

<b>Document Title</b>	Specification of Flash Test
<b>Document Owner</b>	AUTOSAR
<b>Document Responsibility</b>	AUTOSAR
<b>Document Identification No</b>	261

<b>Document Status</b>	Final
<b>Part of AUTOSAR Standard</b>	Classic Platform
<b>Part of Standard Release</b>	4.4.0

<b>Document Change History</b>			
<b>Date</b>	<b>Release</b>	<b>Changed by</b>	<b>Change Description</b>
2018-10-31	4.4.0	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>• FlsTstBlockBgndConfigSet and FlsTstBlockFgndConfigSet removed;</li> <li>• FlsTstEcucPartitionRef configuration parameter added</li> </ul>
2017-12-08	4.3.1	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>• Numeric value definition</li> <li>• minor corrections / clarifications / editorial changes; for details please refer to the ChangeDocumentation</li> </ul>
2016-11-30	4.3.0	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>• ECUC_FlsTst_00172: FlsTstMainFunctionPeriod added;</li> <li>• SWS_FlsTst_00081 removed;</li> <li>• Unresolved references BSW00431, BSW00434, SRS_BSW_00326, SRS_BSW_00435, SRS_BSW_00436 deleted</li> </ul>

Document Change History			
Date	Release	Changed by	Change Description
2015-07-31	4.2.2	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>• Debugging support marked as obsolete</li> <li>• ECUC_FlsTst_00119 set to obsolete;</li> <li>• ECUC_FlsTst_00161 created;</li> <li>• ECUC_FlsTst_00151 modified (pre-compile only);</li> <li>• SWS_FlsTst_00023, SWS_FlsTst_00026, SWS_FlsTst_00133 removed;</li> <li>• SWS_FlsTst_00007: Error code 0x05 modified</li> <li>• SWS_FlsTst_00168 created: Extended production error table with pass/fail criteria;</li> <li>• SWS_FlsTst_00161 modified;</li> <li>• SWS_FlsTst_00167 created;</li> <li>• Renaming from Development Error Tracer to Default Error Tracer; changes in abbreviations, chapter 3.1, SWS_FlsTst_00011;</li> <li>• Template changes; chapters runtime errors and transient faults added</li> </ul>
2014-10-31	4.2.1	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>• Formal text modifications in: SWS_FlsTst_00138, SWS_FlsTst_00140, SWS_FlsTst_00142, SWS_FlsTst_00143, SWS_FlsTst_00071, SWS_FlsTst_00115, SWS_FlsTst_00116, SWS_FlsTst_00117, ECUC_FlsTst_00160, Figure 7/8/9/10</li> <li>• ECUC_FlsTst_00086: configuration FlsTstConfigurationOfOptApiServices added</li> </ul>

Document Change History			
Date	Release	Changed by	Change Description
2013-10-31	4.1.2	AUTOSAR Release Management	<ul style="list-style-type: none"> <li>• SWS_FlStst_00066: VARIABLE_CYCLIC_OR_ON_PRE CONDITION in table removed</li> <li>• Editorial changes</li> <li>• Removed chapter(s) on change documentation</li> </ul>
2013-03-15	4.1.1	AUTOSAR Administration	<ul style="list-style-type: none"> <li>• Rework according to the new SWS_BSWGeneral document</li> <li>• Added Subchapter 3.x due to SWS General Rollout</li> <li>• Chapter 10: scope of configuration parameters are changed to “local”</li> <li>• SWS_FlStst_00003: Rename MemMap.h to FlStst_MemMap.h</li> <li>• SWS_FlStst_00007: production errors removed</li> <li>• New chapters Production Errors and Extended Production Errors created</li> </ul>
2011-12-22	4.0.3	AUTOSAR Administration	<ul style="list-style-type: none"> <li>• SWS_FlStst_00026: minor text change</li> <li>• Figure1: IRQ files removed</li> <li>• SWS_FlStst_00052: parameter range modified</li> <li>• SWS_FlStst_00053: minor text correction</li> </ul>

Document Change History			
Date	Release	Changed by	Change Description
2010-09-30	3.1.5	AUTOSAR Administration	<ul style="list-style-type: none"> <li>• FlsTst_BlockIdFgndType: type change to uint8-32</li> <li>• Limit range of the following parameters to max. value "0xFFFFFFFF"</li> <li>• FlsTstBlockNumberBgnd:</li> <li>• FlsTstBlockNumberFgnd:</li> <li>• FlsTstBlockIndex:</li> <li>• FlsTstBlockSize:</li> <li>• FlsTstNumberOfTestedCells:</li> <li>• FlsTstNumberOfTestedCellsAtomic:</li> <li>• FlsTstTestIntervallIdEndValue:</li> <li>• FlsTst015 removed</li> <li>• ECUC_FlsTst_00119: configuration for each block</li> <li>• ECUC_FlsTst_00158: multiplicity changed to „1“.</li> <li>• FlsTstDemEventParameterRefs table included</li> </ul>
2010-02-02	3.1.4	AUTOSAR Administration	<ul style="list-style-type: none"> <li>• Initial release</li> </ul>

## Disclaimer

This work (specification and/or software implementation) and the material contained in it, as released by AUTOSAR, is for the purpose of information only. AUTOSAR and the companies that have contributed to it shall not be liable for any use of the work.

The material contained in this work is protected by copyright and other types of intellectual property rights. The commercial exploitation of the material contained in this work requires a license to such intellectual property rights.

This work may be utilized or reproduced without any modification, in any form or by any means, for informational purposes only. For any other purpose, no part of the work may be utilized or reproduced, in any form or by any means, without permission in writing from the publisher.

The work has been developed for automotive applications only. It has neither been developed, nor tested for non-automotive applications.

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

## Table of Contents

1	Introduction and functional overview .....	8
2	Acronyms and abbreviations .....	9
3	Related documentation.....	10
3.1	Input documents.....	10
3.2	Related specification .....	10
4	Constraints and assumptions .....	11
4.1	Limitations .....	11
4.2	Applicability to car domains.....	11
5	Dependencies to other modules.....	12
5.1	File structure.....	12
5.1.1	Code file structure.....	12
6	Requirements traceability .....	13
7	Functional specification .....	21
7.1	General behavior .....	21
7.1.1	State Diagram .....	22
7.2	Error Classification .....	23
7.3	Production Errors.....	23
7.4	Extended Production Errors .....	23
7.5	Runtime Errors .....	23
7.6	Transient Faults.....	24
7.7	Error Detection .....	24
7.8	Error Notification.....	24
7.9	Version Check .....	24
7.10	Debugging Support .....	24
8	API specification.....	25
8.1	Imported types.....	25
8.2	Type definitions .....	25
8.2.1	FlsTst_ConfigType.....	25
8.2.2	FlsTst_StateType.....	25
8.2.3	FlsTst_TestResultFgndType.....	26
8.2.4	FlsTst_TestResultBgndType.....	26
8.2.5	FlsTst_BlockIdFgndType .....	27
8.2.6	FlsTst_ErrorDetailsType .....	27
8.2.7	FlsTst_TestSignatureFgndType.....	27
8.2.8	FlsTst_TestSignatureBgndType.....	27
8.2.9	FlsTst_TestResultType .....	28
8.3	Function definitions.....	28
8.3.1	FlsTst_Init .....	28
8.3.2	FlsTst_DelInit.....	29
8.3.3	FlsTst_StartFgnd .....	29
8.3.4	FlsTst_Abort.....	30

8.3.5	FlsTst_Suspend .....	31
8.3.6	FlsTst_Resume .....	31
8.3.7	FlsTst_GetCurrentState .....	32
8.3.8	FlsTst_GetTestResultBgnd .....	33
8.3.9	FlsTst_GetTestResultFgnd .....	33
8.3.10	FlsTst_GetVersionInfo .....	34
8.3.11	FlsTst_GetTestSignatureBgnd .....	34
8.3.12	FlsTst_GetTestSignatureFgnd .....	35
8.3.13	FlsTst_GetErrorDetails .....	35
8.3.14	FlsTst_TestEcc .....	36
8.4	Callback notifications .....	37
8.5	Scheduled functions .....	37
8.5.1	FlsTst_MainFunction .....	37
8.6	Expected Interfaces .....	38
8.6.1	Mandatory Interfaces .....	38
8.6.2	Optional Interfaces .....	38
8.6.3	Configurable interfaces .....	39
8.6.3.1	FlsTst_TestCompleted Notification .....	39
9	Sequence diagrams .....	40
9.1	Initialization .....	40
9.2	De-initialization .....	40
9.3	Background Test .....	41
9.3.1	Test Result Calculation within Flash test driver .....	41
9.3.2	Test signature provided to caller .....	42
9.4	Suspend and Resume Background Testing .....	43
9.5	Foreground Task interrupts Background Task .....	44
10	Configuration specification .....	45
10.1	Specification template for configuration parameters .....	45
10.2	Containers and configuration parameters .....	45
10.2.1	Variants .....	45
10.2.2	FlsTst .....	45
10.2.3	FlsTstGeneral .....	46
10.2.4	FlsTstConfigurationOfOptApiServices .....	48
10.2.5	FlsTstDemEventParameterRefs .....	51
10.2.6	FlsTstBlockBgnd .....	52
10.2.7	FlsTstBlockFgnd .....	53
	Published Information .....	55
11	Not applicable requirements .....	56

## 1 Introduction and functional overview

This specification specifies the functionality, API and the configuration of the AUTOSAR Basic Software module Flash Test driver.

This Flash test module provides algorithm to test invariable memory. Invariable memory can be data/program flash, program SRAM, locked cache and is either embedded in the microcontroller or memory mapped connected to the microcontroller. For simplification the SW module is called Flash Test driver.

The test service can be executed at any time after MCU initialization and it is up to the user of the Flash Test Driver to choose the suitable test algorithm and the right execution place to fulfill the safety requirements of the system. The test service itself is dependant on the storage concept of the system. Therefore, the availability of different test algorithms is configurable.

The Flash Test driver is intended to be integrated in the overall safety concept and will not provide the required diagnostic coverage on its own.



## 2 Acronyms and abbreviations

Acronyms and abbreviations that have a local scope are not contained in the AUTOSAR glossary. These appear in a local glossary below.

<b>Acronym:</b>	<b>Description:</b>
BSW	BasicSoftWare
PC	PreCompile
PB	PostBuild

<b>Abbreviation:</b>	<b>Description:</b>
DEM	Diagnostic Event Manager.
DET	Default Error Tracer.
MCU	Micro Controller Unit.
PLL	Phase Locked Loop.
ISR	Interrupt Service Routine.

The following table lists important Term and Definition, which are used within this document.

<b>Term:</b>	<b>Description:</b>
Background test	Background test is called periodically by a scheduler, and is interruptible. The test is split up over many scheduled tasks.
Foreground test	Foreground test is called via users call.
Flash cell	Smallest entity to be addressed, in this case bytes shall be used
Invariable memory	Invariable memory can be program flash, program SRAM, locked cache and ROM
Test block	Defined memory area to be tested in foreground and background mode.
Test interval	Interval of a complete Flash test in background mode
Test time	Time for partial test defined within one scheduled task.
Signature	Unique calculation result of the content of a specific memory block
Memory block	Defined memory area
Partial test	Test to be executed in one scheduler interval
Test Interval Id	Identifier of a test interval, which shall be incremented by each start of a new test interval

## 3 Related documentation

### 3.1 Input documents

- [1] Layered Software Architecture  
AUTOSAR\_EXP\_LayeredSoftwareArchitecture.pdf
- [2] General Requirements on SPAL  
AUTOSAR\_SRS\_SPALGeneral.pdf
- [3] General Requirements on Basic Software Modules  
AUTOSAR\_SRS\_BSWGeneral.pdf
- [4] Specification of Default Error Tracer  
AUTOSAR\_SWS\_DefaultErrorTracer.pdf
- [5] Specification of MCU Driver  
AUTOSAR\_SWS\_MCUDriver.pdf
- [6] Specification of ECU Configuration,  
AUTOSAR\_TPS\_ECUConfiguration.pdf
- [7] Basic Software Module Description Template,  
AUTOSAR\_TPS\_BSWModuleDescriptionTemplate.pdf
- [8] List of Basic Software Modules  
AUTOSAR\_TR\_BSWModuleList
- [9] General Specification of Basic Software Modules  
AUTOSAR\_SWS\_BSWGeneral.pdf

### 3.2 Related specification

AUTOSAR provides a General Specification on Basic Software modules [9] (SWS BSW General), which is also valid for Flash Test.

Thus, the specification SWS BSW General shall be considered as additional and required specification for Flash Test.

## **4 Constraints and assumptions**

### **4.1 Limitations**

During Flash Test operation, the Flash area under test shall not be modified.

### **4.2 Applicability to car domains**

No restrictions.

## 5 Dependencies to other modules

The Flash Test module depends on the following modules:

- BSW scheduler is required to trigger main function in background mode

### 5.1 File structure

#### 5.1.1 Code file structure

Note: Refer to SWS\_BSWGeneral document [9].

## 6 Requirements traceability

Requirement	Description	Satisfied by
SRS_BSW_00003	All software modules shall provide version and identification information	SWS_FlStst_00166
SRS_BSW_00005	Modules of the $\mu$ C Abstraction Layer (MCAL) may not have hard coded horizontal interfaces	SWS_FlStst_00166
SRS_BSW_00006	The source code of software modules above the $\mu$ C Abstraction Layer (MCAL) shall not be processor and compiler dependent.	SWS_FlStst_00166
SRS_BSW_00007	All Basic SW Modules written in C language shall conform to the MISRA C 2012 Standard.	SWS_FlStst_00166
SRS_BSW_00009	All Basic SW Modules shall be documented according to a common standard.	SWS_FlStst_00166
SRS_BSW_00010	The memory consumption of all Basic SW Modules shall be documented for a defined configuration for all supported platforms.	SWS_FlStst_00166
SRS_BSW_00101	The Basic Software Module shall be able to initialize variables and hardware in a separate initialization function	SWS_FlStst_00017
SRS_BSW_00159	All modules of the AUTOSAR Basic Software shall support a tool based configuration	SWS_FlStst_00166
SRS_BSW_00161	The AUTOSAR Basic Software shall provide a microcontroller abstraction layer which provides a standardized interface to higher software layers	SWS_FlStst_00166
SRS_BSW_00162	The AUTOSAR Basic Software shall provide a hardware abstraction layer	SWS_FlStst_00166
SRS_BSW_00164	The Implementation of interrupt service routines shall be done by the Operating System, complex drivers or modules	SWS_FlStst_00166
SRS_BSW_00167	All AUTOSAR Basic Software Modules shall provide configuration rules and constraints to enable plausibility checks	SWS_FlStst_00166
SRS_BSW_00168	SW components shall be tested by a function defined in a common API in the Basis-SW	SWS_FlStst_00166

SRS_BSW_00170	The AUTOSAR SW Components shall provide information about their dependency from faults, signal qualities, driver demands	SWS_FlsTst_00166
SRS_BSW_00172	The scheduling strategy that is built inside the Basic Software Modules shall be compatible with the strategy used in the system	SWS_FlsTst_00166
SRS_BSW_00300	All AUTOSAR Basic Software Modules shall be identified by an unambiguous name	SWS_FlsTst_00166
SRS_BSW_00301	All AUTOSAR Basic Software Modules shall only import the necessary information	SWS_FlsTst_00166
SRS_BSW_00302	All AUTOSAR Basic Software Modules shall only export information needed by other modules	SWS_FlsTst_00166
SRS_BSW_00304	All AUTOSAR Basic Software Modules shall use the following data types instead of native C data types	SWS_FlsTst_00016
SRS_BSW_00305	Data types naming convention	SWS_FlsTst_00166
SRS_BSW_00306	AUTOSAR Basic Software Modules shall be compiler and platform independent	SWS_FlsTst_00166
SRS_BSW_00307	Global variables naming convention	SWS_FlsTst_00166
SRS_BSW_00308	AUTOSAR Basic Software Modules shall not define global data in their header files, but in the C file	SWS_FlsTst_00166
SRS_BSW_00309	All AUTOSAR Basic Software Modules shall indicate all global data with read-only purposes by explicitly assigning the const keyword	SWS_FlsTst_00166
SRS_BSW_00310	API naming convention	SWS_FlsTst_00166
SRS_BSW_00312	Shared code shall be reentrant	SWS_FlsTst_00166
SRS_BSW_00323	All AUTOSAR Basic Software Modules shall check passed API parameters for validity	SWS_FlsTst_00033
SRS_BSW_00325	The runtime of interrupt service routines and functions that are running in interrupt context shall be kept short	SWS_FlsTst_00166
SRS_BSW_00327	Error values naming convention	SWS_FlsTst_00166
SRS_BSW_00328	All AUTOSAR Basic Software Modules shall avoid the duplication of code	SWS_FlsTst_00166

SRS_BSW_00330	It shall be allowed to use macros instead of functions where source code is used and runtime is critical	SWS_FlsTst_00166
SRS_BSW_00331	All Basic Software Modules shall strictly separate error and status information	SWS_FlsTst_00166
SRS_BSW_00334	All Basic Software Modules shall provide an XML file that contains the meta data	SWS_FlsTst_00166
SRS_BSW_00335	Status values naming convention	SWS_FlsTst_00166
SRS_BSW_00336	Basic SW module shall be able to shutdown	SWS_FlsTst_00027
SRS_BSW_00337	Classification of development errors	SWS_FlsTst_00007
SRS_BSW_00339	Reporting of production relevant error status	SWS_FlsTst_00042, SWS_FlsTst_00060, SWS_FlsTst_00112
SRS_BSW_00341	Module documentation shall contains all needed informations	SWS_FlsTst_00166
SRS_BSW_00342	It shall be possible to create an AUTOSAR ECU out of modules provided as source code and modules provided as object code, even mixed	SWS_FlsTst_00166
SRS_BSW_00343	The unit of time for specification and configuration of Basic SW modules shall be preferably in physical time unit	SWS_FlsTst_00166
SRS_BSW_00344	BSW Modules shall support link-time configuration	SWS_FlsTst_00166
SRS_BSW_00347	A Naming separation of different instances of BSW drivers shall be in place	SWS_FlsTst_00166
SRS_BSW_00348	All AUTOSAR standard types and constants shall be placed and organized in a standard type header file	SWS_FlsTst_00166
SRS_BSW_00350	All AUTOSAR Basic Software Modules shall allow the enabling/disabling of detection and reporting of development errors.	SWS_FlsTst_00166
SRS_BSW_00353	All integer type definitions of target and compiler specific scope shall be placed and organized in a single type header	SWS_FlsTst_00166
SRS_BSW_00357	For success/failure of an API call a standard return type shall be defined	SWS_FlsTst_00063
SRS_BSW_00358	The return type of init() functions implemented by AUTOSAR Basic Software Modules shall be void	SWS_FlsTst_00166

SRS_BSW_00361	All mappings of not standardized keywords of compiler specific scope shall be placed and organized in a compiler specific type and keyword header	SWS_FlsTst_00166
SRS_BSW_00371	The passing of function pointers as API parameter is forbidden for all AUTOSAR Basic Software Modules	SWS_FlsTst_00166
SRS_BSW_00373	The main processing function of each AUTOSAR Basic Software Module shall be named according the defined convention	SWS_FlsTst_00166
SRS_BSW_00375	Basic Software Modules shall report wake-up reasons	SWS_FlsTst_00166
SRS_BSW_00377	A Basic Software Module can return a module specific types	SWS_FlsTst_00048
SRS_BSW_00378	AUTOSAR shall provide a boolean type	SWS_FlsTst_00166
SRS_BSW_00385	List possible error notifications	SWS_FlsTst_00007
SRS_BSW_00386	The BSW shall specify the configuration for detecting an error	SWS_FlsTst_00025, SWS_FlsTst_00033, SWS_FlsTst_00056, SWS_FlsTst_00059, SWS_FlsTst_00062, SWS_FlsTst_00065, SWS_FlsTst_00089, SWS_FlsTst_00091, SWS_FlsTst_00093, SWS_FlsTst_00114
SRS_BSW_00398	The link-time configuration is achieved on object code basis in the stage after compiling and before linking	SWS_FlsTst_00166
SRS_BSW_00401	Documentation of multiple instances of configuration parameters shall be available	SWS_FlsTst_00166
SRS_BSW_00405	BSW Modules shall support multiple configuration sets	SWS_FlsTst_00018, SWS_FlsTst_00019
SRS_BSW_00406	A static status variable denoting if a BSW module is initialized shall be initialized with value 0 before any APIs of the BSW module is called	SWS_FlsTst_00011
SRS_BSW_00407	Each BSW module shall provide a function to read out the version information of a dedicated module implementation	SWS_FlsTst_00044
SRS_BSW_00408	All AUTOSAR Basic Software Modules configuration parameters shall be named according to a specific naming rule	SWS_FlsTst_00166
SRS_BSW_00409	All production code error ID symbols are defined by the Dem module and shall be retrieved by the other BSW modules from Dem configuration	SWS_FlsTst_00007



SRS_BSW_00410	Compiler switches shall have defined values	SWS_FlsTst_00166
SRS_BSW_00411	All AUTOSAR Basic Software Modules shall apply a naming rule for enabling/disabling the existence of the API	SWS_FlsTst_00044
SRS_BSW_00413	An index-based accessing of the instances of BSW modules shall be done	SWS_FlsTst_00166
SRS_BSW_00414	Init functions shall have a pointer to a configuration structure as single parameter	SWS_FlsTst_00166
SRS_BSW_00415	Interfaces which are provided exclusively for one module shall be separated into a dedicated header file	SWS_FlsTst_00166
SRS_BSW_00416	The sequence of modules to be initialized shall be configurable	SWS_FlsTst_00166
SRS_BSW_00417	Software which is not part of the SW-C shall report error events only after the DEM is fully operational.	SWS_FlsTst_00166
SRS_BSW_00419	If a pre-compile time configuration parameter is implemented as "const" it should be placed into a separate c-file	SWS_FlsTst_00166
SRS_BSW_00422	Pre-de-bouncing of error status information is done within the DEM	SWS_FlsTst_00166
SRS_BSW_00423	BSW modules with AUTOSAR interfaces shall be describable with the means of the SW-C Template	SWS_FlsTst_00166
SRS_BSW_00424	BSW module main processing functions shall not be allowed to enter a wait state	SWS_FlsTst_00166
SRS_BSW_00425	The BSW module description template shall provide means to model the defined trigger conditions of schedulable objects	SWS_FlsTst_00166
SRS_BSW_00426	BSW Modules shall ensure data consistency of data which is shared between BSW modules	SWS_FlsTst_00166
SRS_BSW_00427	ISR functions shall be defined and documented in the BSW module description template	SWS_FlsTst_00166
SRS_BSW_00428	A BSW module shall state if its main processing function(s) has to be executed in a specific order or sequence	SWS_FlsTst_00166
SRS_BSW_00429	Access to OS is restricted	SWS_FlsTst_00166
SRS_BSW_00432	Modules should have separate main processing functions for	SWS_FlsTst_00166

	read/receive and write/transmit data path	
SRS_BSW_00433	Main processing functions are only allowed to be called from task bodies provided by the BSW Scheduler	SWS_FlsTst_00166
SRS_BSW_00437	Memory mapping shall provide the possibility to define RAM segments which are not to be initialized during startup	SWS_FlsTst_00166
SRS_BSW_00438	Configuration data shall be defined in a structure	SWS_FlsTst_00018
SRS_BSW_00439	Enable BSW modules to handle interrupts	SWS_FlsTst_00166
SRS_BSW_00440	The callback function invocation by the BSW module shall follow the signature provided by RTE to invoke servers via Rte_Call API	SWS_FlsTst_00166
SRS_FlsTst_14208	Background Flash test shall be interruptible	SWS_FlsTst_00066, SWS_FlsTst_00071
SRS_FlsTst_14209	The memory to be tested shall be split into individual smaller pieces	SWS_FlsTst_00066, SWS_FlsTst_00071, SWS_FlsTst_00139
SRS_FlsTst_14211	Flash test execution status shall be available	SWS_FlsTst_00040, SWS_FlsTst_00041, SWS_FlsTst_00091
SRS_FlsTst_14212	Flash test execution completion shall be provided by a notification mechanism	SWS_FlsTst_00077, SWS_FlsTst_00078
SRS_FlsTst_14213	Calculation signature/checksum of a finalized test shall be provided	SWS_FlsTst_00054, SWS_FlsTst_00055, SWS_FlsTst_00056, SWS_FlsTst_00057, SWS_FlsTst_00058, SWS_FlsTst_00059, SWS_FlsTst_00115, SWS_FlsTst_00116
SRS_FlsTst_14214	Service for Flash test execution result shall be provided.	SWS_FlsTst_00042, SWS_FlsTst_00043, SWS_FlsTst_00093, SWS_FlsTst_00112, SWS_FlsTst_00113, SWS_FlsTst_00114
SRS_FlsTst_14215	Suspend Flash test execution shall be possible	SWS_FlsTst_00034, SWS_FlsTst_00036, SWS_FlsTst_00037, SWS_FlsTst_00088
SRS_FlsTst_14216	Flash test execution shall be resumed when suspended	SWS_FlsTst_00035, SWS_FlsTst_00038, SWS_FlsTst_00039, SWS_FlsTst_00089
SRS_FlsTst_14217	Flash test execution shall be stopped when wanted	SWS_FlsTst_00030, SWS_FlsTst_00031, SWS_FlsTst_00032
SRS_FlsTst_14219	Foreground Flash test shall be available	SWS_FlsTst_00033, SWS_FlsTst_00050, SWS_FlsTst_00051, SWS_FlsTst_00137, SWS_FlsTst_00143, SWS_FlsTst_00149
SRS_FlsTst_14221	Memory Content to Be Tested Should Not be Valid During the Test	SWS_FlsTst_00166
SRS_FlsTst_14223	Flash Test Error details shall be reported	SWS_FlsTst_00060, SWS_FlsTst_00061, SWS_FlsTst_00062
SRS_FlsTst_14224	ECC Circuitry shall be tested	SWS_FlsTst_00063, SWS_FlsTst_00064,

		SWS_FlsTst_00065
SRS_FlsTst_14225	Each Flash test Interval shall have an Identifier	SWS_FlsTst_00153, SWS_FlsTst_00154, SWS_FlsTst_00155
SRS_SPAL_00157	All drivers and handlers of the AUTOSAR Basic Software shall implement notification mechanisms of drivers and handlers	SWS_FlsTst_00040, SWS_FlsTst_00042, SWS_FlsTst_00054, SWS_FlsTst_00057, SWS_FlsTst_00060, SWS_FlsTst_00072, SWS_FlsTst_00073, SWS_FlsTst_00077, SWS_FlsTst_00112
SRS_SPAL_12057	All driver modules shall implement an interface for initialization	SWS_FlsTst_00017, SWS_FlsTst_00020
SRS_SPAL_12064	All driver modules shall raise an error if the change of the operation mode leads to degradation of running operations	SWS_FlsTst_00166
SRS_SPAL_12067	All driver modules shall set their wake-up conditions depending on the selected operation mode	SWS_FlsTst_00166
SRS_SPAL_12068	The modules of the MCAL shall be initialized in a defined sequence	SWS_FlsTst_00166
SRS_SPAL_12069	All drivers of the SPAL that wake up from a wake-up interrupt shall report the wake-up reason	SWS_FlsTst_00166
SRS_SPAL_12075	All drivers with random streaming capabilities shall use application buffers	SWS_FlsTst_00166
SRS_SPAL_12077	All drivers shall provide a non blocking implementation	SWS_FlsTst_00166
SRS_SPAL_12078	The drivers shall be coded in a way that is most efficient in terms of memory and runtime resources	SWS_FlsTst_00166
SRS_SPAL_12092	The driver's API shall be accessed by its handler or manager	SWS_FlsTst_00166
SRS_SPAL_12125	All driver modules shall only initialize the configured resources	SWS_FlsTst_00022
SRS_SPAL_12129	The ISRs shall be responsible for resetting the interrupt flags and calling the according notification function	SWS_FlsTst_00166
SRS_SPAL_12163	All driver modules shall implement an interface for de-initialization	SWS_FlsTst_00027, SWS_FlsTst_00028
SRS_SPAL_12169	All driver modules that provide different operation modes shall provide a service for mode selection	SWS_FlsTst_00166
SRS_SPAL_12265	Configuration data shall be kept constant	SWS_FlsTst_00166
SRS_SPAL_12267	Wakeup sources shall be initialized by MCAL drivers and/or the MCU driver	SWS_FlsTst_00166

SRS_SPAL_12448	All driver modules shall have a specific behavior after a development error detection	SWS_FlsTst_00025, SWS_FlsTst_00033, SWS_FlsTst_00039
SRS_SPAL_12461	Specific rules regarding initialization of controller registers shall apply to all driver implementations	SWS_FlsTst_00166
SRS_SPAL_12462	The register initialization settings shall be published	SWS_FlsTst_00166
SRS_SPAL_12463	The register initialization settings shall be combined and forwarded	SWS_FlsTst_00166

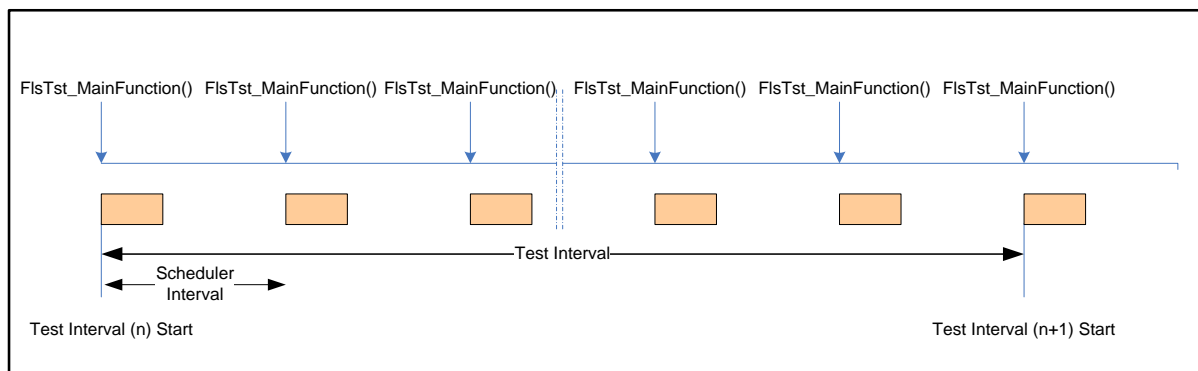
## 7 Functional specification

### 7.1 General behavior

**[SWS\_FlsTst\_00137]** [The Flash test module provides test execution services in background and foreground mode (see also chapter 2). ] (SRS\_FlsTst\_14219)

**[SWS\_FlsTst\_00138]** [The memory blocks to be tested shall be configurable for background and foreground mode separately. ] ( )

**[SWS\_FlsTst\_00139]** [In background mode the test blocks shall be tested in the same order they are configured in configuration structure. When all blocks are tested, one test interval is completed (see Figure 2). In background testing the partial tests shall be triggered via `FlsTst_MainFunction` (see [SWS\\_FlsTst\\_00066](#)). ] (SRS\_FlsTst\_14209)



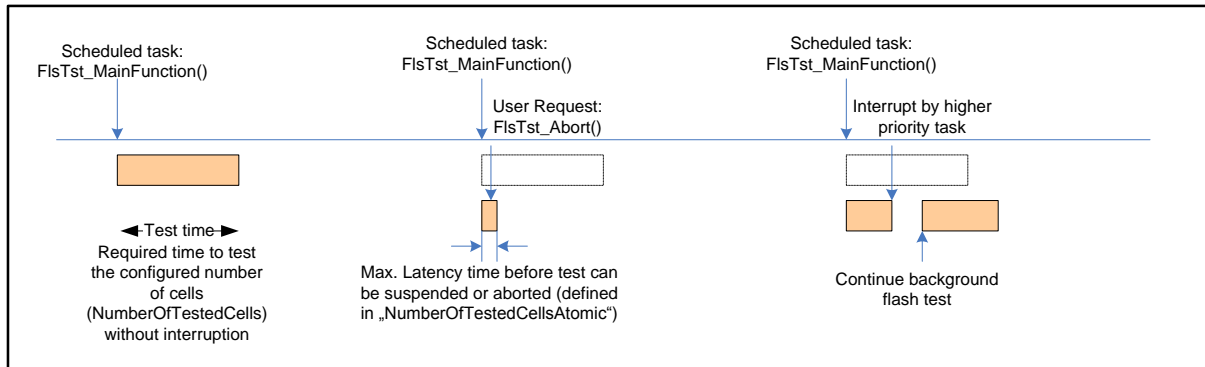
**Figure 2: Background Test: Test Interval**

**[SWS\_FlsTst\_00140]** [The length of a partial test is defined by the number of tested cells, which shall be tested in one scheduled task. (see ECUC\_FlsTst\_00161). The required time for a partial test without interruption is defined as “Test time”. ] ( )

Note: The partial test can be interrupted by a higher priority task at any time, because the Flash test does not require atomic sequences. It is the responsibility of the user to ensure that the interruptible partial test is finished before the scheduler interval is started (See Figure 3).

**[SWS\_FlsTst\_00142]** [A background test shall be aborted or suspended via the API services `FlsTst_Abort()` or `FlsTst_Suspended()`. The maximum latency time until the API call request is processed, shall be configurable (see ECUC\_FlsTst\_00120). ] ( )

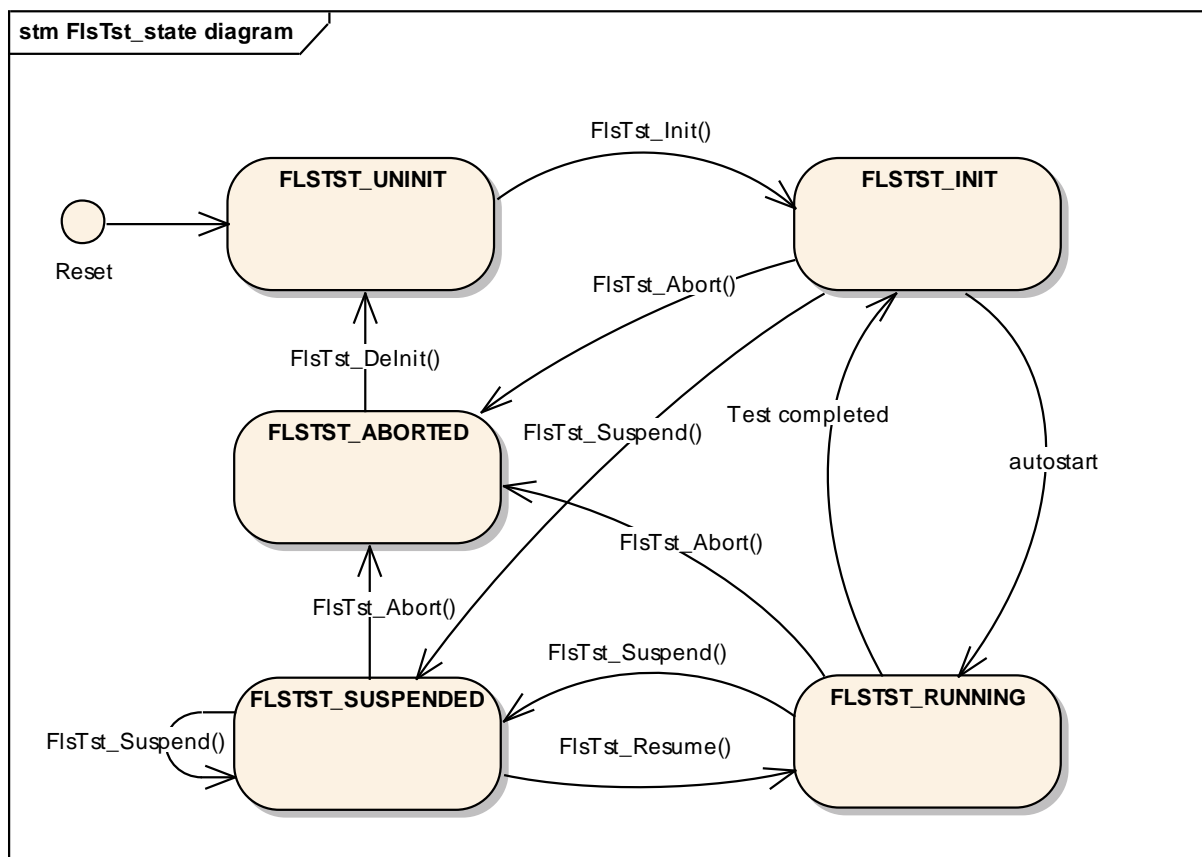
**[SWS\_FlsTst\_00156]** [Each Flash test Interval shall have an Identifier, which shall be incremented by each start of a new test interval in background mode. ] ( )



**Figure 3: Background Test: Test Process**

**7.1.1 State Diagram**

The Flash test driver states in background mode are described in Figure 4. The described states are driver states in background operation mode.



**Figure 4: State Diagram – Background mode**

**[SWS\_FlsTst\_00143]** [ Foreground tests are defined as synchronous tests which shall not be interrupted. The execution of Foreground tests is configurable (see ECUC\_FlsTst\_00086) and can be called after module initialization at any time. ]  
(SRS\_FlsTst\_14219)

## 7.2 Error Classification

**[SWS\_FlsTst\_00007]** [The following errors and exceptions shall be detectable by the Flash Test depending on its build version (development/production mode):

Type of error	Error class	Related error code	Value [hex]
Failure within Flash Test execution state	Development	FLSTST_E_STATE_FAILURE	0x01
API parameter out of specified range	Development	FLSTST_E_PARAM_INVALID	0x02
API service used without module initialization	Development	FLSTST_E_UNINIT	0x03
Flash Test module is already initialized	Development	FLSTST_E_ALREADY_INITIALIZED	0x04
For Variant PB: Configuration pointer is a NULL pointer For Variant PC: Configuration pointer is NOT a NULL pointer	Development	FLSTST_E_INIT_FAILED	0x05
Pointer is a NULL pointer	Development	FLSTST_E_PARAM_POINTER	0x06

To get more details concerning error detection, refer to chapter 8 [API specification](#). ] (SRS\_BSW\_00337, SRS\_BSW\_00409, SRS\_BSW\_00385)

## 7.3 Production Errors

The Flash Test module does not specify any production errors.

## 7.4 Extended Production Errors

**[SWS\_FlsTst\_00168]**

<b>Error Name:</b>	FLSTST_E_FLSTST_FAILURE	
<b>Short Description:</b>	Failure detection in background mode	
<b>Long Description:</b>	This Extended Production Error shall be issued in case a failure is detected in background mode within a test interval.	
<b>Detection Criteria:</b>	Fail	At least one block within a test interval in background mode is NOT OK (see SWS_FlsTst_00167)
	Pass	All blocks within a test interval in background mode are tested with the result OK.(see SWS_FlsTst_00161)
<b>Secondary Parameters:</b>	N/A	
<b>Time Required:</b>	N/A	
<b>Monitor Frequency</b>	continuous	

] ()

## 7.5 Runtime Errors

The Flash Test module does not specify any runtime errors.

## 7.6 Transient Faults

The Flash Test module does not specify any transient faults.

## 7.7 Error Detection

**[SWS\_FlsTst\_00011]** [The function `FlsTst_Init` shall be called first before calling any other Flash Test functions except the function `FlsTst_GetCurrentState`. If this sequence is not respected, the error code `FLSTST_E_UNINIT` shall be reported to the Default Error Tracer (if development error detection is enabled). ]  
(SRS\_BSW\_00406)

## 7.8 Error Notification

Note: Refer to SWS\_BSWGeneral document [9].

## 7.9 Version Check

Note: Refer to SWS\_BSWGeneral document [9].

## 7.10 Debugging Support

No requirement defined.



## 8 API specification

### 8.1 Imported types

This chapter lists data type definitions for the included variables and constants.

#### [SWS\_FlsTst\_00016] [

Module	Header File	Imported Type
Dem	Rte_Dem_Type.h	Dem_EventIdType
	Rte_Dem_Type.h	Dem_EventStatusType
Std_Types	StandardTypes.h	Std_ReturnType
	StandardTypes.h	Std_VersionInfoType

] (SRS\_BSW\_00304)

### 8.2 Type definitions

#### 8.2.1 FlsTst\_ConfigType

#### [SWS\_FlsTst\_00018] [

<b>Name:</b>	FlsTst_ConfigType	
<b>Type:</b>	Structure	
<b>Range:</b>	implementation specific	implementation specific
<b>Description:</b>	This type of external data structure shall contain the initialization data for the Flash Test.	
<b>Available via:</b>	FlsTst.h	

] (SRS\_BSW\_00405, SRS\_BSW\_00438)

[SWS\_FlsTst\_00019] [The type `FlsTst_ConfigType` shall denote the external data structure which contains the configuration data for the Flash Test module.

List of mandatory configuration parameters:

- Memory block definition to test in foreground mode
- Memory block definition to test in background mode
- Test sequence indication in background mode
- Hardware specific configuration] (SRS\_BSW\_00405)

#### 8.2.2 FlsTst\_StateType

#### [SWS\_FlsTst\_00048] [

<b>Name:</b>	FlsTst_StateType		
<b>Type:</b>	Enumeration		
<b>Range:</b>	FLSTST_UNINIT	0x00	The Flash Test is not initialized or not usable.
	FLSTST_INIT	0x01	The Flash Test is initialized and ready to be started.
	FLSTST_RUNNING	0x02	The Flash Test is currently running.
	FLSTST_ABORTED	0x03	The Flash Test is aborted.

	FLSTST_SUSPENDED	0x04	The Flash Test is waiting to be resumed or is waiting to start foreground mode test
<b>Description:</b>	This is a state value returned by the API service FlsTst_GetCurrentState().		
<b>Available via:</b>	FlsTst.h		

] (SRS\_BSW\_00377)

[SWS\_FlsTst\_00049] [For the type FlsTst\_StateType, the enumeration value FLSTST\_UNINIT shall be the default value after a reset.] ( )

### 8.2.3 FlsTst\_TestResultFgndType

[SWS\_FlsTst\_00052] [

<b>Name:</b>	FlsTst_TestResultFgndType		
<b>Type:</b>	Enumeration		
<b>Range:</b>	FLSTST_NOT_TESTED	0x00	There is no result available.
	FLSTST_OK	0x01	The last Flash Test has been tested with OK result.
	FLSTST_NOT_OK	0x02	The last Flash Test has been tested with NOT_OK result.
<b>Description:</b>	Return type of API service FlsTst_GetResultFgnd().		
<b>Available via:</b>	FlsTst.h		

] ( )

[SWS\_FlsTst\_00053] [For the type FlsTst\_TestResultFgndType, the enumeration value FLSTST\_NOT\_TESTED shall be the default value after a reset.] ( )

### 8.2.4 FlsTst\_TestResultBgndType

[SWS\_FlsTst\_00153] [

<b>Name:</b>	FlsTst_TestResultBgndType		
<b>Type:</b>	Structure		
<b>Element:</b>	uint8, uint16, uint32	0..<FlsTstTestIntervalIdEndValue>	current value of FlsTstTestIntervalId, which is incremented by each new start of an test interval.
	FlsTst_TestResultType	result	--
<b>Description:</b>	Return type of API service FlsTst_GetTestResultBgnd().		
<b>Available via:</b>	FlsTst.h		

] (SRS\_FlsTst\_14225)

[SWS\_FlsTst\_00154] [For the type FlsTst\_TestResultBgndType, the enumeration value FLSTST\_RESULT\_NOT\_TESTED shall be the default value after a reset.] (SRS\_FlsTst\_14225)

### 8.2.5 FlsTst\_BlockIdFgndType

[SWS\_FlsTst\_00100] [

<b>Name:</b>	FlsTst_BlockIdFgndType		
<b>Type:</b>	uint8, uint16, uint32		
<b>Range:</b>	0..<FlsTstBlockNumberFgnd > -1	--	The range is dependent on the number of Foreground Flash blocks defined in the configuration structure. The type shall be chosen depending on the MCU platform for best performance.
<b>Description:</b>	This type specifies the identification (ID) for a Flash block to be tested in foreground mode, which is configured in the configuration structure.		
<b>Available via:</b>	FlsTst.h		

] ()

### 8.2.6 FlsTst\_ErrorDetailsType

[SWS\_FlsTst\_00108] [

<b>Name:</b>	FlsTst_ErrorDetailsType		
<b>Type:</b>	Structure		
<b>Range:</b>	implementation specific		implementation specific
<b>Description:</b>	This type shall specify implementation specific error information monitored in the Flash test module.		
<b>Available via:</b>	FlsTst.h		

] ()

### 8.2.7 FlsTst\_TestSignatureFgndType

[SWS\_FlsTst\_00109] [

<b>Name:</b>	FlsTst_TestSignatureFgndType		
<b>Type:</b>	Structure		
<b>Range:</b>	implementation specific		Implementation specific type
<b>Description:</b>	Type for test signature in foreground mode		
<b>Available via:</b>	FlsTst.h		

] ()

### 8.2.8 FlsTst\_TestSignatureBgndType

[SWS\_FlsTst\_00155] [

<b>Name:</b>	FlsTst_TestSignatureBgndType		
<b>Type:</b>	Structure		
<b>Element:</b>	uint8, uint16, uint32	0..<FlsTstTestIntervalIdEndValue>	current value of FlsTstTestIntervalId, which is incremented by each new start of an test interval.
	uint8, uint16, uint32	Implementation specific	It represents the signature value of the last completed test interval. Value might

			be generated from several block signatures.
<b>Description:</b>	Type for test signature in background mode.		
<b>Available via:</b>	FlsTst.h		

] (SRS\_FlsTst\_14225)

## 8.2.9 FlsTst\_TestResultType

### [SWS\_FlsTst\_00164] [

<b>Name:</b>	FlsTst_TestResultType		
<b>Type:</b>	Enumeration		
<b>Range:</b>	FLSTST_RESULT_NOT_TESTED	0x00	There is no test result available.
	FLSTST_RESULT_OK	0x01	The last Flash Test interval has been tested with OK result
	FLSTST_RESULT_NOT_OK	0x02	The last Flash Test interval has been tested with NOT-OK result.
<b>Description:</b>	--		
<b>Available via:</b>	FlsTst.h		

] ()

## 8.3 Function definitions

This is a list of functions provided for upper layer modules.

### 8.3.1 FlsTst\_Init

#### [SWS\_FlsTst\_00017] [

<b>Service name:</b>	FlsTst_Init		
<b>Syntax:</b>	void FlsTst_Init( const FlsTst_ConfigType* ConfigPtr )		
<b>Service ID[hex]:</b>	0x00		
<b>Sync/Async:</b>	Synchronous		
<b>Reentrancy:</b>	Non Reentrant		
<b>Parameters (in):</b>	ConfigPtr	Pointer to configuration set	
<b>Parameters (inout):</b>	None		
<b>Parameters (out):</b>	None		
<b>Return value:</b>	None		
<b>Description:</b>	Service for Flash Test initialization.		
<b>Available via:</b>	FlsTst.h		

] (SRS\_BSW\_00101, SRS\_SPAL\_12057)

[SWS\_FlsTst\_00020] [The function FlsTst\_Init shall initialize all Flash Test relevant registers and global variables and change the execution state to FLSTST\_INIT. ] (SRS\_SPAL\_12057)

**[SWS\_FlsTst\_00022]** [The function `FlsTst_Init` shall only initialize the configured resources and shall not touch resources that are not configured in the configuration file. ] (SRS\_SPAL\_12125)

**[SWS\_FlsTst\_00025]** [If development error detection is enabled, calling the routine `FlsTst_Init` while the Flash Test driver is already initialized shall cause development error `FLSTST_E_ALREADY_INITIALIZED`. The function shall be left without any action. ] (SRS\_BSW\_00386, SRS\_SPAL\_12448)

Note: The `FlsTst_Init` function shall be called only once after a reset, unless an `FlsTst_DeInit` call is made before calling `FlsTst_Init` again.

### 8.3.2 FlsTst\_DeInit

**[SWS\_FlsTst\_00027]** [

<b>Service name:</b>	FlsTst_DeInit
<b>Syntax:</b>	<code>void FlsTst_DeInit(     void )</code>
<b>Service ID[hex]:</b>	0x01
<b>Sync/Async:</b>	Synchronous
<b>Reentrancy:</b>	Non Reentrant
<b>Parameters (in):</b>	None
<b>Parameters (inout):</b>	None
<b>Parameters (out):</b>	None
<b>Return value:</b>	None
<b>Description:</b>	Service for Flash Test De-Initialization.
<b>Available via:</b>	FlsTst.h

] (SRS\_BSW\_00336, SRS\_SPAL\_12163)

**[SWS\_FlsTst\_00028]** [The function `FlsTst_DeInit` shall de-initialize all Flash Test relevant registers and global variables that were initialized by `FlsTst_Init`. ] (SRS\_SPAL\_12163)

**[SWS\_FlsTst\_00029]** [The function `FlsTst_DeInit` shall set the Flash Test module state to `FLSTST_UNINIT`. ] ( )

### 8.3.3 FlsTst\_StartFgnd

**[SWS\_FlsTst\_00149]** [

<b>Service name:</b>	FlsTst_StartFgnd
<b>Syntax:</b>	<code>Std_ReturnType FlsTst_StartFgnd(     FlsTst_BlockIdFgndType FgndBlockId )</code>
<b>Service ID[hex]:</b>	0x02
<b>Sync/Async:</b>	Synchronous
<b>Reentrancy:</b>	Non Reentrant

<b>Parameters (in):</b>	FgndBlockId	Number of the foreground test to be executed. This is dependent on configuration.
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	E_OK: Foreground test processed E_NOT_OK: Foreground test not accepted
<b>Description:</b>	Service for executing foreground Flash Test.	
<b>Available via:</b>	FlsTst.h	

] (SRS\_FlsTst\_14219)

**[SWS\_FlsTst\_00050]** [The function `FlsTst_StartFgnd` is only applicable for Foreground mode Flash Test operation. ] (SRS\_FlsTst\_14219)

**[SWS\_FlsTst\_00051]** [The function `FlsTst_StartFgnd` shall be pre compile time configurable On/Off by the configuration parameter: `FlsTst_StartFgndApi`. ] (SRS\_FlsTst\_14219)

**[SWS\_FlsTst\_00033]** [If development error detection is enabled and the parameter `FgndBlockId` is out of range, the DET error value `FLSTST_E_PARAM_INVALID` shall be raised and the function shall return without any action with return value `E_NOT_OK`. ] (SRS\_BSW\_00323, SRS\_BSW\_00386, SRS\_SPAL\_12448, SRS\_FlsTst\_14219)

### 8.3.4 FlsTst\_Abort

**[SWS\_FlsTst\_00030]** [

<b>Service name:</b>	FlsTst_Abort	
<b>Syntax:</b>	<pre>void FlsTst_Abort(     void )</pre>	
<b>Service ID[hex]:</b>	0x03	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	None	
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	None	
<b>Description:</b>	Service for aborting the Flash Test.	
<b>Available via:</b>	FlsTst.h	

] (SRS\_FlsTst\_14217)

**[SWS\_FlsTst\_00031]** [This function shall abort Flash test background operation and set the state to `FLSTST_ABORTED`. When the `FlsTst_Abort` function is called, `FlsTst_MainFunction` shall finish the current atomic sequence it is running. ] (SRS\_FlsTst\_14217)

**[SWS\_FlsTst\_00032]** [After an `FlsTst_Abort` call, `FlsTst_MainFunction` shall not begin testing again when called by the scheduler until after a complete re-initialization of the Flash test module. ] (SRS\_FlsTst\_14217)

### 8.3.5 FlsTst\_Suspend

**[SWS\_FlsTst\_00034]** [

<b>Service name:</b>	FlsTst_Suspend
<b>Syntax:</b>	void FlsTst_Suspend( void )
<b>Service ID[hex]:</b>	0x04
<b>Sync/Async:</b>	Synchronous
<b>Reentrancy:</b>	Non Reentrant
<b>Parameters (in):</b>	None
<b>Parameters (inout):</b>	None
<b>Parameters (out):</b>	None
<b>Return value:</b>	None
<b>Description:</b>	Service for suspending current operation of the Flash Test, until <code>FlsTst_Resume</code> is called.
<b>Available via:</b>	FlsTst.h

] (SRS\_FlsTst\_14215)

**[SWS\_FlsTst\_00036]** [The function `FlsTst_Suspend` is only applicable for Background mode Flash Test operation. ] (SRS\_FlsTst\_14215)

**[SWS\_FlsTst\_00037]** [The function `FlsTst_Suspend` shall set the Flash Test execution state to `FLSTST_SUSPENDED` in case the execution state was `FLSTST_RUNNING` or `FLSTST_INIT`. ] (SRS\_FlsTst\_14215)

**[SWS\_FlsTst\_00088]** [The function `FlsTst_Suspend` shall be pre compile time configurable On/Off by the configuration parameter: `FlsTst_SuspendResumeApi`. ] (SRS\_FlsTst\_14215)

### 8.3.6 FlsTst\_Resume

**[SWS\_FlsTst\_00035]** [

<b>Service name:</b>	FlsTst_Resume
<b>Syntax:</b>	void FlsTst_Resume( void )
<b>Service ID[hex]:</b>	0x05
<b>Sync/Async:</b>	Synchronous
<b>Reentrancy:</b>	Non Reentrant
<b>Parameters (in):</b>	None
<b>Parameters (inout):</b>	None

<b>Parameters (out):</b>	None
<b>Return value:</b>	None
<b>Description:</b>	Service for continuing the Flash Test at the point it was suspended.
<b>Available via:</b>	FlsTst.h

] (SRS\_FlsTst\_14216)

**[SWS\_FlsTst\_00038]** [The function `FlsTst_Resume` shall change the execution state to `FLSTST_RUNNING` when commanded to continue and the current execution state is `FLSTST_SUSPENDED`. ] (SRS\_FlsTst\_14216)

**[SWS\_FlsTst\_00039]** [If development error detection is enabled and the execution state of the Flash Test module is not `FLSTST_SUSPENDED`, the Flash Test module shall report the error value `FLSTST_E_STATE_FAILURE` to the DET, and then immediately return from the function. ] (SRS\_SPAL\_12448, SRS\_FlsTst\_14216)

**[SWS\_FlsTst\_00162]** [The function `FlsTst_Resume` is only applicable for Background mode Flash Test operation. ] ( )

**[SWS\_FlsTst\_00089]** [The function `FlsTst_Resume` shall be pre compile time configurable On/Off by the configuration parameter: `FlsTst_SuspendResumeApi`. ] (SRS\_BSW\_00386, SRS\_FlsTst\_14216)

### 8.3.7 FlsTst\_GetCurrentState

**[SWS\_FlsTst\_00040]** [

<b>Service name:</b>	FlsTst_GetCurrentState	
<b>Syntax:</b>	<code>FlsTst_StateType FlsTst_GetCurrentState ( void )</code>	
<b>Service ID[hex]:</b>	0x06	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	None	
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	<code>FlsTst_StateType</code>	<code>FLSTST_UNINIT</code> The Flash Test is not initialized or not usable. <code>FLSTST_INIT</code> The Flash Test is initialized and ready to be started. <code>FLSTST_RUNNING</code> The Flash Test is currently running. <code>FLSTST_ABORTED</code> The Flash Test is aborted. <code>FLSTST_SUSPENDED</code> The Flash Test is waiting to be resumed or is waiting to start foreground mode test
<b>Description:</b>	Service returns the current Flash Test execution state.	
<b>Available via:</b>	FlsTst.h	

] (SRS\_SPAL\_00157, SRS\_FlsTst\_14211)

**[SWS\_FlsTst\_00041]** [The function `FlsTst_GetCurrentState` shall return the current Flash Test execution state. ] (SRS\_FlsTst\_14211)



**[SWS\_FlsTst\_00091]** [The function `FlsTst_GetCurrentState` shall be pre compile time configurable On/Off by the configuration parameter: `FlsTst_GetCurrentStateApi`.] (SRS\_BSW\_00386, SRS\_FlsTst\_14211)

### 8.3.8 FlsTst\_GetTestResultBgnd

**[SWS\_FlsTst\_00042]** [

<b>Service name:</b>	FlsTst_GetTestResultBgnd	
<b>Syntax:</b>	FlsTst_TestResultBgndType FlsTst_GetTestResultBgnd( void )	
<b>Service ID[hex]:</b>	0x07	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	None	
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	FlsTst_TestResultBgndType	See type definition
<b>Description:</b>	Service returns the Background Flash Test result.	
<b>Available via:</b>	FlsTst.h	

] (SRS\_BSW\_00339, SRS\_SPAL\_00157, SRS\_FlsTst\_14214)

**[SWS\_FlsTst\_00043]** [The function `FlsTst_GetTestResultBgnd` shall return the Flash test result and Test Interval Id of the last background test. ] (SRS\_FlsTst\_14214)

**[SWS\_FlsTst\_00093]** [The function `FlsTst_GetTestResultBgnd` shall be pre compile time configurable On/Off by the configuration parameter: `FlsTst_GetTestResultBgndApi`.] (SRS\_BSW\_00386, SRS\_FlsTst\_14214)

### 8.3.9 FlsTst\_GetTestResultFgnd

**[SWS\_FlsTst\_00112]** [

<b>Service name:</b>	FlsTst_GetTestResultFgnd	
<b>Syntax:</b>	FlsTst_TestResultFgndType FlsTst_GetTestResultFgnd( void )	
<b>Service ID[hex]:</b>	0x0f	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	None	
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	FlsTst_TestResultFgndType	See type definition
<b>Description:</b>	Service returns the Foreground Flash Test result.	
<b>Available via:</b>	FlsTst.h	

] (SRS\_BSW\_00339, SRS\_SPAL\_00157, SRS\_FlsTst\_14214)

**[SWS\_FlsTst\_00113]** [The function `FlsTst_GetTestResultFgnd` shall return the Flash test result of the last foreground test. ] (SRS\_FlsTst\_14214)

**[SWS\_FlsTst\_00114]** [The function `FlsTst_GetTestResultFgnd` shall be pre compile time configurable On/Off by the configuration parameter: `FlsTst_GetTestResultFgndApi`. ] (SRS\_BSW\_00386, SRS\_FlsTst\_14214)

### 8.3.10 FlsTst\_GetVersionInfo

**[SWS\_FlsTst\_00044]** [

<b>Service name:</b>	FlsTst_GetVersionInfo	
<b>Syntax:</b>	<pre>void FlsTst_GetVersionInfo(     Std_VersionInfoType* versioninfo )</pre>	
<b>Service ID[hex]:</b>	0x08	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	None	
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	versioninfo	Pointer to where to store the version information of this module.
<b>Return value:</b>	None	
<b>Description:</b>	Service returns the version information of this module.	
<b>Available via:</b>	FlsTst.h	

] (SRS\_BSW\_00407, SRS\_BSW\_00411)

### 8.3.11 FlsTst\_GetTestSignatureBgnd

**[SWS\_FlsTst\_00054]** [

<b>Service name:</b>	FlsTst_GetTestSignatureBgnd	
<b>Syntax:</b>	<pre>FlsTst_TestSignatureBgndType FlsTst_GetTestSignatureBgnd(     void )</pre>	
<b>Service ID[hex]:</b>	0x09	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	None	
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	FlsTst_TestSignatureBgndType	See type definition
<b>Description:</b>	Service returns the Flash Test result in background mode.	
<b>Available via:</b>	FlsTst.h	

] (SRS\_FlsTst\_14213, SRS\_SPAL\_00157) **[SWS\_FlsTst\_00055]** [The function `FlsTst_GetTestSignatureBgnd` shall return the signature and Test Interval Id of the last background test. ] (SRS\_FlsTst\_14213)

**[SWS\_FlsTst\_00056]** [The function `FlsTst_GetTestSignatureBgnd` shall be pre compile time configurable On/Off by the configuration parameter: `FlsTst_GetTestSignatureBgndApi`. ] (SRS\_BSW\_00386, SRS\_FlsTst\_14213)

**[SWS\_FlsTst\_00115]** [If no signature is available, the function `FlsTst_GetTestSignatureBgnd` shall return the default signature value “0x0”. ] (SRS\_FlsTst\_14213)

### 8.3.12 FlsTst\_GetTestSignatureFgnd

**[SWS\_FlsTst\_00057]** [

<b>Service name:</b>	FlsTst_GetTestSignatureFgnd	
<b>Syntax:</b>	FlsTst_TestSignatureFgndType FlsTst_GetTestSignatureFgnd( void )	
<b>Service ID[hex]:</b>	0x0a	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	None	
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	FlsTst_TestSignatureFgndType	See type definition
<b>Description:</b>	Service returns the Flash Test result in foreground mode.	
<b>Available via:</b>	FlsTst.h	

] (SRS\_FlsTst\_14213, SRS\_SPAL\_00157)

**[SWS\_FlsTst\_00058]** [The function `FlsTst_GetTestSignatureFgnd` shall return the signature of the last foreground test. ] (SRS\_FlsTst\_14213)

**[SWS\_FlsTst\_00059]** [The function `FlsTst_GetTestSignatureFgnd` shall be pre compile time configurable On/Off by the configuration parameter: `FlsTst_GetTestSignatureFgndApi`. ] (SRS\_BSW\_00386, SRS\_FlsTst\_14213)

**[SWS\_FlsTst\_00116]** [If no signature is available, the function `FlsTst_GetTestSignatureFgnd` shall return the default signature value “0x0”. ] (SRS\_FlsTst\_14213)

### 8.3.13 FlsTst\_GetErrorDetails

**[SWS\_FlsTst\_00060]** [

<b>Service name:</b>	FlsTst_GetErrorDetails	
<b>Syntax:</b>	FlsTst_ErrorDetailsType FlsTst_GetErrorDetails( void )	
<b>Service ID[hex]:</b>	0x0b	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Reentrant	
<b>Parameters (in):</b>	None	
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	FlsTst_ErrorDetailsType	See type definition
<b>Description:</b>	Service returns error details monitored from the Flash module.	
<b>Available via:</b>	FlsTst.h	

] (SRS\_BSW\_00339, SRS\_SPAL\_00157, SRS\_FlsTst\_14223)

**[SWS\_FlsTst\_00061]** [The function `FlsTst_GetErrorDetails` shall return the error details monitored from the Flash Test driver. ] (SRS\_FlsTst\_14223)

**[SWS\_FlsTst\_00062]** [The function `FlsTst_GetErrorDetails` shall be pre compile time configurable On/Off by the configuration parameter: `FlsTst_GetErrorDetailsApi`. ] (SRS\_BSW\_00386, SRS\_FlsTst\_14223)

### 8.3.14 FlsTst\_TestEcc

**[SWS\_FlsTst\_00063]** [

<b>Service name:</b>	FlsTst_TestEcc	
<b>Syntax:</b>	Std_ReturnType FlsTst_TestEcc( void )	
<b>Service ID[hex]:</b>	0x0c	
<b>Sync/Async:</b>	Synchronous	
<b>Reentrancy:</b>	Non Reentrant	
<b>Parameters (in):</b>	None	
<b>Parameters (inout):</b>	None	
<b>Parameters (out):</b>	None	
<b>Return value:</b>	Std_ReturnType	see type definition
<b>Description:</b>	Service executes a test of ECC hardware. This is only applicable in case the hardware provides such functionality.	
<b>Available via:</b>	FlsTst.h	

] (SRS\_BSW\_00357, SRS\_FlsTst\_14224) **[SWS\_FlsTst\_00064]** [The function `FlsTst_TestEcc` shall execute a test of the ECC circuitry. ] (SRS\_FlsTst\_14224)

**[SWS\_FlsTst\_00065]** [The function `FlsTst_TestEcc` shall be pre compile time configurable On/Off by the configuration parameter: `FlsTst_TestEccApi`. ] (SRS\_BSW\_00386, SRS\_FlsTst\_14224)

## 8.4 Callback notifications

Since the Flash Test is a driver module, it does not provide any callback functions for lower layer modules.

## 8.5 Scheduled functions

### 8.5.1 FlsTst\_MainFunction

**[SWS\_FlsTst\_00066]** [

<b>Service name:</b>	FlsTst_MainFunction
<b>Syntax:</b>	void FlsTst_MainFunction( void )
<b>Service ID[hex]:</b>	0x0d
<b>Description:</b>	Service for executing the Flash Test in background mode.
<b>Available via:</b>	SchM FlsTst.h

] (SRS\_FlsTst\_14208, SRS\_FlsTst\_14209)

**[SWS\_FlsTst\_00067]** [The function `FlsTst_MainFunction` shall test the defined flash blocks in background mode, starting with the first flash block in the `FlsTstConfigParams`. ] ()

**[SWS\_FlsTst\_00068]** [The function `FlsTst_MainFunction` shall set the Flash Test execution state from `FLSTST_INIT` to `FLSTST_RUNNING` when calling the function the first time after initialization or after a complete test interval. ] ()

**[SWS\_FlsTst\_00069]** [When `FlsTstTestResultSignature` is true, the function `FlsTst_MainFunction` shall provide the test signatures of all blocks within a test interval. ] ()

**[SWS\_FlsTst\_00161]** [When `FlsTstTestResultSignature` is disabled, the function `FlsTst_MainFunction` shall set the overall result status for a test interval to `FLSTST_RESULT_OK` if all blocks are tested with result status OK. Additionally the `DEM_FLSTST_E_FLSTST_FAILURE` shall be triggered with the detection criteria "Pass". ] ()

**[SWS\_FlsTst\_00167]** [ When `FlsTstTestResultSignature` is disabled, the function `FlsTst_MainFunction` shall set the overall result status for a test interval to `FLSTST_RESULT_NOT_OK` if at least one block test result is not ok regardless whether all blocks are already tested or not. Additionally the `DEM_FLSTST_E_FLSTST_FAILURE` shall be triggered with the detection criteria "Fail". ] ()

**[SWS\_FlsTst\_00070]** [After the function `FlsTst_MainFunction` has completed testing all flash blocks, the next call of the function `FlsTst_MainFunction` shall restart the test from the beginning. ] ( )

**[SWS\_FlsTst\_00071]** [The function `FlsTst_MainFunction` shall test a defined number of flash cells within one call. The defined number is specified by configuration (see `ECUC_FlsTst_00161`). ] (SRS\_FlsTst\_14208, SRS\_FlsTst\_14209)

**[SWS\_FlsTst\_00117]** [The function `FlsTst_MainFunction` shall test a defined number of flash cells without checking user request for Abort or Suspend. The defined number is specified by configuration (see `ECUC_FlsTst_00120`). ] ( )

**[SWS\_FlsTst\_00121]** [The function `FlsTst_MainFunction` shall increment the Test Interval Id by 1 before start of a new test interval. The first test interval shall have the Test Interval Id = "0". If the end value = `FlsTstIntervalIdEndValue` is reached, Test Interval Id shall start with value "0" again. The value shall be provided as part of the return values of `FlsTst_GetTestResultBgnd` and `FlsTst_GetTestSignatureBgnd`. ] ( )

## 8.6 Expected Interfaces

In this chapter, all interfaces required from other modules are listed.

### 8.6.1 Mandatory Interfaces

This chapter defines all interfaces that are required to fulfill the core functionality of the module.

**[SWS\_FlsTst\_00072]** [

<i>API function</i>	<i>Header File</i>	<i>Description</i>
<code>Dem_SetEventStatus</code>	<code>Dem.h</code>	Called by SW-Cs or BSW modules to report monitor status information to the Dem. BSW modules calling <code>Dem_SetEventStatus</code> can safely ignore the return value.

] (SRS\_SPAL\_00157)

### 8.6.2 Optional Interfaces

This chapter defines all interfaces that are required to fulfill an optional functionality of the module.

**[SWS\_FlsTst\_00073]** [

<i>API function</i>	<i>Header File</i>	<i>Description</i>
<code>Det_ReportError</code>	<code>Det.h</code>	Service to report development errors.

] (SRS\_SPAL\_00157)

### 8.6.3 Configurable interfaces

In this chapter, all interfaces are listed where the target function could be configured. The target function is usually a callback function. The names of these kinds of interfaces are not fixed because they are configurable.

**[SWS\_FlsTst\_00074]** [The callback notifications shall be configurable as function pointers within the initialization data structure (FlsTst\_ConfigType). ] ( )

**[SWS\_FlsTst\_00075]** [The callback notifications shall have no parameters and no return value. ] ( )

**[SWS\_FlsTst\_00076]** [If a callback notification is configured as null pointer, the Flash Test module shall not execute the callback. ] ( )

#### 8.6.3.1 FlsTst\_TestCompleted Notification

**[SWS\_FlsTst\_00077]** [

<b>Service name:</b>	FlsTst_TestCompletedNotification
<b>Syntax:</b>	void FlsTst_TestCompletedNotification( void )
<b>Service ID[hex]:</b>	0x0e
<b>Sync/Async:</b>	Synchronous
<b>Reentrancy:</b>	Don't care
<b>Parameters (in):</b>	None
<b>Parameters (inout):</b>	None
<b>Parameters (out):</b>	None
<b>Return value:</b>	None
<b>Description:</b>	The function FlsTst_TestCompleted shall be called every time when a complete test cycle had been tested.
<b>Available via:</b>	FlsTst.h

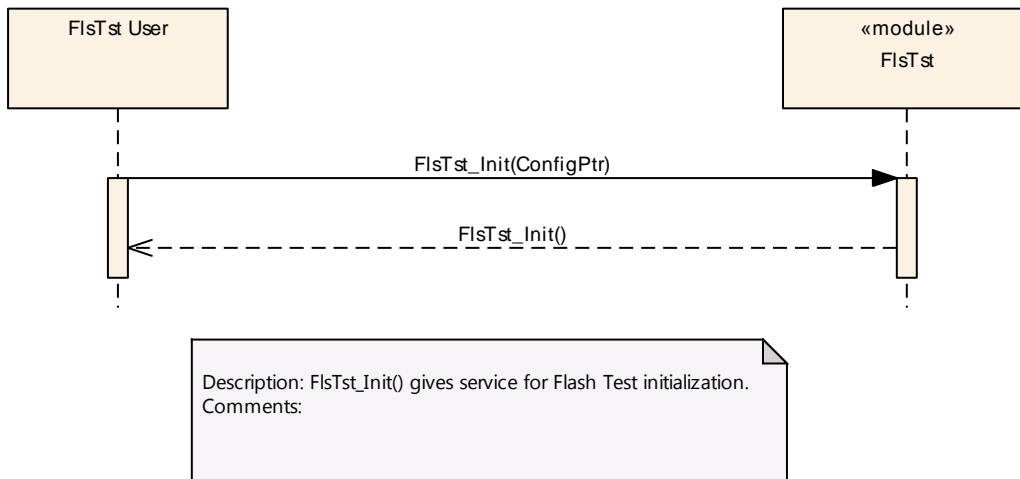
] (SRS\_SPAL\_00157, SRS\_FlsTst\_14212)

**[SWS\_FlsTst\_00078]** [The Flash Test module shall call the callback notification FlsTst\_TestCompleted every time when it has tested a complete test cycle of a flash test in background mode. ] (SRS\_FlsTst\_14212)

**[SWS\_FlsTst\_00159]** [The call of function FlsTst\_TestCompleted shall be pre compile time configurable On/Off by the configuration parameter FlsTstTestCompletedNotificationSupported. ] ( )

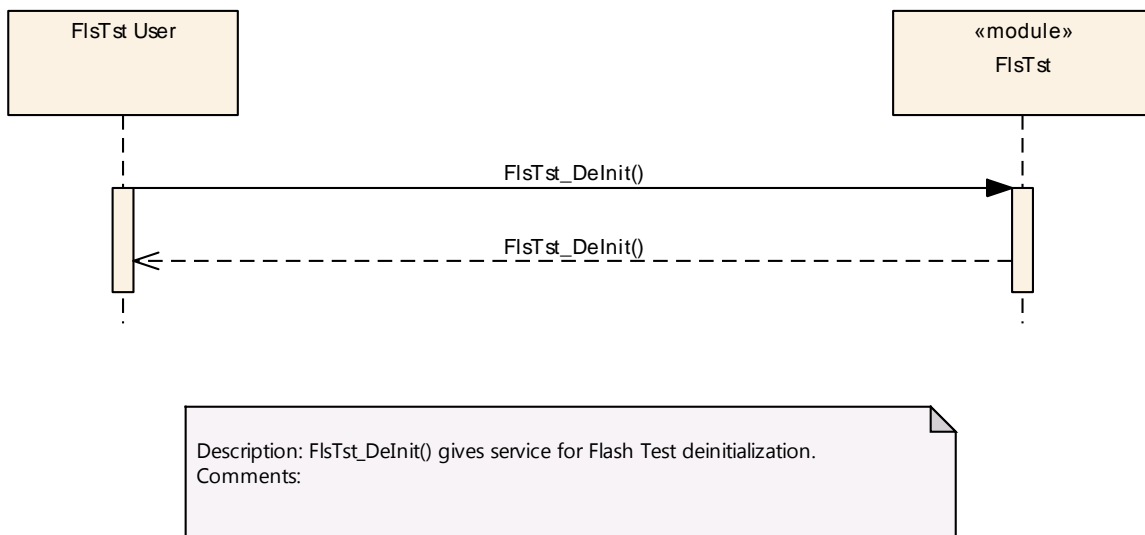
## 9 Sequence diagrams

### 9.1 Initialization



**Figure 5: Flash test driver initialization**

### 9.2 De-initialization



**Figure 6: Flash test driver de-initialization**



### 9.3 Background Test

#### 9.3.1 Test Result Calculation within Flash test driver

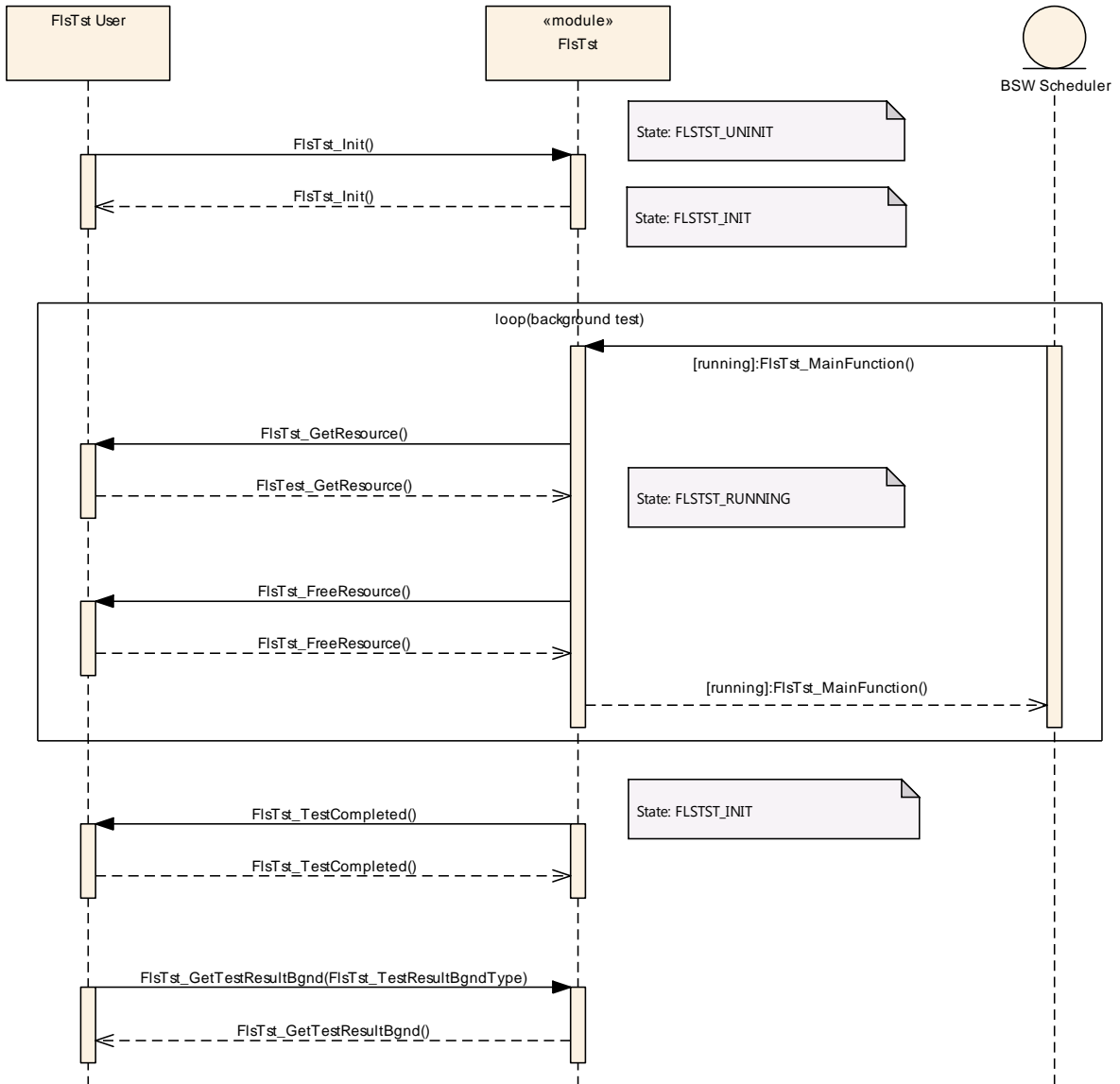
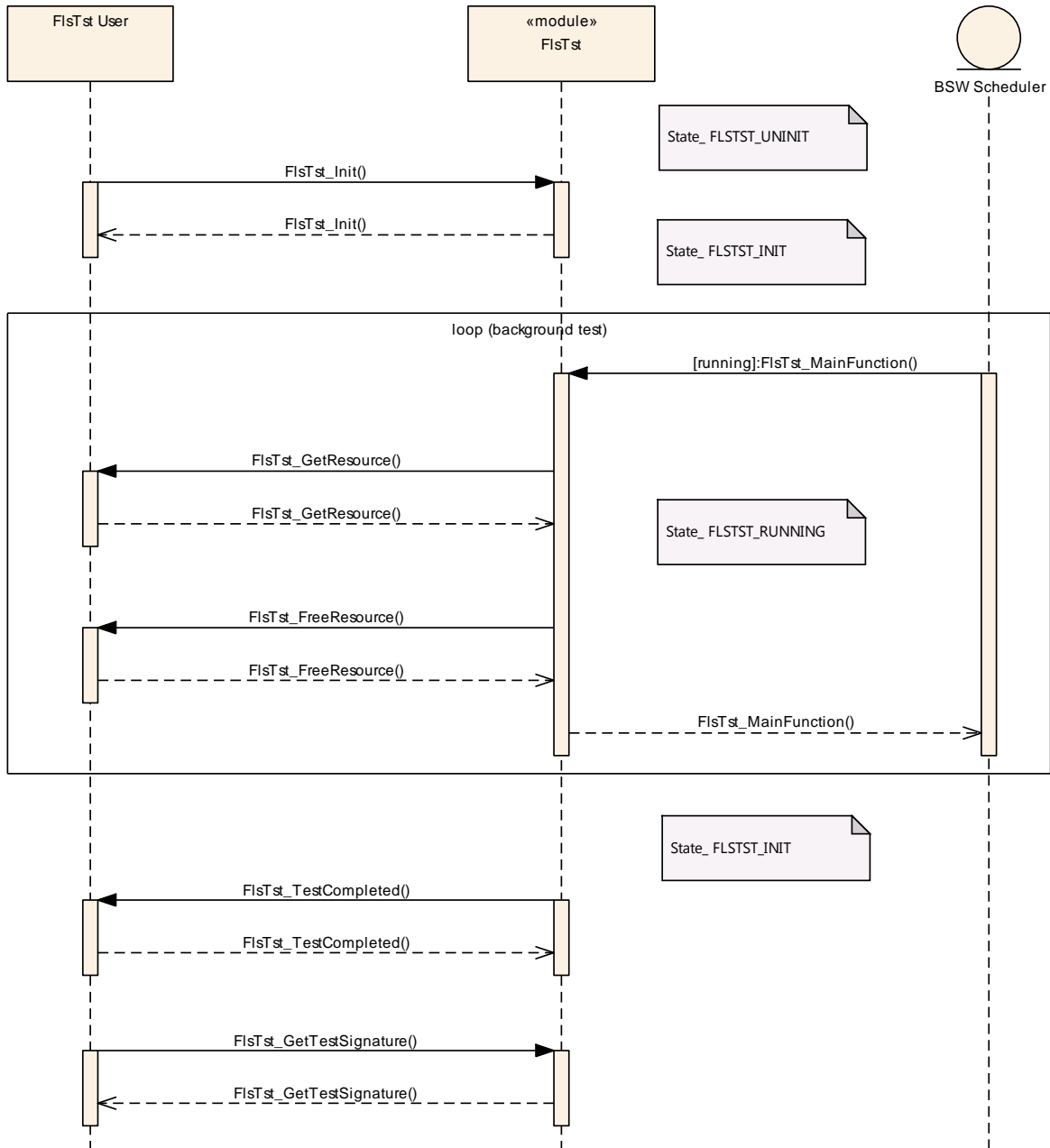


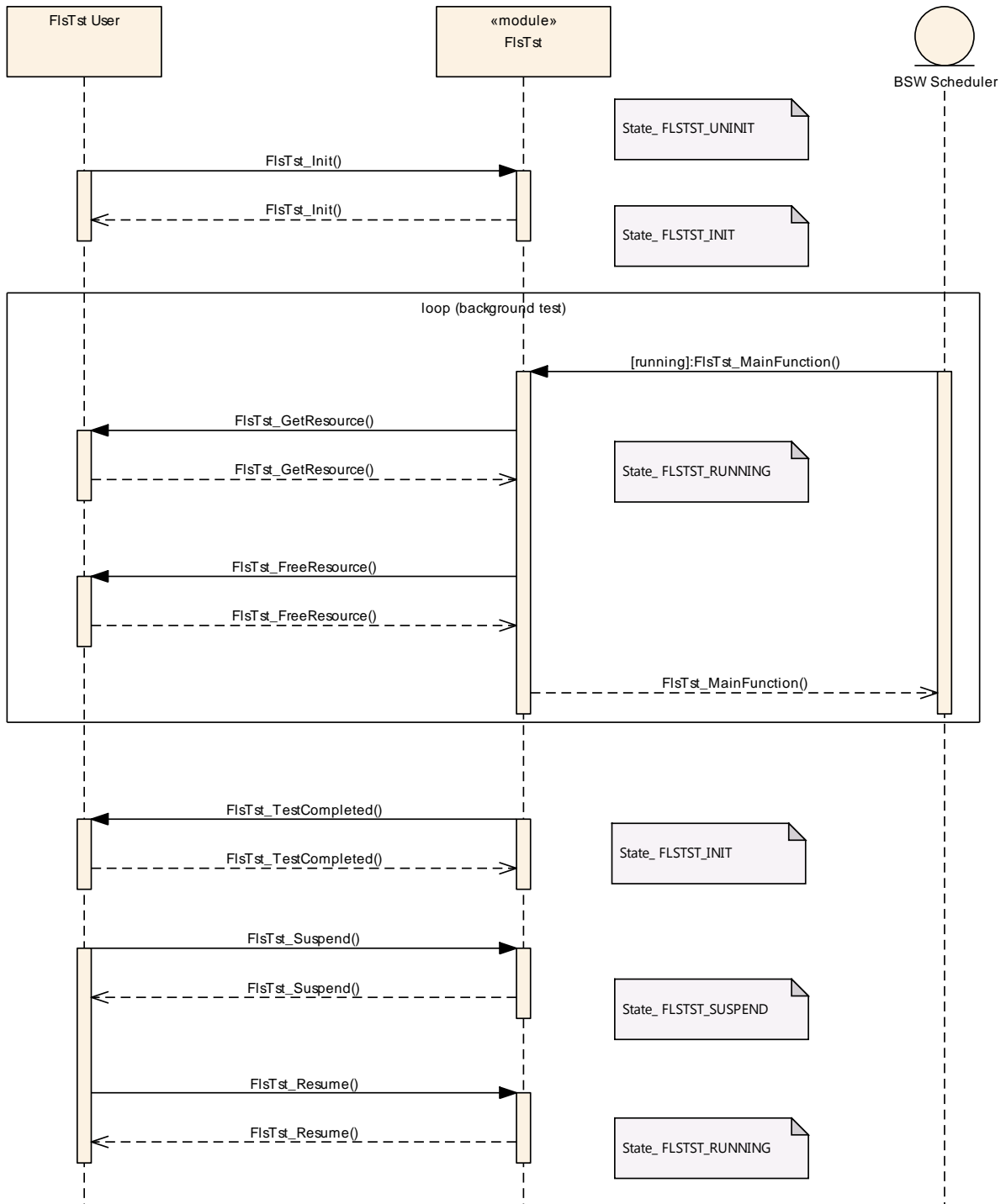
Figure 7: Background Test – Test result calculation in Flash test driver

**9.3.2 Test signature provided to caller**



**Figure 8: Background Test – Test signature provided to caller**

### 9.4 Suspend and Resume Background Testing



**Figure 9: Suspend and Resume Background Testing**

### 9.5 Foreground Task interrupts Background Task

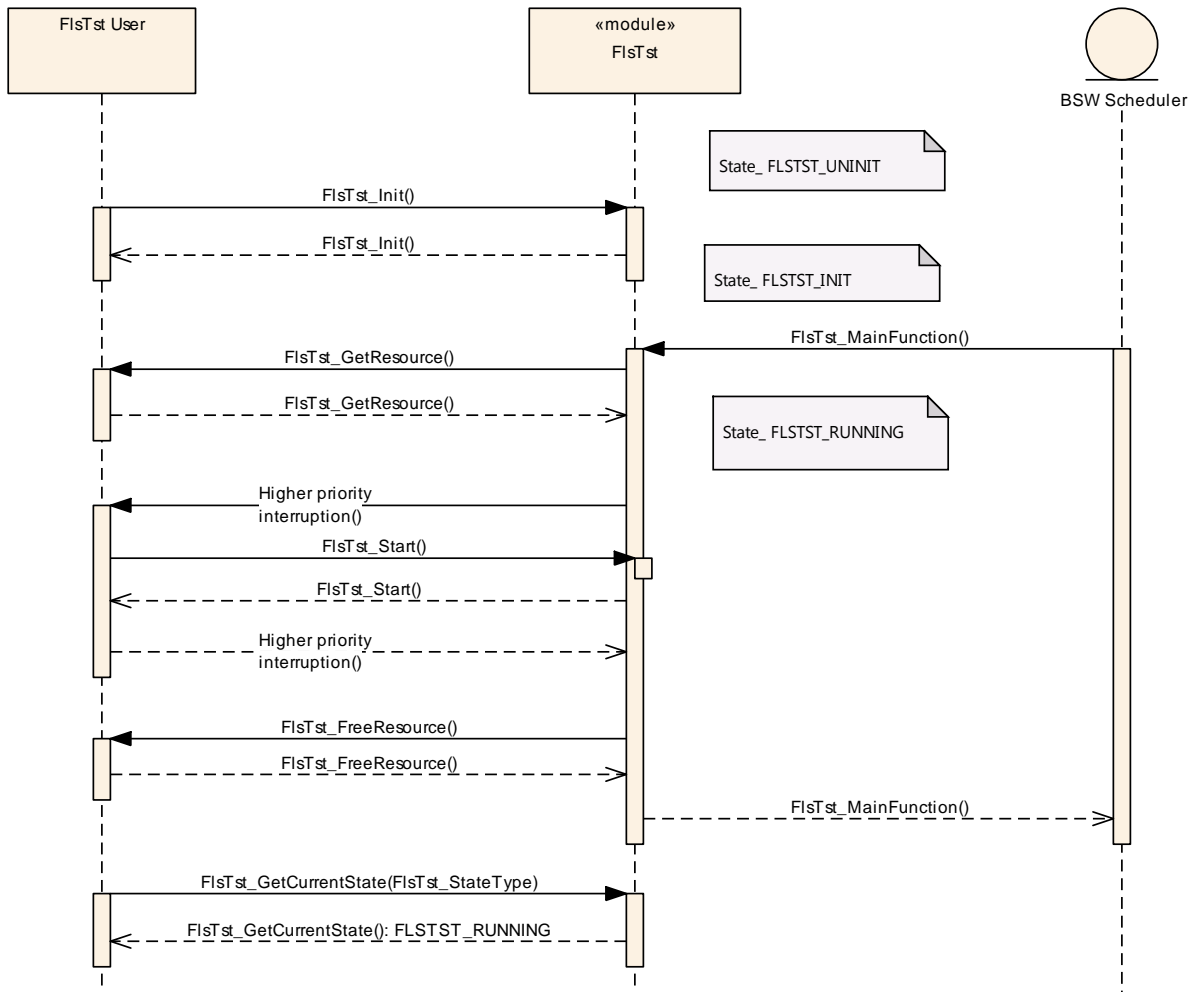


Figure 10: Foreground task interrupts Background Task

## 10 Configuration specification

### 10.1 Specification template for configuration parameters

<b>Label</b>	<b>Description</b>
x	The configuration parameter shall be of configuration class <i>Pre-compile time</i> .
--	The configuration parameter shall never be of configuration class <i>Pre-compile time</i> .

Link time - specifies whether the configuration parameter shall be of configuration class *Link time* or not

<b>Label</b>	<b>Description</b>
x	The configuration parameter shall be of configuration class <i>Link time</i> .
--	The configuration parameter shall never be of configuration class <i>Link time</i> .

Post Build - specifies whether the configuration parameter shall be of configuration class *Post Build* or not

<b>Label</b>	<b>Description</b>
x	The configuration parameter shall be of configuration class <i>Post Build</i> and no specific implementation is required.
L	<i>Loadable</i> - the configuration parameter shall be of configuration class <i>Post Build</i> and only one configuration parameter set resides in the ECU.
M	<i>Multiple</i> - the configuration parameter shall be of configuration class <i>Post Build</i> and is selected out of a set of multiple parameters by passing a dedicated pointer to the init function of the module.
--	The configuration parameter shall never be of configuration class <i>Post Build</i> .

### 10.2 Containers and configuration parameters

The following chapters summarize all configuration parameters. The detailed meanings of the parameters describe Chapters [Functional specification](#) and Chapter [API specification](#).

#### 10.2.1 Variants

See AUTOSAR\_SWS\_BSWGeneral for definition of variants.

#### 10.2.2 FlsTst

<b>SWS Item</b>	ECUC_FlsTst_00135 :
<b>Module Name</b>	FlsTst
<b>Module Description</b>	--
<b>Post-Build Variant Support</b>	true
<b>Supported Config Variants</b>	VARIANT-POST-BUILD, VARIANT-PRE-COMPILE

<b>Included Containers</b>		
<b>Container Name</b>	<b>Multiplicity</b>	<b>Scope / Dependency</b>
FlsTstConfigSet	1	This container contains the configuration parameters and sub containers of the AUTOSAR FlsTst module.
FlsTstConfigurationOfOptApiServices	1	--
FlsTstDemEventParameterRefs	0..1	Container for the references to DemEventParameter elements which shall be invoked using the API Dem_SetEventStatus in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId symbolic value. The standardized errors are provided in this container and can be extended by vendor-specific error references.
FlsTstGeneral	1	--

### 10.2.3 FlsTstGeneral

<b>SWS Item</b>	<b>ECUC_FlsTst_00082 :</b>
<b>Container Name</b>	FlsTstGeneral
<b>Description</b>	--
<b>Configuration Parameters</b>	

<b>SWS Item</b>	<b>ECUC_FlsTst_00083 :</b>		
<b>Name</b>	FlsTstDevErrorDetect		
<b>Parent Container</b>	FlsTstGeneral		
<b>Description</b>	Switches the development error detection and notification on or off. <ul style="list-style-type: none"> <li>• true: detection and notification is enabled.</li> <li>• false: detection and notification is disabled.</li> </ul>		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00172 :</b>		
<b>Name</b>	FlsTstMainFunctionPeriod		
<b>Parent Container</b>	FlsTstGeneral		
<b>Description</b>	Determines the frequency at which the FlsTstMainFunction is called in [s].		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucFloatParamDef		
<b>Range</b>	]0 .. INF[		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	

	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU		

<b>SWS Item</b>	<b>ECUC_FlsTst_00161 :</b>		
<b>Name</b>	FlsTstNumberOfTestedCells		
<b>Parent Container</b>	FlsTstGeneral		
<b>Description</b>	Configures the Number of cells to be tested in background mode during one scheduled task (FlsTst_MainFunction() call).		
<b>Multiplicity</b>	0..1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 4294967295		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00120 :</b>		
<b>Name</b>	FlsTstNumberOfTestedCellsAtomic		
<b>Parent Container</b>	FlsTstGeneral		
<b>Description</b>	Configures the Number of cells to be tested in background mode without checking user requests (Abort, Suspend).		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 4294967295		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00084 :</b>		
<b>Name</b>	FlsTstTestCompletedNotificationSupported		
<b>Parent Container</b>	FlsTstGeneral		
<b>Description</b>	Switch to indicate that the notification is supported.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	true		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00158 :</b>		
<b>Name</b>	FlsTstTestIntervalIdEndValue		
<b>Parent Container</b>	FlsTstGeneral		
<b>Description</b>	Defines the end value of the Test Interval Id.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 4294967295		
<b>Default value</b>	--		

<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00160 :</b>		
<b>Name</b>	FlsTstTestResultSignature		
<b>Parent Container</b>	FlsTstGeneral		
<b>Description</b>	Configures the result of the test in background mode: True: Test Result is a signature (see SWS_FlsTst_00155, SWS_FlsTst_00054) False: Test Result is ok/not ok (see SWS_FlsTst_00153, SWS_FlsTst_00042)		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00175 :</b>		
<b>Name</b>	FlsTstEcucPartitionRef		
<b>Parent Container</b>	FlsTstGeneral		
<b>Description</b>	Maps the Flash test driver to zero or one ECUC partition to make the driver API available in this partition. <b>Tags:</b> atp.Status=draft		
<b>Multiplicity</b>	0..1		
<b>Type</b>	Reference to [ EcucPartition ]		
<b>Post-Build Variant Multiplicity</b>	true		
<b>Post-Build Variant Value</b>	true		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: ECU		

**No Included Containers**

#### 10.2.4 FlsTstConfigurationOfOptApiServices

<b>SWS Item</b>	<b>ECUC_FlsTst_00085 :</b>		
<b>Container Name</b>	FlsTstConfigurationOfOptApiServices		
<b>Description</b>	--		
<b>Configuration Parameters</b>			



<b>SWS Item</b>	<b>ECUC_FlsTst_00092 :</b>		
<b>Name</b>	FlsTstGetCurrentStateApi		
<b>Parent Container</b>	FlsTstConfigurationOfOptApiServices		
<b>Description</b>	Adds / removes the service FlsTst_GetCurrentState() from the code.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00098 :</b>		
<b>Name</b>	FlsTstGetErrorDetailsApi		
<b>Parent Container</b>	FlsTstConfigurationOfOptApiServices		
<b>Description</b>	Adds / removes the service FlsTst_GetErrorDetails() from the code.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00094 :</b>		
<b>Name</b>	FlsTstGetTestResultBgndApi		
<b>Parent Container</b>	FlsTstConfigurationOfOptApiServices		
<b>Description</b>	Adds / removes the service FlsTst_GetTestResultBgnd() from the code.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00150 :</b>		
<b>Name</b>	FlsTstGetTestResultFgndApi		
<b>Parent Container</b>	FlsTstConfigurationOfOptApiServices		
<b>Description</b>	Adds / removes the service FlsTst_GetTestResultFgnd() from the code.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00096 :</b>		
-----------------	----------------------------	--	--

<b>Name</b>	FlsTstGetTestSignatureBgndApi		
<b>Parent Container</b>	FlsTstConfigurationOfOptApiServices		
<b>Description</b>	Adds / removes the service FlsTst_GetTestSignatureBgnd() from the code.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00097 :</b>		
<b>Name</b>	FlsTstGetTestSignatureFgndApi		
<b>Parent Container</b>	FlsTstConfigurationOfOptApiServices		
<b>Description</b>	Adds / removes the service FlsTst_GetTestSignatureFgnd() from the code.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00086 :</b>		
<b>Name</b>	FlsTstStartFgndApi		
<b>Parent Container</b>	FlsTstConfigurationOfOptApiServices		
<b>Description</b>	Adds / removes the service FlsTst_StartFgnd() from the code.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00087 :</b>		
<b>Name</b>	FlsTstSuspendResumeApi		
<b>Parent Container</b>	FlsTstConfigurationOfOptApiServices		
<b>Description</b>	Adds / removes the services FlsTst_Suspend() and FlsTst_Resume() from the code.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00099 :</b>		
<b>Name</b>	FlsTstTestEccApi		

<b>Parent Container</b>	FlsTstConfigurationOfOptApiServices		
<b>Description</b>	Adds / removes the service FlsTst_TestEcc() from the code.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00095 :</b>		
<b>Name</b>	FlsTstVersionInfoApi		
<b>Parent Container</b>	FlsTstConfigurationOfOptApiServices		
<b>Description</b>	Adds / removes the service FlsTst_GetVersionInfo() from the code.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucBooleanParamDef		
<b>Default value</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

**No Included Containers**

### 10.2.5 FlsTstDemEventParameterRefs

<b>SWS Item</b>	<b>ECUC_FlsTst_00170 :</b>		
<b>Container Name</b>	FlsTstDemEventParameterRefs		
<b>Description</b>	Container for the references to DemEventParameter elements which shall be invoked using the API Dem_SetEventStatus in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId symbolic value. The standardized errors are provided in this container and can be extended by vendor-specific error references.		
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_FlsTst_00171 :</b>		
<b>Name</b>	FLSTST_E_FLSTST_FAILURE		
<b>Parent Container</b>	FlsTstDemEventParameterRefs		
<b>Description</b>	Reference to the DemEventParameter which shall be issued when the error "Flash Failure" has occurred.		
<b>Multiplicity</b>	0..1		
<b>Type</b>	Symbolic name reference to [ DemEventParameter ]		
<b>Post-Build Variant Multiplicity</b>	false		
<b>Post-Build Variant Value</b>	false		
<b>Multiplicity Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants

	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

**No Included Containers**

### 10.2.6 FlsTstBlockBgnd

<b>SWS Item</b>	<b>ECUC_FlsTst_00173 :</b>		
<b>Container Name</b>	FlsTstBlockBgnd		
<b>Description</b>	This container specifies configuration parameters for an individual background test block.		
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_FlsTst_00174 :</b>		
<b>Name</b>	FlsTstBgndBlockIndex		
<b>Parent Container</b>	FlsTstBlockBgnd		
<b>Description</b>	The scheduling for background test shall follow an order defined by this index. '0' means highest priority.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
<b>Range</b>	0 .. 4294967295		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00106 :</b>		
<b>Name</b>	FlsTstBlockBaseAddress		
<b>Parent Container</b>	FlsTstBlockBgnd		
<b>Description</b>	Start Address of the Flash block.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 18446744073709551615		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	--	
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00107 :</b>		
<b>Name</b>	FlsTstBlockSize		
<b>Parent Container</b>	FlsTstBlockBgnd		
<b>Description</b>	This parameter shall represent the Flash Test block size.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 .. 4294967295		

<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	--	
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00123 :</b>		
<b>Name</b>	FlsTstSignatureAddress		
<b>Parent Container</b>	FlsTstBlockBgnd		
<b>Description</b>	Address of the signature reference value of the Flash test block.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucIntegerParamDef		
<b>Range</b>	0 ..		
	18446744073709551615		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	--	
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00101 :</b>		
<b>Name</b>	FlsTstTestAlgorithm		
<b>Parent Container</b>	FlsTstBlockBgnd		
<b>Description</b>	This is the configuration of the test algorithm. The availability of algorithm is implementation specific.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucEnumerationParamDef		
<b>Range</b>	FLSTST_16BIT_CRC		--
	FLSTST_32BIT_CRC		--
	FLSTST_8BIT_CRC		--
	FLSTST_CHECKSUM		--
	FLSTST_DUPLICATED_MEMORY		--
	FLSTST_ECC		--
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	--	
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

**No Included Containers**

### 10.2.7 FlsTstBlockFgnd

<b>SWS Item</b>	<b>ECUC_FlsTst_00105 :</b>		
<b>Container Name</b>	FlsTstBlockFgnd		
<b>Description</b>	This container specifies configuration parameters for an individual foreground test block.		
<b>Configuration Parameters</b>			

<b>SWS Item</b>	<b>ECUC_FlsTst_00106 :</b>		
<b>Name</b>	FlsTstBlockBaseAddress		
<b>Parent Container</b>	FlsTstBlockFgnd		
<b>Description</b>	Start Address of the Flash block.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcuIntegerParamDef		
<b>Range</b>	0 .. 18446744073709551615		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	--	
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00107 :</b>		
<b>Name</b>	FlsTstBlockSize		
<b>Parent Container</b>	FlsTstBlockFgnd		
<b>Description</b>	This parameter shall represent the Flash Test block size.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcuIntegerParamDef		
<b>Range</b>	0 .. 4294967295		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	--	
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00151 :</b>		
<b>Name</b>	FlsTstFgndBlockIndex		
<b>Parent Container</b>	FlsTstBlockFgnd		
<b>Description</b>	Index identifies block to be tested by FlsTst_StartFgnd();		
<b>Multiplicity</b>	1		
<b>Type</b>	EcuIntegerParamDef (Symbolic Name generated for this parameter)		
<b>Range</b>	0 .. 4294967295		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	false		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	All Variants
	<b>Link time</b>	--	
	<b>Post-build time</b>	--	
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00123 :</b>		
<b>Name</b>	FlsTstSignatureAddress		
<b>Parent Container</b>	FlsTstBlockFgnd		
<b>Description</b>	Address of the signature reference value of the Flash test block.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcuIntegerParamDef		
<b>Range</b>	0 .. 18446744073709551615		
<b>Default value</b>	--		
<b>Post-Build Variant Value</b>	true		

<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	--	
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

<b>SWS Item</b>	<b>ECUC_FlsTst_00101 :</b>		
<b>Name</b>	FlsTstTestAlgorithm		
<b>Parent Container</b>	FlsTstBlockFgnd		
<b>Description</b>	This is the configuration of the test algorithm. The availability of algorithm is implementation specific.		
<b>Multiplicity</b>	1		
<b>Type</b>	EcucEnumerationParamDef		
<b>Range</b>	FLSTST_16BIT_CRC	--	
	FLSTST_32BIT_CRC	--	
	FLSTST_8BIT_CRC	--	
	FLSTST_CHECKSUM	--	
	FLSTST_DUPLICATED_MEMORY	--	
	FLSTST_ECC	--	
<b>Post-Build Variant Value</b>	true		
<b>Value Configuration Class</b>	<b>Pre-compile time</b>	X	VARIANT-PRE-COMPILE
	<b>Link time</b>	--	
	<b>Post-build time</b>	X	VARIANT-POST-BUILD
<b>Scope / Dependency</b>	scope: local		

**No Included Containers**

## Published Information

Note: The content of this chapter is specified in SWS\_BSWGeneral document [9].

## 11 Not applicable requirements

**[SWS\_FlsTst\_00166]** [These requirements are not applicable to this specification.]

(SRS\_BSW\_00344, SRS\_BSW\_00159, SRS\_BSW\_00167, SRS\_BSW\_00170, SRS\_BSW\_00419, SRS\_BSW\_00398, SRS\_BSW\_00375, SRS\_BSW\_00416, SRS\_BSW\_00168, SRS\_BSW\_00423, SRS\_BSW\_00424, SRS\_BSW\_00425, SRS\_BSW\_00426, SRS\_BSW\_00427, SRS\_BSW\_00428, SRS\_BSW\_00429, SRS\_BSW\_00432, SRS\_BSW\_00433, SRS\_BSW\_00422, SRS\_BSW\_00417, SRS\_BSW\_00161, SRS\_BSW\_00162, SRS\_BSW\_00005, SRS\_BSW\_00415, SRS\_BSW\_00164, SRS\_BSW\_00325, SRS\_BSW\_00342, SRS\_BSW\_00343, SRS\_BSW\_00007, SRS\_BSW\_00300, SRS\_BSW\_00413, SRS\_BSW\_00347, SRS\_BSW\_00305, SRS\_BSW\_00307, SRS\_BSW\_00310, SRS\_BSW\_00373, SRS\_BSW\_00327, SRS\_BSW\_00335, SRS\_BSW\_00350, SRS\_BSW\_00408, SRS\_BSW\_00410, SRS\_BSW\_00348, SRS\_BSW\_00353, SRS\_BSW\_00361, SRS\_BSW\_00301, SRS\_BSW\_00302, SRS\_BSW\_00328, SRS\_BSW\_00312, SRS\_BSW\_00006, SRS\_BSW\_00378, SRS\_BSW\_00306, SRS\_BSW\_00308, SRS\_BSW\_00309, SRS\_BSW\_00371, SRS\_BSW\_00358, SRS\_BSW\_00414, SRS\_BSW\_00330, SRS\_BSW\_00331, SRS\_BSW\_00009, SRS\_BSW\_00401, SRS\_BSW\_00172, SRS\_BSW\_00010, SRS\_BSW\_00003, SRS\_BSW\_00341, SRS\_BSW\_00334, SRS\_BSW\_00437, SRS\_BSW\_00439, SRS\_BSW\_00440, SRS\_SPAL\_12267, SRS\_SPAL\_12461, SRS\_SPAL\_12462, SRS\_SPAL\_12463, SRS\_SPAL\_12068, SRS\_SPAL\_12069, SRS\_SPAL\_12169, SRS\_SPAL\_12075, SRS\_SPAL\_12129, SRS\_SPAL\_12064, SRS\_SPAL\_12067, SRS\_SPAL\_12077, SRS\_SPAL\_12078, SRS\_SPAL\_12092, SRS\_SPAL\_12265, SRS\_FlsTst\_14221)