Document Title	Specification of Flash Test
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	261
Document Classification	Standard

Document Version	1.2.0
Document Status	Final
Part of Release	4.0
Revision	3

Document Change History

Date	Version	Changed by	Change Description
05.10.2011	1.2.0	AUTOSAR Administration	• FlsTst026: minor text change • Figure1: IRQ files removed • FlsTst052: parameter range modified • FlsTst053: minor text correction

22.11.2010	1.1.0	AUTOSAR Administration	• FlsTst_BlockIdFgndType: type change to uint8-32
			• Limit range of the following parameters to max. value "0xFFFFFFFF"
			  ◦ FlsTstBlockNumberBgnd:
			  ◦ FlsTstBlockNumberFgnd:
			  ◦ FlsTstBlockIndex:
			  ◦ FlsTstBlockSize:
			  ◦ FlsTstNumberOfTestedCells:
			  ◦ FlsTstNumberOfTestedCellsAtomic:
			  ◦ FlsTstTestIntervalIdEndValue:
			• FlsTst015 removed
			• FlsTst119_Conf: configuration for each block
			• FlsTst158_Conf: multiplicity changed to „1“.
			• FlsTstDemEventParameterRefs table included

07.12.2009	1.0.0	AUTOSAR Administration	Initial release

Disclaimer

Advice for users

AUTOSAR Specification Documents may contain exemplary items (exemplary reference models, "use cases", and/or references to exemplary technical solutions, devices, processes or software).

Any such exemplary items are contained in the Specification Documents for illustration purposes only, and they themselves are not part of the AUTOSAR Standard. Neither their presence in such Specification Documents, nor any later documentation of AUTOSAR conformance of products actually implementing such exemplary items, imply that intellectual property rights covering such exemplary items are licensed under the same rules as applicable to the AUTOSAR Standard.

Table of Contents

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.

Acronym:	*Description:*
BSW	BasicSoftWare
PC	PreCompile
PB	PostBuild

Abbreviation:	*Description:*
DEM	Diagnostic Event Manager.
DET	Development 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.

Term:	*Description:*
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.
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 Development Error Tracer
AUTOSAR_SWS_DevelopmentErrorTracer.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

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:

- DET: Development Error Tracer: DET services will be called in case of Development errors.
- Production Errors will be reported to Diagnostic Event Manager (DEM)
- BSW scheduler is required to trigger main function in background mode

5.1 File structure

5.1.1 Code file structure

[FlsTst002] ⌈The code file structure for the Flash Test module shall not be defined within this specification. ⌋(BSW00380, BSW00346, BSW158, BSW00314, BSW00370)

5.1.2 Header file structure

[FlsTst003] ⌈The include structure for the source code of the Flash Test module shall be as follows:

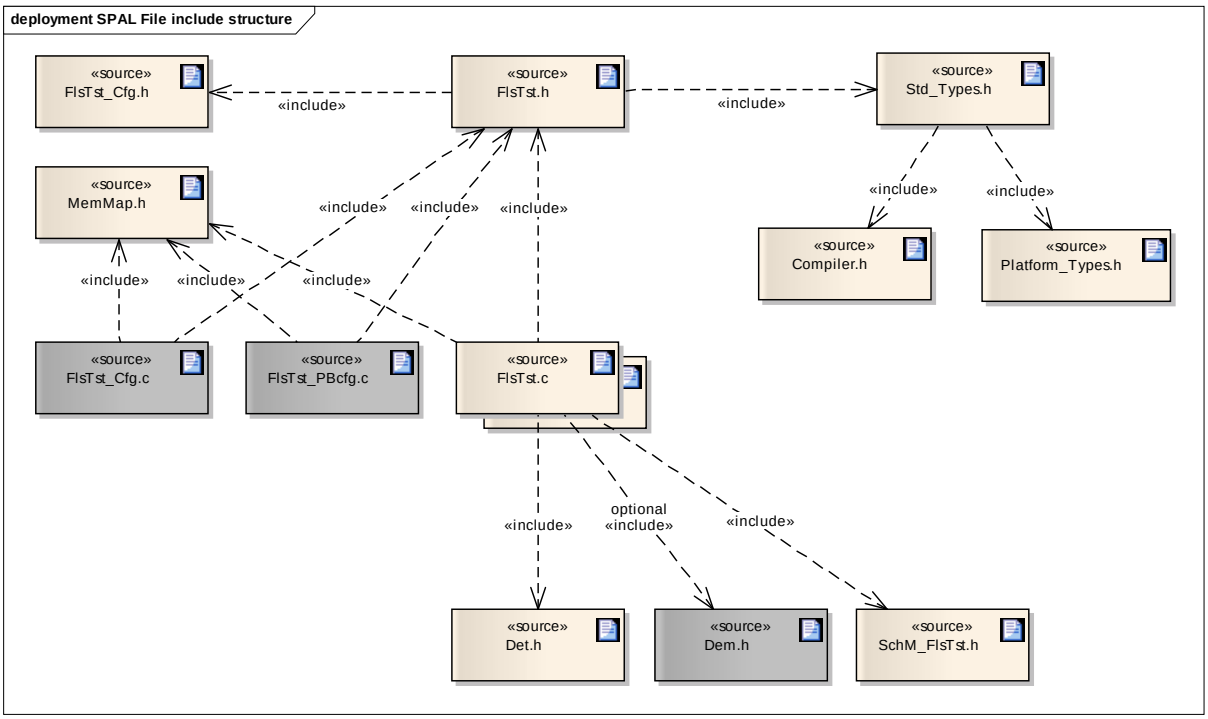

Figure 1: Header file structure

⌋(BSW00380, BSW00381, BSW00412, BSW00383, BSW00435, BSW00436)

[FlsTst004] ⌈The module shall include the `Dem.h` file. By this inclusion, the APIs to report errors as well as the required Event Id symbols are included. This specification

defines the name of the Event Id symbols which are provided by XML to the DEM configuration tool. The DEM configuration tool assigns ECU dependent values to the Event Id symbols and publishes the symbols in `Dem_IntErrId.h`. ⌋()

6 Requirements traceability

Requirement	Satisfied by
-	FlsTst145
-	FlsTst049
-	FlsTst029
-	FlsTst013
-	FlsTst070
-	FlsTst052
-	FlsTst067
-	FlsTst156
-	FlsTst161
-	FlsTst147
-	FlsTst148
-	FlsTst008
-	FlsTst026
-	FlsTst144
-	FlsTst047
-	FlsTst068
-	FlsTst159
-	FlsTst138
-	FlsTst076
-	FlsTst108
-	FlsTst121
-	FlsTst012
-	FlsTst074
-	FlsTst010
-	FlsTst045
-	FlsTst142
-	FlsTst004
-	FlsTst075
-	FlsTst146
-	FlsTst162
-	FlsTst006
-	FlsTst117
-	FlsTst053
-	FlsTst164
-	FlsTst014
-	FlsTst140
BSW003	FlsTst166
BSW00300	FlsTst166

BSW00301	FlsTst166
BSW00302	FlsTst166
BSW00304	FlsTst016
BSW00305	FlsTst166
BSW00306	FlsTst166
BSW00307	FlsTst166
BSW00308	FlsTst166
BSW00309	FlsTst166
BSW00310	FlsTst166
BSW00312	FlsTst166
BSW00314	FlsTst002
BSW00323	FlsTst023, FlsTst133, FlsTst033
BSW00325	FlsTst166
BSW00326	FlsTst166
BSW00327	FlsTst166
BSW00328	FlsTst166
BSW00330	FlsTst166
BSW00331	FlsTst166
BSW00334	FlsTst166
BSW00335	FlsTst166
BSW00336	FlsTst027
BSW00337	FlsTst007
BSW00338	FlsTst009
BSW00339	FlsTst042, FlsTst060, FlsTst112, FlsTst009
BSW00341	FlsTst166
BSW00342	FlsTst166
BSW00343	FlsTst166
BSW00344	FlsTst166
BSW00346	FlsTst002
BSW00347	FlsTst166
BSW00348	FlsTst166
BSW00350	FlsTst166
BSW00353	FlsTst166
BSW00355	FlsTst100, FlsTst109
BSW00357	FlsTst063
BSW00358	FlsTst166
BSW00361	FlsTst166
BSW00369	FlsTst009
BSW00370	FlsTst002
BSW00371	FlsTst166
BSW00373	FlsTst166

BSW00375	FlsTst166
BSW00376	FlsTst066
BSW00377	FlsTst048
BSW00378	FlsTst166
BSW00380	FlsTst003, FlsTst002
BSW00381	FlsTst003
BSW00383	FlsTst003
BSW00385	FlsTst007
BSW00386	FlsTst093, FlsTst091, FlsTst023, FlsTst025, FlsTst133, FlsTst033, FlsTst046, FlsTst056, FlsTst059, FlsTst062, FlsTst065, FlsTst114, FlsTst089
BSW00398	FlsTst166
BSW004	FlsTst134
BSW00401	FlsTst166
BSW00405	FlsTst019, FlsTst018
BSW00406	FlsTst011
BSW00407	FlsTst044
BSW00408	FlsTst166
BSW00409	FlsTst007
BSW00410	FlsTst166
BSW00411	FlsTst044
BSW00412	FlsTst003
BSW00413	FlsTst166
BSW00414	FlsTst166
BSW00415	FlsTst166
BSW00416	FlsTst166
BSW00417	FlsTst166
BSW00419	FlsTst166
BSW00421	FlsTst069, FlsTst009
BSW00422	FlsTst166
BSW00423	FlsTst166
BSW00424	FlsTst166
BSW00425	FlsTst166
BSW00426	FlsTst166
BSW00427	FlsTst166
BSW00428	FlsTst166
BSW00429	FlsTst166
BSW00431	FlsTst166
BSW00432	FlsTst166
BSW00433	FlsTst166
BSW00434	FlsTst166
BSW00435	FlsTst003

BSW00436	FlsTst003
BSW00437	FlsTst166
BSW00438	FlsTst018
BSW00439	FlsTst166
BSW00440	FlsTst166
BSW005	FlsTst166
BSW006	FlsTst166
BSW007	FlsTst166
BSW009	FlsTst166
BSW010	FlsTst166
BSW101	FlsTst017
BSW12057	FlsTst017, FlsTst020
BSW12064	FlsTst166
BSW12067	FlsTst166
BSW12068	FlsTst166
BSW12069	FlsTst166
BSW12075	FlsTst166
BSW12077	FlsTst166
BSW12078	FlsTst166
BSW12092	FlsTst166
BSW12125	FlsTst022
BSW12129	FlsTst166
BSW12163	FlsTst027, FlsTst028
BSW12169	FlsTst166
BSW12265	FlsTst166
BSW12267	FlsTst166
BSW12448	FlsTst023, FlsTst025, FlsTst133, FlsTst039, FlsTst033
BSW12461	FlsTst166
BSW12462	FlsTst166
BSW12463	FlsTst166
BSW14208	FlsTst066, FlsTst071
BSW14209	FlsTst139, FlsTst066, FlsTst071
BSW14211	FlsTst091, FlsTst040, FlsTst041
BSW14212	FlsTst078, FlsTst077
BSW14213	FlsTst058, FlsTst057, FlsTst056, FlsTst055, FlsTst059, FlsTst054, FlsTst116, FlsTst115
BSW14214	FlsTst093, FlsTst042, FlsTst043, FlsTst113, FlsTst112, FlsTst114
BSW14215	FlsTst037, FlsTst036, FlsTst034, FlsTst088
BSW14216	FlsTst039, FlsTst038, FlsTst035, FlsTst089
BSW14217	FlsTst031, FlsTst032, FlsTst030
BSW14219	FlsTst149, FlsTst143, FlsTst137, FlsTst033, FlsTst050, FlsTst051

BSW14221	FlsTst166
BSW14223	FlsTst061, FlsTst060, FlsTst062
BSW14224	FlsTst063, FlsTst065, FlsTst064
BSW14225	FlsTst153, FlsTst154, FlsTst155
BSW157	FlsTst040, FlsTst042, FlsTst057, FlsTst054, FlsTst060, FlsTst077, FlsTst073, FlsTst072, FlsTst112
BSW158	FlsTst002
BSW159	FlsTst166
BSW161	FlsTst166
BSW162	FlsTst166
BSW164	FlsTst166
BSW167	FlsTst166
BSW168	FlsTst166
BSW170	FlsTst166
BSW172	FlsTst166

7 Functional specification

7.1 General behavior

[FlsTst137] ⌈The Flash test module provides test execution services in background and foreground mode (see also chapter 2). ⌋(BSW14219)

[FlsTst138] ⌈The memory blocks to be tested shall be configurable for background and foreground mode separately (see FlsTst103, FlsTst104). ⌋( )

[FlsTst139] ⌈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 FlsTst066). ⌋(BSW14209)

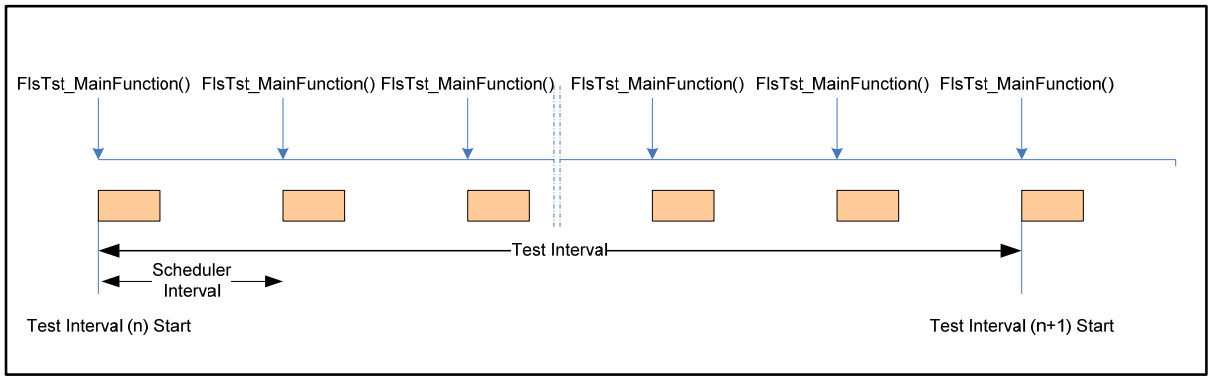

Figure 2: Background Test: Test Interval

[FlsTst140] ⌈The length of a partial test is defined by the number of tested cells, which shall be tested in one scheduled task. (see FlsTst119). 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).

[FlsTst142] ⌈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 FlsTst120). ⌋( )

[FlsTst156] ⌈Each Flash test Interval shall have an Identifier, which shall be incremented by each start of a new test interval in background mode. ⌋( )

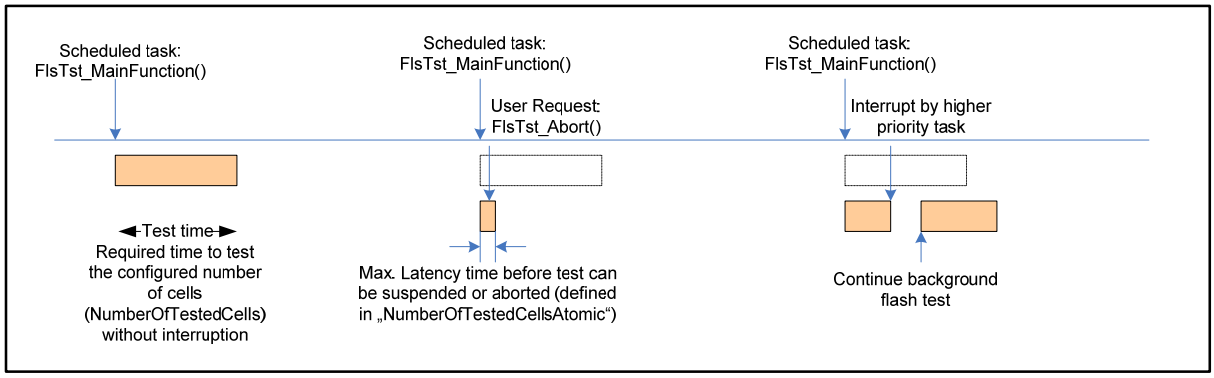

Figure 3: Background Test: Test Process

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.

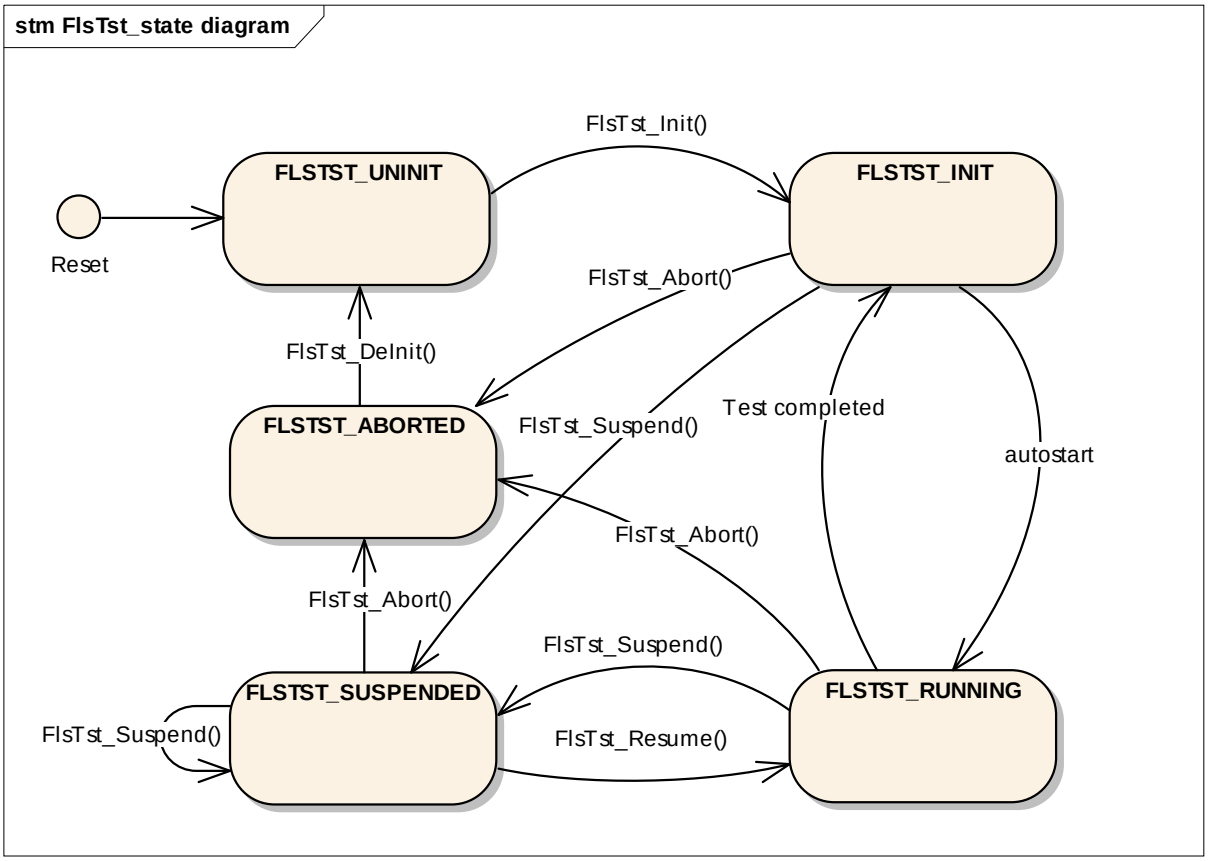

Figure 4: State Diagram – Background mode

[FlsTst143] ⌈Foreground tests are defined as synchronous tests which shall not be interrupted. The execution of Foreground tests is configurable (see FlsTst086) and can be called after module initialization at any time. ⌋(BSW14219)

7.2 Error classification

[FlsTst006] ⌈Development error values are of type uint8. ⌋ ( )

[FlsTst007] ⌈The following errors and exceptions shall be detectable by the Flash Test depending on its build version (development/production mode):

Type of error	*Relevance*	*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	Development	`FLSTST_E_PARAM_CONFIG`	`0x05`
Pointer is a NULL pointer	Development	`FLSTST_E_PARAM_POINTER`	`0x06`
Flash Failure	Production	`FLSTST_E_FLSTST_FAILURE`	`Assigned externally`

To get more details concerning error detection, refer to chapter 8 API specification. ⌋ (BSW00337, BSW00409, BSW00385)

7.3 Error Detection

[FlsTst008] ⌈The detection of development errors is configurable (*ON* / *OFF*) at pre-compile time. The switch `FlsTstDevErrorDetect` (see chapter 10) shall activate or deactivate the detection of all development errors. ⌋ ( )

[FlsTst009] ⌈If the `FlsTstDevErrorDetect` switch is enabled, API parameter checking is enabled. The detailed description of the detected errors can be found in chapter Error classification and chapter API specification. ⌋ (BSW00338, BSW00369, BSW00339, BSW00421)

[FlsTst010] ⌈The detection of production code errors cannot be switched off. ⌋ ( )

[FlsTst011] ⌈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 Development Error Tracer (if development error detection is enabled). ⌋ (BSW00406)

7.4 Error Notification

[FlsTst013] ⌈Production errors shall be reported to Diagnostic Event Manager (DEM) via the `Dem_ReportErrorStatus` API. ⌋()

[FlsTst014] ⌈Detected development errors shall be reported to the `Det_ReportError` service of the Development Error Tracer (DET) if the pre-processor switch `FlsTstDevErrorDetect` is set (see chapter 10). ⌋()

[FlsTst012] ⌈Additional errors that are detected because of specific implementation and/or specific hardware properties shall be added in the Flash device specific implementation specification. The classification and enumeration shall be compatible to the errors listed above in FlsTst007. ⌋()

7.5 Version check

[FlsTst134] ⌈The Flash test module shall perform Inter Module Checks to avoid integration of incompatible files. Preprocessing directives shall check the imported included files.

The following version numbers shall be verified:
- <MODULENAME>_AR_RELEASE_MAJOR_VERSION
- <MODULENAME>_AR_RELEASE_MINOR_VERSION

Where <MODULENAME> is the module short name of the other (external) modules which provide header files included by the Flash test module.

If the values are not identical to the expected values, an error shall be reported. ⌋(BSW004)

7.6 Debugging Support

The following requirements deal with the definition of variables and the description of debug information.

[FlsTst144] ⌈Each variable that shall be accessible by AUTOSAR Debugging, shall be defined as global variable. ⌋()

[FlsTst145] ⌈All type definitions of variables which shall be debugged, shall be accessible by the header file FlsTst.h. ⌋()

[FlsTst146] ⌈The declaration of variables in the header file shall allow to calculate the size of the variables by C-"sizeof".⌋ ( )

[FlsTst147] ⌈Variables available for debugging shall be described in the respective Basic Software Module Description⌋ ( )

[FlsTst148] ⌈The state described in FlsTst048 shall be available for debugging. ⌋ ( )

8 API specification

8.1 Imported types

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

[FlsTst016] ⌈

Module	*Imported Type*
Dem	Dem_EventIdType
	Dem_EventStatusType
Std_Types	Std_ReturnType
	Std_VersionInfoType

⌋(BSW00304)

8.2 Type definitions

8.2.1 FlsTst_ConfigType

[FlsTst018] ⌈

Name:	`FlsTst_ConfigType`	
Type:	`Structure`	
Range:	`implementation specific`	implementation specific
Description:	This type of external data structure shall contain the initialization data for the Flash Test.	

⌋(BSW00405, BSW00438)

[FlsTst019] ⌈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⌋(BSW00405)

8.2.2 FlsTst_StateType

[**FlsTst048**] ⌈

Name:	`FlsTst_StateType`	
Type:	`Enumeration`	
Range:	`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
Description:	This is a state value returned by the API service FlsTst_GetCurrentState().	

⌋ (BSW00377)

[FlsTst049] ⌈For the type `FlsTst_StateType`, the enumeration value `FLSTST_UNINIT` shall be the default value after a reset. This enumeration value shall have the numeric value 0. ⌋ ( )

8.2.3 FlsTst_TestResultFgndType

[FlsTst052] ⌈

Name:	`FlsTst_TestResultFgndType`	
Type:	`Enumeration`	
Range:	`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.
Description:	Return type of API service FlsTst_GetResultFgnd().	

⌋ ( )

[FlsTst053] ⌈For the type `FlsTst_TestResultFgndType`, the enumeration value `FLSTST_NOT_TESTED` shall be the default value after a reset. This enumeration value shall have the numeric value 0. ⌋ ( )

8.2.4 FlsTst_TestResultBgndType

[FlsTst153] ⌈

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

⌋ (BSW14225)

[FlsTst154] ⌈For the type `FlsTst_TestResultBgndType`, the enumeration value `FLSTST_RESULT_NOT_TESTED` shall be the default value after a reset. This enumeration value shall have the numeric value 0. ⌋ (BSW14225)

8.2.5 FlsTst_BlockIdFgndType

[FlsTst100] ⌈

Name:	`FlsTst_BlockIdFgndType`		
Type:	`uint8, uint16, uint32`		
Range:	`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.
Description:	This type specifies the identification (ID) for a Flash block to be tested in foreground mode, which is configured in the configuration structure.		

⌋ (BSW00355)

8.2.6 FlsTst_ErrorDetailsType

[FlsTst108] ⌈

Name:	`FlsTst_ErrorDetailsType`	
Type:	`Structure`	
Range:	`implementation specific`	implementation specific
Description:	This type shall specify implementation specific error information monitored in the Flash test module.	

⌋ ( )

8.2.7 FlsTst_TestSignatureFgndType

[FlsTst109] ⌈

Name:	`FlsTst_TestSignatureFgndType`	
Type:	`Structure`	
Range:	`implementation specific`	Implementation specific type
Description:	Type for test signature in foreground mode	

⌋ (BSW00355)

8.2.8 FlsTst_TestSignatureBgndType

[FlsTst155] ⌈

Name:	`FlsTst_TestSignatureBgndType`		
Type:	`Structure`		
Element:	`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.
Description:	Type for test signature in background mode.		

⌋ (BSW14225)

8.2.9 FlsTst_TestResultType

[FlsTst164] ⌈

Name:	`FlsTst_TestResultType`	
Type:	`Enumeration`	
Range:	`FLSTST_RESULT_NOT_TESTED`	There is no test result available.
	`FLSTST_RESULT_OK`	The last Flash Test interval has been tested with OK result
	`FLSTST_RESULT_NOT_OK`	The last Flash Test interval has been tested with NOT-OK result.
Description:	--	

⌋ ( )

8.3 Function definitions

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

8.3.1 FlsTst_Init

[FlsTst017] ⌈

Service name:	FlsTst_Init	

Syntax:	`void FlsTst_Init(`
	`  const FlsTst_ConfigType* ConfigPtr`
	`)`	

Service ID[hex]:	0x00	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	ConfigPtr	Pointer to configuration set
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Service for Flash Test initialization.	

⌋ (BSW101, BSW12057)

[FlsTst020] ⌈The function `FlsTst_Init` shall initialize all Flash Test relevant registers and global variables and change the execution state to `FLSTST_INIT`. ⌋ (BSW12057)

[FlsTst022] ⌈The function `FlsTst_Init` shall only initialize the configured resources and shall not touch resources that are not configured in the configuration file. ⌋ (BSW12125)

[FlsTst023] ⌈If development error detection is enabled for the Flash Test module, the function `FlsTst_Init` shall raise development error `FLSTST_E_PARAM_CONFIG` if `ConfigPtr` is a null pointer. This is applicable for Variant PB only (see also

FlsTst026). The function shall be left without any action. ⌋ (BSW00323, BSW00386, BSW12448)

[FlsTst025] ⌈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. ⌋ (BSW00386, BSW12448)

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.

[FlsTst026] ⌈For Variant PC a NULL pointer shall be passed to the initialization routine. In this case the check for this NULL pointer has to be omitted. ⌋ ( )

8.3.2 FlsTst_DeInit

[FlsTst027] ⌈

Service name:	FlsTst_DeInit

Syntax:	`void FlsTst_DeInit(`
	`  void`
	`)`

Service ID[hex]:	0x01
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None
Parameters (inout):	None
Parameters (out):	None
Return value:	None
Description:	Service for Flash Test De-Initialization.

⌋ (BSW00336, BSW12163)

[FlsTst028] ⌈The function `FlsTst_DeInit` shall de-initialize all Flash Test relevant registers and global variables that were initialized by `FlsTst_Init.` ⌋ (BSW12163)

[FlsTst029] ⌈The function `FlsTst_DeInit` shall set the Flash Test module state to `FLSTST_UNINIT`. ⌋ ( )

8.3.3 FlsTst_StartFgnd

[FlsTst149] ⌈

Service name:	FlsTst_StartFgnd

Syntax:	`Std_ReturnType FlsTst_StartFgnd(`
	`  FlsTst_BlockIdFgndType FgndBlockId`
	`)`

Service ID[hex]:	0x02
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant

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

⌋ (BSW14219)

[FlsTst050] ⌈The function `FlsTst_StartFgnd` is only applicable for Foreground mode Flash Test operation. ⌋ (BSW14219)

[FlsTst051] ⌈The function `FlsTst_StartFgnd` shall be pre compile time configurable On/Off by the configuration parameter: `FlsTst_StartFgndApi`. ⌋ (BSW14219)

[FlsTst033] ⌈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`. ⌋ (BSW00323, BSW00386, BSW12448, BSW14219)

8.3.4 FlsTst_Abort

[FlsTst030] ⌈

Service name:	FlsTst_Abort

Syntax:	`void FlsTst_Abort(`
	`  void`
	`)`

Service ID[hex]:	0x03
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None
Parameters (inout):	None
Parameters (out):	None
Return value:	None
Description:	Service for aborting the Flash Test.

⌋ (BSW14217)

[FlsTst031] ⌈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. ⌋ (BSW14217)

[FlsTst032] ⌈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. ⌋ (BSW14217)

8.3.5 FlsTst_Suspend

[FlsTst034] ⌈

Service name:	FlsTst_Suspend

Syntax:	`void FlsTst_Suspend(`
	`  void`
	`)`

Service ID[hex]:	0x04
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None
Parameters (inout):	None
Parameters (out):	None
Return value:	None
Description:	Service for suspending current operation of the Flash Test, until FlsTst_Resume is called.

⌋ (BSW14215)

[FlsTst036] ⌈The function `FlsTst_Suspend` is only applicable for Background mode Flash Test operation. ⌋ (BSW14215)

[FlsTst037] ⌈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.` ⌋ (BSW14215)

[FlsTst088] ⌈The function `FlsTst_Suspend` shall be pre compile time configurable On/Off by the configuration parameter: `FlsTst_SuspendResumeApi.` ⌋ (BSW14215)

8.3.6 FlsTst_Resume

[FlsTst035] ⌈

Service name:	FlsTst_Resume

Syntax:	`void FlsTst_Resume(`
	`  void`
	`)`

Service ID[hex]:	0x05
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None
Parameters (inout):	None
Parameters (out):	None
Return value:	None
Description:	Service for continuing the Flash Test at the point it was suspended.

⌋ (BSW14216)

[FlsTst038] ⌈The function `FlsTst_Resume` shall change the execution state to `FLSTST_RUNNING` when commanded to continue and the current execution state is `FLSTST_SUSPENDED.` ⌋ (BSW14216)

[FlsTst039] ⌈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.⌋(BSW12448, BSW14216)

[FlsTst162] ⌈The function `FlsTst_Resume` is only applicable for Background mode Flash Test operation.⌋( )

[FlsTst089] ⌈The function `FlsTst_Resume` shall be pre compile time configurable On/Off by the configuration parameter: `FlsTst_SuspendResumeApi`.⌋(BSW00386, BSW14216)

8.3.7 FlsTst_GetCurrentState

[FlsTst040] ⌈

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

⌋(BSW157, BSW14211)

[FlsTst041] ⌈The function `FlsTst_GetCurrentState` shall return the current Flash Test execution state.⌋(BSW14211)

[FlsTst091] ⌈The function `FlsTst_GetCurrentState` shall be pre compile time configurable On/Off by the configuration parameter: `FlsTst_GetCurrentStateApi`.⌋(BSW00386, BSW14211)

8.3.8 FlsTst_GetTestResultBgnd

[FlsTst042] ⌈

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

」(BSW00339, BSW157, BSW14214)

[FlsTst043] 「The function `FlsTst_GetTestResultBgnd` shall return the Flash test result and Test Interval Id of the last background test. 」(BSW14214)

[FlsTst093] 「The function `FlsTst_GetTestResultBgnd` shall be pre compile time configurable On/Off by the configuration parameter: `FlsTst_GetTestResultBgndApi`. 」(BSW00386, BSW14214)

8.3.9 FlsTst_GetTestResultFgnd

[FlsTst112] 「

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

」(BSW00339, BSW157, BSW14214)

[FlsTst113] 「The function `FlsTst_GetTestResultFgnd` shall return the Flash test result of the last foreground test. 」(BSW14214)

[FlsTst114] 「The function `FlsTst_GetTestResultFgnd` shall be pre compile time configurable On/Off by the configuration parameter: `FlsTst_GetTestResultFgndApi`. 」(BSW00386, BSW14214)

8.3.10 FlsTst_GetVersionInfo

[FlsTst044] ⌈

Service name:	FlsTst_GetVersionInfo

Syntax:	`void FlsTst_GetVersionInfo(`
	`  Std_VersionInfoType* versioninfo`
	`)`

Service ID[hex]:	0x08
Sync/Async:	Synchronous
Reentrancy:	Reentrant
Parameters (in):	None
Parameters (inout):	None
Parameters (out):	versioninfo Pointer to where to store the version information of this module.
Return value:	None
Description:	Service returns the version information of this module.

⌋ (BSW00407, BSW00411)

[FlsTst045] ⌈The function `FlsTst_GetVersionInfo` shall return the version information of this module. The version information includes:

- Module Id
- Vendor Id
- Vendor specific version numbers (BSW00407). ⌋ ( )

[FlsTst046] ⌈The function `FlsTst_GetVersionInfo` shall be pre compile time configurable On/Off by the configuration parameter: `FlsTst_VersionInfoApi`. ⌋ (BSW00386)

[FlsTst047] ⌈If source code for caller and callee of `FlsTst_GetVersionInfo` is available, the Flash Test module should realize `FlsTst_GetVersionInfo` as a macro, defined in the module's header file. ⌋ ( )

[FlsTst133] ⌈If development error detection is enabled for the Flash Test module, the function `FlsTst_GetVersionInfo` shall raise development error `FLSTST_E_PARAM_POINTER` if parameter versioninfo is a null pointer. ⌋ (BSW00323, BSW00386, BSW12448)

8.3.11 FlsTst_GetTestSignatureBgnd

[FlsTst054] ⌈

Service name:	FlsTst_GetTestSignatureBgnd

Syntax:	`FlsTst_TestSignatureBgndType FlsTst_GetTestSignatureBgnd(`
	`  void`
	`)`

Service ID[hex]:	0x09
Sync/Async:	Synchronous
Reentrancy:	Reentrant
Parameters (in):	None
Parameters (inout):	None
Parameters (out):	None

Return value:	FlsTst_TestSignatureBgndType	See type definition
Description:	Service returns the Flash Test result in background mode.	

⌋(BSW14213, BSW157)

[FlsTst055] ⌈The function `FlsTst_GetTestSignatureBgnd` shall return the signature and Test Interval Id of the last background test. ⌋(BSW14213)

[FlsTst056] ⌈The function `FlsTst_GetTestSignatureBgnd` shall be pre compile time configurable On/Off by the configuration parameter: `FlsTst_GetTestSignatureBgndApi`. ⌋(BSW00386, BSW14213)

[FlsTst115] ⌈If no signature is available, the function `FlsTst_GetTestSignatureBgnd` shall return the default value “0x0”. ⌋(BSW14213)

8.3.12 FlsTst_GetTestSignatureFgnd

[FlsTst057] ⌈

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

⌋(BSW14213, BSW157)

[FlsTst058] ⌈The function `FlsTst_GetTestSignatureFgnd` shall return the signature of the last foreground test. ⌋(BSW14213)

[FlsTst059] ⌈The function `FlsTst_GetTestSignatureFgnd` shall be pre compile time configurable On/Off by the configuration parameter: `FlsTst_GetTestSignatureFgndApi`. ⌋(BSW00386, BSW14213)

[FlsTst116] ⌈If no signature is available, the function `FlsTst_GetTestSignatureFgnd` shall return the default value “0x0”. ⌋(BSW14213)

8.3.13 FlsTst_GetErrorDetails

[FlsTst060] ⌈

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

⌋ (BSW00339, BSW157, BSW14223)

[FlsTst061] ⌈The function `FlsTst_GetErrorDetails` shall return the error details monitored from the Flash Test driver. ⌋ (BSW14223)

[FlsTst062] ⌈The function `FlsTst_GetErrorDetails` shall be pre compile time configurable On/Off by the configuration parameter: `FlsTst_GetErrorDetailsApi`. ⌋ (BSW00386, BSW14223)

8.3.14 FlsTst_TestEcc

[FlsTst063] ⌈

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

⌋ (BSW00357, BSW14224)

[FlsTst064] ⌈The function `FlsTst_TestEcc` shall execute a test of the ECC circuitry. ⌋ (BSW14224)

[FlsTst065] ⌈The function `FlsTst_TestEcc` shall be pre compile time configurable On/Off by the configuration parameter: `FlsTst_TestEccApi`. ⌋ (BSW00386, BSW14224)

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

The Basic Software Scheduler calls these functions directly. The following functions shall have no return value and no parameter. All functions shall be non-reentrant.

Terms and definitions:

Fixed cyclic: Fixed cyclic means that one cycle time is defined at configuration and shall not be changed because functionality is requiring a fixed timing (e.g. filters).

Variable cyclic: Variable cyclic means that the cycle times are defined at configuration but might be mode dependent and therefore vary during runtime.

On pre-condition: On pre-condition means that no cycle time can be defined. The function is called when the conditions are fulfilled. Alternatively, the function may be called cyclically, however the cycle time is assigned dynamically during runtime by other modules.

8.5.1 FlsTst_MainFunction

[FlsTst066] ⌈

Service name:	FlsTst_MainFunction

Syntax:	`void FlsTst_MainFunction(`
	`  void`
	`)`

Service ID[hex]:	0x0d
Timing:	VARIABLE_CYCLIC_OR_ON_PRECONDITION
Description:	Service for executing the Flash Test in background mode.

⌋ (BSW00376, BSW14208, BSW14209)

[FlsTst067] ⌈The function `FlsTst_MainFunction` shall test the defined flash blocks in background mode, starting with the first flash block in the FlsTstConfigParams. ⌋ ()

[FlsTst068] ⌈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. ⌋ ()

[FlsTst069] ⌈When FlsTstTestResultSignature is true, the function `FlsTst_MainFunction` shall provide the test signatures of all blocks within a test interval. ⌋ (BSW00421)

[FlsTst161] ⌈When FlsTstTestResultSignature is disabled, the function `FlsTst_MainFunction` shall set the overall result status to `FLSTST_RESULT_OK` if all blocks are tested with result status OK. If at least one block test result is not ok, then the function shall set the overall test result status to `FLSTST_RESULT_NOT_OK` regardless whether all blocks are already tested or not and report the production error `FLSTST_E_FLSTST_FAILURE` to the DEM. ⌋ ( )

[FlsTst070] ⌈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. ⌋ ( )

[FlsTst071] ⌈The function `FlsTst_MainFunction` shall test a defined number of flash cells within one call. The defined number is specified by configuration (see FlsTst119). ⌋ (BSW14208, BSW14209)

[FlsTst117] ⌈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 FlsTst120). ⌋ ( )

[FlsTst121] ⌈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 which are required to fulfill the core functionality of the module.

[FlsTst072] ⌈

API function	*Description*
Dem_ReportErrorStatus	Queues the reported events from the BSW modules (API is only used by BSW modules). The interface has an asynchronous behavior, because the processing of the event is done within the Dem main function.

⌋(BSW157)

8.6.2 Optional Interfaces

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

[FlsTst073] ⌈

API function	*Description*
Det_ReportError	Service to report development errors.

⌋(BSW157)

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.

[FlsTst074] ⌈The callback notifications shall be configurable as function pointers within the initialization data structure (FlsTst_ConfigType). ⌋( )

[FlsTst075] ⌈The callback notifications shall have no parameters and no return value. ⌋( )

[FlsTst076] ⌈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

[FlsTst077] ⌈

Service name:	FlsTst_TestCompletedNotification

Syntax:	`void FlsTst_TestCompletedNotification(`
	`  void`
	`)`

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

⌋(BSW157, BSW14212)

[FlsTst078] ⌈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. ⌋(BSW14212)

[FlsTst159] ⌈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

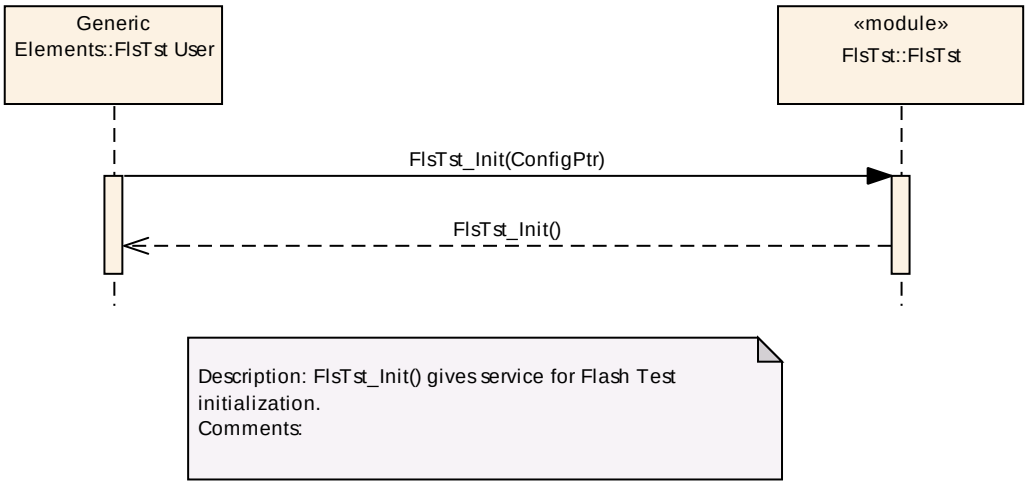

Figure 5: Flash test driver initialization

9.2 De-initialization

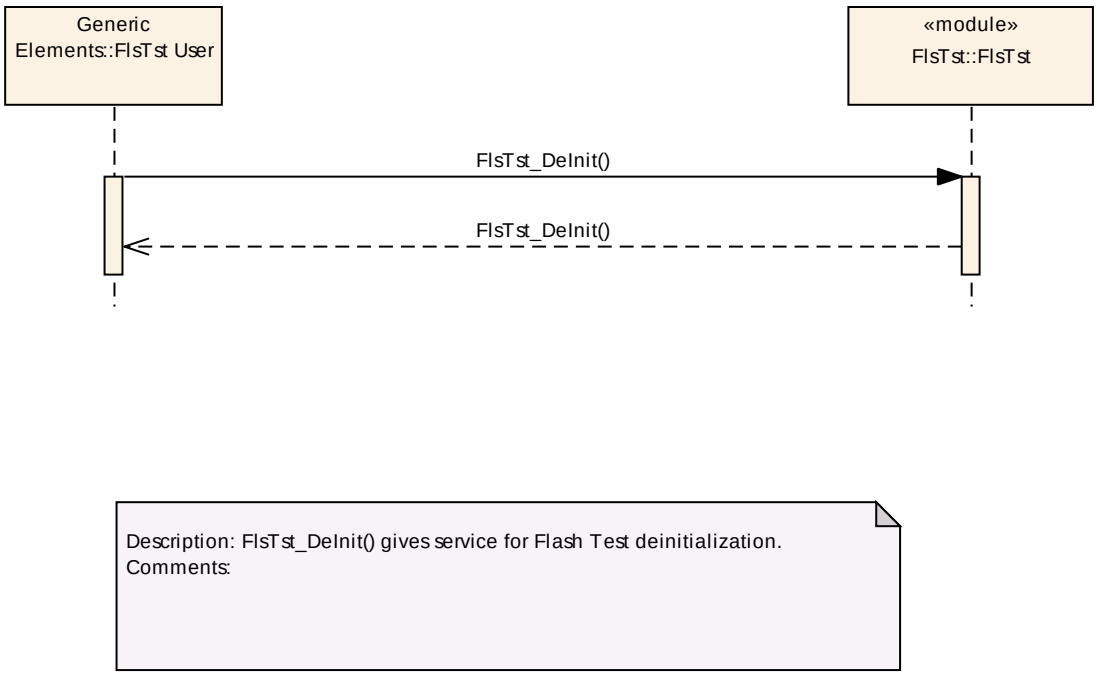

Figure 6: Flash test driver de-initialization

9.3 Background Test

9.3.1 Test Result Calculation within Flash test driver

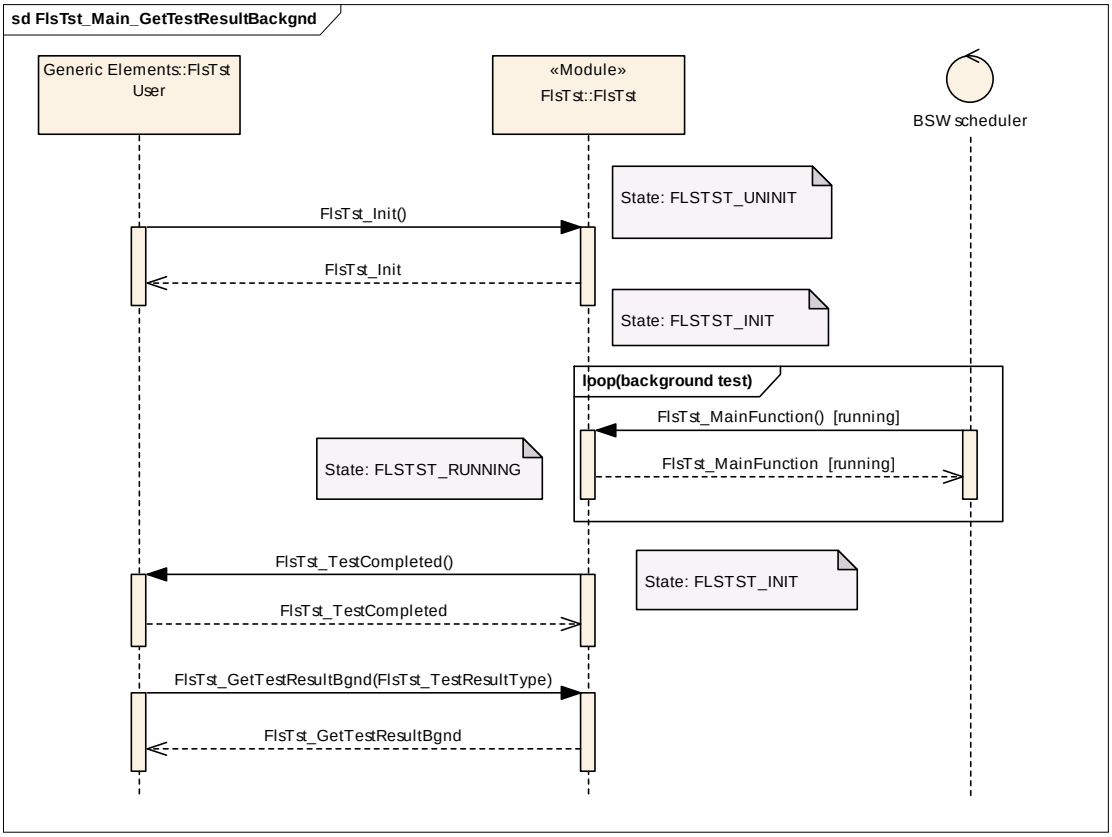

Figure 7: Background Test – Test result calculation in Flash test driver

9.3.2 Test signature provided to caller

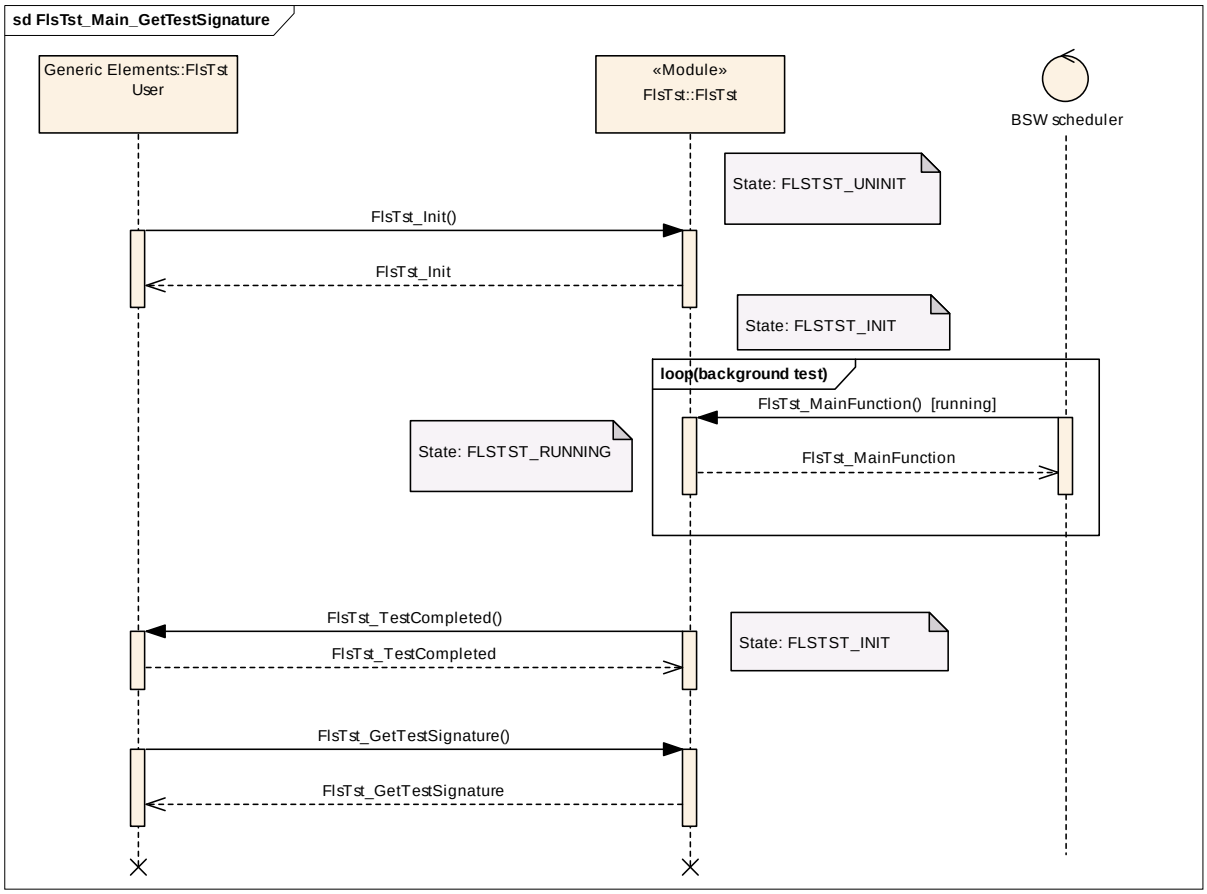

Figure 8: Background Test – Test signature provided to caller

9.4 Suspend and Resume Background Testing

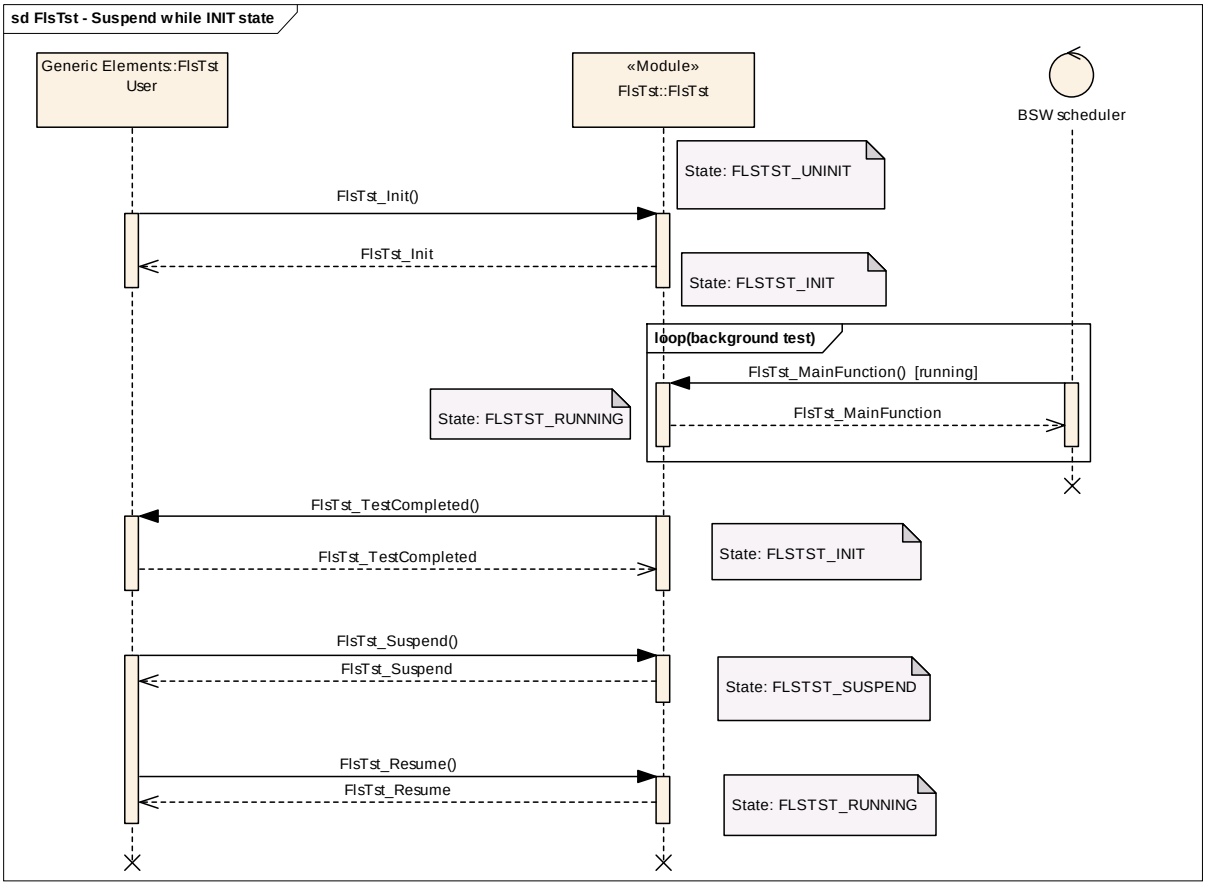

Figure 9: Suspend and Resume Background Testing

9.5 Foreground Task interrupts Background Task

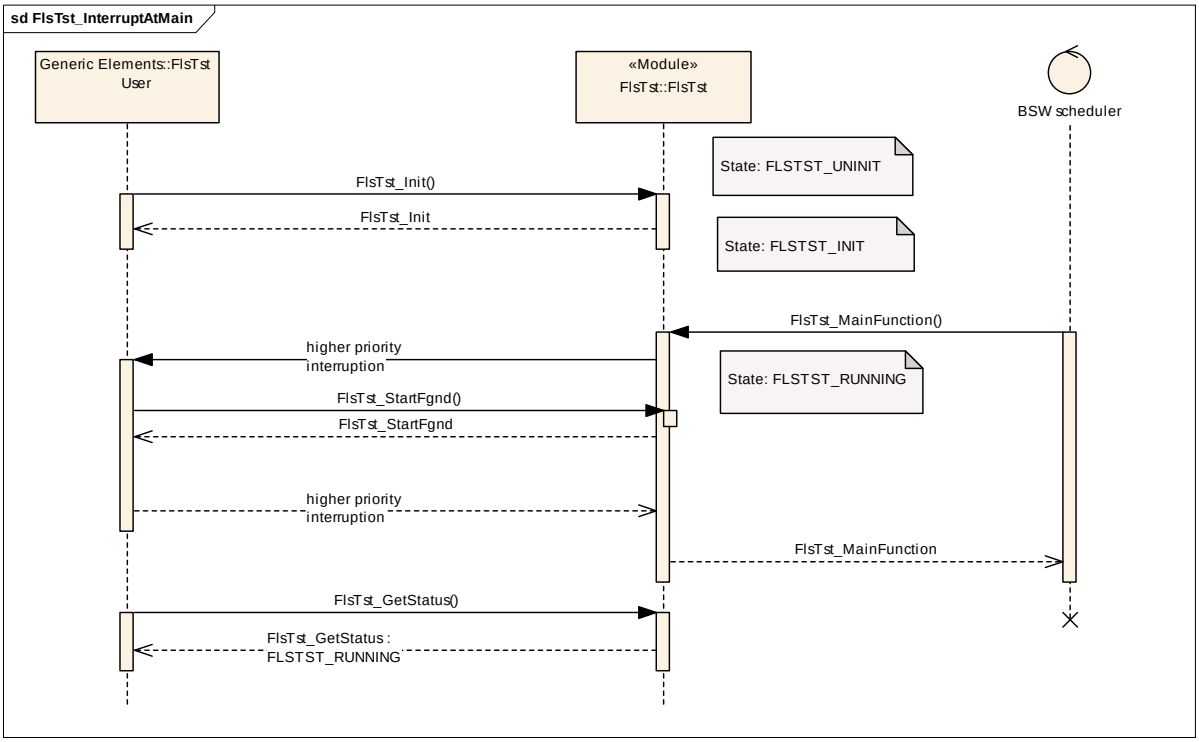

Figure 10: Foreground task interrupts Background Task

10 Configuration specification

This chapter defines configuration parameters and their clustering into containers.

10.1 How to read this chapter

In addition to this section, it is highly recommended to read the documents:

- AUTOSAR Layered Software Architecture [1]
- AUTOSAR ECU Configuration Specification [6]
  This document describes the AUTOSAR configuration methodology and the AUTOSAR configuration Meta model in detail.

The following is only a short survey of the topic and it will not replace the ECU Configuration Specification document.

10.1.1 Configuration and configuration parameters

Configuration parameters define the variability of the generic part(s) of an implementation of a module. This means that only generic or configurable module implementation can be adapted to the environment (software/hardware) in use during system and/or ECU configuration.

The configuration of parameters can be achieved at different times during the software process: before compile time, before link time or after build time. In the following, the term “configuration class” (of a parameter) shall be used in order to refer to a specific configuration point in time.

10.1.2 Containers

Containers structure the set of configuration parameters. This means:

- *all* configuration parameters are kept in containers.
- (sub-) containers can reference (sub-) containers. It is possible to assign a multiplicity to these references. The multiplicity then defines the possible number of instances of the contained parameters.

10.1.3 Specification template for configuration parameters

The following tables consist of three sections:

- the general section
- the configuration parameter section
- the section of included/referenced containers

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

Label	***Description***
x	The configuration parameter shall be of configuration class *Pre-compile time*.
--	The configuration parameter shall never be of configuration class *Pre-compile time*.

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

Label	*Description*
x	The configuration parameter shall be of configuration class *Link time*.
--	The configuration parameter shall never be of configuration class *Link time*.

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

Label	*Description*
x	The configuration parameter shall be of configuration class *Post Build* and no specific implementation is required.
L	*Loadable* - the configuration parameter shall be of configuration class *Post Build* and only one configuration parameter set resides in the ECU.
M	*Multiple* - the configuration parameter shall be of configuration class *Post Build* 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 *Post Build*.

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

[FlsTst079] ⌈Variant PC: This variant is limited to pre-compile-configuration parameters only. The intention of this variant is to optimize the parameters configuration for a source code delivery. ⌋ ( )

[FlsTst081] ⌈Variant PB: This variant allows a mix of pre-compile time-, post build-time configuration parameters. The intention of this variant is to optimize the parameters configuration for a re-loadable binary⌋ ( )

10.2.2 FlsTst

SWS Item	**FlsTst135_Conf :**
Module Name	*FlsTst*
Module Description	--

Included Containers		
Container Name	*Multiplicity*	*Scope / Dependency*
FlsTstConfigSet	1	Multiple Configuration Set Container

FlsTstConfigurationOfOptApiServices	1	--
FlsTstDemEventParameterRefs	0..1	Container for the references to DemEventParameter elements which shall be invoked using the API Dem_ReportErrorStatus API in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId value. The standardized errors are provided in the container and can be extended by vendor specific error references.
FlsTstGeneral	1	--

10.2.3 FlsTstGeneral

SWS Item	**FlsTst082_Conf :**
Container Name	FlsTstGeneral
Description	--
Configuration Parameters	

SWS Item	**FlsTst083_Conf :**		
Name	FlsTstDevErrorDetect {FLSTST_DEV_ERROR_DETECT}		
Description	Switch for enabling the development error detection.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: module		

SWS Item	**FlsTst120_Conf :**		
Name	FlsTstNumberOfTestedCellsAtomic {FLSTST_NUMBER_OF_TESTED_CELLS_ATOMIC}		
Description	Configures the Number of cells to be tested in background mode without checking user requests (Abort, Suspend).		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 4294967295		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: module		

SWS Item	**FlsTst084_Conf :**		
Name	FlsTstTestCompletedNotificationSupported {FLSTST_TEST_COMPLETED_NOTIFICATION_SUPPORTED}		
Description	Switch to indicate that the notification is supported.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	true		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	

Scope / Dependency	scope: module

SWS Item	**FlsTst158_Conf :**		
Name	FlsTstTestIntervalIdEndValue {FLSTST_TEST_INTERVAL_ID_END_VALUE}		
Description	Defines the end value of the Test Interval Id.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 4294967295		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: module		

SWS Item	**FlsTst160_Conf :**		
Name	FlsTstTestResultSignature {FLSTST_TEST_RESULT_SIGNATURE}		
Description	Configures the result of the test in background mode: True: Test Result is a signature (see FlsTst155, FlsTst054) False: Test Result is ok/not ok (see FlsTst153, FlsTst042)		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: module		

No Included Containers

10.2.4 FlsTstConfigurationOfOptApiServices

SWS Item	**FlsTst085_Conf :**
Container Name	FlsTstConfigurationOfOptApiServices
Description	--
Configuration Parameters	

SWS Item	**FlsTst092_Conf :**		
Name	FlsTstGetCurrentStateApi {FLSTST_GET_CURRENT_STATE_API}		
Description	Adds / removes the service FlsTst_GetCurrentState() from the code.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: module		

SWS Item	**FlsTst098_Conf :**
Name	FlsTstGetErrorDetailsApi {FLSTST_GET_ERROR_DETAILS_API}
Description	Adds / removes the service FlsTst_GetErrorDetails() from the code.
Multiplicity	1
Type	EcucBooleanParamDef

Default value	false		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: module		

SWS Item	**FlsTst094_Conf :**		
Name	FlsTstGetTestResultBgndApi {FLSTST_GET_TEST_RESULT_BGND_API}		
Description	Adds / removes the service FlsTst_GetTestResultBgnd() from the code.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: module		

SWS Item	**FlsTst150_Conf :**		
Name	FlsTstGetTestResultFgndApi {FLSTST_GET_TEST_RESULT_FGND_API}		
Description	Adds / removes the service FlsTst_GetTestResultFgnd() from the code.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: module		

SWS Item	**FlsTst096_Conf :**		
Name	FlsTstGetTestSignatureBgndApi {FLSTST_GET_TEST_SIGNATURE_BGND_API}		
Description	Adds / removes the service FlsTst_GetTestSignatureBgnd() from the code.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: module		

SWS Item	**FlsTst097_Conf :**		
Name	FlsTstGetTestSignatureFgndApi {FLSTST_GET_TEST_SIGNATURE_FGND_API}		
Description	Adds / removes the service FlsTst_GetTestSignatureFgnd() from the code.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: module		

SWS Item	FlsTst086_Conf :		
Name	FlsTstStartFgndApi {FLSTST_START_FGND_API}		
Description	Adds / removes the service FlsTst_StartFgnd() from the code.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
ConfigurationClass	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: module		

SWS Item	FlsTst087_Conf :		
Name	FlsTstSuspendResumeApi {FLSTST_SUSPEND_RESUME_API}		
Description	Adds / removes the services FlsTst_Suspend() and FlsTst_Resume() from the code.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
ConfigurationClass	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: module		

SWS Item	FlsTst099_Conf :		
Name	FlsTstTestEccApi {FLSTST_TEST_ECC_API}		
Description	Adds / removes the service FlsTst_TestEcc() from the code.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
ConfigurationClass	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: module		

SWS Item	FlsTst095_Conf :		
Name	FlsTstVersionInfoApi {FLSTST_VERSION_INFO_API}		
Description	Adds / removes the service FlsTst_GetVersionInfo() from the code.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	false		
ConfigurationClass	Pre-compile time	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: module		

No Included Containers

10.2.5 FlsTstDemEventParameterRefs

SWS Item	**FlsTst170_Conf :**
Container Name	FlsTstDemEventParameterRefs
Description	Container for the references to DemEventParameter elements which shall be invoked using the API Dem_ReportErrorStatus API in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId value. The standardized errors are provided in the container and can be extended by vendor specific error references.
Configuration Parameters	

SWS Item	**FlsTst171_Conf :**		
Name	FLSTST_E_FLSTST_FAILURE		
Description	Reference to the DemEventParameter which shall be issued when the error "Flash Failure" has occurred.		
Multiplicity	0..1		
Type	Reference to [DemEventParameter]		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency			

No Included Containers

10.2.6 FlsTstConfigSet

SWS Item	**FlsTst152_Conf :**
Container Name	FlsTstConfigSet [Multi Config Container]
Description	Multiple Configuration Set Container
Configuration Parameters	

SWS Item	**FlsTst122_Conf :**		
Name	FlsTstBlockNumberBgnd {FLSTST_BLOCK_NUMBER_BGND}		
Description	This parameter shall represent the number of test blocks available for the background test. calculationFormula = Number of configured FlsTstBlocks in the FlsTstBlockBgndConfigSet (or 0 if no FlsTstBlocks are configured).		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 4294967295		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	--	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: module		

SWS Item	**FlsTst124_Conf :**
Name	FlsTstBlockNumberFgnd {FLSTST_BLOCK_NUMBER_FGND}
Description	This parameter shall represent the number of test blocks available

	for the foreground test. calculationFormula = Number of configured FlsTstBlocks in the FlsTstBlockFgndConfigSet (or 0 if no FlsTstBlocks are configured).		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 4294967295		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	--	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: module		

SWS Item	**FlsTst102_Conf :**		
Name	FlsTstTestCompletedNotification {FLSTST_TEST_COMPLETED_NOTIFICATION}		
Description	Pointer to function, which shall be called after finishing the background Flash test interval.		
Multiplicity	1		
Type	EcucFunctionNameDef		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	--	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: module		

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
FlsTstBlockBgndConfigSet	0..1	This container defines the blocks in background mode.
FlsTstBlockFgndConfigSet	0..1	This container defines the blocks in foreground mode.

10.2.7 FlsTstBlockBgndConfigSet

SWS Item	**FlsTst103_Conf :**
Container Name	FlsTstBlockBgndConfigSet
Description	This container defines the blocks in background mode.
Configuration Parameters	

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
FlsTstBlock	1..*	This container specifies configuration parameters for an individual test block.

10.2.8 FlsTstBlockFgndConfigSet

SWS Item	**FlsTst104_Conf :**
Container Name	FlsTstBlockFgndConfigSet
Description	This container defines the blocks in foreground mode.
Configuration Parameters	

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
FlsTstBlock	1..*	This container specifies configuration parameters for an individual test block.

10.2.9 FlsTstBlock

SWS Item	**FlsTst105_Conf :**
Container Name	FlsTstBlock
Description	This container specifies configuration parameters for an individual test block.
Configuration Parameters	

SWS Item	**FlsTst106_Conf :**			
Name	FlsTstBlockBaseAddress {FLSTST_BLOCK_BASE_ADDRESS}			
Description	Start Address of the Flash block.			
Multiplicity	1			
Type	EcucIntegerParamDef			
Range	0 .. 18446744073709551615			
Default value	--			
ConfigurationClass	***Pre-compile time***		X	VARIANT-PRE-COMPILE
	Link time		--	
	Post-build time		X	VARIANT-POST-BUILD
Scope / Dependency	scope: module			

SWS Item	**FlsTst151_Conf :**			
Name	FlsTstBlockIndex {FLSTST_BLOCK_INDEX}			
Description	Foreground Test: Index identifies block to be tested by FlsTst_StartFgnd(); Background Test: The scheduling for background test shall follow an order defined by this index. '0' means highest priority.			
Multiplicity	1			
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)			
Range	0 .. 4294967295			
Default value	--			
ConfigurationClass	***Pre-compile time***		X	VARIANT-PRE-COMPILE
	Link time		--	
	Post-build time		X	VARIANT-POST-BUILD
Scope / Dependency	scope: module			

SWS Item	**FlsTst107_Conf :**
Name	FlsTstBlockSize {FLSTST_BLOCK_SIZE}

Description	This parameter shall represent the Flash Test block size.			
Multiplicity	1			
Type	EcucIntegerParamDef			
Range	0 .. 4294967295			
Default value	--			
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE	
	Link time	--		
	Post-build time	X	VARIANT-POST-BUILD	
Scope / Dependency	scope: module			

SWS Item	**FlsTst119_Conf :**			
Name	FlsTstNumberOfTestedCells {FLSTST_NUMBER_OF_TESTED_CELLS}			
Description	Configures the Number of cells to be tested in background mode during one scheduled task (FlsTst_MainFunction() call).			
Multiplicity	1			
Type	EcucIntegerParamDef			
Range	0 .. 4294967295			
Default value	--			
ConfigurationClass	***Pre-compile time***	X	All Variants	
	Link time	--		
	Post-build time	--		
Scope / Dependency	scope: module			

SWS Item	**FlsTst123_Conf :**			
Name	FlsTstSignatureAddress {FLSTST_SIGNATURE_ADDRESS}			
Description	Address of the signature reference value of the Flash test block.			
Multiplicity	1			
Type	EcucIntegerParamDef			
Range	0 .. 18446744073709551615			
Default value	--			
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE	
	Link time	--		
	Post-build time	X	VARIANT-POST-BUILD	
Scope / Dependency	scope: module			

SWS Item	**FlsTst101_Conf :**		
Name	FlsTstTestAlgorithm {FLSTST_TEST_ALGORITHM}		
Description	This is the configuration of the test algorithm for foreground mode and background mode. The availability of algorithm is implementation specific.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	FLSTST_16BIT_CRC	--	
	FLSTST_32BIT_CRC	--	
	FLSTST_8BIT_CRC	--	
	FLSTST_CHECKSUM	--	
	FLSTST_DUPLICATED_MEMORY	--	
	FLSTST_ECC	--	
ConfigurationClass	***Pre-compile time***	X	VARIANT-

			PRE-COMPILE
Link time		--	
Post-build time		X	VARIANT-POST-BUILD
Scope / Dependency	scope: module		

No Included Containers

10.3 Published Information

[FlsTst136] ⌈The standardized common published parameters as required by BSW00402 in the General Requirements on Basic Software Modules [3] shall be published within the header file of this module and need to be provided in the BSW Module Description. The according module abbreviation can be found in the List of Basic Software Modules [8].⌋(BSW00402, BSW00374, BSW00379, BSW00318, BSW00321)

Additional module-specific published parameters are listed below if applicable.
Not applicable requirementsific published parameters are listed below if applicable.

11 Not applicable requirements

[FlsTst166] ⌈These requirements are not applicable to this specification.⌋ (BSW00344, BSW159, BSW167, BSW170, BSW00419, BSW00398, BSW00375, BSW00416, BSW168, BSW00423, BSW00424, BSW00425, BSW00426, BSW00427, BSW00428, BSW00429, BSW00431, BSW00432, BSW00433, BSW00434, BSW00422, BSW00417, BSW161, BSW162, BSW005, BSW00415, BSW164, BSW00325, BSW00326, BSW00342, BSW00343, BSW007, BSW00300, BSW00413, BSW00347, BSW00305, BSW00307, BSW00310, BSW00373, BSW00327, BSW00335, BSW00350, BSW00408, BSW00410, BSW00348, BSW00353, BSW00361, BSW00301, BSW00302, BSW00328, BSW00312, BSW006, BSW00378, BSW00306, BSW00308, BSW00309, BSW00371, BSW00358, BSW00414, BSW00330, BSW00331, BSW009, BSW00401, BSW172, BSW010, BSW003, BSW00341, BSW00334, BSW00437, BSW00439, BSW00440, BSW12267, BSW12461, BSW12462, BSW12463, BSW12068, BSW12069, BSW12169, BSW12075, BSW12129, BSW12064, BSW12067, BSW12077, BSW12078, BSW12092, BSW12265, BSW14221)