID to be store in memory
Description	<p>Callback is called: during reprogramming process to update the NBID of the downloaded application stored in NVM.</p> <p>Callback shall implement: Writting in non volatile memory of the NBID value</p>	

4.6.2.4.298. PROG_SetNetworkStatus

Purpose	Inform of network status change.	
Synopsis	<code>void PROG_SetNetworkStatus (u16 uwNetworkStatus);</code>	
Parameters (in)	uwNetworkStatus	new network status



Description	This function is called on change of the network status
-------------	---

4.6.2.4.299. PROG_SetProgrammingStatus

Purpose	Update the programming status.	
Synopsis	<code>void PROG_SetProgrammingStatus (u32 ulProgrammingStatusMask , tProgBoolean ubStatus);</code>	
Parameters (in)	ulProgrammingStatusMask	the mask for a specific failure
	ubStatus	failure status (0 - present, 1 - not present)

4.6.2.4.300. PROG_SetRequestSeedCounter

Purpose	Set the current Request seed counter value.	
Synopsis	<code>void PROG_SetRequestSeedCounter (u8 ubRequestSeedCounter);</code>	
Parameters (in)	ubRequestSeedCounter	request seed counter to set

4.6.2.4.301. PROG_SetRequestSeedTimer

Purpose	Set the current Request seed Timer.	
Synopsis	<code>void PROG_SetRequestSeedTimer (u32 ulRequestSeedTimer);</code>	
Parameters (in)	ulRequestSeedTimer	request seed timer to set

4.6.2.4.302. PROG_SetRsDelayTimer

Purpose	Set the current Request seed delay timer value.	
Synopsis	<code>void PROG_SetRsDelayTimer (u32 ulRsDelayTimer);</code>	
Parameters (in)	ulRsDelayTimer	request seed delay timer value to set

4.6.2.4.303. PROG_SetSBIFlagValue

Purpose	Set the SBI flag value in secure memory.	
---------	--	--



Synopsis	<code>void PROG_SetSBIFlagValue (u8 ubSBI);</code>	
Parameters (in)	ubSBI	New value of the SBI flag to be stored in memory
Description	<p>Callback is called: during startup to update the SBI flag for the SBA ticket stored in Secure RAM.</p> <p>Callback shall implement: Writing in secure volatile memory of the SBI flag value</p>	

4.6.2.4.304. PROG_SignatureCheck

Purpose	
Synopsis	<code>void PROG_SignatureCheck (void);</code>

4.6.2.4.305. PROG_SignerInfoCheck

Purpose	
Synopsis	<code>void PROG_SignerInfoCheck (void);</code>

4.6.2.4.306. PROG_SimulateExtendedSessionNoResponse

Purpose	Initialize PROG_SimulateExtendedSessionNoResponse function.
Synopsis	<code>void PROG_SimulateExtendedSessionNoResponse (void);</code>

4.6.2.4.307. PROG_SimulateOpenProgSession

Purpose	Request to simulation a programming session opening.
Synopsis	<code>void PROG_SimulateOpenProgSession (void);</code>

4.6.2.4.308. PROG_SkipPage

Purpose	Called by the Flash driver to know if the page can be written.
Synopsis	<code>u8 PROG_SkipPage (u32 * uAddr);</code>



Parameters (in)	uAddr	pointer on Address of the page to write
Return Value	Result of the check	
	PROG_NO_SKIP_PAGE	page can be written
	others	write is not allowed
Description	This API is called by the Flash driver to know if it is authorized to write a Flash pages. The API will forward the request to the framework	

4.6.2.4.309. PROG_Streaming

Purpose	Called in Streaming state.
Synopsis	<code>void PROG_Streaming (void);</code>

4.6.2.4.310. PROG_StreamingFrameReceived

Purpose	Reception of a streaming frame.	
Synopsis	<code>void PROG_StreamingFrameReceived (u16 ulReceivedDataLength , u8 * aubData);</code>	
Parameters (in)	ulReceivedDataLength	Data Length in the received frame
	aubData	Pointer to buffer where received data are located
Description	This function is called on reception of a streaming frame	

4.6.2.4.311. PROG_SwitchApplicationMode

Purpose	
Synopsis	<code>void PROG_SwitchApplicationMode (void);</code>

4.6.2.4.312. PROG_SwitchApplicationModeInd

Purpose	Called before Bootloader perform a jump to application.
Synopsis	<code>void PROG_SwitchApplicationModeInd (void);</code>
Description	Callback is called: Before jumping to Application



	Callback shall implement: customer code that shall be executed before jumping to application
--	--

4.6.2.4.313. PROG_TD

Purpose	Called in TD state.	
Synopsis	<code>tProgStatus PROG_TD (void);</code>	
Return Value		

4.6.2.4.314. PROG_TD_Impl80

Purpose	API called in TD state.	
Synopsis	<code>void PROG_TD_Impl80 (void);</code>	

4.6.2.4.315. PROG_TpRxInd

Purpose	called on frame reception	
Synopsis	<code>void PROG_TpRxInd (tTpMsgIdx uMsgIdx , u8 ebStatus);</code>	
Parameters (in)	uMsgIdx	Identifier of the transmission frame
	ebStatus	status of the transmission
Description	This function is called on a diagnostic frame reception	

4.6.2.4.316. PROG_TpStartOfReceptionInd

Purpose	called on frame reception	
Synopsis	<code>void PROG_TpStartOfReceptionInd (u8 ubStatus);</code>	
Parameters (in)	ubStatus	status of the transmission
Description	This function is called on a diagnostic frame reception	

4.6.2.4.317. PROG_TpTxConf

Purpose	called on confirmation of frame transmission	
---------	--	--



Synopsis	<code>void PROG_TpTxConf (tTpMsgIdx uMsgIdx , u8 ebStatus);</code>	
Parameters (in)	uMsgIdx	Identifier of the transmission frame
	ebStatus	status of the transmission
Description	This function is called on confirmation of a diagnostic response transmission	

4.6.2.4.318. PROG_TransferData

Purpose	UDS callback for TransferData.	
Synopsis	<code>tUdsStatus PROG_TransferData (PduLengthType * pulLen , u8 * aubUdsData);</code>	
Parameters (in,out)	pulLen	pointer on data length
	aubUdsData	pointer on data
Return Value	UDS status	
	UDS_ACK	positive response
	UDS_NRC_xxx	negative response
	UDS_NRC_31	Input pointer parameters NULL value
Description	This function handles the TransferData UDS request. It shall be configured in UDS Tresos Studio plugin for the request TD (0x36) with this exact name and with Callback_Origin set to OTHER.	

4.6.2.4.319. PROG_TxConfNotification

Purpose	TxConf notification for the Prog-Crypto Interface layer.	
Synopsis	<code>void PROG_TxConfNotification (void);</code>	
Description	This function is called to indicate the Txconf of the transmitted message, especially for the notification of the NRC78 message transmission.	

4.6.2.4.320. PROG_UpdatePSI

Purpose	UDS callback for UpdatePSI.	
Synopsis	<code>tUdsStatus PROG_UpdatePSI (PduLengthType * pulLen , u8 * aubUdsData);</code>	
Parameters (in,out)	pulLen	pointer on data length



	aubUdsData	pointer on data
Return Value	UDS status	
	UDS_ACK	positive response
	UDS_NRC_xxx	negative response
	UDS_NRC_31	Input pointer parameters NULL value
Description	This function handles the UpdatePSI UDS request. It shall be configured in UDS Tresos Studio plugin for the request XXXXXX with this exact name and with Call-back_Origin set to OTHER.	

4.6.2.4.321. PROG_VerificationOnTheFly

Purpose	Called in CRC state.	
Synopsis	<code>tProgStatus PROG_VerificationOnTheFly (void);</code>	
Return Value		

4.6.2.4.322. PROG_VerificationOnTheFly_Impl80

Purpose	API called on the TD CRC state.	
Synopsis	<code>void PROG_VerificationOnTheFly_Impl80 (void);</code>	

4.6.2.4.323. PROG_VerifySectionCrc

Purpose	Callback verifying the application is correctly downloaded and setting the application validity marker.	
Synopsis	<code>tProgCompleteStatus PROG_VerifySectionCrc (void);</code>	
Return Value	Result of check	
	TRUE	Application is valid
	FALSE	Application is not valid or not present
Description	<p>Callback is called: on CheckApplicationValidation routine at the end of the programming sequence.</p> <p>Callback shall implement: verification that each segment is fully programmed and setting of the validity marker if necessary. It then returns the status of the check.</p>	



4.6.2.4.324. PROG_Write

Purpose	Called in Write state.
Synopsis	<code>void PROG_Write (void);</code>

4.6.2.4.325. PROG_WriteCheck

Purpose	Called in WriteCheck state.
Synopsis	<code>void PROG_WriteCheck (void);</code>

4.6.2.4.326. PROG_WriteCheck_Impl80

Purpose	API called on the do of the write check state.
Synopsis	<code>void PROG_WriteCheck_Impl80 (void);</code>

4.6.2.4.327. PROG_WriteFingerprintCheck

Purpose	Called on loop in WriteFingerprint state.
Synopsis	<code>void PROG_WriteFingerprintCheck (void);</code>

4.6.2.4.328. PROG_Write_Impl80

Purpose	API called on the do of the write state.
Synopsis	<code>void PROG_Write_Impl80 (void);</code>

4.6.2.4.329. Prog_CustomGetAdditionalProgrammingConditionalFlags

Purpose	get additional programming conditional flags	
Synopsis	<code>void Prog_CustomGetAdditionalProgrammingConditionalFlags (u8 * pubFlag);</code>	
Parameters (out)	pubFlag	pointer to a variable to get the additional programming conditional flag
Description	This API is called during the processing of the check programming dependencies routine	



4.6.2.4.330. Prog_CustomGetECUInternalProgrammingFlag

Purpose	get the programming conditions flag	
Synopsis	<code>void Prog_CustomGetECUInternalProgrammingFlag (u8 * pubFlag);</code>	
Parameters (out)	pubFlag	pointer to a variable to get the programming conditions flag
Description	This API is called during the processing of the check programming dependencies routine	

4.6.2.4.331. Prog_CustomGetProgrammingConditionsFlag

Purpose	get the ecu internal programming flags	
Synopsis	<code>void Prog_CustomGetProgrammingConditionsFlag (u8 * pubFlag);</code>	
Parameters (out)	pubFlag	pointer to a variable to get the ecu internal programming flags
Description	This API is called during the processing of the check programming dependencies routine	

4.6.2.4.332. Prog_CustomGetProgrammingTolerantConditionsFlag

Purpose	get the programming tolerant conditions flag	
Synopsis	<code>void Prog_CustomGetProgrammingTolerantConditionsFlag (u8 * pubFlag);</code>	
Parameters (out)	pubFlag	pointer to a variable to get the programming tolerant conditions flag
Description	This API is called during the processing of the check programming dependencies routine	

4.6.2.4.333. Prog_CustomIsProdKeyPresent

Purpose	Check if the production key is written or not.
----------------	--



Synopsis	boolean Prog_CustomIsProdKeyPresent (void);
Return Value	boolean TRUE : Key exists, FALSE : Key does not exist
Description	<p>Callback is called: before writing or reading the production key</p> <p>Callback shall implement: check if the key exists or not</p>

4.6.2.4.334. **Prog_CustomReadKeyChecksum**

Purpose	Get the checksum of the key.	
Synopsis	void Prog_CustomReadKeyChecksum (u8 * aubKeyChecksum);	
Parameters (out)	aubKeyChecksum	key checksum
Description	<p>Callback is called: On receiving RDBI of the production key</p> <p>Callback shall implement: get the key checksum from the non volatile memory</p>	

4.6.2.4.335. **Prog_CustomWriteKey**

Purpose	write the production key used in signature verification	
Synopsis	void Prog_CustomWriteKey (u32 ulExponent , u8 * aubModulus , u8 * aubKeyChecksum);	
Parameters (in)	ulExponent	key exponent
	aubModulus	key modulus
	aubKeyChecksum	key checksum
Description	<p>Callback is called: On receiving WDBI for the production key</p> <p>Callback shall implement: write the key and its checksum in non volatile memory</p>	

4.6.2.4.336. **Prog_GetAPPL_VBTLength**

Purpose	Get APPL/CALIB VBT Length.	
Synopsis	tProgStatus Prog_GetAPPL_VBTLength (u8 ubBlockIndex , u32 * pulVBTLength);	
Parameters (in)	ubBlockIndex	index of the logical block in Prog plugin configuration



Parameters (out)	pulVBTLength	length of the verification block table of Application and calibration
Return Value	eProgStatus success of the operation	
	PROG_E_OK	Information is available
	PROG_E_NOT_OK	Information is not available
Description	<p>Callback is called: On RD to get Verification Block Table location length for application and calibration</p> <p>Callback shall implement: return length of the verification block table of the request logical block in APPL/CALIB</p>	

4.6.2.4.337. Prog_GetEssApplicationStartAddress

Purpose	Get the address of the jump to the application.	
Synopsis	<pre>void Prog_GetEssApplicationStartAddress (u32 * ulApplicationStartAddress);</pre>	
Parameters (out)	ulApplicationStartAddress	Application start address
Description	<p>Callback is called: On switching to application mode</p> <p>Callback shall implement: get the application start address from the ESS</p>	

4.6.2.4.338. Prog_GetEssLength

Purpose	Get the length of the ESS.	
Synopsis	<pre>void Prog_GetEssLength (u32 * ulEssLength);</pre>	
Parameters (out)	ulEssLength	ESS length
Description	<p>Callback is called: On calculating MAC of ESS</p> <p>Callback shall implement: get the length of the ESS</p>	

4.6.2.4.339. Prog_GetEssLogicalBlockId

Purpose	Get ESS logical block Identifier.
----------------	-----------------------------------



Synopsis	<code>tProgStatus Prog_GetEssLogicalBlockId (u8 ubBlockIndex , u16 * pulBlockIdent);</code>	
Parameters (in)	ubBlockIndex	Block identifier
Parameters (out)	pulBlockIdent	identifier of the request logical block in ESS
Return Value	eProgStatus success of the operation	
	PROG_E_OK	Information is available
	PROG_E_NOT_OK	Information is not available
Description	Callback is called: On RD or Erase routine to identify the downloaded block Callback shall implement: return identifier of the request logical block in ESS	

4.6.2.4.340. Prog_GetEssLogicalBlockLength

Purpose	Get ESS logical block Length.	
Synopsis	<code>tProgStatus Prog_GetEssLogicalBlockLength (u8 ubBlockIndex , u32 * pulBlockLength);</code>	
Parameters (in)	ubBlockIndex	Block identifier
Parameters (out)	pulBlockLength	length of the request logical block in ESS
Return Value	eProgStatus success of the operation	
	PROG_E_OK	Information is available
	PROG_E_NOT_OK	Information is not available
Description	Callback is called: On RD or Erase routine to identify the downloaded block Callback shall implement: return length of the request logical block in ESS	

4.6.2.4.341. Prog_GetEssLogicalBlockNbr

Purpose	Get ESS number of logical block.	
Synopsis	<code>tProgStatus Prog_GetEssLogicalBlockNbr (u8 * pubBlockNbr);</code>	
Parameters (out)	pubBlockNbr	number of logical block in ESS
Return Value	eProgStatus success of the operation	
	PROG_E_OK	Information is available



	PROG_E_NOT_OK	Information is not available
Description	Callback is called: On RD or Erase routine to identify the downloaded block	
	Callback shall implement: return number of logical block in ESS	

4.6.2.4.342. `Prog_GetEssLogicalBlockStartAddr`

Purpose	Get ESS logical block address.	
Synopsis	<code>tProgStatus Prog_GetEssLogicalBlockStartAddr (u8 ubBlockIndex, u32 * pulBlockAddr);</code>	
Parameters (in)	ubBlockIndex	Block identifier
Parameters (out)	pulBlockAddr	address of the request logical block in ESS
Return Value	eProgStatus success of the operation	
	PROG_E_OK	Information is available
	PROG_E_NOT_OK	Information is not available
Description	Callback is called: On RD or Erase routine to identify the downloaded block	
	Callback shall implement: return address of the request logical block in ESS	

4.6.2.4.343. `Prog_GetEssLogicalBlockVerifTable`

Purpose	Get ESS logical block VBT address.	
Synopsis	<code>tProgStatus Prog_GetEssLogicalBlockVerifTable (u8 ubBlockIndex, u32 * pulVBTAddress);</code>	
Parameters (in)	ubBlockIndex	Block identifier
Parameters (out)	pulVBTAddress	address of the verification block table of the request logical block in ESS
Return Value	eProgStatus success of the operation	
	PROG_E_OK	Information is available
	PROG_E_NOT_OK	Information is not available
Description	Callback is called: On RD to get Verification Block Table location	
	Callback shall implement: return address of the verification block table of the request logical block in ESS	



4.6.2.4.344. Prog_GetEssStartAddr

Purpose	Get the start address of the ESS.	
Synopsis	<code>void Prog_GetEssStartAddr (u32 * ulEssStartAddress);</code>	
Parameters (out)	ulEssStartAddress	ESS start address
Description	<p>Callback is called: On calculating MAC</p> <p>Callback shall implement: get the start address of the ESS</p>	

4.6.2.4.345. Prog_GetEssValidityStatus

Purpose	Get ESS validity status.	
Synopsis	<code>tProgStatus Prog_GetEssValidityStatus (void);</code>	
Return Value	eProgStatus success of the operation	
	PROG_E_OK	ESS is valid
	PROG_E_NOT_OK	is not valid
Description	<p>Callback is called: Before using information from ESS</p> <p>Callback shall implement: ESS validity status</p>	

4.6.2.4.346. Prog_GetEssVerifTable

Purpose	Get ESS VBT address.	
Synopsis	<code>tProgStatus Prog_GetEssVerifTable (u32 * pulVBTAddress);</code>	
Parameters (out)	pulVBTAddress	address of the verification block table of the request logical block in ESS
Return Value	eProgStatus success of the operation	
	PROG_E_OK	Information is available
	PROG_E_NOT_OK	Information is not available
Description	<p>Callback is called: On RD to get Verification Block Table location</p> <p>Callback shall implement: return address of the verification block table of the request logical block in ESS</p>	



4.6.2.4.347. Prog_GetEss_VBTLength

Purpose	Get ESS VBT Length.	
Synopsis	<code>tProgStatus Prog_GetEss_VBTLength (u32 * pulVBTLength);</code>	
Parameters (out)	pulVBTLength	length of the verification block table in ESS
Return Value	eProgStatus success of the operation	
	PROG_E_OK	Information is available
	PROG_E_NOT_OK	Information is not available
Description	<p>Callback is called: On RD to get Verification Block Table location length</p> <p>Callback shall implement: return length of the verification block table of the request logical block in ESS</p>	

4.6.2.4.348. Prog_GetLogicalBlockSignature

Purpose	Get logical block Signature address.	
Synopsis	<code>tProgStatus Prog_GetLogicalBlockSignature (u8 ubBlockIndex , u32 * pulsSignatureAddress);</code>	
Return Value	eProgStatus success of the operation	
	PROG_E_OK	Information is available
	PROG_E_NOT_OK	Information is not available
Description	<p>Callback is called: On request Validate Application to get address of Signature of the requested logical block</p> <p>Callback shall implement: return address of the Signature</p>	

4.6.2.4.349. Prog_GetLogicalBlockVerifStructure

Purpose	Get logical block VS address.	
Synopsis	<code>tProgStatus Prog_GetLogicalBlockVerifStructure (u8 ubBlockIndex , u32 * pulVSAddress);</code>	



Return Value	eProgStatus success of the operation	
	PROG_E_OK	Return value shall always be E_OK
Description	<p>Callback is called: On RD to get Verification Structure Address</p> <p>Callback shall implement: return address of the verification Structure of the requested logical block</p>	

4.6.2.4.350. Prog_HSMTransferData

Purpose	Request to HSM Processor for Update.	
Synopsis	<pre>tProgStatus Prog_HSMTransferData (u8 * paubHsmBuffer , Pdu- LengthType ulHsmBufferLength , u16 * uwHsmPEC);</pre>	
Parameters (in)	paubHsmBuffer	pointer to a HSM buffer
	ulHsmBufferLength	Length of HSM Buffer
	uwHsmPEC	Programming Error Code received from HSM
Return Value	Result	
	PROG_E_OK	Read Data From buffer by HSM Processor succeeded
	PROG_E_NOT_OK	Read Data From buffer by HSM Processor Failed
	PROG_E_BUSY	HSM Read Data From buffer by HSM Processor InProgress
Description	<p>Callback is called: To transfer Data from buffer to HSM Processor</p> <p>Callback shall implement: the reading of status to check if the HSM Processor Read Data From buffer</p>	

4.6.3. Integration notes

4.6.3.1. Exclusive areas

Exclusive areas information is not available for this module.

4.6.3.2. Production errors

Production errors information is not available for this module.

4.6.3.3. Memory mapping

General information about memory mapping is provided in the EB tresos AutoCore Generic documentation. Refer to the section [Memory mapping and compiler abstraction](#) in the [Integration notes](#) section for details.

Memory mapping information is not available for this module.

4.6.3.4. Integration requirements

WARNING**Integration requirements list is not exhaustive**

The following list of integration requirements helps you to integrate your product. However, this list is not exhaustive. You also require information from the user guide, release notes, and EB tresos AutoCore known issues to successfully integrate your product.

Integration requirements are not listed for the Prog module.

4.7. SA

4.7.1. Configuration parameters

Containers included		
Container name	Multiplicity	Description
CommonPublishedInformation	1..1	Label: Common Published Information Common container, aggregated by all modules. It contains published information about vendor and versions.
PublishedInformation	1..1	Label: EB Published Information Additional published parameters not covered by Common-PublishedInformation container.



Containers included

General	1..1	This container describes the general properties of the node.
-------------------------	------	--

4.7.1.1. CommonPublishedInformation

Parameters included

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

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

Parameter Name	ArMinorVersion
Label	AUTOSAR Minor Version
Description	Minor version number of AUTOSAR specification on which the appropriate implementation is based on.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0



Configuration class	PublishedInformation:	
Origin	Elektrobit Automotive GmbH	

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

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

Parameter Name	SwMinorVersion
Label	Software Minor Version
Description	Minor version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	16
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	SwPatchVersion
Label	Software Patch Version



Description	Patch level version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	ModuleId
Label	Numeric Module ID
Description	Module ID of this module from Module List
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	VendorId
Label	Vendor ID
Description	Vendor ID of the dedicated implementation of this module according to the AU-TOSAR vendor list
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	1
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

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



4.7.1.2. PublishedInformation

Parameters included	
Parameter name	Multiplicity
PbcfgMSupport	1..1

Parameter Name	PbcfgMSupport
Label	PbcfgM support
Description	Specifies whether or not the SA can use the PbcfgM module for post-build support.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

4.7.1.3. General

Containers included		
Container name	Multiplicity	Description
CsmRandomGenerate	1..1	
CsmRandomSeed	1..1	
SignatureVerify	1..1	
CsmMACVerification_Ms-gAuth	1..1	
CsmMACGenerate_MsgAuth	1..1	
CsmAES_Encryption	1..1	
CsmAES_Decryption	1..1	
CsmMACVerification_Poo	1..1	
CsmMACGenerate_Poo	1..1	

Parameters included	
Parameter name	Multiplicity
MANAGE_PERIOD	1..1



Parameters included

Seed_Type	1..1
Security_Algorithm_Type	1..1
Security_Access_Seed_Length	1..1
Enable_Static_Seed	1..1
Compare_Key_Type	1..1
Security_Access_Key_Length	1..1
Security_Access_Rs_Delay_Timer	1..1
Enable_Antiscanning	1..1
Enable_Request_Seed_Limit	1..1
Security_Access_As_Timer	1..1
Security_Access_As_Retry_Counter	1..1
Security_Access_RS_Retry_Counter	1..1
Security_Access_RS_Timer	1..1
Decomp_Out_Buffer_size	1..1
Static_Key_0	1..1
Static_Key_1	1..1
Static_Key_2	1..1
Static_Key_3	1..1
Static_Key_4	1..1
SA_AM_TYPE	1..1
SA2_Awaiting_MaxTime	1..1

Parameter Name	MANAGE_PERIOD
Label	SA Manage Period
Description	Period of the periodical SA task. This period must be multiple of EB periodical value in EB module configuration.
Multiplicity	1..1
Type	INTEGER
Range	>=1
Origin	EB

Parameter Name	Seed_Type
Label	Seed Type



Description	Specify the type of seed that should be used.
	<ul style="list-style-type: none"> ▶ Case Standard: Seed is free timer based. ▶ Case Cryptographic_Random: Seed is generated using cryptographic random. ▶ Case Cryptographic_PUN: Seed is PUN based.
Multiplicity	1..1
Type	ENUMERATION
Default value	Standard
Range	<ul style="list-style-type: none"> Standard Cryptographic_Random Cryptographic_PUN
Origin	EB

Parameter Name	Security_Algorithm_Type
Label	Security Algorithm Type
Description	Specify the security algorithm that should be used.
	<ul style="list-style-type: none"> ▶ Case Standard: Standard security algorithm of the OEM will be used. ▶ Case Custom: custom algorithm implemented by the user in a callback will be used.
Multiplicity	1..1
Type	ENUMERATION
Default value	Standard
Range	<ul style="list-style-type: none"> Standard Custom
Origin	EB

Parameter Name	Security_Access_Seed_Length
Label	Security Access Seed Length
Description	Specify the size of the seed for SecurityAccess service.
Multiplicity	1..1
Type	INTEGER
Default value	3
Origin	EB



Parameter Name	Enable_Static_Seed
Label	Enable Static Seed
Description	<p>Specify if the Static Seed should be used.</p> <ul style="list-style-type: none"> ▶ Case Disabled: A new Seed is generated on each time a GetSeed request is received. ▶ Case Enabled: A new Seed is generated only if the precedent one was used by the tester to generate a key and send an request. Else the last generated seed is returned as response to the GetSeed request.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Compare_Key_Type
Label	Compare Key Type
Description	<p>Specify the type of key comparison that should be used.</p> <ul style="list-style-type: none"> ▶ Case Standard: Byte to byte key compare. ▶ Case Verify_Signature: Key signature verification. ▶ Case SA_AM_0001: When SA Authentication Method 0001 feature is enabled. ▶ Case PUN: PUN extracted from received key compared with stored PUN.
Multiplicity	1..1
Type	ENUMERATION
Default value	Standard
Range	Standard Verify_Signature SA_AM_0001 PUN
Origin	EB

Parameter Name	Security_Access_Key_Length
Label	Security Access Key Length
Description	Specify the size of the key for SecurityAccess service.
Multiplicity	1..1



Type	INTEGER
Default value	3
Origin	EB

Parameter Name	Security_Access_Rs_Delay_Timer
Label	Security Access RS Delay Timer
Description	Specify the value of the Request Seed Delay Timer SecurityAccess service. If the value is greater than 0, the feature is enabled. Else, the feature is disabled.
Multiplicity	1..1
Type	INTEGER
Default value	0
Origin	EB

Parameter Name	Enable_Antiscanning
Label	Enable Antiscanning
Description	Specify if the Anti-scanning is enabled.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Enable_Request_Seed_Limit
Label	Enable Request Seed Limit
Description	Specify if the number of Request Seed requests shall be limited within a period of time.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Security_Access_As_Timer
Label	Security Access As Timer
Description	Specify the value of the Anti-scanning Lock Timer SecurityAccess service.
Multiplicity	1..1
Type	INTEGER



Default value	10000
Origin	EB

Parameter Name	Security_Access_As_Retry_Counter
Label	Security Access As Retry Counter
Description	Specify the value of the Anti-scanning Retry Counter for SecurityAccess service.
Multiplicity	1..1
Type	INTEGER
Default value	3
Origin	EB

Parameter Name	Security_Access_RS_Retry_Counter
Label	Security Access RS Retry Counter
Description	Specify the value of the Request Seed limit Counter for SecurityAccess service.
Multiplicity	1..1
Type	INTEGER
Default value	2
Origin	EB

Parameter Name	Security_Access_RS_Timer
Label	Security_Access_RS_Timer
Description	Specify the value in milliseconds of the Request seed limit Timer for SecurityAccess service.
Multiplicity	1..1
Type	INTEGER
Default value	5000
Origin	EB

Parameter Name	Decomp_Out_Buffer_size
Label	Decompression Output Buffer Size
Description	Size of the Output decompression buffer. This buffer is used to store the data decompressed and will be used for write operation.
Multiplicity	1..1



Type	INTEGER
Default value	0x400
Range	>=0
Origin	EB

Parameter Name	Static_Key_0
Description	First byte of the constant security key.
Multiplicity	1..1
Type	INTEGER
Default value	0x04
Range	>=0
	<=255
Origin	EB

Parameter Name	Static_Key_1
Description	Second byte of the constant security key.
Multiplicity	1..1
Type	INTEGER
Default value	0x03
Range	>=0
	<=255
Origin	EB

Parameter Name	Static_Key_2
Description	Third byte of the constant security key.
Multiplicity	1..1
Type	INTEGER
Default value	0x02
Range	>=0
	<=255
Origin	EB

Parameter Name	Static_Key_3
Description	Fouth byte of the constant security key.



Multiplicity	1..1
Type	INTEGER
Default value	0x01
Range	>=0 <=255
Origin	EB

Parameter Name	Static_Key_4
Description	Fifth byte of the constant security key.
Multiplicity	1..1
Type	INTEGER
Default value	0x00
Range	>=0 <=255
Origin	EB

Parameter Name	SA_AM_TYPE
Label	Security Access Authentication Method Type
Description	Specify the Security Authentication Method that shall be used for the Security Access Authentication. Note: Currently Bootloader shall support only SA Authentication Method 0001. <ul style="list-style-type: none"> ▶ Case Authentication Method 0001: Enables the use of SA Authentication Method 0001 procedure for the Security Access Authentication. ▶ Case Authentication Method Disable: Disables the SA Authentication Method usage.
Multiplicity	1..1
Type	ENUMERATION
Default value	Authentication Method Disable
Range	Authentication Method 0001 Authentication Method Disable
Origin	EB

Parameter Name	SA2_Awaiting_MaxTime
Label	SA2 Awaiting Time



Description	Max allowed time for sending the SA2 request after the transmission of the SA1 response. This time unit and the actual SA2 wait time shall depend on 'SA Manage Period'. (Formula: Actual SA2 Wait time = Wait time entered in the field 'SA2 Awaiting Time' * SA Manage Period).
Multiplicity	1..1
Type	INTEGER
Range	>=1
Origin	EB

4.7.1.4. CsmRandomGenerate

Parameters included	
Parameter name	Multiplicity
CsmRandomGenerateConfigId	1..1
Parameter Name	
Label	CsmRandomGenerateConfigId
Description	Reference a <i>CsmRandomGenerate</i> Dependencies: ▶ Reference shall be valid
Multiplicity	1..1
Type	REFERENCE
Origin	Elektrobit Automotive GmbH

4.7.1.5. CsmRandomSeed

Parameters included	
Parameter name	Multiplicity
CsmRandomSeedConfigId	1..1
Allow2Cancel_OngoingJobs	1..1
Parameter Name	
Label	CsmRandomSeedConfigId



Description	Reference a <i>CsmRandomSeed</i> Dependencies: ▶ Reference shall be valid
Multiplicity	1..1
Type	REFERENCE
Origin	Elektrobit Automotive GmbH

Parameter Name	Allow2Cancel_OngoingJobs
Label	Allow cancellation of any ongoing Crypto job
Description	If enabled, any ongoing or active Crypto job operations shall be cancelled at the start of this job operation. This option is applicable only for the integration of Crypto ASR 4.3 or higher versions, for older version this feature is realized using 'Enable the cancelation of ongoing requests' in the Csm/Cry modules.
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

4.7.1.6. SignatureVerify

Parameters included	
Parameter name	Multiplicity
CsmSignatureVerifyConfigId	1..1
Allow2CustomCsmStartPreprocess	1..1

Parameter Name	CsmSignatureVerifyConfigId
Label	CsmSignatureVerifyConfigId
Description	Reference a <i>CsmSignatureVerify</i> Dependencies: ▶ Reference shall be valid
Multiplicity	1..1
Type	REFERENCE



Origin	Elektrobit Automotive GmbH
Parameter Name	Allow2CustomCsmStartPreprocess
Label	Custom CSM start pre-processing
Description	<p>This will allow the integrator to perform the encoding (eg. Base64, DER etc.) on the Key stored at the host side at the start of the Crypto operation</p> <p>This option is applicable only for the integration of Crypto ASR 4.3 or higher versions.</p> <p>Note: The key encoding shall be performed only if the Crypto driver expects the key to be encoded in an expected format.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

4.7.1.7. CsmMACVerification_MsgAuth

Parameters included	
Parameter name	Multiplicity
CsmMACVerification_MsgAuthConfigId	1..1
Allow2Cancel_OngoingJobs	1..1
Allow2SetCryptoKey	1..1

Parameter Name	CsmMACVerification_MsgAuthConfigId
Label	CsmMacVerify_MsgAuthConfigId
Description	<p>Reference a <i>CsmMacVerifyConfig</i> for the Verification of Proof of Ownership.</p> <p>Dependencies:</p> <ul style="list-style-type: none"> ▶ Reference shall be valid
Multiplicity	1..1
Type	REFERENCE
Origin	Elektrobit Automotive GmbH

Parameter Name	Allow2Cancel_OngoingJobs
Label	Allow cancellation of any ongoing Crypto job



Description	If enabled, any ongoing or active Crypto job operations shall be cancelled at the start of this job operation. This option is applicable only for the integration of Crypto ASR 4.3 or higher versions, for older version this feature is realized using 'Enable the cancelation of ongoing requests' in the Csm/Cry modules.
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	Allow2SetCryptoKey
Label	Allow to fetch the stored Key
Description	This will allow the integrator to pass the desired key and its elements at the start of the Crypto operation. This option is applicable only for the integration of Crypto ASR 4.3 or higher versions.
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

4.7.1.8. CsmMACGenerate_MsgAuth

Parameters included	
Parameter name	Multiplicity
CsmMACGenerate_MsgAuthConfigId	1..1
Allow2SetCryptoKey	1..1

Parameter Name	CsmMACGenerate_MsgAuthConfigId
Label	CsmMacGen_MsgAuthConfigId
Description	Reference a <i>CsmMacGenerateConfig</i> for the Generation of Mac SA Message Authentication Code. Dependencies: ▶ Reference shall be valid



Multiplicity	1..1
Type	REFERENCE
Origin	Elektrobit Automotive GmbH

Parameter Name	Allow2SetCryptoKey
Label	Allow to fetch the stored Key
Description	<p>This will allow the integrator to pass the desired key and its elements at the start of the Crypto operation.</p> <p>This option is applicable only for the integration of Crypto ASR 4.3 or higher versions.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

4.7.1.9. CsmAES_Encryption

Parameters included	
Parameter name	Multiplicity
CsmAES_EncryptionConfigId	1..1
AllowECB2CTR	1..1
Allow2SetCryptoKey	1..1

Parameter Name	CsmAES_EncryptionConfigId
Label	CsmAES_EncryptionConfigId
Description	<p>Reference a <i>CsmSymEncrypt</i> for the Encryption of Proof of Ownership.</p> <p>Dependencies:</p> <ul style="list-style-type: none"> ▶ Reference shall be valid
Multiplicity	1..1
Type	REFERENCE
Origin	Elektrobit Automotive GmbH

Parameter Name	AllowECB2CTR
Label	Enable ECB2CTR Conversion



Description	This will allow the integrator to enable ECB2CTR conversion. This option is applicable only for the integration of Crypto ASR 4.3 or higher versions.
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	Allow2SetCryptoKey
Label	Allow to fetch the stored Key
Description	This will allow the integrator to pass the desired key and its elements at the start of the Crypto operation. This option is applicable only for the integration of Crypto ASR 4.3 or higher versions.
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

4.7.1.10. CsmAES_Decryption

Parameters included	
Parameter name	Multiplicity
CsmAES_DecryptionConfigId	1..1
AllowECB2CTR	1..1
Allow2SetCryptoKey	1..1

Parameter Name	CsmAES_DecryptionConfigId
Label	CsmAES_DecryptionConfigId
Description	Reference a <i>CsmSymDecrypt</i> for the Decryption of Proof of Ownership. Dependencies: ▶ Reference shall be valid
Multiplicity	1..1



Type	REFERENCE
Origin	Elektrobit Automotive GmbH

Parameter Name	AllowECB2CTR
Label	Enable ECB2CTR Conversion
Description	<p>This will allow the integrator to enable ECB2CTR conversion.</p> <p>This option is applicable only for the integration of Crypto ASR 4.3 or higher versions.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	Allow2SetCryptoKey
Label	Allow to fetch the stored Key
Description	<p>This will allow the integrator to pass the desired key and its elements at the start of the Crypto operation.</p> <p>This option is applicable only for the integration of Crypto ASR 4.3 or higher versions.</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

4.7.1.11. CsmMACVerification_Poo

Parameters included	
Parameter name	Multiplicity
CsmMACVerification_PooConfigId	1..1
Allow2Cancel_OngoingJobs	1..1
Allow2SetCryptoKey	1..1

Parameter Name	CsmMACVerification_PooConfigId
Label	CsmMacVerify_PooConfigId



Description	Reference a <i>CsmMacVerifyConfig</i> for the Verification of Proof of Ownership. Dependencies: <ul style="list-style-type: none">▶ Reference shall be valid
Multiplicity	1..1
Type	REFERENCE
Origin	Elektrobit Automotive GmbH

Parameter Name	Allow2Cancel_OngoingJobs
Label	Allow cancellation of any ongoing Crypto job
Description	If enabled, any ongoing or active Crypto job operations shall be cancelled at the start of this job operation. This option is applicable only for the integration of Crypto ASR 4.3 or higher versions, for older version this feature is realized using 'Enable the cancelation of ongoing requests' in the Csm/Cry modules.
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	Allow2SetCryptoKey
Label	Allow to fetch the stored Key
Description	This will allow the integrator to pass the desired key and its elements at the start of the Crypto operation. This option is applicable only for the integration of Crypto ASR 4.3 or higher versions.
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

4.7.1.12. CsmMACGenerate_Poo

Parameters included	
Parameter name	Multiplicity



Parameters included

CsmMACGenerate_PooConfigId	1..1
Allow2SetCryptoKey	1..1

Parameter Name	CsmMACGenerate_PooConfigId
Label	CsmMacGen_PooConfigId
Description	Reference a <i>CsmMacGenerateConfig</i> for the Generation of Proof of Ownership. Dependencies: <ul style="list-style-type: none">▶ Reference shall be valid
Multiplicity	1..1
Type	REFERENCE
Origin	Elektrobit Automotive GmbH

Parameter Name	Allow2SetCryptoKey
Label	Allow to fetch the stored Key
Description	This will allow the integrator to pass the desired key and its elements at the start of the Crypto operation. This option is applicable only for the integration of Crypto ASR 4.3 or higher versions.
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

4.7.2. Application programming interface (API)

4.7.2.1. Type definitions

4.7.2.1.1. tAntiscanInfo

Purpose	
----------------	--



Type	struct
Members	u8 ubRetryCnt
	u32 ulLockTimer
	tSaBoolean ubAsLocked

4.7.2.1.2. tDecompressStateType

Purpose	
Type	u8

4.7.2.1.3. tLimitReqSeedInfo

Purpose	
Type	struct
Members	u8 ubRequestCounter
	u32 ultimer

4.7.2.1.4. tSACsmJobConf

Purpose	
Type	struct
Members	u32 ubCsmJobId
	u32 ubCsmKeyID
	u32 ubCryptoElementID
	u32 ubCryptoKeyLength
	boolean ubAllowJobCancellation
	boolean ubAllowKeySet
	boolean ubAllowCustCsmPreProc

4.7.2.1.5. tSA_AESCTRAlgo

Purpose	
---------	--



Type	struct
Members	<pre>Csm_SymKeyType const * CsmKeyP- tr u32 IVLen u32 CTLen u32 PTLen u32 * PTLenptr u32 * CTLenptr u8 * IVptr u8 * PTptr u8 * CTptr u8 CsmCfdID</pre>

4.7.2.1.6. tSA.AuthenticationState

Purpose	
Type	u32

4.7.2.1.7. tSA.ConcatenatePar

Purpose	
Type	struct
Members	<pre>u32 DataLength u8 * DataPtr</pre>

4.7.2.1.8. tSA.MACAlgo

Purpose	
Type	struct
Members	<pre>Csm_SymKeyType const * CsmKeyP- tr u32 DataLen u32 MACLen</pre>



	u32 * MACLenPtr	
	u8 * DataPtr	
	u8 * MACPtr	
	u8 CsmCfdID	

4.7.2.1.9. tSaBoolean

Purpose	
Type	u8

4.7.2.1.10. tSaCsmState

Purpose	
Type	u8

4.7.2.1.11. tSaStatus

Purpose	
Type	u8

4.7.2.2. Macro constants

4.7.2.2.1. CSM_E_BUSY

Purpose	
Value	CRYPTO_E_BUSY

4.7.2.2.2. CSM_E_NOT_OK

Purpose	
---------	--



Value	E_NOT_OK
--------------	----------

4.7.2.2.3. CSM_E_OK

Purpose	
Value	E_OK

4.7.2.2.4. CSM_E_VER_NOT_OK

Purpose	
Value	CRYPTO_E_VER_NOT_OK

4.7.2.2.5. CSM_E_VER_OK

Purpose	
Value	CRYPTO_E_VER_OK

4.7.2.2.6. Csm_ReturnType

Purpose	
Value	Std_ReturnType

4.7.2.2.7. Csm_VerifyResultType

Purpose	
Value	Crypto_VerifyResultType

4.7.2.2.8. LZSS_BREAK EVEN

Purpose	
Value	(u8)((1U + LZSS_INDEX_BIT_COUNT + LZSS_LENGTH_BIT_COUNT) / 9U)



4.7.2.2.9. LZSS_END_OF_STREAM

Purpose	
Value	(u8)0U

4.7.2.2.10. LZSS_INDEX_BIT_COUNT

Purpose	
Value	(u8)10U

4.7.2.2.11. LZSS_LENGTH_BIT_COUNT

Purpose	
Value	(u8)4U

4.7.2.2.12. LZSS_MOD_WINDOW

Purpose	
Value	(u16)((a) & (LZSS_WINDOW_SIZE - 1U))

4.7.2.2.13. LZSS_WINDOW_SIZE

Purpose	
Value	(u16)((u16)(1U) << LZSS_INDEX_BIT_COUNT)

4.7.2.2.14. SA1_AESCTR_CTOFFSET_REQ

Purpose	
Value	22U

4.7.2.2.15. SA1_AESCTR_CTOFFSET_RES

Purpose	
---------	--



Value	20U
--------------	-----

4.7.2.2.16. SA1_AESCTR_IVOFFSET

Purpose	
Value	6U

4.7.2.2.17. SA1_AESCTR_IVOFFSET_RES

Purpose	
Value	4U

4.7.2.2.18. SA1_ENCRYPTION_DATALEN

Purpose	
Value	32U

4.7.2.2.19. SA1_MSGID

Purpose	
Value	0x0001U

4.7.2.2.20. SA1_MSGREQ_SIZE

Purpose	
Value	54U

4.7.2.2.21. SA1_MSGRES_SIZE

Purpose	
Value	68U



4.7.2.2.22. SA1_MSGSIZE_WITHOUT_MACVALUE

Purpose	
Value	(SA1_MSGREQ_SIZE - SA_MSGAUTHCODE_MACSIZE)

4.7.2.2.23. SA1_RESPONSE_MSGID

Purpose	
Value	0x0002U

4.7.2.2.24. SA1_RESSIZE_WITHOUT_MACVALUE

Purpose	
Value	(SA1_MSGRES_SIZE - SA_MSGAUTHCODE_MACSIZE)

4.7.2.2.25. SA2_AESCTR_CTOFFSET

Purpose	
Value	20U

4.7.2.2.26. SA2_AESCTR_IVOFFSET

Purpose	
Value	4U

4.7.2.2.27. SA2_AESCTR_PTLEN

Purpose	
Value	16U

4.7.2.2.28. SA2_MAX_AWAIT_TIME

Purpose	
---------	--



Value	[!"num:i(num:i(General/SA2_Awaiting_MaxTime) * num:i(General/MAN-AGE_PERIOD))"!]U
--------------	---

4.7.2.2.29. SA2_MSGID

Purpose	
Value	0x0003U

4.7.2.2.30. SA2_MSGREQ_SIZE

Purpose	
Value	52U

4.7.2.2.31. SA2_MSGRES_SIZE

Purpose	
Value	2U

4.7.2.2.32. SA2_MSGSIZE_WITHOUT_MACVALUE

Purpose	
Value	(SA2_MSGREQ_SIZE - SA_MSGAUTHCODE_MACSIZE)

4.7.2.2.33. SA_AESCTR_BLOCKSIZE

Purpose	
Value	16U

4.7.2.2.34. SA_AESCTR_IVLEN

Purpose	
----------------	--



Value	16U
--------------	-----

4.7.2.2.35. SA_AES_DECRYPT_FINISH

Purpose	
Value	0x16U

4.7.2.2.36. SA_AES_DECRYPT_START

Purpose	
Value	0x18U

4.7.2.2.37. SA_AES_DECRYPT_UPDATE

Purpose	
Value	0x17U

4.7.2.2.38. SA_AES_ENCRYPT_FINISH

Purpose	
Value	0x19U

4.7.2.2.39. SA_AES_ENCRYPT_START

Purpose	
Value	0x1BU

4.7.2.2.40. SA_AES_ENCRYPT_UPDATE

Purpose	
Value	0x1AU



4.7.2.2.41. SA_AM0001_IDLE

Purpose	
Value	0x00000000U

4.7.2.2.42. SA_AM0001_MAXBUF

Purpose	
Value	64U

4.7.2.2.43. SA_AM0001_PROCESSING_FAILED

Purpose	
Value	0x09U

4.7.2.2.44. SA_AM0001_STATE_BUSY

Purpose	
Value	0x40000000U

4.7.2.2.45. SA_AM0001_STATE_FAILED

Purpose	
Value	0x80000000U

4.7.2.2.46. SA_ANTISCANNING_ENABLED

Purpose	
Value	STD_ON

4.7.2.2.47. SA_AS_LOCK_TIMER

Purpose	
---------	--



Value	[!"num:i(General/Security_Access_As_Timer div General/MANAGE_PERIOD)"!]U
--------------	--

4.7.2.2.48. SA_AS_MAX_NB_RETRY

Purpose	
Value	[!"num:i(General/Security_Access_As_Retry_Counter)"!]U

4.7.2.2.49. SA_AUTHENTICATION_0001

Purpose	
Value	0x0001U

4.7.2.2.50. SA_AUTHENTICATION_METHOD

Purpose	
Value	SA_AUTHENTICATION_0001

4.7.2.2.51. SA_AUTHENTICATION_OFF

Purpose	
Value	0xFFFFFU

4.7.2.2.52. SA_AWAITS A2_TIMEEXPIRED

Purpose	
Value	0x80000200U

4.7.2.2.53. SA_AWAIT_SA2REQ

Purpose	
Value	0x00000100U



4.7.2.2.54. SA_AWAIT_SA2REQ_BUSY

Purpose	
Value	0x40000200U

4.7.2.2.55. SA_CHALLENGE_BIT

Purpose	
Value	64U

4.7.2.2.56. SA_CLIENT_POO_SIZE

Purpose	
Value	16U

4.7.2.2.57. SA_CLIENT_RNDNUM_SIZE

Purpose	
Value	16U

4.7.2.2.58. SA_COMPARE_KEY_AM0001

Purpose	
Value	0x02U

4.7.2.2.59. SA_COMPARE_KEY_STANDARD

Purpose	
Value	0x00U

4.7.2.2.60. SA_COMPARE_KEY_TYPE

Purpose	
---------	--



Value	[!WS "20"] [!IF "General/Compare_Key_Type = 'Verify_Signature'"!]SA_COMPARE_KEY_VERIFY_SIGNATURE
--------------	--

4.7.2.2.61. SA_COMPARE_KEY_VERIFY_SIGNATURE

Purpose	
Value	0x01U

4.7.2.2.62. SA_COMPRESSION_DISABLED

Purpose	
Value	0x02U

4.7.2.2.63. SA_COMPRESSION_ENABLED

Purpose	
Value	0x01U

4.7.2.2.64. SA_COMPRESSION_STATE

Purpose	
Value	SA_COMPRESSION_ENABLED

4.7.2.2.65. SA_CRYPTO_AESDECRYPT_KEYELEID

Purpose	
Value	[!WS "12"] [!"(as:ref(as:ref(as:ref(as:ref(as:ref(as:ref(as:modconf('SA')/General/CsmAES_Decryption/CsmaES_DecryptionConfigId)/CsmJobKeyRef)/CsmKeyRef)/CryIfKeyRef)/CryptoKeyTypeRef)/CryptoKeyElementRef/*[number(\$KEYELEPOSNUM)])]/CryptoKeyElementId)"]U

4.7.2.2.66. SA_CRYPTO_AESDECRYPT_KEYSIZE

Purpose	
----------------	--



Value	[!WS "13"!][!"(as:ref(as:ref(as:ref(as:ref(as:ref(as:ref(as:modconf('SA')/General/CsmAES_Decryption/CsmAES_DecryptionConfigId)/CsmJobKeyRef)/CsmKeyRef/CryIfKeyRef)/CryptoKeyTypeRef)/CryptoKeyElementRef/*[number(\$KEYELEPONUM)])/CryptoKeyElementSize)"!]U
--------------	---

4.7.2.2.67. SA_CRYPTO_AESENCRYPT_KEYELEID

Purpose	
Value	[!WS "12"!][!"(as:ref(as:ref(as:ref(as:ref(as:ref(as:ref(as:modconf('SA')/General/CsmAES_Encryption/CsmAES_EncryptionConfigId)/CsmJobKeyRef)/CsmKeyRef/CryIfKeyRef)/CryptoKeyTypeRef)/CryptoKeyElementRef/*[number(\$KEYELEPONUM)])/CryptoKeyElementId)"!]U

4.7.2.2.68. SA_CRYPTO_AESENCRYPT_KEYSIZE

Purpose	
Value	[!WS "13"!][!"(as:ref(as:ref(as:ref(as:ref(as:ref(as:ref(as:modconf('SA')/General/CsmAES_Encryption/CsmAES_EncryptionConfigId)/CsmJobKeyRef)/CsmKeyRef/CryIfKeyRef)/CryptoKeyTypeRef)/CryptoKeyElementRef/*[number(\$KEYELEPONUM)])/CryptoKeyElementSize)"!]U

4.7.2.2.69. SA_CRYPTO_MACGENPOO_KEYELEID

Purpose	
Value	[!WS "12"!][!"(as:ref(as:ref(as:ref(as:ref(as:ref(as:ref(as:modconf('SA')/General/CsmMACGenerate_Poo/CsmMACGenerate_PooConfigId)/CsmJobKeyRef)/CsmKeyRef/CryIfKeyRef)/CryptoKeyTypeRef)/CryptoKeyElementRef/*[number(\$KEYELEPONUM)])/CryptoKeyElementId)"!]U

4.7.2.2.70. SA_CRYPTO_MACGENPOO_KEYSIZE

Purpose	
Value	[!WS "13"!][!"(as:ref(as:ref(as:ref(as:ref(as:ref(as:ref(as:modconf('SA')/General/CsmMACGenerate_Poo/CsmMACGenerate_PooConfigId)/CsmJobKeyRef)/CsmKeyRef/CryIfKeyRef)/CryptoKeyTypeRef)/CryptoKeyElementRef/*[number(\$KEYELEPONUM)])/CryptoKeyElementSize)"!]U



4.7.2.2.71. SA_CRYPTO_MACGEN_KEYELEID

Purpose	
Value	[!WS "12"!][!"(as:ref(as:ref(as:ref(as:ref(as:ref(as:modconf('SA')/General/CsmMACGenerate_MsgAuth/CsmMACGenerate_MsgAuthConfigId)/CsmJobKeyRef)/CsmKeyRef)/CryIfKeyRef)/CryptoKeyTypeRef)/CryptoKeyElementRef/*[number(\$KEYELEPONUM)])]/CryptoKeyElementId)"!]U

4.7.2.2.72. SA_CRYPTO_MACGEN_KEYSIZE

Purpose	
Value	[!WS "13"!][!"(as:ref(as:ref(as:ref(as:ref(as:ref(as:modconf('SA')/General/CsmMACGenerate_MsgAuth/CsmMACGenerate_MsgAuthConfigId)/CsmJobKeyRef)/CsmKeyRef)/CryIfKeyRef)/CryptoKeyTypeRef)/CryptoKeyElementRef/*[number(\$KEYELEPONUM)])]/CryptoKeyElementSize)"!]U

4.7.2.2.73. SA_CRYPTO_MACVERIFYPOO_KEYELEID

Purpose	
Value	[!WS "12"!][!"(as:ref(as:ref(as:ref(as:ref(as:ref(as:modconf('SA')/General/CsmMACVerification_Poo/CsmMACVerification_PooConfigId)/CsmJobKeyRef)/CsmKeyRef)/CryIfKeyRef)/CryptoKeyTypeRef)/CryptoKeyElementRef/*[number(\$KEYELEPONUM)])]/CryptoKeyElementId)"!]U

4.7.2.2.74. SA_CRYPTO_MACVERIFYPOO_KEYSIZE

Purpose	
Value	[!WS "13"!][!"(as:ref(as:ref(as:ref(as:ref(as:ref(as:modconf('SA')/General/CsmMACVerification_Poo/CsmMACVerification_PooConfigId)/CsmJobKeyRef)/CsmKeyRef)/CryIfKeyRef)/CryptoKeyTypeRef)/CryptoKeyElementRef/*[number(\$KEYELEPONUM)])]/CryptoKeyElementSize)"!]U

4.7.2.2.75. SA_CRYPTO_MACVERIFY_KEYELEID

Purpose	
---------	--



Value	[!WS "12"!][!"(as:ref(as:ref(as:ref(as:ref(as:ref(as:ref(as:modconf('SA')/General/CsmMACVerification_MsgAuth/CsmMACVerification_MsgAuthConfigId)/CsmJobKeyRef)/CsmKeyRef)/CryIfKeyRef)/CryptoKeyTypeRef)/CryptoKeyElementRef/*[number(\$KEYELEPOSPNUM)])]/CryptoKeyElementId)"!]U
--------------	---

4.7.2.2.76. SA_CRYPTO_MACVERIFY_KEYSIZE

Purpose	
Value	[!WS "13"!][!"(as:ref(as:ref(as:ref(as:ref(as:ref(as:ref(as:modconf('SA')/General/CsmMACVerification_MsgAuth/CsmMACVerification_MsgAuthConfigId)/CsmJobKeyRef)/CsmKeyRef)/CryIfKeyRef)/CryptoKeyTypeRef)/CryptoKeyElementRef/*[number(\$KEYELEPOSPNUM)])]/CryptoKeyElementSize)"!]U

4.7.2.2.77. SA_CRYPTO_SIGN_KEYELEID

Purpose	
Value	[!"(as:ref(as:ref(as:ref(as:ref(as:ref(as:ref(as:modconf('SA')/General/SignatureVerify/CsmSignatureVerifyConfigId)/CsmJobKeyRef)/CsmKeyRef)/CryIfKeyRef)/CryptoKeyTypeRef)/CryptoKeyElementRef/*[number(\$KEYELEPOSPNUM)])]/CryptoKeyElementId)"!]U

4.7.2.2.78. SA_CRY_EXPONENT_ENABLED

Purpose	
Value	STD_OFF

4.7.2.2.79. SA_CSMASR43_USED

Purpose	
Value	STD_ON

4.7.2.2.80. SA_CSM_AESCTR_DECRYPTION_USED

Purpose	
Value	STD_ON



4.7.2.2.81. SA_CSM_AESCTR_DECRYPT_ID

Purpose	
Value	[!"(as:ref(as:modconf('SA')/General/CsmAES_Decryption/CsmAES_DecryptionConfigId)/CsmJobId)"]

4.7.2.2.82. SA_CSM_AESCTR_ENCRYPTION_USED

Purpose	
Value	STD_ON

4.7.2.2.83. SA_CSM_AESCTR_ENCRYPT_ID

Purpose	
Value	[!"(as:ref(as:modconf('SA')/General/CsmAES_Encryption/CsmAES_EncryptionConfigId)/CsmJobId)"]

4.7.2.2.84. SA_CSM_AESDECRYPT_KEYID

Purpose	
Value	[!WS "18"]![!"node:value(as:ref(as:ref(as:modconf('SA')/General/CsmAES_Decryption/CsmAES_DecryptionConfigId)/CsmJobKeyRef)/CsmKeyId)"]U

4.7.2.2.85. SA_CSM_AESENCRYPT_KEYID

Purpose	
Value	[!WS "18"]![!"node:value(as:ref(as:ref(as:modconf('SA')/General/CsmAES_Encryption/CsmAES_EncryptionConfigId)/CsmJobKeyRef)/CsmKeyId)"]U

4.7.2.2.86. SA_CSM_ALLJOBS_COUNT

Purpose	
Value	[!"num:i(count(as:modconf('Csm')/CsmJobs/CsmJob/*))"]U



4.7.2.2.87. SA_CSM_CANCELJOB_ENABLED

Purpose	
Value	STD_ON

4.7.2.2.88. SA_CSM_INDEX

Purpose	
Value	SA_CSMTABID_##job_name

4.7.2.2.89. SA_CSM_MACGENPOO_KEYID

Purpose	
Value	[!WS "18"]![!"node:value(as:ref(as:ref(as:modconf('SA')/General/CsmMACGenerate_Poo/CsmMACGenerate_PooConfigId)/CsmJobKeyRef)/CsmKeyId)"!]U

4.7.2.2.90. SA_CSM_MACGEN_KEYID

Purpose	
Value	[!WS "18"]![!"node:value(as:ref(as:ref(as:modconf('SA')/General/CsmMACGenerate_MsgAuth/CsmMACGenerate_MsgAuthConfigId)/CsmJobKeyRef)/CsmKeyId)"!]U

4.7.2.2.91. SA_CSM_MACVERIFYPOO_KEYID

Purpose	
Value	[!WS "18"]![!"node:value(as:ref(as:ref(as:modconf('SA')/General/CsmMACVerification_Poo/CsmMACVerification_PooConfigId)/CsmJobKeyRef)/CsmKeyId)"!]U

4.7.2.2.92. SA_CSM_MACVERIFY_KEYID

Purpose	
Value	[!WS "18"]![!"node:value(as:ref(as:ref(as:modconf('SA')/General/CsmMACVerification_MsgAuth/CsmMACVerification_MsgAuthConfigId)/CsmJobKeyRef)/CsmKeyId)"!]U



4.7.2.2.93. SA_CSM_MAC_GENERATION_USED

Purpose	
Value	STD_ON

4.7.2.2.94. SA_CSM_MAC_MSGCODEGEN_ID

Purpose	
Value	[!"(as:ref(as:modconf('SA')/General/CsmMACGenerate_MsgAuth/CsmMACGenerate_MsgAuthConfigId)/CsmJobId)"]

4.7.2.2.95. SA_CSM_MAC_MSGCODEVERIFY_ID

Purpose	
Value	[!"(as:ref(as:modconf('SA')/General/CsmMACVerification_MsgAuth/CsmMACVerification_MsgAuthConfigId)/CsmJobId)"]

4.7.2.2.96. SA_CSM_MAC_POOGEN_ID

Purpose	
Value	[!"(as:ref(as:modconf('SA')/General/CsmMACGenerate_Poo/CsmMACGenerate_PooConfigId)/CsmJobId)"]

4.7.2.2.97. SA_CSM_MAC_POOVRFY_ID

Purpose	
Value	[!"(as:ref(as:modconf('SA')/General/CsmMACVerification_Poo/CsmMACVerification_PooConfigId)/CsmJobId)"]

4.7.2.2.98. SA_CSM_MAC_VERIFICATION_USED

Purpose	
Value	STD_ON



4.7.2.2.99. SA_CSM_RANDOM_GENERATE_ID

Purpose	
Value	[!"(as:ref(as:modconf('SA')/General/CsmRandomGenerate/CsmRandomGenerateConfigId)/CsmJobId)"!]

4.7.2.2.100. SA_CSM_RANDOM_SEED_ID

Purpose	
Value	[!"(as:ref(as:modconf('SA')/General/CsmRandomSeed/CsmRandomSeedConfigId)/CsmJobId)"!]

4.7.2.2.101. SA_CSM_RANDOM_SEED_KEYID

Purpose	
Value	[!"node:value(as:ref(as:ref(as:modconf('SA')/General/CsmRandomSeed/CsmRandomSeedConfigId)/CsmJobKeyRef)/CsmKeyId)"!]U

4.7.2.2.102. SA_CSM_SETKEY_ENABLED

Purpose	
Value	STD_ON

4.7.2.2.103. SA_CSM_SIG_VERIFY_ID

Purpose	
Value	[!"(as:ref(as:modconf('SA')/General/SignatureVerify/CsmSignatureVerifyConfigId)/CsmJobId)"!]

4.7.2.2.104. SA_CSM_SIG_VERIFY_KEYID

Purpose	



Value	[!"node:value(as:ref(as:ref(as:modconf('SA')/General/SignatureVerify/CsmSignatureVerifyConfigId)/CsmJobKeyRef/CsmKeyId)"!]U
--------------	---

4.7.2.2.105. SA_CSM_STATE_INIT

Purpose	
Value	0x00U

4.7.2.2.106. SA_CUSTOMGETAESCTRKEY

Purpose	
Value	STD_ON

4.7.2.2.107. SA_CUSTOMGETMACKEY

Purpose	
Value	STD_ON

4.7.2.2.108. SA_CUSTOM_CSMSTARTPREPROCESS_ENABLED

Purpose	
Value	STD_ON

4.7.2.2.109. SA_Csm_MainFunction

Purpose	
Value	Crypto_MainFunction

4.7.2.2.110. SA_DECOMP_COMPLETE

Purpose	
----------------	--



Value	0x10U
--------------	-------

4.7.2.2.111. SA_DECOMP_COMPRESSLEN

Purpose	
Value	0x08U

4.7.2.2.112. SA_DECOMP_COMPRESSPOS

Purpose	
Value	0x04U

4.7.2.2.113. SA_DECOMP_FINISH

Purpose	
Value	0x40U

4.7.2.2.114. SA_DECOMP_INIT

Purpose	
Value	0x01U

4.7.2.2.115. SA_DECOMP_IN_PROGRESS

Purpose	
Value	0x20U

4.7.2.2.116. SA_DECOMP_OUT_BUF_SIZE

Purpose	
Value	[!"num:i(General/Decomp_Out_Buffer_size)"!]U



4.7.2.2.117. SA_DECOMP_UNCOMPRESSED

Purpose	
Value	0x02U

4.7.2.2.118. SA_DECRYPT_REQDATA_SA1

Purpose	
Value	0x00000002U

4.7.2.2.119. SA_DECRYPT_REQDATA_SA1_FAILED

Purpose	
Value	0x80000004U

4.7.2.2.120. SA_DECRYPT_REQDATA_SA2

Purpose	
Value	0x00000400U

4.7.2.2.121. SA_DECRYPT_REQDATA_SA2_FAILED

Purpose	
Value	0x80000800U

4.7.2.2.122. SA_ECB2CTR_ENABLED

Purpose	
Value	STD_ON

4.7.2.2.123. SA_ERR_NULL_PTR

Purpose	
---------	--



Value	0x80U
--------------	-------

4.7.2.2.124. SA_E_BUSY

Purpose	
Value	0x03U

4.7.2.2.125. SA_E_NOK_AS_LIMIT_RS

Purpose	
Value	0x06U

4.7.2.2.126. SA_E_NOK_AS_LOCKED

Purpose	
Value	0x05U

4.7.2.2.127. SA_E_NOK_INVALID_KEY

Purpose	
Value	0x04U

4.7.2.2.128. SA_E_NOK_RS_DELAY_LOCKED

Purpose	
Value	0x07U

4.7.2.2.129. SA_E_NOT_OK

Purpose	
Value	0x02U



4.7.2.2.130. SA_E_OK

Purpose	
Value	0x01U

4.7.2.2.131. SA_E_STATUS_UNKNOWN

Purpose	
Value	0x00U

4.7.2.2.132. SA_E_WAITING_SA2_NOT_OK

Purpose	
Value	SA_E_NOT_OK

4.7.2.2.133. SA_FALSE

Purpose	
Value	0U

4.7.2.2.134. SA_IDLE

Purpose	
Value	0x01U

4.7.2.2.135. SA_KEYUNLOCK

Purpose	
Value	0x00000800U

4.7.2.2.136. SA_KEYUNLOCK_FAILED

Purpose	
---------	--



Value	0x80001000U
--------------	-------------

4.7.2.2.137. SA_KEYUNLOCK_SUCCESS

Purpose	
Value	0x00001000U

4.7.2.2.138. SA_KEY_LEN

Purpose	
Value	SA_KEY_LEN_AM0001

4.7.2.2.139. SA_KEY_LEN_AM0001

Purpose	
Value	(SA2_MSGREQ_SIZE - 2U)

4.7.2.2.140. SA_LIMIT_NB_REQUEST_SEED_ENABLED

Purpose	
Value	STD_ON

4.7.2.2.141. SA_MACSIZE

Purpose	
Value	16U

4.7.2.2.142. SA_MAC_GENERATE_FINISH

Purpose	
Value	0x13U



4.7.2.2.143. SA_MAC_GENERATE_START

Purpose	
Value	0x15U

4.7.2.2.144. SA_MAC_GENERATE_UPDATE

Purpose	
Value	0x14U

4.7.2.2.145. SA_MAC_VERIFY_FINISH

Purpose	
Value	0x10U

4.7.2.2.146. SA_MAC_VERIFY_START

Purpose	
Value	0x12U

4.7.2.2.147. SA_MAC_VERIFY_UPDATE

Purpose	
Value	0x11U

4.7.2.2.148. SA_MANAGE_PERIOD

Purpose	
Value	[!"num:i(General/MANAGE_PERIOD)"!]U

4.7.2.2.149. SA_MSGAUTHCODE_MACSIZE

Purpose	
---------	--



Value	16U
--------------	-----

4.7.2.2.150. SA_PUBLIC_KEY_LENGTH

Purpose	
Value	32U

4.7.2.2.151. SA_RANDOM_GEN_STATE_GENERATE

Purpose	
Value	0x06U

4.7.2.2.152. SA_RANDOM_GEN_STATE_STANDBY

Purpose	
Value	0x07U

4.7.2.2.153. SA_RANDOM_NUMBER_LENGTH

Purpose	
Value	16U

4.7.2.2.154. SA_REQUEST_SEED_DELAY_TIMER_ENABLED

Purpose	
Value	STD_ON

4.7.2.2.155. SA_RS_DELAY_TIMER

Purpose	
Value	[!"num:i(General/Security_Access_Rs_Delay_Timer div General/MAN-AGE_PERIOD)"!]U



4.7.2.2.156. SA_RS_LIMIT_COUNTER

Purpose	
Value	[!"num:i(General/Security_Access_RS_Retry_Counter)"!]U

4.7.2.2.157. SA_RS_LIMIT_TIMER

Purpose	
Value	[!"num:i(General/Security_Access_RS_Timer div General/MANAGE_PERIOD)"!]U

4.7.2.2.158. SA_SA2AWATINGTIME_EXPIRED

Purpose	
Value	0x08U

4.7.2.2.159. SA_SECUREDKEYS_STATUSREAD

Purpose	
Value	STD_OFF

4.7.2.2.160. SA_SECURITY_ALGORITHM_CUSTOM

Purpose	
Value	0X01U

4.7.2.2.161. SA_SECURITY_ALGORITHM_STANDARD

Purpose	
Value	0X00U

4.7.2.2.162. SA_SECURITY_ALGORITHM_TYPE

Purpose	
---------	--



Value	[!WS "20"]![!IF "as:modconf('SA')/General/Security_Algorithm_Type = 'Standard'"!]SA_SECURITY_ALGORITHM_STANDARD[!ELSE!]SA_SECURITY_ALGORITHM_CUSTOM[!ENDIF!]
--------------	--

4.7.2.2.163. SA_SECURITY_ALGORITHM_UNDEFINED

Purpose	
Value	0xFFU

4.7.2.2.164. SA_SEEDGEN_MSGAUTHCODE

Purpose	
Value	0x00000040U

4.7.2.2.165. SA_SEEDGEN_MSGAUTHCODE_FAILED

Purpose	
Value	0x800000080U

4.7.2.2.166. SA_SEEDGEN_MSGAUTHCODE_SUCCESS

Purpose	
Value	0x000000080U

4.7.2.2.167. SA_SEEDGEN_RNDIV

Purpose	
Value	0x000000004U

4.7.2.2.168. SA_SEEDGEN_RNDIV_FAILED

Purpose	
----------------	--



Value	0x80000008U
--------------	-------------

4.7.2.2.169. SA_SEEDGEN_RNDSERVER

Purpose	
Value	0x00000008U

4.7.2.2.170. SA_SEEDGEN_RNDSERVER_FAILED

Purpose	
Value	0x80000010U

4.7.2.2.171. SA_SEEDGEN_SERVERPOO

Purpose	
Value	0x00000010U

4.7.2.2.172. SA_SEEDGEN_SERVERPOO_ENCRYBLK2

Purpose	
Value	0x40000020U

4.7.2.2.173. SA_SEEDGEN_SERVERPOO_ENCRYPT

Purpose	
Value	0x00000020U

4.7.2.2.174. SA_SEEDGEN_SERVERPOO_ENCRYPT_FAILED

Purpose	
Value	0x80000040U



4.7.2.2.175. SA_SEEDGEN_SERVERPOO_FAILED

Purpose	
Value	0x80000020U

4.7.2.2.176. SA_SEEDGEN_SERVERPOO_SUCCESS

Purpose	
Value	0x00000020U

4.7.2.2.177. SA_SEED_CSM_RANDOM

Purpose	
Value	0x01U

4.7.2.2.178. SA_SEED_GEN_STATE_INIT

Purpose	
Value	0x02U

4.7.2.2.179. SA_SEED_GEN_STATE_START

Purpose	
Value	0x03U

4.7.2.2.180. SA_SEED_GEN_STATE_UPDATE

Purpose	
Value	0x05U

4.7.2.2.181. SA_SEED_LEN

Purpose	
---------	--



Value	SA_SEED_LEN_AM0001
--------------	--------------------

4.7.2.2.182. SA_SEED_LEN_AM0001

Purpose	
Value	(SA1_MSGRES_SIZE - 2U)

4.7.2.2.183. SA_SEED_STANDARD

Purpose	
Value	0x00U

4.7.2.2.184. SA_SEED_TYPE

Purpose	
Value	[!WS "20"]![!IF "as:modconf('SA')/General/Seed_Type = 'Cryptographic_Random'"!]SA_SEED_CSM_RANDOM[!ELSE!]SA_SEED_STANDARD[!ENDIF!]

4.7.2.2.185. SA_SEND_SA1RES

Purpose	
Value	0x00000080U

4.7.2.2.186. SA_SEND_SA1RES_AWAIT_TXCONF

Purpose	
Value	0x40000100U

4.7.2.2.187. SA_SEND_SA1RES_FAILED

Purpose	
----------------	--



Value	0x80000100U
--------------	-------------

4.7.2.2.188. SA_SERVER_RNDNUM_SIZE

Purpose	
Value	16U

4.7.2.2.189. SA_SIGNATURE_CHECK_FINISH

Purpose	
Value	0x0AU

4.7.2.2.190. SA_SIGNATURE_CHECK_STANDBY

Purpose	
Value	0x0CU

4.7.2.2.191. SA_SIGNATURE_CHECK_START

Purpose	
Value	0x08U

4.7.2.2.192. SA_SIGNATURE_CHECK_UPDATE

Purpose	
Value	0x09U

4.7.2.2.193. SA_SIGNATURE_LENGTH

Purpose	
Value	SA_KEY_LEN



4.7.2.2.194. SA_STATIC_KEY_LEN

Purpose	
Value	5U

4.7.2.2.195. SA_STATIC_SEED_ENABLED

Purpose	
Value	STD_ON

4.7.2.2.196. SA_TRUE

Purpose	
Value	1U

4.7.2.2.197. SA_UDS_ACK

Purpose	
Value	0x40U

4.7.2.2.198. SA_USE_CRYPTO

Purpose	
Value	STD_ON

4.7.2.2.199. SA_VRFYMSG_AUTHCODE_SA1

Purpose	
Value	0x00000001U

4.7.2.2.200. SA_VRFYMSG_AUTHCODE_SA1_FAILED

Purpose	
---------	--



Value	0x80000002U
--------------	-------------

4.7.2.2.201. SA_VRFYMSG_AUTHCODE_SA2

Purpose	
Value	0x00000200U

4.7.2.2.202. SA_VRFYMSG_AUTHCODE_SA2_FAILED

Purpose	
Value	0x80000400U

4.7.2.3. Objects

4.7.2.3.1. m_astSACsmJobConf

Purpose	
Type	const tSACsmJobConf

4.7.2.3.2. m_aubSAPublicModulus

Purpose	
Type	const u8

4.7.2.3.3. m_aubSaAesKeyData

Purpose	
Type	const Csm_SymKeyType

4.7.2.3.4. m_aubSaMacKeyData

Purpose	
----------------	--



Type	const Csm_SymKeyType
------	----------------------

4.7.2.3.5. m_eSaCsmState

Purpose	
Type	tSaCsmState

4.7.2.3.6. m_eSaStatus

Purpose	
Type	tSaStatus

4.7.2.3.7. m_pubSAReqResptr

Purpose	
Type	u8 *

4.7.2.3.8. m_stAESCTRpar

Purpose	
Type	tSA_AESCTRAlgo

4.7.2.3.9. m_stLimitReqSeed

Purpose	
Type	tLimitReqSeedInfo

4.7.2.3.10. m_stMACpar

Purpose	
---------	--



Type	tSA_MACAlgo
------	-----------------------------

4.7.2.3.11. m_ubMacVerificationResult

Purpose	
Type	u8

4.7.2.3.12. m_ubSaAllKeysStatus

Purpose	
Type	tSaStatus

4.7.2.3.13. m_uISAPublicExponent

Purpose	
Type	const u32

4.7.2.4. Functions

4.7.2.4.1. SA_AM0001_GetState

Purpose	Retrieve the Current State.
Synopsis	tSA_AuthenticationState SA_AM0001_GetState (void);
Return Value	
Description	This function retrieves the current state of Authentication Method 0001 procedure.

4.7.2.4.2. SA_AM0001_SetState

Purpose	Set the Authentication Method 0001 State.
Synopsis	void SA_AM0001_SetState (tSA_AuthenticationState ulAuthentication0001State);
Description	This function sets the current state of the Authentication Method 0001 procedure.



4.7.2.4.3. SA_AM0001_SetStatus

Purpose	Set the status of the current state and set the response to be sent in the event of failure.
Synopsis	<code>void SA_AM0001_SetStatus (tSaStatus eSaStatus);</code>
Description	This function sets/resets the status of the current state by resetting the Busy status and, in the event of failure sets the State status to failure and decides the type of response to be sent.

4.7.2.4.4. SA_AuthMethod0001

Purpose	Manages the Authentication Method 0001 Procedure Sequence and Transitions.
Synopsis	<code>void SA_AuthMethod0001 (void);</code>
Description	This function triggers start of operation for every state of Authentication Method 0001, also sets the next state of operation.

4.7.2.4.5. SA_CompareKey

Purpose	Compare the received and calculated key.	
Synopsis	<code>tProgStatus SA_CompareKey (u8 * aubReceivedKey);</code>	
Parameters (in)	aubReceivedKey	input buffer with the received key from the network
Return Value	Result of comparison	
	PROG_STATUS_OK	Both keys are the same
	PROG_STATUS_NOT_OK	Both keys are different
Description	This function is called upon correct SA2 request. It compares the received key from the diagnostic request to the one calculated into SA_ComputeSeed. It then returns the result.	

4.7.2.4.6. SA_ConcatenateAndStoreData

Purpose	Concatenate the input block of data and store in a given output buffer.
Synopsis	<code>void SA_ConcatenateAndStoreData (tSA_ConcatenatePar * aubInputData , u32 ulInputbufCount , u8 * aubStore);</code>
Description	This function concatenates the input block of data and stores in a given output buffer.



4.7.2.4.7. SA_CsmNotification

Purpose	API is a unique callback called by CSM module to treat random generation, encryption, decryption, hash and signature verification.	
Synopsis	<pre>Std_ReturnType SA_CsmNotification (Std_ReturnType eCsmResult);</pre>	
Parameters (in)	eCsmResult	Csm treatment result
Return Value		

4.7.2.4.8. SA_CustomCsmStrtPreproc

Purpose	This Optional Callback used for the Custom Csm Implementation, called before the Csm Start mode of operation.							
Synopsis	<pre>Csm_ReturnType SA_CustomCsmStrtPreproc (const u8 ** ubKeyPtr , u32 ulCsmJobId);</pre>							
Parameters (in)	ulCsmJobId	is the Csm Job ID for which the key pointer could be updated or read						
Parameters (in,out)	ubKeyPtr	is the pointer to the address of the key buffer						
Return Value	<table border="1"> <tr> <td>state</td> <td></td> </tr> <tr> <td>E_OK</td><td>Treatment finished successfully</td> </tr> <tr> <td>E_NOT_OK</td><td>Error happened during treatment</td> </tr> </table>		state		E_OK	Treatment finished successfully	E_NOT_OK	Error happened during treatment
state								
E_OK	Treatment finished successfully							
E_NOT_OK	Error happened during treatment							
Description	<p>Callback is called: Before the start of Csm operation, after all successful job cancel operations(if cancellation is enabled)</p> <p>Callback shall implement: the Custom operation such as Key DER encoding or set IV or just the notification about the start of the crypto operation or any custom operation needed for the Integration.</p>							

4.7.2.4.9. SA_CustomGetAESCTRKey

Purpose	Get the AES key for AES Encryption and Decryption.	
Synopsis	<pre>void SA_CustomGetAESCTRKey (Csm_SymKeyType const * paubKeyData , u8 pubCsmCfgID);</pre>	
Parameters (out)	paubKeyData	Pointer to AES key modulus array



	pubCsmCfgID	for which the key shall be retrieved
--	-------------	--------------------------------------

4.7.2.4.10. SA_CustomGetAsymPublicKey

Purpose	Get the public key when using asymmetric cryptography. Used in SA_InitCrypto.	
Synopsis	<code>void SA_CustomGetAsymPublicKey (const u8 ** paubPublicModulus , u32 * pulPublicExponent);</code>	
Parameters (out)	paubPublicModulus	Pointer to public key modulus array
	pulPublicExponent	Pointer to public key exponent

4.7.2.4.11. SA_CustomGetLastRandomNumber

Purpose	Get the last random number stored.	
Synopsis	<code>void SA_CustomGetLastRandomNumber (u8 * aubDataRandomNumber);</code>	
Parameters (out)	aubDataRandomNumber	Generated random number array
Description	Called in SA_GenerateRandomCallback when Csm is in SA_SEED_GEN_STATE--START state	

4.7.2.4.12. SA_CustomGetMACKey

Purpose	Get the MAC key for MAC Verification and MAC Generation.	
Synopsis	<code>void SA_CustomGetMACKey (Csm_SymKeyType const * paubKeyData , u8 pubCsmCfgID);</code>	
Parameters (out)	paubKeyData	Pointer to MAC key modulus array
	pubCsmCfgID	for which the key shall be retrieved

4.7.2.4.13. SA_CustomRestoreAsRetryCnt

Purpose	API that restores the security access anti-scanning retry counter.	
Synopsis	<code>u8 SA_CustomRestoreAsRetryCnt (void);</code>	
Return Value	Value of security access anti-scanning retry counter	
Description	This callback is called at Bootloader startup (if anti-scanning feature is activated) to get the retry counter value from non-volatile memory.	



4.7.2.4.14. SA_CustomStoreAsRetryCnt

Purpose	API that stores the security access anti-scanning retry counter.	
Synopsis	<code>void SA_CustomStoreAsRetryCnt (u8 RetryCntValue);</code>	
Parameters (in)	RetryCntValue	Value of security access anti-scanning retry counter
Description	This callback is called on reception of SecurityAccess service in case anti-scanning feature is activated. The counter value provided as parameter shall be stored in non-volatile memory.	

4.7.2.4.15. SA_CustomStoreRandomNumber

Purpose	Store the generated random number.	
Synopsis	<code>void SA_CustomStoreRandomNumber (u8 * pubDataRandomNumber);</code>	
Parameters (in)	pubDataRandomNumber	Generated random number array
Description	Called in SA_GenerateRandomCallback when Csm is in SA_RANDOM_GEN_STATE_GENERATE state	

4.7.2.4.16. SA_DecomplInputParamInit

Purpose	reinit decompression input param	
Synopsis	<code>void SA_DecomplInputParamInit (void);</code>	
Description	This function Reinitializes the input decompression parameter after each TD	

4.7.2.4.17. SA_DecomplWriteDataConfirmation

Purpose	freed the written data from output buffer	
Synopsis	<code>void SA_DecomplWriteDataConfirmation (u16 uwLength);</code>	
Parameters (in)	uwLength	length written with sucess that shall be freed from output buffer
Description	This function allows to freed the written data from output buffer	



4.7.2.4.18. SA_DecompressData

Purpose	store the data to be decompress	
Synopsis	<code>void SA_DecompressData (u8 * pubData , PduLengthType ulDataLength);</code>	
Parameters (in)	pubData	Input buffer where the data to decompress are stored
	ulDataLength	Length of data to decompress from the buffer
Description	This function is used to store the data received from the network to an input buffer. The decompression will be done asynchronously afterward.	

4.7.2.4.19. SA_DecompressInit

Purpose	Decompression variable initialization.
Synopsis	<code>void SA_DecompressInit (void);</code>
Description	This function is called at init and for each request Download to intialize all decompression variable.

4.7.2.4.20. SA_DecompressManage

Purpose	decompress a byte of the input buffer
Synopsis	<code>void SA_DecompressManage (void);</code>
Description	This function Decompress a byte of the input buffer data with LZSS algorithm. Each decompressed byte is then store into an Output buffer

4.7.2.4.21. SA_GetDecompressedData

Purpose	Accessor to get the decompressed data.	
Synopsis	<code>tDecompressStateType SA_GetDecompressedData (u8 ** pubDecompData , PduLengthType * pulDecompressLength);</code>	
Parameters (out)	pubDecompData	Output buffer where the data decompressed are copied
	pulDecompressLength	Total length of data decompressed



Return Value	
Description	This function allows to get the decompressed data from the buffer

4.7.2.4.22. SA_GetSecuredKeysStatus

Purpose	Retrieves SA programmed Key status.
Synopsis	<code>tSaStatus SA_GetSecuredKeysStatus (void);</code>
Return Value	
Description	This function retrieves the status of the Stored secure-keys used for the SA Authentication Method 0001 procedure

4.7.2.4.23. SA_GetSeed

Purpose	Compute the key from an static key and a random Seed.	
Synopsis	<code>tSaStatus SA_GetSeed (u8 * aubSeed);</code>	
Parameters (out)	<code>aubSeed</code>	Output buffer to send back the random Seed
Return Value	Result of GetSeed action	
Description	This function is called upon correct SA1 request. It first calls SA_GetRandomValue to get a random value. Then it computes the key following an algorithm from this random seed and a static key predefined. The random generated Seed is then sent back to lower layer.	

4.7.2.4.24. SA_GetStatus

Purpose	API called by PROG module to get the SecurityAccess status.	
Synopsis	<code>tSaStatus SA_GetStatus (void);</code>	
Return Value		

4.7.2.4.25. SA_Init

Purpose	Initialize layer.
----------------	-------------------



Synopsis	<code>void SA_Init (void);</code>
Description	This function initializes the SA layer, shall be called only once at ECU startup

4.7.2.4.26. SA_Manage

Purpose	Manage the SA layer periodic task.
Synopsis	<code>void SA_Manage (void);</code>
Description	This function is the periodic function managing the SA layer

4.7.2.4.27. SA_SetSecuredKeyStatus

Purpose	Sets the SA programmed Key status.
Synopsis	<code>void SA_SetSecuredKeyStatus (Std_ReturnType eCsmResult);</code>
Description	This function sets the SA programmed Key status of the Authentication Method 0001 procedure.

4.7.3. Integration notes

4.7.3.1. Exclusive areas

Exclusive areas information is not available for this module.

4.7.3.2. Production errors

Production errors information is not available for this module.

4.7.3.3. Memory mapping

General information about memory mapping is provided in the EB tresos AutoCore Generic documentation. Refer to the section [Memory mapping and compiler abstraction](#) in the [Integration notes](#) section for details.

Memory mapping information is not available for this module.

4.7.3.4. Integration requirements

WARNING



Integration requirements list is not exhaustive

The following list of integration requirements helps you to integrate your product. However, this list is not exhaustive. You also require information from the user guide, release notes, and EB tresos AutoCore known issues to successfully integrate your product.

Integration requirements are not listed for the SA module.

4.8. Uds

4.8.1. Configuration parameters

Containers included		
Container name	Multiplicity	Description
CommonPublishedInformation	1..1	Label: Common Published Information Common container, aggregated by all modules. It contains published information about vendor and versions.
PublishedInformation	1..1	Label: EB Published Information Additional published parameters not covered by CommonPublishedInformation container.
General	1..1	
Session	8..8	Allows the configuration of 8 sessions. The "Name" column is a fixed value which has to be compliant with naming below: <ul style="list-style-type: none">▶ DEFAULT▶ PROGRAMMING▶ EXTENDED▶ SUPPLIER▶ OTHER_01▶ OTHER_02▶ OTHER_03▶ OTHER_04



Containers included		
		A total of 8 sessions can be configured. It is mandatory that these sessions are defined in the list.
<u>Service</u>	0..n	This container contains the standard service configuration
<u>Supplier_Services</u>	1..1	
<u>Service_DID</u>	0..n	This container contains the RDBI, WDBI, IOCBI and RSDBI service configuration.
<u>Routine_Control</u>	0..n	This container contains the Routine Control services configuration. This container contains the Routine Control services configuration.
<u>Service_OBD</u>	0..n	This container contains the OBD services configuration.

4.8.1.1. CommonPublishedInformation

Parameters included	
Parameter name	Multiplicity
<u>ArMajorVersion</u>	1..1
<u>ArMinorVersion</u>	1..1
<u>ArPatchVersion</u>	1..1
<u>SwMajorVersion</u>	1..1
<u>SwMinorVersion</u>	1..1
<u>SwPatchVersion</u>	1..1
<u>ModuleId</u>	1..1
<u>VendorId</u>	1..1
<u>Release</u>	1..1

Parameter Name	ArMajorVersion
Label	AUTOSAR Major Version
Description	Major version number of AUTOSAR specification on which the appropriate implementation is based on.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0



Configuration class	PublishedInformation:	
Origin	Elektrobit Automotive GmbH	

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

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

Parameter Name	SwMajorVersion
Label	Software Major Version
Description	Major version number of the vendor specific implementation of the module.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	3
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	SwMinorVersion
Label	Software Minor Version



Description	Minor version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	17
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	SwPatchVersion
Label	Software Patch Version
Description	Patch level version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	ModuleId
Label	Numeric Module ID
Description	Module ID of this module from Module List
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

Parameter Name	VendorId
Label	Vendor ID
Description	Vendor ID of the dedicated implementation of this module according to the AU-TOSAR vendor list
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	1



Configuration class	PublishedInformation:	
Origin	Elektrobit Automotive GmbH	

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

4.8.1.2. PublishedInformation

Parameters included	
Parameter name	Multiplicity
PbcfgMSupport	1..1

Parameter Name	PbcfgMSupport
Label	PbcfgM support
Description	Specifies whether or not the Uds can use the PbcfgM module for post-build support.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Configuration class	PublishedInformation:
Origin	Elektrobit Automotive GmbH

4.8.1.3. General

Parameters included	
Parameter name	Multiplicity
Standard	1..1
UDS_MANAGE_PERIOD	1..1



Parameters included

RDBI_MAX_RESPONSE_LENGTH	1..1
SecurityCheck	1..1
UDS_TIMEOUT_CHECK	1..1
SecurityFunction	1..1
RC_NRC_IMPLEMENTATION	1..1
DID_NRC_IMPLEMENTATION	1..1
UDS_MAX_DID_MULTI_RDBI	1..1
SPREC_IN_RESPONSE	1..1
RESPONSE_PENDING	1..1
TIMER_RESPONSE_PENDING_CHECK	1..1
RELOAD_TSTOPDIAG	1..1
Ext_ResponsePending_Manage_Call	1..1
UDS_P2_ADJUST	0..1
P2STAR_ADJUST	0..1

Parameter Name Standard

Description	Notify the variants OEM in which the plugin is used.
Multiplicity	1..1
Type	STRING
Default value	ISO
Range	ISO
Origin	EB

Parameter Name UDS_MANAGE_PERIOD

Description	Specifies the period of the manage task in ms. This period must be multiple of EB periodical value in EB module configuration.
Multiplicity	1..1
Type	INTEGER
Default value	2
Range	>=1
Origin	EB

Parameter Name RDBI_MAX_RESPONSE_LENGTH



Description	Specifies the max response length of any DID configured. This must be less the configured reception buffer length configured in BIPduR. Note: This parameter affects the stack size.
Multiplicity	1..1
Type	INTEGER
Default value	255
Range	>=1
Origin	EB

Parameter Name	SecurityCheck
Description	Security check feature: <ul style="list-style-type: none"> ▶ Activated: Enable API and internal code to manage NRC_33 (Security Access Denied) ▶ Deactivated: Disable API and internal code to manage NRC_33 (Security Access Denied)
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	UDS_TIMEOUT_CHECK
Description	This entry defines if a delay has to be waited before checking the application Validity. Note: <ul style="list-style-type: none"> ▶ it allows to start a new reprog session even if the application is valid, by receiving a DSC02 during this delay.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	SecurityFunction
Description	The name of the function which will be in charge to get the current security level. Note:



	<ul style="list-style-type: none"> ▶ Its prototype shall be: u8 SecurityFunction_Name(void); <p>Example:</p> <ul style="list-style-type: none"> ▶ PROG_GetSecurityLevel ▶ TELE_GetSecurityLevel ▶ ...
Multiplicity	1..1
Type	STRING
Origin	EB

Parameter Name	RC_NRC_IMPLEMENTATION
Description	<p>Only for Routine Control service (\$31)</p> <p>Define NRC used in case of Service and Sub-service are implemented but not in current session</p> <p>Range: [0x00 ; 0xFF]</p> <p>Default: NRC_31</p>
Multiplicity	1..1
Type	INTEGER
Default value	49
Range	<p><=255</p> <hr/> <p>>=0</p>
Origin	EB

Parameter Name	DID_NRC_IMPLEMENTATION
Description	<p>Only for service with DID</p> <p>Define NRC used in case of Service and DID are implemented but not in current session</p> <p>Range: [0x00 ; 0xFF]</p> <p>Default: NRC_31</p>
Multiplicity	1..1
Type	INTEGER
Default value	49
Range	<=255



	>=0
Origin	EB

Parameter Name	UDS_MAX_DID_MULTI_RDBI
Description	Define the maximum number of DIDs allowed into a request. Example: <ul style="list-style-type: none"> ▶ If a maximum of 4 DID is accepted into a request, then UDS_MAX_DID_MULTI_RDBI shall be set to 4.
Multiplicity	1..1
Type	INTEGER
Default value	1
Range	<=1000 >=1
Origin	EB

Parameter Name	SPREC_IN_RESPONSE
Description	SPREC (Session Parameter REcord) functionality SPREC (Session Parameter REcord) refers to UDS_P2 and UDS_P2STAR values <ul style="list-style-type: none"> ▶ Activated: SPREC parameter is sent within the DSC (\$10) positive response ▶ Deactivated: SPREC parameter not present in DSC (\$10) positive response
Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	RESPONSE_PENDING
Description	Response pending functionality <ul style="list-style-type: none"> ▶ Activated: Enable API and internal code to manage NRC_78 (response pending) ▶ Deactivated: Disable API and internal code to manage NRC_78 (response pending)



Multiplicity	1..1
Type	BOOLEAN
Default value	true
Origin	EB

Parameter Name	TIMER_RESPONSE_PENDING_CHECK
Description	<p>External NRC_78 (response pending) timeout increment, allowing increment under interrupt using hardware timer</p> <ul style="list-style-type: none"> ▶ Activated: NRC_78 timeout increment is managed externally by calling UDS_ResponsePending_TimCntManage() function ▶ Deactivated: NRC_78 timeout increment is managed internally in UDS_Manage call
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	RELOAD_TSTOPDIAG
Description	<p>Reload session timeout functionality</p> <p>API: void UDS_ReloadTStopDiag (void)</p> <p>This function can be called by the customer application, in order to maintain a non standard session opened</p> <p>It allows the ECU to maintain the current session for a duration equal to T_Stop_Diag (5 s).</p> <p>This functionality can be used for the NRC78 periodic answers, if the request duration is very long.</p> <ul style="list-style-type: none"> ▶ Activated: API UDS_ReloadTStopDiag available ▶ Deactivated: API UDS_ReloadTStopDiag unavailable
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	Ext_ResponsePending_Manage_Call
-----------------------	--



Label	External Response Pending Manage Call
Description	<p>This parameter allow the user to call the API UDS_ResponsePending_Manage by himself</p> <p>This allow to manage the response pending timer separately than the basic scheduling (e.g specific os task with higher priority)</p>
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

Parameter Name	UDS_P2_ADJUST
Label	P2 Adjust
Description	<p>This parameter is used to guarantee that the diagnostic response is available on the bus before reaching P2 by adjusting the current P2ServerMax</p> <p>$P2Timing = P2ServerMax - P2Adjust$</p> <p>Bootloader will send a response within P2Timing ms after receiving the request.</p> <p>The parameter value is defined in ms and must be a multiple of UDS_MANAGER_PERIOD.</p> <p>This timing is common to all sessions.</p>
Multiplicity	0..1
Type	INTEGER
Default value	0
Origin	EB

Parameter Name	P2STAR_ADJUST
Label	P2* Adjust
Description	<p>This parameter is used to guarantee that the diagnostic response is available on the bus before reaching P2Star by adjusting the current P2StarServerMax</p> <p>$P2StarTiming = P2StarServerMax - P2StarAdjust$</p> <p>Bootloader will send a response within P2StarTiming ms after the previous NRC78.</p> <p>The parameter value is defined in ms and must be a multiple of UDS_MANAGER_PERIOD.</p>



	This timing is common to all sessions.
Multiplicity	0..1
Type	INTEGER
Default value	0
Origin	EB

4.8.1.4. Session

Parameters included	
Parameter name	Multiplicity
Identifier	1..1
UDS_P2	1..1
UDS_P2STAR	1..1

Parameter Name	Identifier
Description	Specifies the hexadecimal value of the session Range:[0x00 ; 0xFF]
Multiplicity	1..1
Type	INTEGER
Origin	EB

Parameter Name	UDS_P2
Description	First transmission timing of the NRC78 after the diagnostic request. (ms) This time must be a multiple of UDS_MANAGE_PERIOD. This timing is common to all sessions.
Multiplicity	1..1
Type	INTEGER
Default value	50
Origin	EB

Parameter Name	UDS_P2STAR
Description	Periodic transmission of the NRC78. (ms)



	This time must be a multiple of UDS_MANAGE_PERIOD. This timing is common to all sessions.
Multiplicity	1..1
Type	INTEGER
Default value	5000
Origin	EB

4.8.1.5. Service

Parameters included	
Parameter name	Multiplicity
Service	1..1
SubService	1..1
Mode	1..1
Default	1..1
Programming	1..1
Extended	1..1
Supplier	1..1
Other_1	1..1
Other_2	1..1
Other_3	1..1
Other_4	1..1
Length	1..1
SecurityLevel	1..1
Callback	1..1
Callback_Origin	1..1

Parameter Name	Service
Description	<p>Specifies the service.</p> <ul style="list-style-type: none"> ▶ DSC:Diagnostic Session Control (\$10) ▶ ER:ECU Reset (\$11) ▶ SA:Security Access (\$27) ▶ CC:Communication Control (\$28)



	<ul style="list-style-type: none"> ▶ TP:Tester Present (\$3E) ▶ RTE:Request Transfert Exit (\$37) ▶ TD:Transfert Data (\$36) ▶ RU:Request Upload (\$35) ▶ RD:Request Download (\$34) ▶ RMBA:Read Memory By Address (\$23) ▶ WMBA:Write Memory By Address (\$3D) ▶ RDTCI:Read DTC Information (\$19) ▶ CDTCI:Clear Diagnostic Information (\$14) ▶ CDTCS:Control DTC Setting (\$85) ▶ LC:Link Control (\$87)
Multiplicity	1..1
Type	ENUMERATION
Default value	DSC
Range	DSC ER SA CC TP RTE TD RU RD RMBA WMBA RDTCI CDTCI CDTCS LC
Origin	EB

Parameter Name	SubService
Description	Specifies the sub-service.



	Range:[0x00 ; 0xFF]
Multiplicity	1..1
Type	INTEGER
Default value	0
Range	>=0 <=0xFF
Origin	EB

Parameter Name	Mode
Description	Addressing mode <ul style="list-style-type: none"> ▶ Physical: only physical addressing available for this request ▶ Functional: only functional addressing available for this request (broadcast) ▶ Both: physical and functional addressing available for this request
Multiplicity	1..1
Type	STRING
Default value	Physical
Range	Physical Functional Both
Origin	EB

Parameter Name	Default
Description	Switches ON if the Service/sub-service is implemented in default session.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0

Parameter Name	Programming
Description	Switches ON if the Service/sub-service is implemented in Programming session.
Multiplicity	1..1
Type	BOOLEAN
Default value	false



Origin	AUTOSAR_ECUC V1.0.0
Parameter Name	Extended
Description	Switches ON if the Service/sub-service is implemented in Extended session.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0
Parameter Name	Supplier
Description	Switches ON if the Service/sub-service is implemented in Supplier session.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0
Parameter Name	Other_1
Description	Switches ON if the Service/sub-service is implemented in Other_1 session.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0
Parameter Name	Other_2
Description	Switches ON if the Service/sub-service is implemented in Other_2 session.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0
Parameter Name	Other_3
Description	Switches ON if the Service/sub-service is implemented in Other_3 session.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0



Parameter Name	Other_4
Description	Switches ON if the Service/sub-service is implemented in Other_4 session.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0

Parameter Name	Length
Description	Specifies the length of the request. A null length disable the length filtering.
Multiplicity	1..1
Type	INTEGER
Default value	0
Range	>=0
Origin	EB

Parameter Name	SecurityLevel
Description	Specifies the security level required for the request. <ul style="list-style-type: none"> ▶ 0: no check required ▶ 1: SA1 / SA2 ▶ 2: SA3 / SA4 ▶ ...
Multiplicity	1..1
Type	INTEGER
Default value	0
Range	>=0 <=63
Origin	EB

Parameter Name	Callback
Description	Specifies the callback name. this callback will be called by UDS if the request is valid (session allowed + length correct + parameters correct)



Multiplicity	1..1
Type	STRING
Default value	
Origin	EB

Parameter Name	Callback_Origin
Description	Select the layer where the callback is defined: EB_cbk, APP_cbk or other one. <ul style="list-style-type: none"> ▶ EB: callback is defined in EB_Prg.c ▶ APP: callback is defined in APP_Prg.c ▶ OTHER: callback is defined in other file (not APP or EB)
Multiplicity	1..1
Type	STRING
Default value	APP
Range	EB APP OTHER
Origin	EB

4.8.1.6. Supplier_Services

Parameters included	
Parameter name	Multiplicity
BA	1..1
BB	1..1
BC	1..1
BD	1..1
BE	1..1

Parameter Name	BA
Description	The service with SID (service identifier) 0xBA is supplier defined. If you need to use this service, activate the current configuration variable. Then the UDS_CustomSupplier_BA callback will be provided to manage it.
Multiplicity	1..1



Type	BOOLEAN
Default value	false
Origin	EB
Parameter Name	BB
Description	The service with SID (service identifier) 0xBB is supplier defined. If you need to use this service, activate the current configuration variable. Then the UDS_CustomSupplier_BB callback will be provided to manage it.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB
Parameter Name	BC
Description	The service with SID (service identifier) 0xBC is supplier defined. If you need to use this service, activate the current configuration variable. Then the UDS_CustomSupplier_BC callback will be provided to manage it.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB
Parameter Name	BD
Description	The service with SID (service identifier) 0xBD is supplier defined. If you need to use this service, activate the current configuration variable. Then the UDS_CustomSupplier_BD callback will be provided to manage it.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB
Parameter Name	BE
Description	The service with SID (service identifier) 0xBE is supplier defined. If you need to use this service, activate the current configuration variable. Then the UDS_CustomSupplier_BE callback will be provided to manage it.



Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	EB

4.8.1.7. Service_DID

Parameters included	
Parameter name	Multiplicity
Service	1..1
DID	1..1
Mode	1..1
Default	1..1
Programming	1..1
Extended	1..1
Supplier	1..1
Other_1	1..1
Other_2	1..1
Other_3	1..1
Other_4	1..1
Length	1..1
SecurityLevel	1..1
Callback	1..1
Callback_Origin	1..1

Parameter Name	Service
Description	Specifies the service. <ul style="list-style-type: none"> ▶ RDBI:ReadDataByIdentifier (\$22) ▶ WDBI:WriteDataByIdentifier (\$2E) ▶ IOCB:InputOutputControlByIdentifier (\$2F) ▶ RSDBI:ReadScalingDataByIdentifier (\$24)
Multiplicity	1..1
Type	ENUMERATION



Default value	RDBI
Range	RDBI WDBI IOCB RSDBI
Origin	EB

Parameter Name	DID
Description	Specifies the DID. Range:[0x0000 ; 0xFFFF]
Multiplicity	1..1
Type	INTEGER
Default value	0
Range	>=0 <=0xFFFF
Origin	EB

Parameter Name	Mode
Description	Addressing mode <ul style="list-style-type: none"> ▶ Physical: only physical addressing available for this request ▶ Functional: only functional addressing available for this request (broadcast) ▶ Both: physical and functional addressing available for this request
Multiplicity	1..1
Type	STRING
Default value	Physical
Range	Physical Functional Both
Origin	EB

Parameter Name	Default
Description	Switches ON if the Service/sub-service is implemented in default session.



Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0

Parameter Name	Programming
Description	Switches ON if the Service/sub-service is implemented in Programming session.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0

Parameter Name	Extended
Description	Switches ON if the Service/sub-service is implemented in Extended session.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0

Parameter Name	Supplier
Description	Switches ON if the Service/sub-service is implemented in Supplier session.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0

Parameter Name	Other_1
Description	Switches ON if the Service/sub-service is implemented in Other_1 session.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0

Parameter Name	Other_2
Description	Switches ON if the Service/sub-service is implemented in Other_2 session.
Multiplicity	1..1



Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0

Parameter Name	Other_3
Description	Switches ON if the Service/sub-service is implemented in Other_3 session.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0

Parameter Name	Other_4
Description	Switches ON if the Service/sub-service is implemented in Other_4 session.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0

Parameter Name	Length
Description	Specifies the expected length to accept the request (SID + DID = 3). A null length disable the length filtering.
Multiplicity	1..1
Type	INTEGER
Default value	3
Range	>=0
Origin	EB

Parameter Name	SecurityLevel
Description	Specifies the security level required for the request. <ul style="list-style-type: none"> ▶ 0: no check required ▶ 1: SA1 / SA2 ▶ 2: SA3 / SA4 ▶ ...
Multiplicity	1..1



Type	INTEGER
Default value	0
Range	>=0 <=63
Origin	EB

Parameter Name	Callback
Description	Specifies the callback name. this callback will be called by UDS if the request is valid (session allowed + length correct + parameters correct)
Multiplicity	1..1
Type	STRING
Default value	
Origin	EB

Parameter Name	Callback_Origin
Description	Select the layer where the callback is defined: EB_cbk, APP_cbk or other one. <ul style="list-style-type: none"> ▶ EB: callback is defined in EB_Prg.c ▶ APP: callback is defined in APP_Prg.c ▶ OTHER: callback is defined in other file (not APP or EB)
Multiplicity	1..1
Type	STRING
Default value	APP
Range	EB APP OTHER
Origin	EB

4.8.1.8. Routine_Control

Parameters included	
Parameter name	Multiplicity
<u>SubService</u>	1..1



Parameters included

Mode	1..1
Routine_Identifier	1..1
Default	1..1
Programming	1..1
Extended	1..1
Supplier	1..1
Other_1	1..1
Other_2	1..1
Other_3	1..1
Other_4	1..1
Length	1..1
SecurityLevel	1..1
Callback	1..1
Callback_Origin	1..1

Parameter Name

SubService

Description	Only for PSA RoutineControlType ▶ 0x01: Start Routine ▶ 0x02: Stop Routine ▶ 0x03: Status Request
Multiplicity	1..1
Type	INTEGER
Default value	0
Range	>=0
Origin	EB

Parameter Name

Mode

Description	Addressing mode ▶ Physical: only physical addressing available for this request ▶ Functional: only functional addressing available for this request (broadcast)
--------------------	--



	► Both: physical and functional addressing available for this request
Multiplicity	1..1
Type	STRING
Default value	Physical
Range	Physical
	Functional
	Both
Origin	EB
Parameter Name	Routine_Identifier
Description	Specifies the routine identifier. Range:[0x0000 ; 0xFFFF] Specifies the routine identifier.
Multiplicity	1..1
Type	INTEGER
Default value	0
Range	>=0
	<=0xFFFF
Origin	EB
Parameter Name	Default
Description	Switches ON if the Service/sub-service is implemented in default session.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0
Parameter Name	Programming
Description	Switches ON if the Service/sub-service is implemented in Programming session.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0
Parameter Name	Extended



Description	Switches ON if the Service/sub-service is implemented in Extended session.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0
Parameter Name	Supplier
Description	Switches ON if the Service/sub-service is implemented in Supplier session.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0
Parameter Name	Other_1
Description	Switches ON if the Service/sub-service is implemented in Other_1 session.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0
Parameter Name	Other_2
Description	Switches ON if the Service/sub-service is implemented in Other_2 session.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0
Parameter Name	Other_3
Description	Switches ON if the Service/sub-service is implemented in Other_3 session.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0
Parameter Name	Other_4
Description	Switches ON if the Service/sub-service is implemented in Other_4 session.



Multiplicity	1..1
Type	BOOLEAN
Default value	false
Origin	AUTOSAR_ECUC V1.0.0

Parameter Name	Length
Description	Specifies the routine control length to accept the request (SID + subfonction + RI = 4). A null length disable the length filtering.
Multiplicity	1..1
Type	INTEGER
Default value	4
Range	>=0
Origin	EB

Parameter Name	SecurityLevel
Description	Specifies the security level required for the request. <ul style="list-style-type: none"> ▶ 0: no check required ▶ 1: SA1 / SA2 ▶ 2: SA3 / SA4 ▶ ...
Multiplicity	1..1
Type	INTEGER
Default value	0
Range	>=0 <=63
Origin	EB

Parameter Name	Callback
Description	Specifies the callback name. this callback will be called by UDS if the request is valid (session allowed + length correct + parameters correct)
Multiplicity	1..1
Type	STRING



Default value	
Origin	EB
Parameter Name	Callback_Origin
Description	Select the layer where the callback is defined: EB_cbk, APP_cbk or other one. <ul style="list-style-type: none"> ▶ EB: callback is defined in EB_Prg.c ▶ APP: callback is defined in APP_Prg.c ▶ OTHER: callback is defined in other file (not APP or EB)
Multiplicity	1..1
Type	STRING
Default value	APP
Range	EB APP OTHER
Origin	EB

4.8.1.9. Service_OBD

Parameters included	
Parameter name	Multiplicity
Service	1..1
Mode	1..1
Length	1..1
Callback	1..1
Callback_Origin	1..1

Parameter Name	Service
Description	Specifies the OBD service. <ul style="list-style-type: none"> ▶ OBD_SID_00 (\$00) ▶ OBD_SID_01 (\$01) ▶ OBD_SID_02 (\$02) ▶ OBD_SID_03 (\$03) ▶ OBD_SID_04 (\$04)



	<ul style="list-style-type: none">▶ OBD_SID_05 (\$05)▶ OBD_SID_06 (\$06)▶ OBD_SID_07 (\$07)▶ OBD_SID_08 (\$08)▶ OBD_SID_09 (\$09)▶ OBD_SID_0A (\$0A)▶ OBD_SID_0B (\$0B)▶ OBD_SID_0C (\$0C)▶ OBD_SID_0D (\$0D)▶ OBD_SID_0E (\$0E)▶ OBD_SID_0F (\$0F)
Multiplicity	1..1
Type	ENUMERATION
Default value	OBD_SID_00
Range	OBD_SID_00 OBD_SID_01 OBD_SID_02 OBD_SID_03 OBD_SID_04 OBD_SID_05 OBD_SID_06 OBD_SID_07 OBD_SID_08 OBD_SID_09 OBD_SID_10 OBD_SID_11 OBD_SID_12 OBD_SID_13 OBD_SID_14 OBD_SID_15
Origin	EB
Parameter Name	Mode



Description	Addressing mode <ul style="list-style-type: none"> ▶ Physical: only physical addressing available for this request ▶ Functional: only functional addressing available for this request (broadcast) ▶ Both: physical and functional addressing available for this request
Multiplicity	1..1
Type	STRING
Default value	Physical
Range	Physical Functional Both
Origin	EB

Parameter Name	Length
Description	Specifies the exact length to accept the request. A null length disable the length filtering.
Multiplicity	1..1
Type	INTEGER
Default value	0
Range	>=0
Origin	EB

Parameter Name	Callback
Description	Specifies the callback name. this callback will be called by UDS if the request is valid (session allowed + length correct + parameters correct)
Multiplicity	1..1
Type	STRING
Default value	
Origin	EB

Parameter Name	Callback_Origin
Description	Select the layer where the callback is defined: EB_cbk, APP_cbk or other one.



	<ul style="list-style-type: none">▶ EB: callback is defined in EB_Prg.c▶ APP: callback is defined in APP_Prg.c▶ OTHER: callback is defined in other file (not APP or EB)
Multiplicity	1..1
Type	STRING
Default value	APP
Range	<ul style="list-style-type: none">EBAPPOTHER
Origin	EB

4.8.2. Application programming interface (API)

4.8.2.1. Objects

4.8.2.1.1. m_astDiagSrvCfg1

Purpose	structure for Diagnostic services configuration 1, stored in ROM
Type	const tUdsSrvCfg1

4.8.2.1.2. m_astDiagSrvCfg2

Purpose	structure for Diagnostic services configuration 2, stored in ROM
Type	const tUdsSrvCfg2

4.8.2.1.3. m_astDiagSrvCfg3

Purpose	structure for Diagnostic services configuration 3, stored in ROM
Type	const tUdsSrvCfg3



4.8.2.1.4. m_astDiagSrvCfg5

Purpose	structure for Diagnostic services configuration 5, stored in ROM
Type	const tUdsSrvCfg5

4.8.2.2. Functions

4.8.2.2.1. UDS_CbkOnRxRequestInd

Purpose	Callback for the diagnostic request (configuration).	
Synopsis	<code>tUdsStatus UDS_CbkOnRxRequestInd (PduLengthType * pulLen , u8 * aubUdsData);</code>	
Parameters (in,out)	pulLen	pointer on data length
	aubUdsData	pointer on data
Return Value	Diagnostic response status	
	UDS_ACK	Positive response
	UDS_NRC_xxx	Negative response code.
Description	The diagnostic request is valid (filtering completed). The callback configured for this service is used.	

4.8.2.2.2. UDS_CustomIsBMCCountTimeout

Purpose	Request if ECU has started normally or not.	
Synopsis	<code>tUdsBoolean UDS_CustomIsBMCCountTimeout (const u8 * aubUdsData);</code>	
Parameters (in,out)	aubUdsData	pointer on data
Return Value	eUdsBoolean : UDS_TRUE/UDS_FALSE	
	UDS_TRUE	: when BMTimeoutCount is not equal to zero and diagnosticSessionType is equal to programmingSession and if BMTimeoutCount is equal to zero
	UDS_FALSE	: Under abnormal startup if BMTimeoutCount is equal to zero and diagnosticSes-



	sionType is not equal to programmingSession
Description	This function is called at startup to know if normal or abnormal startup has been done. In case of abnormal startup ECU stay in Bootloader mode during a configured time before jumping to application (if valid). The ECU shall ignore any request other than the DiagnosticSessionControl service with diagnosticSessionType equal to programmingSession to force the Boot mode during the 20 ms time window after power on.

4.8.2.2.3. UDS_CustomPositiveAnswerInd

Purpose	Notification of positive answer.	
Synopsis	<code>void UDS_CustomPositiveAnswerInd (const PduLengthType * pullen , const u8 * aubUdsData);</code>	
Parameters (in)	pullen	received data length
	aubUdsData	received data pointer
Description	Notification in order to give possibility to the user to execute an action before the positive answer transmission.	

4.8.2.2.4. UDS_CustomSupplier_BA

Purpose	API that check if a supplier request has been received.	
Synopsis	<code>tUdsStatus UDS_CustomSupplier_BA (PduLengthType * pullen , u8 * aubUdsData , tUdsAddrMode eUdsAddrMode , tUdsAddrType eUdsAddrType);</code>	
Parameters (in)	eUdsAddrMode	Addressing mode information (PHYSICAL 0x01U , FUNCTIONAL 0x02U , PHYSICAL_FUNCTIONAL 0x03U)
	eUdsAddrType	Addressing type information (DIAG 0x00U , OBD 0x01U , DIAG_OBD 0x02U)
Parameters (in,out)	pullen	received data length, response length (no response if 0)
	aubUdsData	received data pointer, data to transmit
Return Value	Result of check	
	UDS_ACK	BA supplier request has been received successfully



	UDS_NRC_xxx	Negative response code.
	UDS_ERR_COHE	SDF error detected (no response)
Description	Callback is called: receiving a BA supplier request. Callback shall implement: BA supplier service	

4.8.2.2.5. UDS_CustomSupplier_BB

Purpose	API that check if a supplier request has been received.	
Synopsis	<code>tUdsStatus UDS_CustomSupplier_BB (PduLengthType * pulLen , u8 * aubUdsData , tUdsAddrMode eUdsAddrMode , tUdsAddrType eUdsAddrType);</code>	
Parameters (in)	eUdsAddrMode	Addressing mode information (PHYSICAL 0x01U / FUNCTIONAL 0x02U / PHYSICAL_FUNCTIONAL 0x03U)
	eUdsAddrType	Addressing type information (DIAG 0x00U / OBD 0x01U / DIAG_OBD 0x02U)
Parameters (in,out)	pulLen	received data length, response length (no response if 0)
	aubUdsData	received data pointer, data to transmit
Return Value	Result of check	
	UDS_ACK	BB supplier request has been received successfully
	UDS_NRC_xxx	Negative response code.
	UDS_ERR_COHE	SDF error detected (no response)
Description	Callback is called: receiving a BB supplier request. Callback shall implement: BB supplier service	

4.8.2.2.6. UDS_CustomSupplier_BC

Purpose	API that check if a supplier request has been received.	
Synopsis	<code>tUdsStatus UDS_CustomSupplier_BC (PduLengthType * pulLen , u8 * aubUdsData , tUdsAddrMode eUdsAddrMode , tUdsAddrType eUdsAddrType);</code>	



Parameters (in)	eUdsAddrMode	Addressing mode information (PHYSICAL 0x01U / FUNCTIONAL 0x02U / PHYSICAL_FUNCTIONAL 0x03U)
	eUdsAddrType	Addressing type information (DIAG 0x00U / OBD 0x01U / DIAG_OBD 0x02U)
Parameters (in,out)	pulLen	received data length, response length (no response if 0)
	aubUdsData	received data pointer, data to transmit
Return Value	Result of check	
	UDS_ACK	BC supplier request has been received successfully
	UDS_NRC_xxx	Negative response code.
	UDS_ERR_COHE	SDF error detected (no response)
Description	<p>Callback is called: receiving a BC supplier request.</p> <p>Callback shall implement: BC supplier service</p>	

4.8.2.2.7. UDS_CustomSupplier_BD

Purpose	API that check if a supplier request has been received.	
Synopsis	<pre>tUdsStatus UDS_CustomSupplier_BD (PduLengthType * pulLen , u8 * aubUdsData , tUdsAddrMode eUdsAddrMode , tUdsAddrType eUdsAddrType);</pre>	
Parameters (in)	eUdsAddrMode	Addressing mode information (PHYSICAL 0x01U / FUNCTIONAL 0x02U / PHYSICAL_FUNCTIONAL 0x03U)
	eUdsAddrType	Addressing type information (DIAG 0x00U / OBD 0x01U / DIAG_OBD 0x02U)
Parameters (in,out)	pulLen	received data length, response length (no response if 0)
	aubUdsData	received data pointer, data to transmit
Return Value	Result of check	
	UDS_ACK	BD supplier request has been received successfully
	UDS_NRC_xxx	Negative response code.
	UDS_ERR_COHE	SDF error detected (no response)



Description	Callback is called: receiving a BD supplier request. Callback shall implement: BD supplier service
--------------------	---

4.8.2.2.8. UDS_CustomSupplier_BE

Purpose	API that check if a supplier request has been received.	
Synopsis	<code>tUdsStatus UDS_CustomSupplier_BE (PduLengthType * pulLen , u8 * aubUdsData , tUdsAddrMode eUdsAddrMode , tUdsAddrType eUdsAddrType);</code>	
Parameters (in)	<code>eUdsAddrMode</code>	Addressing mode information (PHYSICAL 0x01U / FUNCTIONAL 0x02U / PHYSICAL_FUNCTIONAL 0x03U)
	<code>eUdsAddrType</code>	Addressing type information (DIAG 0x00U / OBD 0x01U / DIAG_OBD 0x02U)
Parameters (in,out)	<code>pulLen</code>	received data length, response length (no response if 0)
	<code>aubUdsData</code>	received data pointer, data to transmit
Return Value	Result of check	
	<code>UDS_ACK</code>	BE supplier request has been received successfully
	<code>UDS_NRC_XXX</code>	Negative response code.
	<code>UDS_ERR_COHE</code>	SDF error detected (no response)
Description	Callback is called: receiving a BE supplier request. Callback shall implement: BE supplier service	

4.8.2.2.9. UDS_GetCurrentSession

Purpose	Accessor for the current session.	
Synopsis	<code>tUdsSessionType UDS_GetCurrentSession (void);</code>	
Return Value	Session type	
	<code>UDS_SESSION_DEFAULT</code>	Default session
	<code>UDS_SESSION_PROGRAMMING</code>	Programming session



	UDS_SESSION_EXTENDED_DIAG	Extended session
	UDS_SESSION_SUPPLIER	Supplier session
	UDS_SESSION_OTHER_0x	Other session (01/02/03/04)
Description	Accessor for the current session.	

4.8.2.2.10. UDS_Init

Purpose	Initialize layer.
Synopsis	<code>void UDS_Init (void);</code>
Description	This function initializes UDS layer, shall be called only once at ECU startup. Current session is set to DEFAULT session. Session timeout is deactivated.

4.8.2.2.11. UDS_IsOBDSERVICE

Purpose	Test if the current service is OBD type.	
Synopsis	<code>tUdsBoolean UDS_IsOBDSERVICE (void);</code>	
Return Value	Result of treatment	
	UDS_FALSE	Current service is not OBD type
	UDS_TRUE	Current service is OBD type

4.8.2.2.12. UDS_IsPending

Purpose	Response pending treatment: transmission acknowledgement for intermediate NRC_-78 or final response.	
Synopsis	<code>tUdsStatus UDS_IsPending (void);</code>	
Return Value	Result of treatment	
	UDS_FALSE	NRC_78 response is not previously sent
	UDS_TRUE	NRC_78 response was previously sent
Description	This API is useful for response pending management. It shall be called on Receive indication of received tester present to check if last processed request was response pending. The aim is to ensure that S3 timer is not reloaded when a functional tester present is received after a response pending,	



4.8.2.2.13. UDS_LongRequestEnd

Purpose	Response pending completion: final response received from the application.	
Synopsis	<pre>void UDS_LongRequestEnd (PduLengthType ulLen , u8 * aubUdsData , , tUdsStatus eStatus);</pre>	
Parameters (in)	ulLen	data length
	aubUdsData	pointer on response data
	eStatus	Diagnostic response status (UDS_ACK for positive response, UDS_NRC_xxx for negative response)
Description	<p>This API shall be called in order to provide an answer after an long processing request. This suppose that a callback (configured on a service) has generated a NRC_78 answer.</p> <p>The complete data buffer must have been updated by the application starting from index0. The 3 first data bytes have been written with "7F <service> 78" on first NACK_78 response.</p>	

4.8.2.2.14. UDS_LongRequestRespTxConf

Purpose	Response pending treatment: transmission acknowledgement for intermediate NRC_78 or final response.
Synopsis	<pre>void UDS_LongRequestRespTxConf (void);</pre>
Description	This API is useful for response pending management. It shall be called on transmission acknowledgement for intermediate NRC_78 message. The aim is to ensure that the transport protocol layer is not already treating NRC_78 message, when the final response is given by the application. If a NRC_78 message is under transmission, the final response transmission is delayed until TxConf notification.

4.8.2.2.15. UDS_LongRequestResponseInd

Purpose	Response pending management: request for intermediate NRC_78 or final response transmission.	
Synopsis	<pre>tUdsStatus UDS_LongRequestResponseInd (PduLengthType ulLen , u8 * aubUdsData);</pre>	
Parameters (in)	ulLen	final length for response



Parameters (in,out)	aubUdsData	pointer on data
Return Value	Diagnostic response status	
	UDS_ACK	Positive response: the transmission is pending
	UDS_NRC_xxx	Negative response code (just need to be different from UDS_ACK) : the transmission request is rejected for undetailed reason.
Description	Response pending is in progress. The UDS layer ensure cyclic NRC_78 transmissions and final response transmission. This callback is used to request transmission.	

4.8.2.2.16. UDS_Manage

Purpose	Regular tick of the layer.
Synopsis	<code>void UDS_Manage (void);</code>
Description	Ensure cyclic tasks of the layer: it manages the session counter and throws timeout notification. If session timeout occurs, the layer automatically switches to the default diagnostic session.

4.8.2.2.17. UDS_P2AboutToExpireInd

Purpose	Notification just before the P2/P2_STAR timeout.
Synopsis	<code>void UDS_P2AboutToExpireInd (void);</code>
Description	Notification in order to give possibility to the application to execute an action before P2/P2_STAR timeout.

4.8.2.2.18. UDS_ReloadTStopDiag

Purpose	Reload session timer.
Synopsis	<code>void UDS_ReloadTStopDiag (void);</code>
Description	This function checks the state of the current session (standard or other). If the current session is non standard (diagnostic or programming session for example), the timer T_Stop_Diag is reloaded with its maximum value: m_uUdsSessTimeout. It allows the ECU to maintain the current session for a duration equal to T_Stop_Diag (5 s). This



	functionality can be used for the NRC78 periodic answers, if the request treatment needs more than T_Stop_Diag to be completed.
--	---

4.8.2.2.19. UDS_ResponsePending_Manage

Purpose	Regular tick of the layer.
Synopsis	<code>void UDS_ResponsePending_Manage (void);</code>
Description	Ensure cyclic tasks of the layer: response pending, it ensures cyclic NRC_78 message transmissions and final response transmission.

4.8.2.2.20. UDS_ResponsePending_TimCntManage

Purpose	Response Pending Manage from ISR of STM Timer.
Synopsis	<code>void UDS_ResponsePending_TimCntManage (void);</code>
Description	External NRC_78 (response pending) timeout increment, allowing increment under interrupt using hardware timer. This API is called from ISR of STM timer if TIMER_RESPONSE_PENDING_CHECK is enabled in UDS configuration

4.8.2.2.21. UDS_RxRequest

Purpose	Treatment of diagnostic request.	
Synopsis	<code>tUdsBoolean UDS_RxRequest (PduLengthType * pulLen , u8 * aubUdsData);</code>	
Parameters (in,out)	pulLen	pointer on data length (request)
	aubUdsData	pointer on data (request)
Return Value	Result of treatment	
	UDS_FALSE	pulLen and/or aubUdsData are NULL pointers
	UDS_TRUE	when pulLen is different than zero
Description	This function performs the processing of a received diagnostic request. It automatically handles the TesterPresent service as well as the status request of routine control request (RC). In the case of a configured service, the function calls the corresponding callback m_astDiagServiceCfg1[].pfuRxRequestInd, m_astDiagServiceCfg2[].pfuRxRequestInd, m_astDiagServiceCfg3[].pfuRxRequestInd or m_astDi-	



	agServiceCfg5[].pfuRxRequestInd In the case of a service not configured, the function automatically sends a negative appropriated response.
--	---

4.8.2.2.22. UDS_RxRequestWithAddrMode

Purpose	Treatment of diagnostic request with addressing mode.	
Synopsis	<code>tUdsBoolean UDS_RxRequestWithAddrMode (PduLengthType * pulLen , u8 * aubUdsData , tUdsAddrMode eUdsAddrMode , tUdsAddrType eUdsAddrType);</code>	
Parameters (in)	<code>eUdsAddrMode</code>	addressing mode (request) (UDS_ADDR_PHYSICAL, UDS_ADDR_FUNCTIONAL)
	<code>eUdsAddrType</code>	diagnostic type (request) (UDS_TYPE_DIAG, UDS_TYPE_OBD, UDS_TYPE_DIAG_OBD)
Parameters (in,out)	<code>pulLen</code>	pointer on data length
	<code>aubUdsData</code>	pointer on data
Return Value	Result of treatment	
	<code>UDS_FALSE</code>	<code>pulLen</code> and/or <code>aubUdsData</code> are NULL pointers
	<code>UDS_TRUE</code>	when <code>pulLen</code> is different than zero
Description	This function performs the processing of a received diagnostic request with addressing mode. It automatically handles the TesterPresent service as well as the status request of routine control request (RC). In the case of a configured service, the function calls the corresponding callback <code>m_astDiagServiceCfg1[].pfuRxRequestInd</code> , <code>m_astDiagServiceCfg2[].pfuRxRequestInd</code> , <code>m_astDiagServiceCfg3[].pfuRxRequestInd</code> or <code>m_astDiagServiceCfg5[].pfuRxRequestInd</code> In the case of a service not configured with physical addressing mode, the function automatically sends a negative appropriated response. In the case of a service not configured with functional addressing mode, a negative answer is transmitted in specific cases only. Addressing mode comply to 14229-1:2005(E) specification. The diagnostic type is also evaluated to reject some services not defined in OBD diagnostic.	

4.8.2.2.23. UDS_SessionStatusInd

Purpose	Notification for diagnostic session transition.
----------------	---



Synopsis	<code>void UDS_SessionStatusInd (tUdsSessionType eUdsNewSessType , tUdsSessionType eUdsOldSessType , tUdsChangeReason eUdsChangingCause);</code>	
Parameters (in)	eUdsNewSessType	new session
	eUdsOldSessType	old session
	eUdsChangingCause	explicit request (UDS_SESSION_CHANGE_REQUESTED) or session timeout (UDS_SESSION_TIMEOUT)
Description	It provides old and new sessions, with the reason for the transition.	

4.8.2.2.24. UDS_StopNRC78Timer

Purpose	Response pending management: Stop NRC78 timer.
Synopsis	<code>void UDS_StopNRC78Timer (void);</code>
Description	API used internally in bootloader to stop NRC78 timer while FLASH erasing treatment (code execution from RAM).

4.8.2.2.25. UDS_StopSessionTimer

Purpose	Stop session timer.
Synopsis	<code>void UDS_StopSessionTimer (void);</code>
Description	This function allows to deactivate the T_Stop_Diag timer.

4.8.3. Integration notes

4.8.3.1. Exclusive areas

Exclusive areas information is not available for this module.

4.8.3.2. Production errors

Production errors information is not available for this module.

4.8.3.3. Memory mapping

General information about memory mapping is provided in the EB tresos AutoCore Generic documentation. Refer to the section [Memory mapping and compiler abstraction](#) in the [Integration notes](#) section for details.

Memory mapping information is not available for this module.

4.8.3.4. Integration requirements

WARNING

Integration requirements list is not exhaustive



The following list of integration requirements helps you to integrate your product. However, this list is not exhaustive. You also require information from the user guide, release notes, and EB tresos AutoCore known issues to successfully integrate your product.

Integration requirements are not listed for the Uds module.



5. Bibliography