Document Title	Specification of Diagnostic Event Manager
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	019
Document Classification	Standard

Document Version	5.2.0
Document Status	Final
Part of Release	4.1
Revision	3

Document Change History

Date	Version	Changed by	Description
31.03.2014	5.2.0	AUTOSAR Release Management	• Further clarification of event combination • Clarification of DTC groups • Editorial changes
31.10.2013	5.1.0	AUTOSAR Release Management	• Added API table for service interfaces • Clarification of event combination • Editorial changes • Removed chapter(s) on change documentation

08.03.2013	5.0.0	AUTOSAR Administration	• Incorporated new and reporting of several Dem internal data elements • Supported J1939 • Supported OBD • Reworked Dem/Dcm interface • Introduced new debouncing behaviour and debounce counter storage • Extended clear DTC functionality • Supported event suppression • Extended DTC storage behavior • Incorporation of user-controlled warning indicator requestet bit • Incorporation of autostart behaviour for operation cycles
09.12.2011	4.2.0	AUTOSAR Administration	• Introduced multiple formats per DTC • Reworked Dem_ResetEventStatus behavior • Reworked Dlt interaction • Reworked Dem/Dcm interface • Corrected include-structure and RTE interfaces • Refined several aspects on features
18.10.2010	4.1.0	AUTOSAR Administration	• Reworked Dem/Dcm interface • Extended definition of ”Diagnostic Monitor” • Introduced "Event significance" and "DTC suppression" • Reworked OBD (esp. interface for service $02, readiness, and permanent memory) • Reworked file-structure • Finalization of issues on Revision 1

04.12.2009	4.0.0	AUTOSAR Administration	• Improved and extended of DEM functional description (especially: status bit handling, event displacement, debouncing, fault confirmation and indicator handling, event combination, enable- & storage conditions, event related data, operation cycle management) • Introduced the approach for functional diagnostics of SW-Cs • Added Dlt interaction and debugging interface • Document structure reworked and extended • Legal disclaimer revised
02.08.2008	3.0.0	AUTOSAR Administration	• Document structure reworked and extended • Add APIs and configuration parameters for OBD support • Improve interaction between DCM and software components • Legal disclaimer revised
28.11.2007	2.2.0	AUTOSAR Administration	• Improvement of RTE compliance • Improvement of configuration part • Improvement of document structure • Rework and tightening of data type useage • New API added • Document meta information extended • Small layout adaptations made
24.01.2007	2.1.1	AUTOSAR Administration	• “Advice for users” revised • “Revision Information” added

05.12.2006	2.1.0	AUTOSAR Administration	• Complete rework and extension of debouncing part • Dem_ClearGroupOfDTC and Dem_ClearSingleDTC replaced by Dem_SingleDTC • Dem_GetNextFilteredDTC AndFDC, Dem_SetDTCFilterForRecords, Dem_GetSizeOfFreezeFrame, Dem_SetValueByOemId, Dem_SetEnableCondition, Xxx_DemGetFaultDetection Counter added • DTCTranslationType replaced by DTCKind in several APIs • Chapter "Service DEM" added • Function IDs reworked • File Structure reworked and extended • Configuration chapter reworked and extended • Dem_GetNextFilteredDTC reworked • Legal disclaimer revised
29.06.2006	2.0.0	AUTOSAR Administration	• Layout Adaptations

Disclaimer

This specification 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 specification.

The material contained in this specification is protected by copyright and other types of Intellectual Property Rights. The commercial exploitation of the material contained in this specification requires a license to such Intellectual Property Rights.

This specification 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 specification may be utilized or reproduced, in any form or by any means, without permission in writing from the publisher.

The AUTOSAR specifications have been developed for automotive applications only. They have neither been developed, nor tested for non-automotive applications.

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

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

The service component Diagnostic Event Manager (Dem) is responsible for processing and storing diagnostic events (errors) and associated data. Further, the Dem provides fault information to the Dcm (e.g. read all stored DTCs from the event memory). The Dem offers interfaces to the application layer and to other BSW modules.

The basic target of the Dem specification document is to define the ability for a common approach of a "diagnostic fault memory" for automotive manufacturers and component suppliers

This specification defines the functionality, API and the configuration of the AUTOSAR basic software module Diagnostic Event Manager (Dem). Parts of the internal behavior are manufacturer specific and described in the Limitations chapter.

2 Acronyms and Abbreviations

The glossary below includes acronyms and abbreviations relevant to the CAN Interface module that are not included in the [1, AUTOSAR glossary].

Acronym	Description
Aging	Unlearning/deleting of a no longer failed event/DTC after a defined number of operation cycles from event memory
Aging Counter	The "Aging Counter" or "Aging Cycle Counter" or "DTC Aging Counter" specifies the counter which is used to perform Aging. It counts the number of operation cycles until an event/DTC is removed from event memory.
Combined DTC	Normal DTC, but referenced by multiple events reported by several monitors (e.g. ECU Defect, consisting of different HW defects).
Debounce counter	Internal counter for counter-based debouncing algorithm(s).
Dem-internal data value	Some data values (e.g. the occurrence counter) are calculated by the Dem module itself internally.
Denominator	The denominator of a specific monitor m (Denominatorm) is a counter indicating the number of vehicle driving events, taking into account conditions specific to that specific monitor.
Dependend / Secondary ECUs	Dependend / Secondary (or dep. / sec.) ECUs are always related to a Master or a Primary ECU.
Event combination	Event combination is a method to merge several events to one specific combined DTC. It is used to adapt different monitor results to one significant fault, which is clearly evaluable in a service station.
Event debouncing	Debouncing is a specific mechanism (e.g. counter-based) to evaluate, if the diagnostic event gets qualified. This works on top of potential signal debouncing and can be done within the SW-C or inside the Dem.
Event qualification	A diagnostic event is qualified in case of a passed or a failed result is set (Dem-internal or reported from another BSW module or SW-C).
Event confirmation	A diagnostic event is confirmed in case of repeated detection of qualified events over cycles or time evaluated by means of fault confirmation counters. Therefore, also the UDS DTC Status bit 3 (ConfirmedDTC) is set.
Event memory	An event memory (e.g. Primary memory) consists of several event memory entries.

Acronym	Description
Event memory entry	The event memory entry is a single storage container for an event and its event related data. The event memory entry is dynamically assigned to specific events.
Event related data	Event related data is additional data, e.g. sensor values or time stamp/mileage, which is stored with an event in an event memory. ISO defines two types of event related data: freeze frames (snapshot records) and extended data.
Event memory overflow indication	The event memory overflow indication indicates, if this specific event memory is full and the next event occurs to be stored in this event memory.
Extended data record	An extended data record is a record to store specific information assigned to a fault.
Fault Detection Counter	sint8 value as used in ISO and FDC-APIs.
Freeze frame	Freeze frame is defined as a record of data (DIDs/PIDs). Freeze frames are the same as SnapShotRecords in [2, ISO 14229-1].
General Denominator	The general denominator is a counter indicating the number of times a vehicle has been operated, taking into account general conditions.
Healing	Switching off the warning indicator including the handling of reported passed results over a period of time / several operation cycles
In-Use performance ratio	The in-use performance ratio (IUPR) of a specific monitor m of the OBD system is: IUPRm = Numeratorm / Denominatorm
Master ECU	As a primary ECU a Master ECU stores “it’s own” and “reported errors” of related dep. / sec ECUs in it’s Event Memory. Beside this a Master has to fulfill special Master tasks as MIL Master or provision of “general nominator” information.
Monitor	A diagnostic monitor is a routine entity determining the proper functionality of a component. Alternatively the term “diagnostic function” can be used.
Numerator	The numerator of a specific monitor m (Numeratorm) is a counter indicating the number of times a vehicle has been operated such that all monitoring conditions necessary for that specific monitor to detect a malfunction have been encountered.
Operating cycle	An ‘Operating cycle’ is the base of the event qualifying and also Dem scheduling (e.g. ignition key off-on cycles, driving cycles, etc.)

Acronym	Description
OBD	On-Board Diagnostics, or OBD is a generic term referring to a vehicle's self-diagnostic and reporting capability. OBD systems give the vehicle owner or a repair technician access to state of health information for various vehicle sub-systems.
OBD ECUs	"In a vehicle there can be 3 different types of OBD ECUs: • Master ECU (one per vehicle) • Primary ECU (several per vehicle) • Dependend / Secondary ECUs (several per vehicle)
P-Code	Power train code
PossibleErrors	PossibleErrors means the ApplicationErrors as defined in meta model
Primary ECU	A primary ECU stores "it's own" and "reported errors" of related dep. / sec ECUs in it's Event Memory
Readiness	The readiness refers to the tested bits TestNotCompletedSinceLastClear (bit 4) and TestNotCompleteThisOperationCycle (bit 6) of the UDS DTC Status Byte.

Abbreviation	Description
API	Application Programming Interface
BSW	Basic Software
CDD	Complex Device Driver
CRC	Cyclic Redundancy Check
Dcm	Diagnostic Communication Manager
Dem	Diagnostic Event Manager
Det	Development Error Tracer
DID	Data Identifier
Dlt	Diagnostic Log and Trace
DTC	Diagnostic Trouble Code
ECU	Electronic Control Unit
EcuM	Electronic Control Unit Manager
FDC	Fault Detection Counter
FiM	Function Inhibition Manager
FMI	Failure Mode Indicator (SAE J1939)
HW	Hardware
ID	Identification/Identifier
ISO	International Standardization Organization
IUMPR	In Use Monitoring Performance Ratio (OBD Term)

Abbreviation	Description
MIL	Malfunction Indicator Light (SAE J1979) or Lamp (SAE J1939)
NVRAM	Non volatile RAM
OBD	Onboard Diagnostics
OC	Occurrence Count (SAE J1939)
OEM	Original Equipment Manufacturer (Automotive Manufacturer)
OS	Operating System
PID	Parameter Identification (SAE J1587 or SAE J1979)
PTO	Power Take Off
RAM	Random Access Memory
ROM	Read-only Memory
RTE	Runtime Environment
SPN	Suspect Parameter Number (SAE J1939)
SSCP	synchronous server call point
SW	Software
SW-C	Software Component
UDS	Unified Diagnostic Services
DYC	OBD Term: Driving Cycle (OBD Term)
PFC cycle	OBD Term: Permanent fault code - driving cycle (OBD Term)
WUC	OBD Term: Warm up cycle (OBD Term)
WIR	Warning Indicator Request
RBM cycle	OBD Term: General Nominator / Rate-based monitoring - driving cycle (OBD Term)

3 Related documentation

3.1 Input documents & related standards and norms

[1] Glossary
AUTOSAR_TR_Glossary

[2] Unified diagnostic services (UDS) - Part 1: Specification and requirements (Release 2006-12)
http://www.iso.org

[3] General Specification of Basic Software Modules
AUTOSAR_SWS_BSWGeneral

[4] Specification of a Diagnostic Communication Manager for SAE J1939
AUTOSAR_SWS_SAEJ1939DiagnosticCommunicationManager

[5] Requirements on Function Inhibition Manager
AUTOSAR_SRS_FunctionInhibitionManager

[6] Specification of Diagnostic Log and Trace
AUTOSAR_SWS_DiagnosticLogAndTrace

[7] Specification of Diagnostic Communication Manager
AUTOSAR_SWS_DiagnosticCommunicationManager

[8] Specification of NVRAM Manager
AUTOSAR_SWS_NVRAMManager

[9] Requirements on Diagnostic
AUTOSAR_SRS_Diagnostic

[10] General Requirements on Basic Software Modules
AUTOSAR_SRS_BSWGeneral

[11] SAE J1939-73 Application Layer - Diagnostics

[12] Communication
http://portal.osek-vdx.org/files/pdf/specs/
osekcom303.pdf

[13] SAE J1979

[14] Road vehicles - Communication between vehicle and external equipment for emission-related diagnostic - Part 5: Emission-related diagnostic services.
http://www.iso.org

[15] Title 13, California Code Regulations, Section 1971.1, On-Board Diagnostic System Requirements for 2010 and Subsequent Model-Year Heavy-Duty Engines (HD OBD)

[16] Title 13, California Code Regulations, Section 1968.2, Malfunction and Diagnos-

tic System Requirements for 2004 and Subsequent Model-Year Passenger Cars, Light-Duty Trucks, and Medium-Duty Vehicles and Engines (OBD II) (Biennial Review MY08-11)
http://www.arb.ca.gov/regact/obdii06/19682clean.pdf

[17] Requirements on Diagnostic Log and Trace
AUTOSAR_SRS_DiagnosticLogAndTrace

3.2 Related specification

AUTOSAR provides a General Specification on Basic Software modules [3, SWS BSW General], which is also valid for Diagnostic Event Manager.

Thus, the specification SWS BSW General shall be considered as additional and required specification for Diagnostic Event Manager.

4 Constraints and assumptions

Some of the synchronous API calls defined within the Dem might take variable time to complete. Thus, this variable time must be considered in the system configuration.

[SWS_Dem_00126] ⌈There shall only be one Dem module available per ECU.⌋ *(SRS_Diag_04002)*

The Dem can have multiple different sections of event memory. The mapping of a DTC to the respective section is done with the parameter DTC Origin. A specific ECU's Dem is only accessible by software components located inside the same ECU.

4.1 Limitations

Timing constrains have to be considered for the whole ECU. If there are explicit needs for faster responses from the Dem than the Dem basic cycle time, special measures have to be implemented, that are not specified in this AUTOSAR document. This is especially the case in ECUs with many events.

The handling of infrastructure errors reported by the RTE during Dem <=> SW-C interactions is missing from the SWS and might have to be taken into account by implementers if they need it.

The Dem is able to support additional event memories (Permanent memory, Mirror memory and Secondary memory), but the specific event memory processing is not defined in detail.

The functional behavior on the callback kind InitMonitorForFunction is not specified.

Some details on the interaction between Dem and specific emission-related SW-C are not specified in this specification, since they are dependent on the SW-C implementation. The following functionality is not defined:

- Malfunction Indicator Lamp (MIL)-activation (MIL handler interaction to Dem, MIL-bulb check, readiness blinking, blinking in case of catalyst damaging misfire, etc.)
- misfire fault handling (debouncing over all cylinders, filtering single / multiple misfire faults)
- support of similar conditions for the specific healing of misfire and fuel system faults

Note: For OBD2, it is required that misfire and fuel system fault shall only be healed (yielding leaving Service $07) under the similar conditions as they have been detected. The "similarity" is derived from a "window" spanned by ranges on engine speed, engine load and temperature conditions being present at the time of fault detection.

For the handling of similar conditions it is assumed that the SW-C modules of misfire detection / fuelsystem diagnostics carry out the necessary computations themselves. That is, it is not globally solved within the Dem.

However, based on specific interface requirements of the respective misfire detection / fuelsystem diagnostics this may result in an extension of the Dem. Any further implementations on that need to be defined for a particular engine control unit project depending on the diagnostics and the Dem implementation.

The DTCStoredDataRecordNumber (absolute freeze frame record addressing functionality) is limited to 0x00 (OBD freeze frame, refer to chapter 7.9.2.2).

The structure of a specific extended data record identified by its record number is unique per ECU.

The Dcm may lock the event memory update (refer to [SWS_Dem_00270]) while processing the “read diagnostic data” service, due to architectural design.

This specification covers a subset of SAE J1939 related diagnostic requirements (see Table 1: Supported DMx messages in [4, SWS J1939 Dcm]). The SPN Conversion Method is limited to Version 4 (CM = 0).

Post build time loadable configuration is not supported by the Dem configuration model and respective parameter set.

4.2 Applicability to car domains

The Dem is designed to fulfill the design demands for ECUs with OBD requirements as well as for ECUs without OBD requirements. The immediate domains of applicability are currently body, chassis and powertrain ECUs. However, there is no reason why the Dem cannot be used to implement ECUs for other car domains like infotainment.

5 Dependencies to other modules

The AUTOSAR **Diagnostic Event Manager (Dem)** has interfaces and dependencies to the following Basic software modules and Software Components:

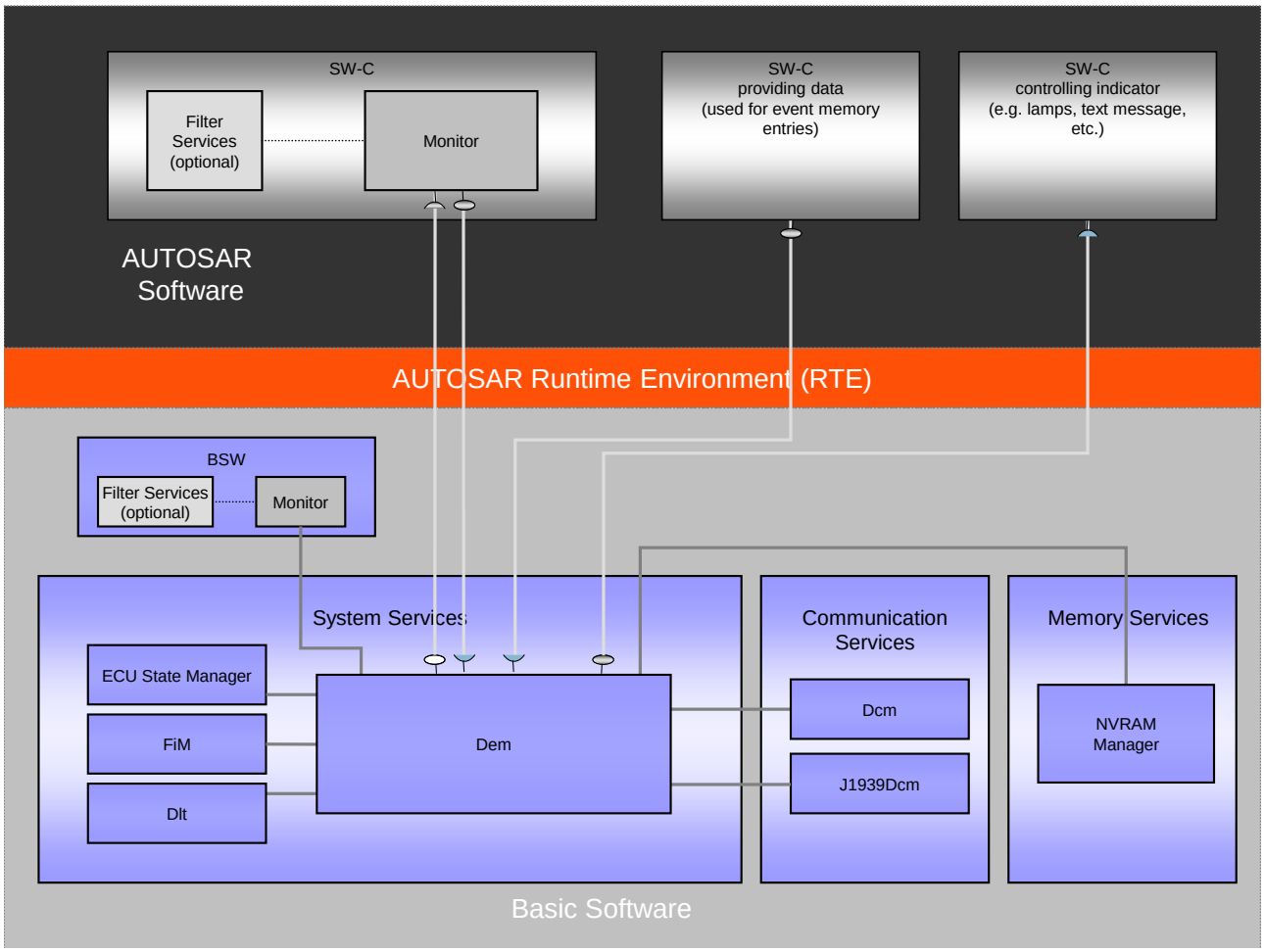

Figure 5.1: Dependencies of the Diagnostic Event Manager (Dem) to other software modules

- The **Function Inhibition Manager (FiM)** (refer to [5, SWS FiM]) stands for the evaluation and assignment of events to the required actions for Software Components (e.g. inhibition of specific “Monitors”). The Dem informs and updates the Function Inhibition Manager (FiM) upon changes of the event status in order to stop or release function entities according to assigned dependencies.
- The **Diagnostic Log and Trace (Dlt)** (refer to [6, SWS Dlt]) provides a generic Logging and Tracing functionality for the Dem. The Dem informs and updates the Diagnostic Log and Trace (Dlt) upon changes of the event status and provides access on the current event related data in order to log and trace this information.
- The **Diagnostic Communication Manager (Dcm)** (refer to [7, SWS Dcm]) is in charge of the UDS and SAE J1979 communication path and execution of diag-

nostic service resulting in the processing of diagnostic requests from an external tester or onboard test system. It forwards requests coming from an external diagnostic scan tool and is further responsible for assembly of response messages (DTC, status information, etc.) which will be transferred to the external diagnostic scan tool afterwards.

- The **Diagnostic Communication Manager for J1939 (J1939Dcm)** (refer to [4, SWS J1939 Dcm]) is in charge of the SAE J1939-73 diagnostics communication protocol.
- **Software-Components (SW-C)** and **Basic Software (BSW) modules** can access the Dem to update and/or retrieve current event status information. SW-Cs and BSW modules can retrieve data from the Dem e.g. to turn the indicator lamps on or off. The monitor is a sub-component of a SW-C / BSW module.
- **Data Provider** SW-Cs and/or BSW modules will provide data (i.e. event related data) required by the Dem, for example, to be able to create event memory entries.
- The **NVRAM Manager (NvM)** (refer to [8]) provides mechanisms to store data blocks in NVRAM. NVRAM blocks (maximum size is a matter of configuration) are assigned to the Dem and used by the Dem to achieve permanent storage of event status information and associated data (e.g. over power-on reset).
- The **ECU State Manager (EcuM)** is responsible for the basic initialization and de-initialization of basic software components including Dem.
- The **RTE** implements scheduling mechanisms for BSW, e.g. assigns priority and memory protection to each BSW module used in an ECU.

5.1 File structure

5.1.1 Code file structure

For details refer to the chapter 5.1.6 “Code File Structure” in SWS_BSWGeneral. (5.1.6.3 Link time configuration source & 5.1.6.4 Post-build time configuration source)

5.1.2 Header file structure

[SWS_Dem_00151] ⌈ The header-file structure shall contain the following files named (in addition to SWS_BSWGeneral):

- Dem_Dcm.h - for Dem APIs used by the Dcm exclusively
- Dem_Types.h - for Dem data types (not defined in Rte_Dem_Type.h)
- Dcm_Types.h - for all imported Dcm types (refer to chapter 8.1)

- J1939Dcm_Types.h - for all imported J1939Dcm types (optional)
- FiM.h - for Function Inhibition Manager symbols (optional)
- Dlt.h - for Diagnostic Log & Trace symbols (optional)
- NvM.h - for NVRAM Manager symbols
- <...>.h - contains all C-callback declarations (refer to DemHeaderFileInclusion in DemGeneral) configured for the Dem

⌋(*SRS_BSW_00447*)

The symbolic names (refer to TPS_ECUC_02108) are generated for configuration containers containing an identifier parameter, like event Id symbols, operation cycles, indicators, enable/storage conditions, etc.

Note, that also SW-C event Ids are published especially for complex device drivers, which may access on the status of specific SW-C events. SW-Cs use different ports to distinguish between different events.

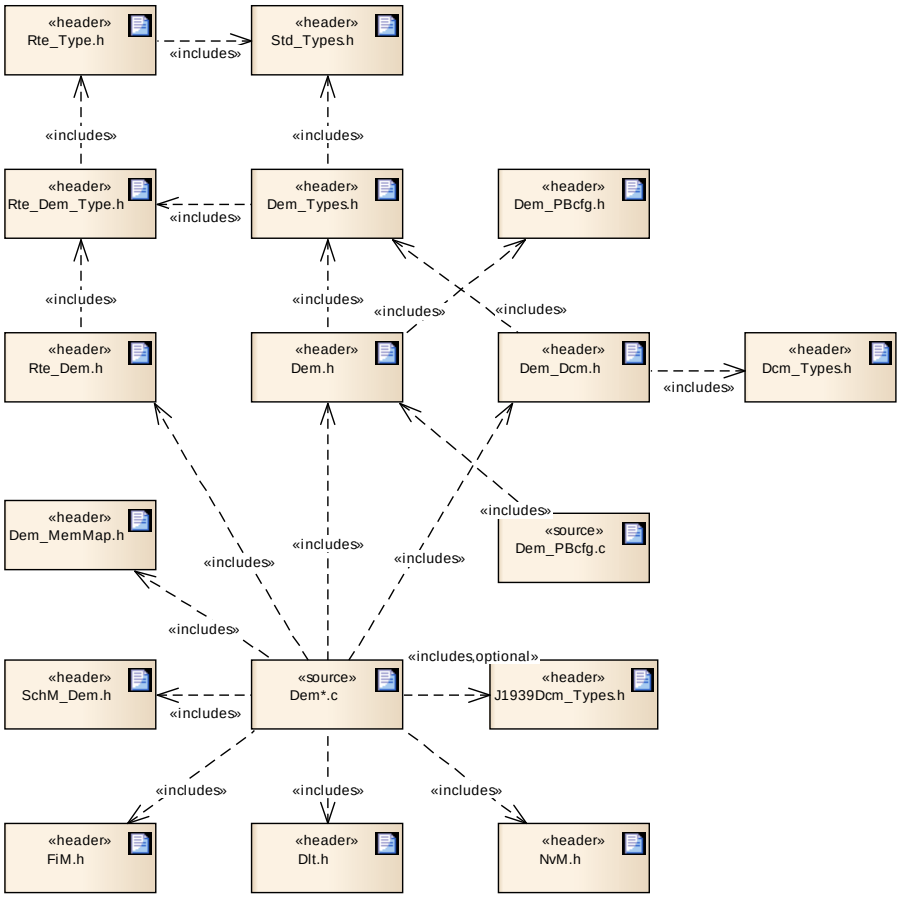

Figure 5.2: Header file structure

6 Requirements traceability

The following table references the requirements specified in [9] as well as [10] and links to the fulfillment of these.

Requirement	Description	Satisfied by
[SRS_BSW_00101]	The Basic Software Module shall be able to initialize variables and hardware in a separate initialization function	[SWS_Dem_00181] [SWS_Dem_00340] [SWS_Dem_00550]
[SRS_BSW_00171]	Optional functionality of a Basic-SW component that is not required in the ECU shall be configurable at pre-compile-time	[SWS_Dem_00255] [SWS_Dem_00579]
[SRS_BSW_00334]	All Basic Software Modules shall provide an XML file that contains the meta data	[SWS_Dem_00267] [SWS_Dem_00268]
[SRS_BSW_00336]	Basic SW module shall be able to shutdown	[SWS_Dem_00102] [SWS_Dem_00182] [SWS_Dem_00341]
[SRS_BSW_00345]	BSW Modules shall support pre-compile configuration	[SWS_Dem_00267] [SWS_Dem_00268]
[SRS_BSW_00396]	The Basic Software Module specifications shall specify one classe (of the three) to be supported	[SWS_Dem_00267] [SWS_Dem_00268]
[SRS_BSW_00397]	The configuration parameters in pre-compile time are fixed before compilation starts	[SWS_Dem_00267] [SWS_Dem_00268]
[SRS_BSW_00398]	The link-time configuration is achieved on object code basis in the stage after compiling and before linking	[SWS_Dem_00267] [SWS_Dem_00268]
[SRS_BSW_00399]	Parameter-sets shall be located in a separate segment and shall be loaded after the code	[SWS_Dem_00267] [SWS_Dem_00268]
[SRS_BSW_00400]	Parameter shall be selected from multiple sets of parameters after code has been loaded and started	[SWS_Dem_00267] [SWS_Dem_00268]
[SRS_BSW_00401]	Documentation of multiple instances of configuration parameters shall be available	[SWS_Dem_00267] [SWS_Dem_00268]
[SRS_BSW_00404]	BSW Modules shall support post-build configuration	[SWS_Dem_00267] [SWS_Dem_00268]

Requirement	Description	Satisfied by
[SRS_BSW_00405]	BSW Modules shall support multiple configuration sets	[SWS_Dem_00267] [SWS_Dem_00268]
[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_Dem_00124] [SWS_Dem_00169] [SWS_Dem_00364]
[SRS_BSW_00417]	Software which is not part of the SW-C shall report error events only after the DEM is fully operational.	[SWS_Dem_00107]
[SRS_BSW_00420]	No description	[SWS_Dem_00107]
[SRS_BSW_00421]	No description	[SWS_Dem_00107]
[SRS_BSW_00447]	Standardizing Include file structure of BSW Modules Implementing Autosar Service	[SWS_Dem_00151]
[SRS_Diag_04000]	The DEM and DCM shall support the Diagnostic Standard UDS (ISO14229-1)	[SWS_Dem_00013] [SWS_Dem_00061] [SWS_Dem_00080] [SWS_Dem_00645] [SWS_Dem_00686]
[SRS_Diag_04001]	The DEM and DCM shall support the Diagnostic Standard OBD (ISO15031-5)	[SWS_Dem_00299] [SWS_Dem_00359] [SWS_Dem_00645] [SWS_Dem_00697] [SWS_Dem_00745] [SWS_Dem_00753] [SWS_Dem_00763]
[SRS_Diag_04002]	The Diagnostic event (fault) management shall be established as Basic SW Module	[SWS_Dem_00126]
[SRS_Diag_04010]	The DEM module and DCM module shall ensure interaction in order to fulfill ISO 14229-1 and ISO 15031-5	[SWS_Dem_00060] [SWS_Dem_00079] [SWS_Dem_00204] [SWS_Dem_00210] [SWS_Dem_00216] [SWS_Dem_00228] [SWS_Dem_00264] [SWS_Dem_00277] [SWS_Dem_00415] [SWS_Dem_00417] [SWS_Dem_00439] [SWS_Dem_00575] [SWS_Dem_00584] [SWS_Dem_00645] [SWS_Dem_00657] [SWS_Dem_00671] [SWS_Dem_00827]
[SRS_Diag_04024]	The DCM module shall be able to access and handle specific data elements and data element groups if requested by an external scan tool	[SWS_Dem_00479]
[SRS_Diag_04030]	The DEM shall provide an interface via the RTE to monitoring SW components for reporting and processing diagnostic test results	[SWS_Dem_00331] [SWS_Dem_00603] [SWS_Dem_00608] [SWS_Dem_00744] [SWS_Dem_00849] [SWS_Dem_00850]

Requirement	Description	Satisfied by
[SRS_Diag_04031]	The DEM shall notify the Function Inhibition Manager (FIM) upon changes of the event status in order to process them according to the SW components dependencies	[SWS_Dem_00029] [SWS_Dem_00658]
[SRS_Diag_04057]	The DEM shall support a classification of events for series production, OBD and expert usage	[SWS_Dem_00057] [SWS_Dem_00518] [SWS_Dem_00593] [SWS_Dem_00649] [SWS_Dem_00906] [SWS_Dem_00988]
[SRS_Diag_04058]	The DCM module shall be able to access different event memories provided by the DEM module	[SWS_Dem_00171]
[SRS_Diag_04061]	The DEM shall provide mechanisms to distinguish between the reported fault of different applications and basic software modules	[SWS_Dem_00153] [SWS_Dem_00602] [SWS_Dem_00616] [SWS_Dem_00621] [SWS_Dem_00745]
[SRS_Diag_04063]	The DEM module shall process a dedicated event identifier (EventId) for each monitoring path to support an autonomous handling of different events/faults	[SWS_Dem_00153] [SWS_Dem_00154]
[SRS_Diag_04065]	The DEM and DCM shall be able to remove a specific event or event groups from the configured event memory	[SWS_Dem_00077] [SWS_Dem_00569] [SWS_Dem_00570] [SWS_Dem_00571] [SWS_Dem_00572] [SWS_Dem_00573] [SWS_Dem_00661] [SWS_Dem_00662] [SWS_Dem_00663] [SWS_Dem_00664] [SWS_Dem_00914] [SWS_Dem_01057] [SWS_Dem_01059] [SWS_Dem_01060] [SWS_Dem_01061]
[SRS_Diag_04066]	The DEM module shall provide different event memories which have to be configurable per event	[SWS_Dem_00010] [SWS_Dem_00162] [SWS_Dem_00212] [SWS_Dem_00236] [SWS_Dem_00238] [SWS_Dem_00239] [SWS_Dem_00240] [SWS_Dem_00241] [SWS_Dem_00548]
[SRS_Diag_04067]	The DCM and DEM shall provide the diagnostic status information according to ISO 14229-1	[SWS_Dem_00011] [SWS_Dem_00016] [SWS_Dem_00053] [SWS_Dem_00059] [SWS_Dem_00060] [SWS_Dem_00330] [SWS_Dem_00333] [SWS_Dem_00523] [SWS_Dem_00524] [SWS_Dem_00530] [SWS_Dem_00566] [SWS_Dem_00700]

Requirement	Description	Satisfied by
[SRS_Diag_04068]	The DEM module shall provide event specific debounce algorithms	[SWS_Dem_00019] [SWS_Dem_00204] [SWS_Dem_00264] [SWS_Dem_00343] [SWS_Dem_00413] [SWS_Dem_00414] [SWS_Dem_00416] [SWS_Dem_00417] [SWS_Dem_00418] [SWS_Dem_00419] [SWS_Dem_00420] [SWS_Dem_00421] [SWS_Dem_00422] [SWS_Dem_00423] [SWS_Dem_00424] [SWS_Dem_00425] [SWS_Dem_00426] [SWS_Dem_00427] [SWS_Dem_00428] [SWS_Dem_00429] [SWS_Dem_00430] [SWS_Dem_00431] [SWS_Dem_00432] [SWS_Dem_00433] [SWS_Dem_00434] [SWS_Dem_00435] [SWS_Dem_00436] [SWS_Dem_00437] [SWS_Dem_00438] [SWS_Dem_00439] [SWS_Dem_00526] [SWS_Dem_00527] [SWS_Dem_00772] [SWS_Dem_00774] [SWS_Dem_00778] [SWS_Dem_00779] [SWS_Dem_00985]
[SRS_Diag_04069]	The DEM module shall provide event specific information of warning indicators requested by SW-Cs or other BSW modules	[SWS_Dem_00046] [SWS_Dem_00500] [SWS_Dem_00504] [SWS_Dem_00506] [SWS_Dem_00510] [SWS_Dem_00511] [SWS_Dem_00533] [SWS_Dem_00535] [SWS_Dem_00568] [SWS_Dem_00701] [SWS_Dem_00707] [SWS_Dem_00865] [SWS_Dem_00886] [SWS_Dem_00896] [SWS_Dem_00967]
[SRS_Diag_04070]	The DEM module shall process the order of the event occurrences in an appropriate and obvious manner	[SWS_Dem_00161] [SWS_Dem_00219] [SWS_Dem_00221] [SWS_Dem_00410] [SWS_Dem_00412] [SWS_Dem_00787] [SWS_Dem_00852]
[SRS_Diag_04071]	The DEM module shall process events according to their defined importance like priority and/or severity	[SWS_Dem_00382] [SWS_Dem_00383] [SWS_Dem_00692]
[SRS_Diag_04073]	DEM shall process combined events which shall consist of several different events	[SWS_Dem_00024] [SWS_Dem_00442] [SWS_Dem_00538] [SWS_Dem_00539] [SWS_Dem_00541]
[SRS_Diag_04074]	The DEM module shall process event related data (e.g. freeze frames and extended data records)	[SWS_Dem_00039] [SWS_Dem_00040] [SWS_Dem_00070] [SWS_Dem_00071] [SWS_Dem_00074] [SWS_Dem_00075] [SWS_Dem_00076] [SWS_Dem_00189] [SWS_Dem_00225] [SWS_Dem_00271] [SWS_Dem_00464] [SWS_Dem_00465] [SWS_Dem_00576] [SWS_Dem_00582] [SWS_Dem_00585] [SWS_Dem_00630] [SWS_Dem_00631] [SWS_Dem_00775] [SWS_Dem_00796] [SWS_Dem_00807] [SWS_Dem_00969] [SWS_Dem_00995] [SWS_Dem_00997]

Requirement	Description	Satisfied by
[SRS_Diag_04076]	The DEM module shall provide a set of system cycles that may qualify the event in an additional manner	[SWS_Dem_00019] [SWS_Dem_00338] [SWS_Dem_00480] [SWS_Dem_00481] [SWS_Dem_00482] [SWS_Dem_00483] [SWS_Dem_00484] [SWS_Dem_00577] [SWS_Dem_00601] [SWS_Dem_00602] [SWS_Dem_00639] [SWS_Dem_00641] [SWS_Dem_00642] [SWS_Dem_00644] [SWS_Dem_00673] [SWS_Dem_00693] [SWS_Dem_00698] [SWS_Dem_00773] [SWS_Dem_00825] [SWS_Dem_00826] [SWS_Dem_00968] [SWS_Dem_00985]
[SRS_Diag_04077]	The DEM uses standard mechanisms provided by NVRAM-Manager	[SWS_Dem_00164] [SWS_Dem_00329]
[SRS_Diag_04079]	The size of a FreezeFrame shall be reported to the DCM by the DEM	[SWS_Dem_00074] [SWS_Dem_00650] [SWS_Dem_00822]
[SRS_Diag_04082]	The diagnostic modules DCM and DEM shall provide standardized interfaces to support OBD services as defined in ISO15031-5 and SAE J1979	[SWS_Dem_00291] [SWS_Dem_00293] [SWS_Dem_00294] [SWS_Dem_00296] [SWS_Dem_00297] [SWS_Dem_00298] [SWS_Dem_00300] [SWS_Dem_00301] [SWS_Dem_00304] [SWS_Dem_00308] [SWS_Dem_00346] [SWS_Dem_00347] [SWS_Dem_00348] [SWS_Dem_00349] [SWS_Dem_00351] [SWS_Dem_00352] [SWS_Dem_00354] [SWS_Dem_00355] [SWS_Dem_00356] [SWS_Dem_00357] [SWS_Dem_00358] [SWS_Dem_00360] [SWS_Dem_00361] [SWS_Dem_00362] [SWS_Dem_00377] [SWS_Dem_00378] [SWS_Dem_00528] [SWS_Dem_00575] [SWS_Dem_00590] [SWS_Dem_00596] [SWS_Dem_00597] [SWS_Dem_00623] [SWS_Dem_00703] [SWS_Dem_00704] [SWS_Dem_00705] [SWS_Dem_00706] [SWS_Dem_00708] [SWS_Dem_00709] [SWS_Dem_00710] [SWS_Dem_00712] [SWS_Dem_00715] [SWS_Dem_00717] [SWS_Dem_00718] [SWS_Dem_00719] [SWS_Dem_00721] [SWS_Dem_00722] [SWS_Dem_00723] [SWS_Dem_00728] [SWS_Dem_00746] [SWS_Dem_00748] [SWS_Dem_00754] [SWS_Dem_00755] [SWS_Dem_00757] [SWS_Dem_00762] [SWS_Dem_00764] [SWS_Dem_00966]
[SRS_Diag_04085]	The Development Error Tracer shall provide an interface to receive error reports	[SWS_Dem_00463]
[SRS_Diag_04092]	Control of event handling	[SWS_Dem_00514] [SWS_Dem_00515] [SWS_Dem_00516] [SWS_Dem_00667] [SWS_Dem_00668] [SWS_Dem_00669]
[SRS_Diag_04093]	Memory Overflow indication	[SWS_Dem_00397] [SWS_Dem_00398] [SWS_Dem_00399] [SWS_Dem_00559]

Requirement	Description	Satisfied by
[SRS_Diag_04095]	The DEM module shall provide the ability to handle event specific enable and storage conditions	[SWS_Dem_00202] [SWS_Dem_00446] [SWS_Dem_00447] [SWS_Dem_00449] [SWS_Dem_00450] [SWS_Dem_00453] [SWS_Dem_00455] [SWS_Dem_00458] [SWS_Dem_00459] [SWS_Dem_00543] [SWS_Dem_00591] [SWS_Dem_00605]
[SRS_Diag_04096]	The DEM module shall support the UDS DTC status bit support & handling according to ISO 14229-1	[SWS_Dem_00006] [SWS_Dem_00036] [SWS_Dem_00053] [SWS_Dem_00333] [SWS_Dem_00379] [SWS_Dem_00385] [SWS_Dem_00386] [SWS_Dem_00387] [SWS_Dem_00388] [SWS_Dem_00389] [SWS_Dem_00390] [SWS_Dem_00391] [SWS_Dem_00392] [SWS_Dem_00393] [SWS_Dem_00394] [SWS_Dem_00395] [SWS_Dem_00421] [SWS_Dem_00431] [SWS_Dem_00498] [SWS_Dem_00525] [SWS_Dem_00539] [SWS_Dem_00823] [SWS_Dem_00824]
[SRS_Diag_04099]	The DEM should forward incoming events to the DLT interface	[SWS_Dem_00517] [SWS_Dem_00632] [SWS_Dem_00633] [SWS_Dem_00634] [SWS_Dem_00635] [SWS_Dem_00636] [SWS_Dem_00637] [SWS_Dem_00992] [SWS_Dem_00993]
[SRS_Diag_04102]	The DEM module shall provide a chronological reporting order of the events located in the configured event memory	[SWS_Dem_00411] [SWS_Dem_00412] [SWS_Dem_00477] [SWS_Dem_00695] [SWS_Dem_00696]
[SRS_Diag_04104]	The DEM module shall support a signal based configuration of event related data (define freeze frames and extended data records)	[SWS_Dem_00469] [SWS_Dem_00779] [SWS_Dem_00821] [SWS_Dem_00995]
[SRS_Diag_04105]	Event memory management	[SWS_Dem_00580] [SWS_Dem_00607] [SWS_Dem_00783] [SWS_Dem_00784] [SWS_Dem_00785] [SWS_Dem_00786] [SWS_Dem_00829] [SWS_Dem_00922] [SWS_Dem_00923]
[SRS_Diag_04106]	The DEM module shall provide an event specific set of configurable debouncing algorithms	[SWS_Dem_00019] [SWS_Dem_00344] [SWS_Dem_00418] [SWS_Dem_00419] [SWS_Dem_00420] [SWS_Dem_00421] [SWS_Dem_00422] [SWS_Dem_00423] [SWS_Dem_00424] [SWS_Dem_00425] [SWS_Dem_00432] [SWS_Dem_00433] [SWS_Dem_00434] [SWS_Dem_00438] [SWS_Dem_00643] [SWS_Dem_00818] [SWS_Dem_00985]
[SRS_Diag_04107]	Defensive behavior of the DEM module	[SWS_Dem_00339]

Requirement	Description	Satisfied by
[SRS_Diag_04109]	The DEM module shall provide an interface to retrieve the number of event memory entries	[SWS_Dem_00651] [SWS_Dem_00652]
[SRS_Diag_04110]	The DEM module shall support SAE J1979-73 lamp status	[SWS_Dem_00858] [SWS_Dem_00859] [SWS_Dem_00860] [SWS_Dem_00861] [SWS_Dem_00862] [SWS_Dem_00863] [SWS_Dem_00866] [SWS_Dem_00868] [SWS_Dem_00872] [SWS_Dem_00873] [SWS_Dem_00883] [SWS_Dem_00884] [SWS_Dem_00885] [SWS_Dem_00887] [SWS_Dem_00889] [SWS_Dem_00893] [SWS_Dem_00894] [SWS_Dem_00895] [SWS_Dem_00897]
[SRS_Diag_04111]	The DEM module shall support SAE J1979 Expanded- / FreezeFrame	[SWS_Dem_00877] [SWS_Dem_00899] [SWS_Dem_00900] [SWS_Dem_00901] [SWS_Dem_00902] [SWS_Dem_00903] [SWS_Dem_00904] [SWS_Dem_00905] [SWS_Dem_00906] [SWS_Dem_00907]
[SRS_Diag_04112]	The DEM module shall support DTCs according to SAE J1979-73	[SWS_Dem_00645] [SWS_Dem_00845] [SWS_Dem_00856] [SWS_Dem_00864] [SWS_Dem_00874]
[SRS_Diag_04113]	The DEM module shall support a set of SAE J1979-73 DM-messages	[SWS_Dem_00880] [SWS_Dem_00881] [SWS_Dem_00882] [SWS_Dem_00913] [SWS_Dem_00921]
[SRS_Diag_04115]	The optional parameter DTCSettingControlOption-Record as part of UDS service ControlDTCSetting shall be limited to GroupOfDTC	[SWS_Dem_00080] [SWS_Dem_00626]
[SRS_Diag_04117]	The DEM shall provide a configurable behavior for the deletion of DTC	[SWS_Dem_00343] [SWS_Dem_00620] [SWS_Dem_00670] [SWS_Dem_00679] [SWS_Dem_00680] [SWS_Dem_00879]
[SRS_Diag_04118]	The DEM shall optionally support event displacement	[SWS_Dem_00382] [SWS_Dem_00383] [SWS_Dem_00400] [SWS_Dem_00401] [SWS_Dem_00402] [SWS_Dem_00403] [SWS_Dem_00404] [SWS_Dem_00405] [SWS_Dem_00406] [SWS_Dem_00407] [SWS_Dem_00408] [SWS_Dem_00409] [SWS_Dem_00443] [SWS_Dem_00692] [SWS_Dem_00694]
[SRS_Diag_04122]	The ClearDTC command in DEM shall be usable for a Complex Device Driver	[SWS_Dem_00514] [SWS_Dem_00515] [SWS_Dem_00516] [SWS_Dem_00569] [SWS_Dem_00570] [SWS_Dem_00571] [SWS_Dem_00572] [SWS_Dem_00573] [SWS_Dem_00659] [SWS_Dem_00660] [SWS_Dem_00661] [SWS_Dem_00662] [SWS_Dem_00663] [SWS_Dem_00664] [SWS_Dem_00665] [SWS_Dem_00667] [SWS_Dem_00668] [SWS_Dem_00669]

Requirement	Description	Satisfied by
[SRS_Diag_04123]	The DEM module shall support harmonized Driving-/WarmUp cycles	[SWS_Dem_00699]
[SRS_Diag_04124]	The DEM shall be able to store the current debounce counter value non-volatile to over a power-down cycle	[SWS_Dem_00674] [SWS_Dem_00675] [SWS_Dem_00676] [SWS_Dem_00782]
[SRS_Diag_04125]	The behaviour of the event debounce counter shall be configurable	[SWS_Dem_00654] [SWS_Dem_00655] [SWS_Dem_00656] [SWS_Dem_00677] [SWS_Dem_00678] [SWS_Dem_00681] [SWS_Dem_00682] [SWS_Dem_00776] [SWS_Dem_00788] [SWS_Dem_00789] [SWS_Dem_00790] [SWS_Dem_00791] [SWS_Dem_00792] [SWS_Dem_00793] [SWS_Dem_00794] [SWS_Dem_00795]
[SRS_Diag_04126]	Configurable suppression of events	[SWS_Dem_00586] [SWS_Dem_00587] [SWS_Dem_00687] [SWS_Dem_00688] [SWS_Dem_00690] [SWS_Dem_00691] [SWS_Dem_00917]
[SRS_Diag_04127]	Configurable record numbers and trigger options for DTCSnapshotRecords and DTCExtendedDataRecords	[SWS_Dem_00334] [SWS_Dem_00337] [SWS_Dem_00461] [SWS_Dem_00471] [SWS_Dem_00581] [SWS_Dem_00797] [SWS_Dem_00798] [SWS_Dem_00799] [SWS_Dem_00800] [SWS_Dem_00801] [SWS_Dem_00802] [SWS_Dem_00803] [SWS_Dem_00804] [SWS_Dem_00805] [SWS_Dem_00806] [SWS_Dem_00808] [SWS_Dem_00809] [SWS_Dem_00810] [SWS_Dem_00811] [SWS_Dem_00812] [SWS_Dem_00813] [SWS_Dem_00814] [SWS_Dem_00815] [SWS_Dem_00816] [SWS_Dem_00817] [SWS_Dem_00819] [SWS_Dem_00820] [SWS_Dem_00991]
[SRS_Diag_04128]	DEM interface to set/reset the WarningIndicatorRequested bit	[SWS_Dem_00606] [SWS_Dem_00749] [SWS_Dem_00831] [SWS_Dem_00832] [SWS_Dem_00833] [SWS_Dem_00834] [SWS_Dem_00835] [SWS_Dem_00836]
[SRS_Diag_04129]	The DCM and DEM shall provide OBD-specific configuration capabilities	[SWS_Dem_00700] [SWS_Dem_00702] [SWS_Dem_00752]
[SRS_Diag_04130]	The DEM shall provide the capability to process a new request	[SWS_Dem_01042]

7 Functional specification

The Diagnostic Event Manager (Dem) handles and stores the events detected by diagnostic monitors in both Software Components (SW-Cs) and Basic software (BSW) modules. The stored event information is available via an interface to other BSW modules or SW-Cs.

Figure 7.1 shows the Dem configuration. DemGeneral contains the global part of the configuration and DemConfigSet contains the multiple configuration part.

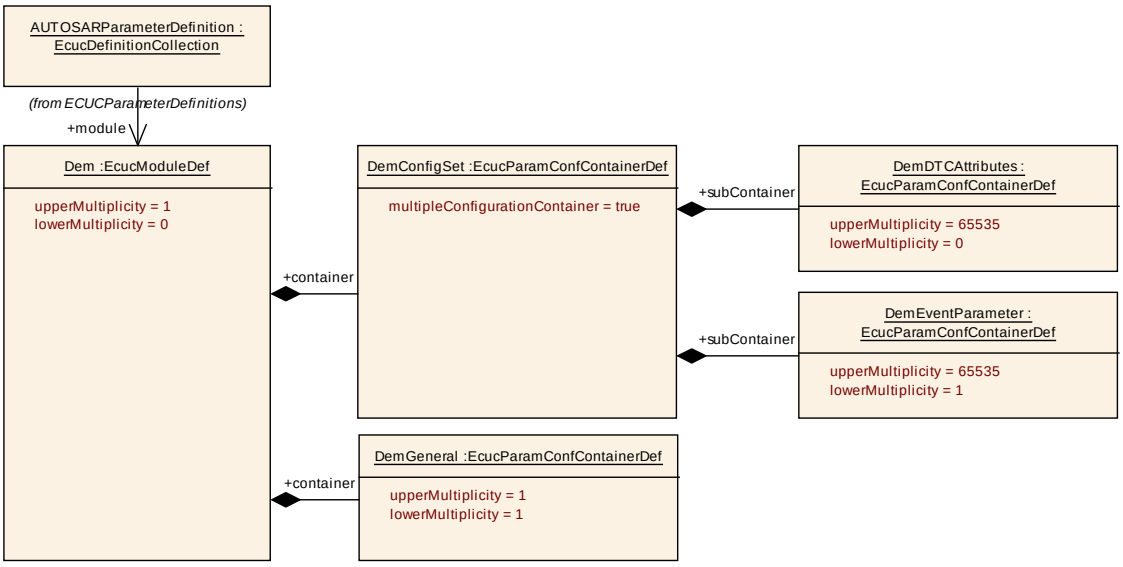

Figure 7.1: Top-level view of Dem configuration

7.1 Diagnostic event definition

A ‘Diagnostic Event’ defines the atomic unit that can be handled by the Dem module. The status of a ‘Diagnostic Event’ represents the result of a monitor (refer to chapter 7.1.6). The Dem receives the result of a monitor from SW-C via the RTE or other BSW modules.

The Dem module uses the EventId to manage the status of the ‘Diagnostic Event’ of a system and performs the required actions for individual test results, e.g. stores the freeze frame.

[SWS_Dem_00153] ⌈ The Dem module shall represent each Diagnostic Event by an EventId and the related EventName. ⌋*(SRS_Diag_04061, SRS_Diag_04063)*

All monitors and BSW modules use the EventId as a symbolic EventName. The Dem configuration tool replaces the symbolic names by numbers.

[SWS_Dem_00154] ⌈ The EventId and the related EventName shall be unique per Dem module represented by the ECU configuration (refer to [SWS_Dem_00126]). ⌋*(SRS_Diag_04063)*

The Dem is not designed to be able to handle the case where more than one monitor shares a single EventId.

The Dem module may use an internal event status. However, when requested by the Dcm, the extended event status will be reported.

The Dem module supports several event-specific configuration parameters as shown in the following figures. For a detailed description, refer to chapter 10 Configuration specification.

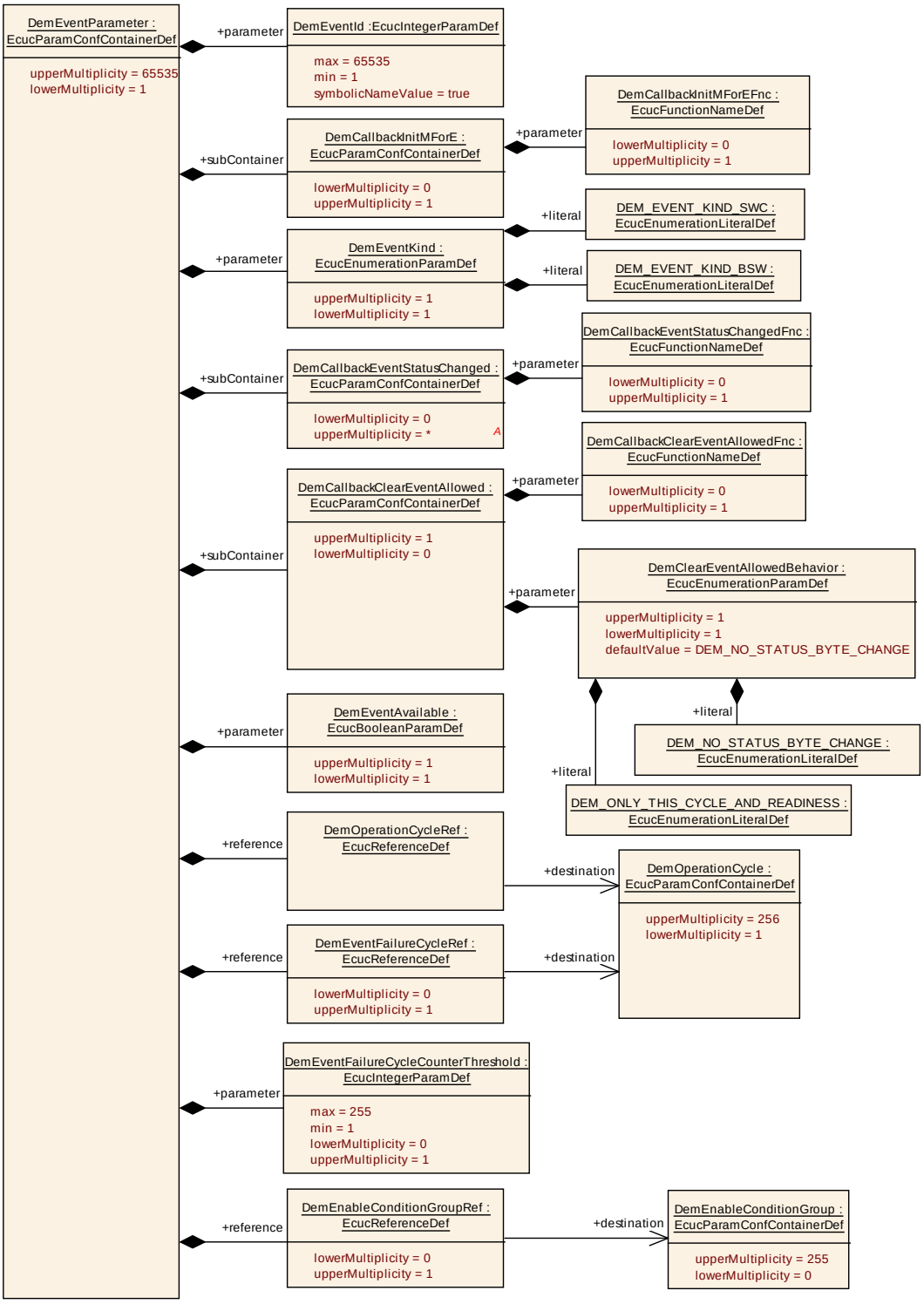

Figure 7.2: Event parameter configuration (part 1)

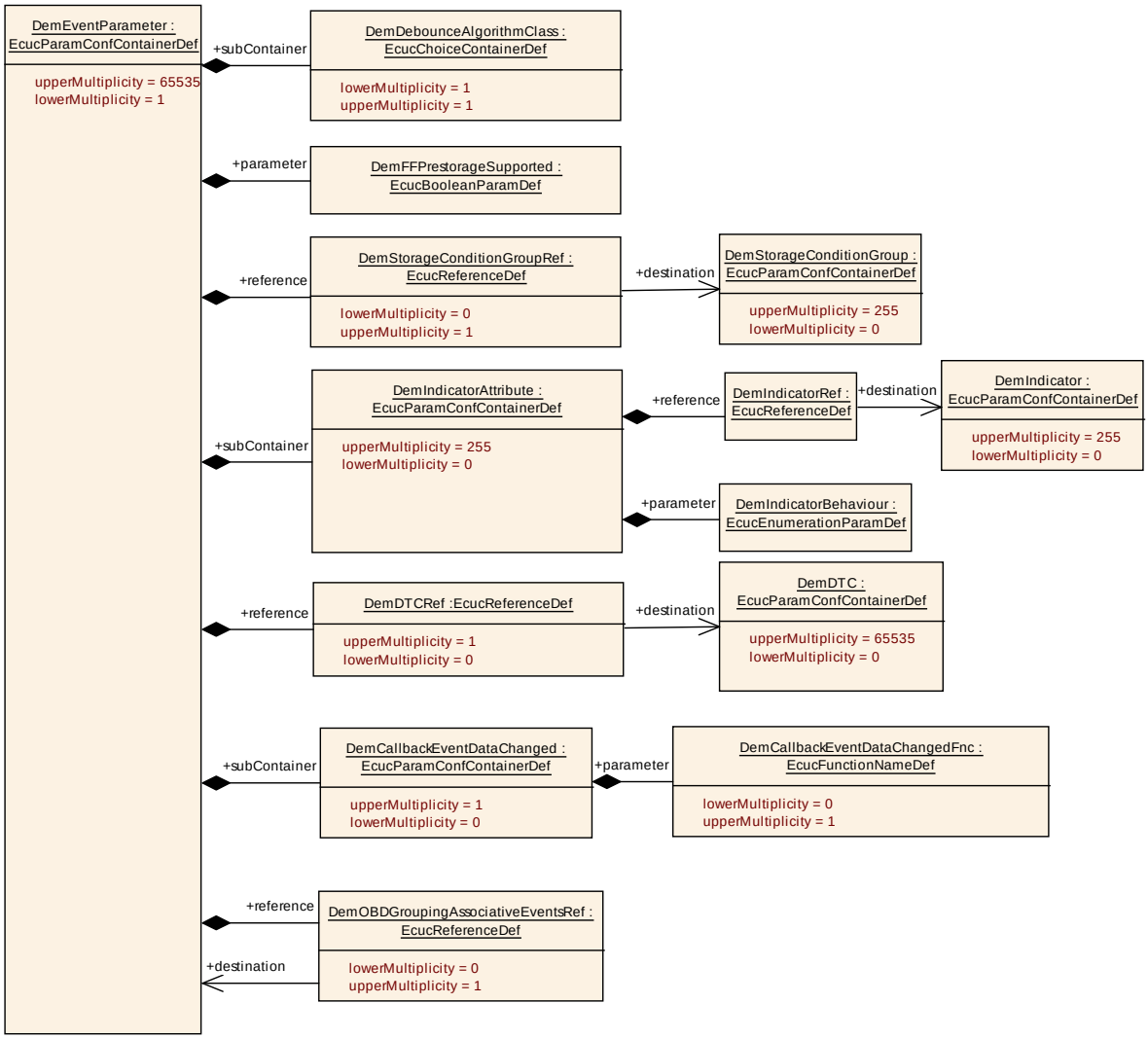

Figure 7.3: Event parameter configuration (part 2)

7.1.1 Event priority

Event priority is defined as a ranking of events based upon level of importance. It is used to determine which fault entries may be removed from the event memory in case the number of stored events exceeds the maximum number of memory entries (event memory is full).

[SWS_Dem_00382] ⌈ Each supported event shall have a priority assigned to it (refer to parameter DemEventPriority in DemDTCAttributes). ⌋*(SRS_Diag_04118, SRS_Diag_04071)*

[SWS_Dem_00383] ⌈ A priority value of 1 shall be the highest priority. Larger priority value shall define lower importance. ⌋*(SRS_Diag_04118, SRS_Diag_04071)*

7.1.2 Event occurrence

[SWS_Dem_00011] ⌈ The Dem module shall provide an occurrence counter per event memory entry. ⌋*(SRS_Diag_04067)*

[SWS_Dem_00523] ⌈ The Dem module shall initialize the occurrence counter with the value one if the related event is entered in the respective event memory. ⌋*(SRS_Diag_04067)*

[SWS_Dem_00524] ⌈ If the configuration parameter DemOccurrenceCounterProcessing (refer to DemGeneral) is DEM_PROCESS_OCCCTR_TF , the Dem module shall increment the occurrence counter by one, triggered by each UDS DTC status bit 0 (TestFailed) transition from 0 to 1, if the related event is already stored in the event memory. ⌋*(SRS_Diag_04067)*

[SWS_Dem_00580] ⌈ If the configuration parameter DemOccurrenceCounterProcessing (refer to DemGeneral) is DEM_PROCESS_OCCCTR_CDTC, the Dem module shall increment the occurrence counter by one, triggered by each UDS DTC status bit 0 (TestFailed) transition from 0 to 1, if the related event is already stored in the event memory and the UDS DTC Status bit 3 (ConfirmedDTC) is equal to 1 (refer to chapter 0). ⌋*(SRS_Diag_04105)*

[SWS_Dem_00625] ⌈ The Dem module shall not increment the event-specific occurrence counter anymore, if it has reached its maximum value (255, refer to [SWS_Dem_00471]). ⌋

7.1.3 Event kind

There are two different types of events:

- BSW-related events (reported via C-API - Dem_ReportErrorStatus)
- SW-C-related events (reported via RTE operation - SetEventStatus)

This kind is configurable per event (refer to DemEventKind in DemEventParameter).

This is necessary because BSW-events may be reported prior to full Dem initialization and need to be buffered (refer to chapter 7.6).

7.1.4 Event significance

There are two different significance levels of events:

- fault: classifies a failure, which relates to the component/ECU itself (and requires for example a repair action)

- occurrence: classifies an issue, which indicates additional information concerning insufficient system behavior (and relates for example to a condition out of the ECU's control)

This significance level is configurable per event (refer to DemEventSignificance in DemDTCAttributes) and can be mapped as a data element (refer to chapter 7.3.7.4, DEM_SIGNIFICANCE).

7.1.5 Event destination

The configuration parameter DemMemoryDestination (refer to DemDTCAttributes) defines the dedicated storage location(s) of the event and its related data (refer to chapter 7.3.7).

The "permanent event memory" assignment is implicitly derived from the related DTC kind (refer to chapter 7.2.1). Emission-related events are automatically assigned to the permanent event memory, since the storage of an event as "permanent DTC" is dynamically derived from its current status (handling is described in chapter 7.7.1.4). In this context the term "permanent" relates to an attribute of emission-related events and does not relate only to persistent storage via NvM, which is done for each event memory type anyway.

The definition and use of the different memory types is OEM specific.

For the Dcm-Dem interface the parameter DTCOrigin is used to distinguish between the different memory areas. The intention is to allow specific operations on the different memory areas (primary, secondary, permanent and mirror memory).

7.1.6 Diagnostic monitor definition

A diagnostic monitor is a routine entity determining the proper functionality of a component. This monitoring function identifies a specific fault type (e.g. short to ground, open load, etc.) for a monitoring path. A monitoring path represents the physical system or a circuit, that is being monitored (e.g. sensor input). Each monitoring path is associated with exactly one diagnostic event.

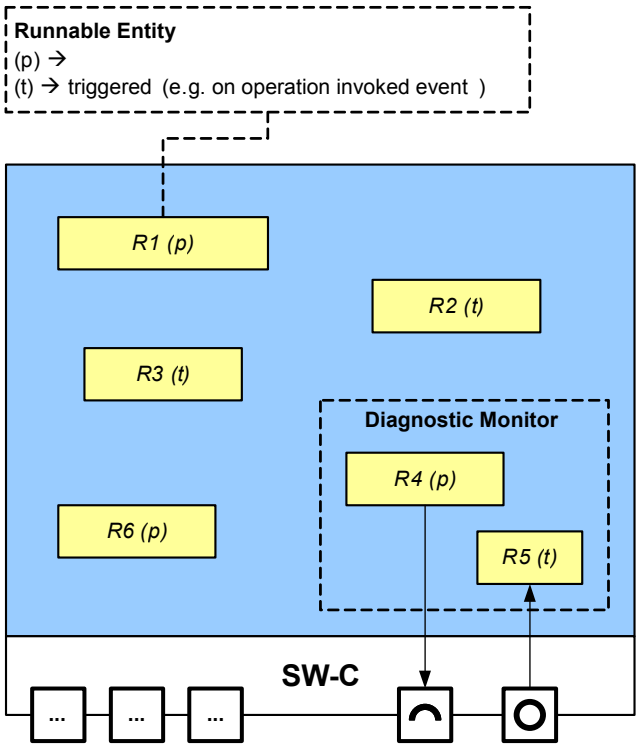

Figure 7.4: Example for a monitor embedded within a SW-C

If the monitor debounces on its own, the reporting API is called only after a qualified result (passed or failed) is available. A report is necessary at least if the result changes. However, usually it is computationally more efficient for the monitor to always call the Dem and should therefore be preferred. Hence, it is implementation specific for the Dem to deal with reports of unchanged results.

If the monitor uses the Dem-internal debouncing mechanism (refer to chapter 7.3.3), the reporting API is called whenever the code with the functional check is executed.

7.2 Diagnostic trouble code definition

A ‘Diagnostic trouble code’ defines a unique identifier (shown to the diagnostic tester) mapped to a ‘Diagnostic event’ of the Dem module. The Dem provides the status of ‘Diagnostic trouble codes’ to the Dcm module (refer to chapter 7.9.2).

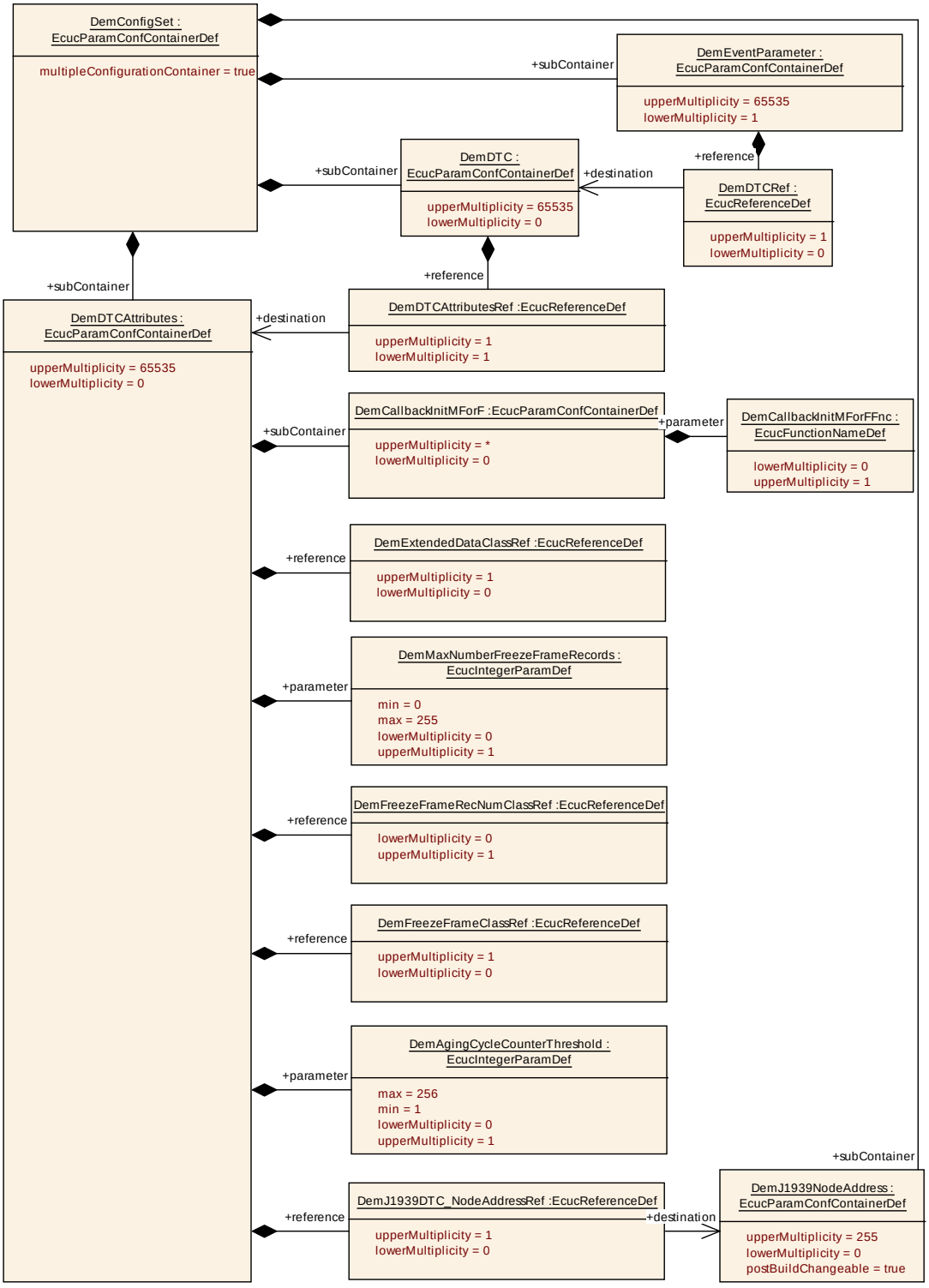

Figure 7.5: DTC configuration (part I)

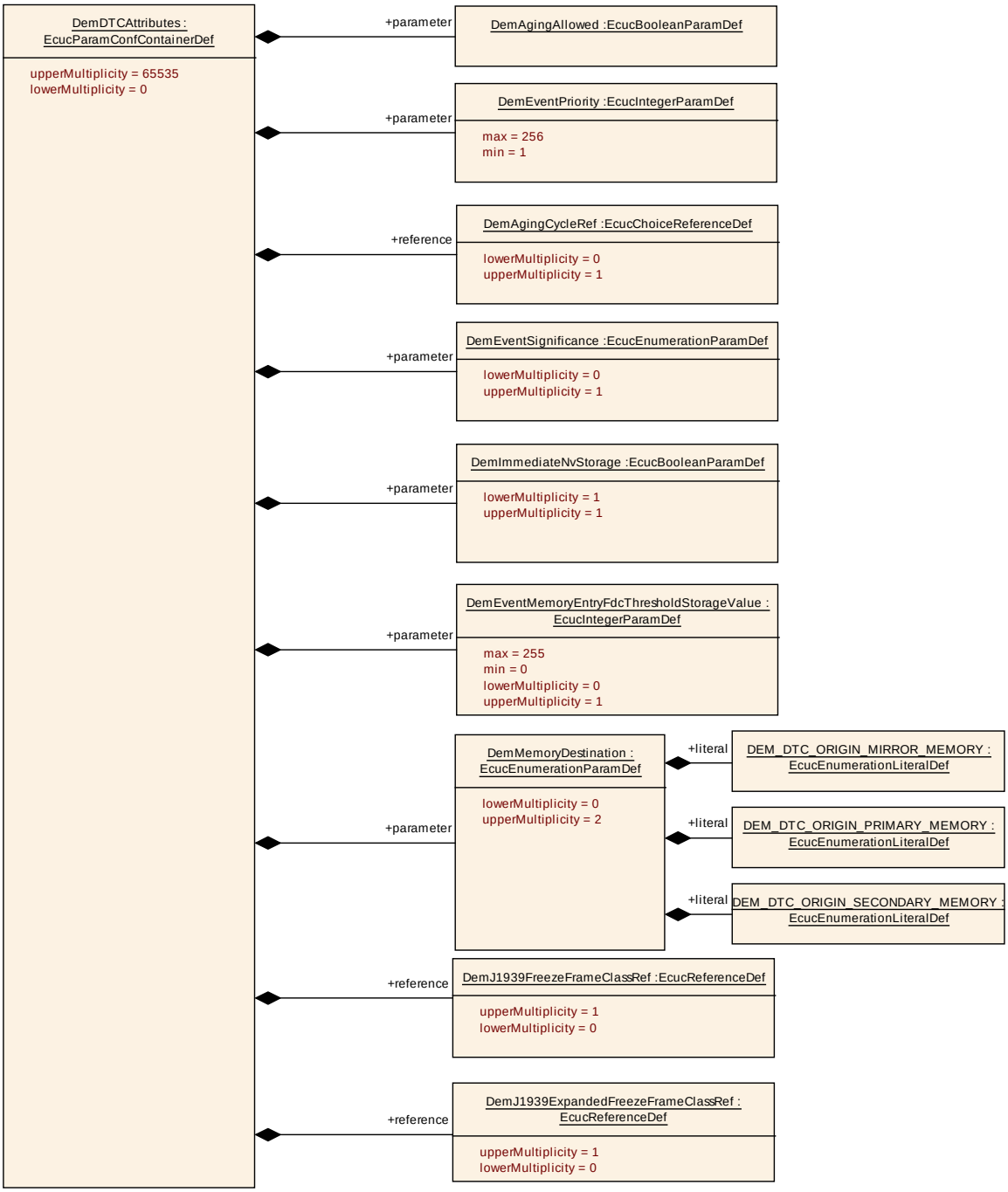

Figure 7.6: DTC configuration (part II)

7.2.1 DTC kind

There are two different kinds of DTCs:

- non OBD-relevant DTCs (UDS DTCs)
- OBD-relevant DTCs

This kind is derived implicitly from the DemDTCAttributes configuration. If the parameter DemObdDTC exists, the DTC and all releated events are OBD-relevant.

Note: The DTC kind (refer to chapter 8.2.1.9) is relevant for the selection of several DTCs used by the diagnostic service ReadDTCInformation and some OBD services (0x19/$03/$07/$0A, refer to Dem_DcmSetDTCFilter), as well as for the diagnostic service ControlDTCSetting (0x85, refer to Dem_DcmDisableDTCSetting and Dem_DcmEnableDTCSetting).

7.2.2 DTC format

[SWS_Dem_00013] ⌈ The Dem module shall support DTC formats for DemDTC according to:

- ISO 14229-1
- ISO 15031-6
- SAE J1939-73
- ISO 11992-4

⌋*(SRS_Diag_04000)*

The configuration parameter DemTypeOfDTCSupported (refer to DemGeneral) is used to select one of the supported DTC formats defined in [SWS_Dem_00013] of the ECU (refer to Dem_DcmGetTranslationType). This is required to determine the DTC format value to be reported for ISO 14229-1 service Read DTC Information (0x19).

[SWS_Dem_00645] ⌈ The Dem module shall support different DTC numbers for UDS, OBD, and J1939 DTCs based on separate configuration parameters (refer to parameters DemDTC and DemObdDTC in DemConfigSet). ⌋*(SRS_Diag_04010, SRS_Diag_04000, SRS_Diag_04001, SRS_Diag_04112)*

A DTC can have any combination of the three formats (UDS, OBD, and J1939), i.e. one, two, or three formats at the same time. The Dem will therefore handle three DTC value lists internally. The reported format depends on the Dem_DTCFormatType (refer to chapter 8.2.1.10) or is defined by the context of the related API.

[SWS_Dem_00277] ⌈ The Dem shall report DTC values as a uint32 with byte 0 = LowByte, byte 1 = MiddleByte, byte 2 = HighByte and byte 3 is unused. For OBD DTC format there are only two bytes (HighByte, LowByte) used. The Dem services shall report these DTCs as a uint32 with byte 1 = LowByte, byte 2 = HighByte, byte 3 is unused and byte 0 = 0x00. ⌋*(SRS_Diag_04010)*

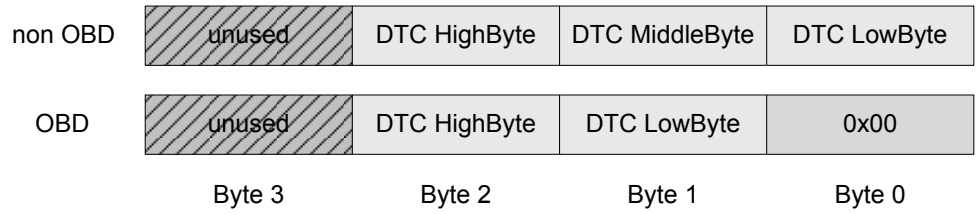

Figure 7.7: DTC Byte Order

[SWS_Dem_00921] ⌈ The SPN of the J1939DTC shall be represented as Intel for all 19 bits in conjunction with the SPN Conversion Method Version 4 (CM = 0). ⌋*(SRS_Diag_04113)*

Refer chapter 5.7.1.11 SPN Conversion Method in [11] for details.

[SWS_Dem_00269] ⌈ The function Dem_GetDTCOfEvent (refer to chapter 8.3.3.16) shall get the DTC which is mapped to EventId by the Dem configuration. ⌋

7.2.3 DTC groups

In addition to single DTC values, groups of DTCs can be configured (refer to DemGroupOfDTC), as defined by ISO 14229-1 [2] - Annex D.1. Each DTC group has its own DTC group value assigned (which must be unique to any other DTC and DTC group value).

[SWS_Dem_01059] ⌈ The DemGroupOfDTC shall represent the values of the DTC Group boundaries. The global boundaries 0x00000100 and 0x00fffe00 shall not be configured. ⌋*(SRS_Diag_04065)*

[SWS_Dem_01060] ⌈ The Dem shall use on requests with DTC groups like ClearDTC and Disable/Enable DTC the higher boundary of a DTCGroup value to identify the DTC group. ⌋*(SRS_Diag_04065)*

[SWS_Dem_01061] ⌈ All the DTCs in the range between the dtc-code of the requested group and the next higher dtcgroup-code shall be treated as belonging to the group. ⌋*(SRS_Diag_04065)*

Note: DTC groups are relevant for the diagnostic service ClearDiagnosticInformation (0x14, refer to Dem_DcmClearDTC), as well as for the diagnostic service ControlDTCSetting (0x85, refer to Dem_DcmDisableDTCSetting and Dem_DcmEnableDTCSetting).

The following DTC groups are provided:

- powertrain DTC group (optional, configurable value)
- chassis DTC group (optional, configurable value)
- body DTC group (optional, configurable value)
- network communication DTC group (optional, configurable value)

- further user-defined DTC groups (optional, configurable value)
- ‘all DTCs’ DTC group (mandatory, fixed value = 0xFFFFFF)

Note, that the DTC group ‘all DTCs’ will not be configured in DemGroupOfDTC, because it has always to be provided by the Dem module.

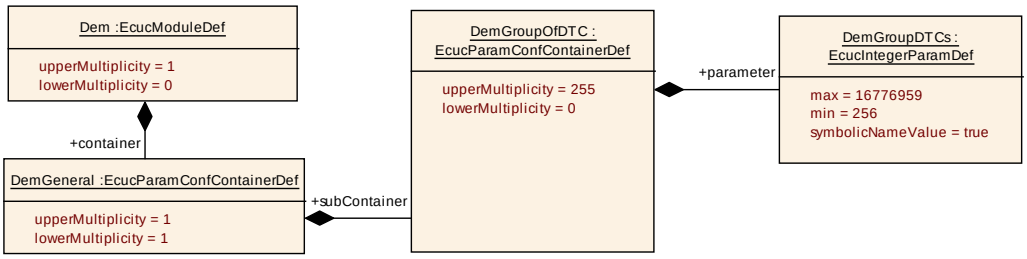

Figure 7.8: DTC group configuration

Taking into consideration that 0xFFFFFFFF is an unconfigurable value meaning “all DTCs group”, each of the DTCs configured for a group “X” let’s say, shall have their ids in the open interval (ID_of_group_X, ID_of_next_configured_group) e.g.:

powertrain DTC group –> 0xFFFFFF00	DTC1 –> 0xFFFFFF04 DTC2 –> 0xFFFFFF06 . . . DTCn –> 0xFFFFFF0E
body DTC group –> 0xFFFFFF0F	... DTCs in the body DTC group ...

As a configuration constraint of the above mentioned approach on DTC groups, if there is one or more DTC groups configured, there is no way to configure group independent DTCs.

7.2.4 DTC severity

[SWS_Dem_00033] ⌈ The Dem module shall provide a severity value per DTC (regarding the importance of the specific events) according to ISO 14229-1, Annex D.3 “DTCSeverityMask and DTCSeverity bit definitions” [2] (refer to Dem_DcmGetSeverityOfDTC paragraph 8.3.4.1.4 and Dem_DcmGetNextFilteredDTCAndSeverity paragraph 8.3.4.1.10), only if configured for at least one DTC. ⌋

The severity is only available for ISO 14229-1 DTCs. It is configurable per DTC optionally (refer to DemDTCSeverity in DemDTCAttributes).

Note: ISO 14229-1, Annex D defines the following severity levels: no severity available, Maintenance, Check at next Halt, Check immediately.

The function Dem_DcmSetDTCFilter (refer to [SWS_Dem_00057]) allows filtering for DTCs with severity information.

7.2.5 Functional unit

[SWS_Dem_00593] ⌈ The Dem module shall provide a functional unit value per DTC (refer to Dem_DcmGetFunctionalUnitOfDTC), only if configured for at least one DTC. ⌋*(SRS_Diag_04057)*

The functional unit value is configurable per DTC (refer to DemDTCFunctionalUnit in DemDTCAttributes).

Note: The functional unit values are required by the Dcm.

7.2.6 DTC suppression

[SWS_Dem_00586] ⌈ The Dem module shall provide the capability to enable or disable the suppression of DTCs dynamically (refer to chapter 8.3.3.26 Dem_SetDTCSuppression), if the configuration parameter DemSuppressionSupport (refer to DemGeneral) is set to DEM_DTC_SUPPRESSION, or DEM_EVENT_AND_DTC_SUPPRESSION. ⌋*(SRS_Diag_04126)*

[SWS_Dem_00587] ⌈ If the suppression for a specific DTC is enabled, this DTC shall be fully invisible. ⌋*(SRS_Diag_04126)*

Note: This means the (external) reporting (refer to chapter 7.9.2) of the disabled DTC is suppressed, but the event processing (refer to chapter 7.3) is not affected (e.g. for functional degradation).

[SWS_Dem_00588] ⌈ The Dem shall report a suppressed Event/DTC if an event memory entry exists. ⌋

Note: If there was a malfunction stored in the event memory at the time DTC/event suppression is activated this malfunction is not hide to the tester. If an event shall not reenter the event memory entry during it is suppressed. The Monitor should not report this event or enable conditions should de used..

[SWS_Dem_00914] ⌈ The Dem module shall provide the capability to enable or disable the suppression of DTC via events dynamically (refer to chapter 8.3.3.27 Dem_SetEventSuppression), if the configuration parameter DemSuppressionSupport (refer to DemGeneral) is set to DEM_EVENT_SUPPRESSION, or DEM_EVENT_AND_DTC_SUPPRESSION. ⌋*(SRS_Diag_04065)*

[SWS_Dem_00915] ⌈ Dem shall suppress a DTC respond if all related Events to this DTC are suppressed. ⌋

Note: If there is a one to one repationship between DTC and Event, the DTC is suppressed if the related Event is suppressed. If the DTC is a combined DTC the DTC shall be suppressed if all combined DTCs are suppressed.

[SWS_Dem_00916] ⌈ Suppression of DTCs and events shall be treated separately (suppressing event and dtc first and afterwards re-enable the DTC will not automatically reenable the event). ⌋

[SWS_Dem_00917] ⌈ If DemSuppressionSupport is set to DEM_EVENT_AND_DTC_SUPPRESSION the DTC suppression and the suppression via Event shall be an OR combination. ⌋*(SRS_Diag_04126)*

7.2.7 Events Suppression

[SWS_Dem_00687] ⌈ The Dem shall provide means to suppress events. Suppressed events are treated as if they do not exist. Their UDS status byte shall not be updated and stay at 0 (not failed, tested). ⌋*(SRS_Diag_04126)*

Note: Suppressed events shall not be considered for computation of service $01 PID $41. They shall not be reported to Dcm (Dem_DcmSetDTCFilter). FiM may ignore suppressed events for computation of fault reactions (though it is considered sufficient to process the event status 0byte 0x00). IUMPR Ratios refering to a suppressed event shall neither be computed nor reported. DTRs refering to the event shall neither be computed nor reported (Dem_SetDtr).

[SWS_Dem_00688] ⌈ The Dem shall provide the postbuild/selectable boolean configuration option DemEventSuppressed per event to configure an event as suppressed. ⌋*(SRS_Diag_04126)*

[SWS_Dem_00689] ⌈ If the configuration parameter DemEventSuppressed was set, the Suppressed event functionaliy shall be activated. ⌋

[SWS_Dem_00690] ⌈ Dem_SetEventStatus) shall ignore events which are configured as suppressed events. ⌋*(SRS_Diag_04126)*

[SWS_Dem_00691] ⌈ Dem_ReportErrorStatus shall ignore events which are configured as suppressed events. ⌋*(SRS_Diag_04126)*

7.3 Event memory description

The ‘Event Memory’ is defined as a set of event records located in a dedicated memory block. The event record includes at least the extended event status and the event related data.

[SWS_Dem_00010] ⌈ The Dem module shall support the primary event memory. ⌋*(SRS_Diag_04066)*

[SWS_Dem_00548] ⌈ If configured (refer to [SWS_Dem_00162]) the Dem module shall support the additional event memories (secondary, mirror, permanent). ⌋*(SRS_Diag_04066)*

The size of the different event memories is configurable in the Dem configuration.

[SWS_Dem_00162] ⌈ The Dem module shall provide configuration parameters to adapt the fault memory size to the ECU memory space available (refer to configuration parameters DemMaxNumberEventEntry<...> in DemGeneral). ⌋*(SRS_Diag_04066)*

Note: If the size is configured to zero, the respective event memory is deactivated.

[SWS_Dem_00329] ⌈ For storing to non-volatile memory the Dem shall use the NVRAM Manager (refer to chapter 7.9.5). ⌋*(SRS_Diag_04077)*

The following figure shows an example of a logical Dem event memory layout.

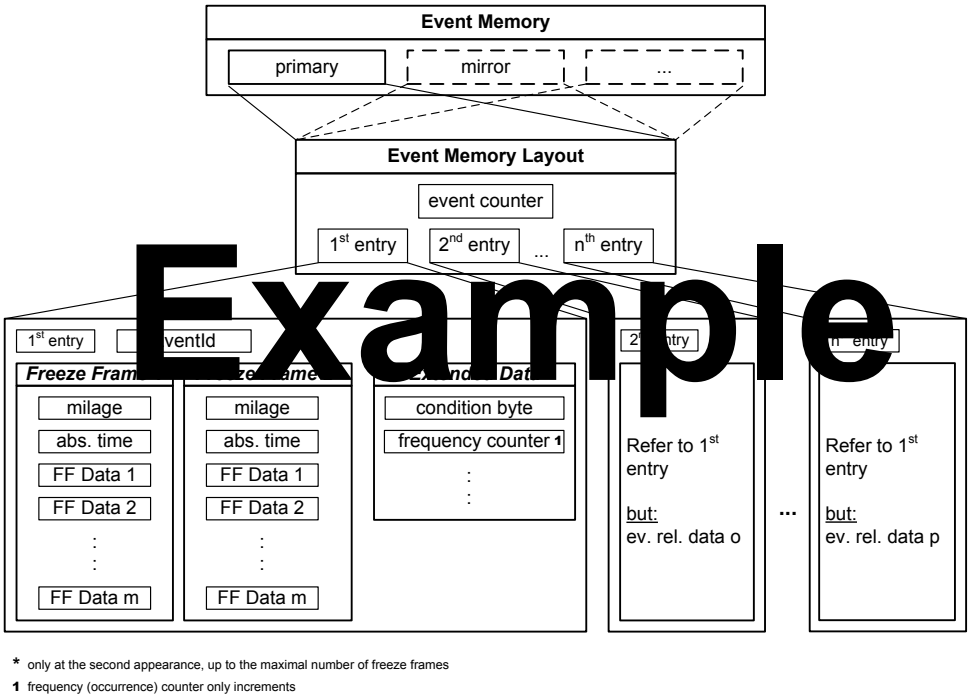

Figure 7.9: Example of a logical Dem event memory layout

If there are limitations of the memory size, it is necessary to provide overflow indication of the event memory and a displacement strategy (refer to chapter 7.3.2).

7.3.1 Event status management

The 'Event Status Management' is the Dem's ability to record and retain events, event status and associated data.

[SWS_Dem_00330] ⌈ The Dem module shall provide the capability to report the status of an event allowing a diagnostic monitor to inform the Dem about the result

of the internal diagnostic test (refer to chapter 8.3.3.2 and [SWS_Dem_00107]). ⌋(*SRS_Diag_04067*)

The monitors, which are located in the application, should call the function (Dem_SetEventStatus) to report an event status as soon as a new test result is available. This will be done independently of the current state of the Dem module (refer to Figure 11).

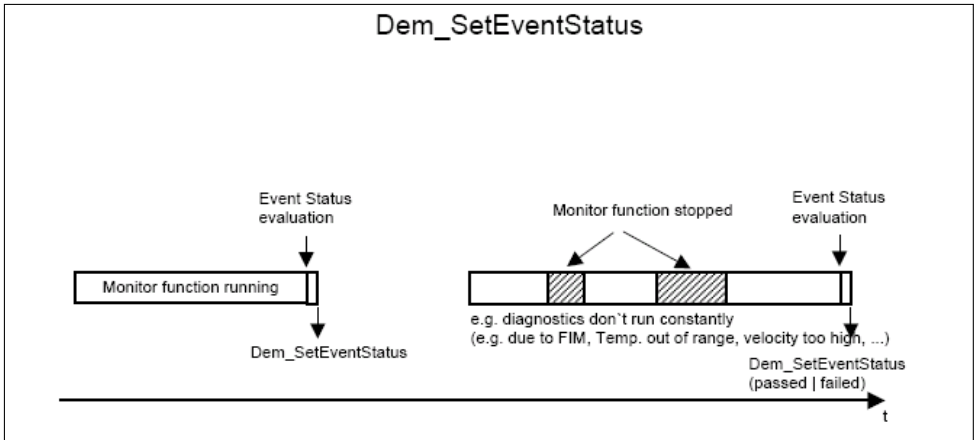

Figure 7.10: Example for using Dem_SetEventStatus

[SWS_Dem_00331] ⌈ The Dem module shall provide the capability to reset the failed status of an event without reporting a passed result (refer to chapter 8.3.3.4). ⌋(*SRS_Diag_04030*)

Note: Dem_ResetEventStatus does not mean an event status report (like done by Dem_SetEventStatus)). After the call, the event status is not qualified or tested.

Monitors will use the function Dem_ResetEventStatus in order to deactivate limp home and switch back to normal operation. At this point in time, the monitor has typically not been OK-tested and therefore Dem_SetEventStatus) with tested and passed cannot be used.

[SWS_Dem_00187] ⌈ The function Dem_ResetEventStatus shall set the UDS DTC status bit 0 (TestFailed) to 0 and reset the Dem-internal debounce algorithm to initial values if configured. ⌋

Note: The function Dem_ResetEventStatus does not change the UDS DTC status bit 6 (TestNotCompletedThisOperationCycle) and does not clear the pre-stored freeze frame.

[SWS_Dem_00638] ⌈ The function Dem_ResetEventStatus shall return E_NOT_OK, if the event was already tested this operation cycle (UDS DTC status bit 6 - TestNotCompletedThisOperationCycle is set to 0). ⌋

[SWS_Dem_00051] ⌈ The Dem module shall provide the capability to retrieve the current UDS DTC status byte of a specific event (refer to chapter 8.3.3.12). ⌋

Note: The function Dem_GetEventStatus is provided to be used by SW-Cs or other BSW modules (e.g. FiM) on event-level. The Dcm uses the function Dem_DcmGetStatusOfDTC on DTC-level instead.

[SWS_Dem_00333] ⌈ The Dem module shall be able to provide the current event failed status (refer to chapter 8.3.3.13) and the current event tested status (refer to chapter 8.3.3.14). ⌋*(SRS_Diag_04067, SRS_Diag_04096)*

[SWS_Dem_00052] ⌈ The function Dem_GetEventFailed shall report the UDS DTC status bit 0 (TestFailed) of the requested diagnostic event. ⌋

[SWS_Dem_00053] ⌈ The function Dem_GetEventTested shall read the negated UDS DTC status bit 6 (TestNotCompletedThisOperationCycle) of the requested diagnostic event. ⌋*(SRS_Diag_04067, SRS_Diag_04096)*

[SWS_Dem_00844] ⌈ The Dem shall provide a function Dem_GetDebouncingOfEvent () that reports the debounce status of an event. The outparameter 'DebouncingState' returns the debouncing incl. intermediate states. One particular OBD specific bit shall support the DTR update trigger if test is complete and debouncing is at its limit while the enable and storage conditions are met. ⌋

7.3.1.1 Status bit support

[SWS_Dem_00006] ⌈ The Dem module shall by default implement the current set of all UDS DTC status bits according to the definition in ISO 14229-1 [2] for each diagnostic event. ⌋*(SRS_Diag_04096)*

Note: For this specification, the UDS DTC status byte of ISO 14229-1 for events is represented by the Dem data type Dem_UdsStatusByteType (defined in chapter 8.2.1.6). If particular DTC status bits do not need to be supported (refer also to [SWS_Dem_00060]), such optimizations or limitations may be implemented vendor-specific.

7.3.1.2 Status bit update

It is a system design decision if synchronous or asynchronous event processing is used

[SWS_Dem_00036] ⌈ In case an event is qualifying, either by reporting (passed / failed) or by reaching the debounce counter thresholds, the Dem shall perform the event status transition synchronously in the reporting function (e.g. Dem_SetEventStatus)) at least for the following status bits:

- Bit 0 TestFailed
- Bit 1 TestFailedThisOperationCycle
- Bit 6 TestNotCompletedThisOperationCycle

⌋(*SRS_Diag_04096*)

[SWS_Dem_00379] ⌈ Depending on the design decision (synchronous or asynchronous event processing) the status update of the following bits:

- Bit 2 PendingDTC
- Bit 3 ConfirmedDTC
- Bit 7 WarningIndicatorRequested

may occur at a later point in time. ⌋(*SRS_Diag_04096*)

Rationale: the immediate status updates may be relevant for immediate fault reactions like update of the function inhibition (FIM).

Note: If the status update of the bits described in [SWS_Dem_00379] is implemented asynchronously, a queuing mechanism is needed which ensures that all the changes applied to bit 0, 1, 4, 5 and 6 will be considered when calculating bit 2, 3 and 7. Similarly, all processing and data storage associated with bit 2, 3 and 7 needs to be queued (refer to Figure 12 describing different approaches of synchronous and asynchronous event processing design).

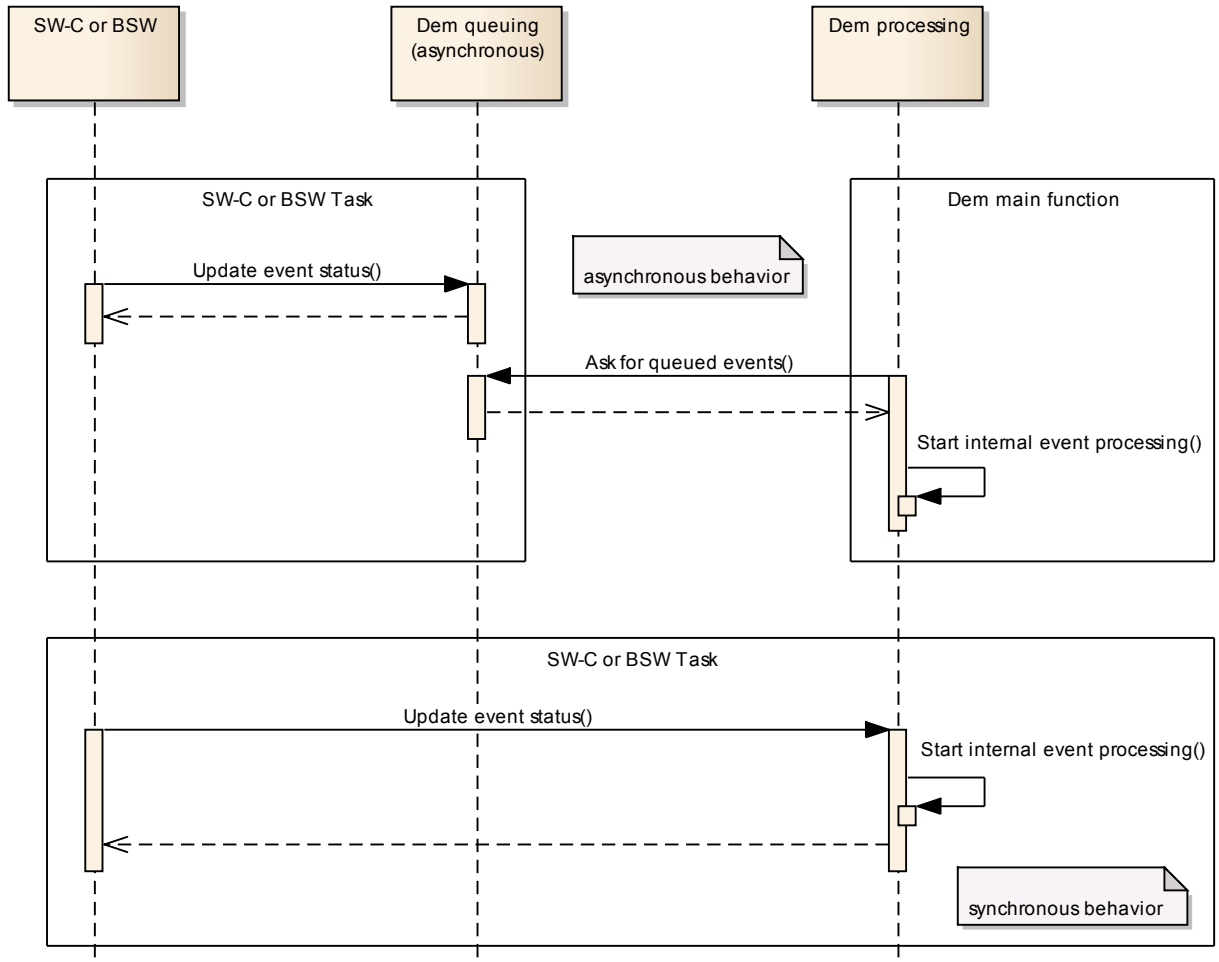

Figure 7.11: Synchronous and asynchronous event processing

Note: Refer to chapter 7.3.2 for the event memory management.

7.3.1.3 Status bit transitions

This section describes the behavior of the individual status bits of the UDS DTC status byte (refer to [2], UDS Status Byte - bit transitions).

[SWS_Dem_00385] ⌈ After a clear command has been applied to a specific DTC (refer to chapter 7.3.2.2) the Dem module shall set the UDS status byte to 0x50 (Readiness bits 4 and 6 set to 1, and all others are set to zero). ⌋*(SRS_Diag_04096)*

Note: The value of the UDS status byte after a clear command has been applied represents the delivery status (initial state) of this byte as well.

[SWS_Dem_00386] ⌈ The Dem module shall implement the status bit transitions for UDS DTC status bit 0 (TestFailed) according to figure 7.12. ⌋*(SRS_Diag_04096)*

[SWS_Dem_00387] ⌈ The Dem module shall support the configuration parameter DemStatusBitStorageTestFailed (refer to DemGeneral) to determine whether the information for UDS DTC status bit 0 (TestFailed) is stored volatile or non-volatile. ⌋*(SRS_Diag_04096)*

[SWS_Dem_00388] ⌈ If the configuration parameter DemStatusBitStorageTestFailed is set to False, the Dem module shall not retain the information for UDS DTC status bit 0 (TestFailed) over power cycles (volatile) ⌋*(SRS_Diag_04096)*

[SWS_Dem_00525] ⌈ If the configuration parameter DemStatusBitStorageTestFailed is set to True, the Dem module shall retain the information for UDS DTC status bit 0 (TestFailed) over power cycles (non-volatile). ⌋*(SRS_Diag_04096)*

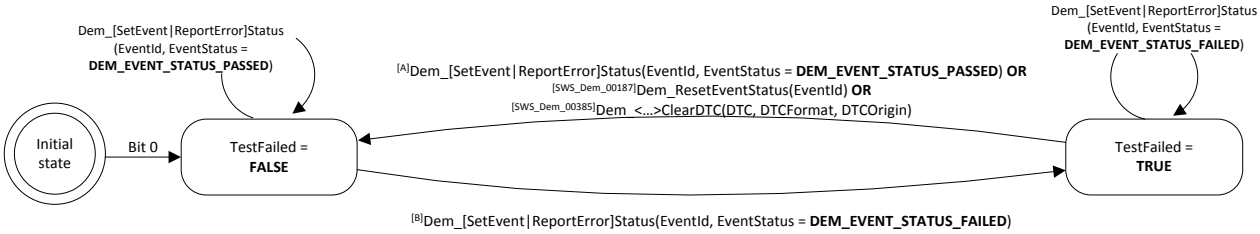

Figure 7.12: DTC status bit 0 TestFailed logic

[SWS_Dem_00389] ⌈ The Dem module shall implement the status bit transition for UDS DTC status bit 1 (TestFailedThisOperationCycle) according to figure 7.13. ⌋*(SRS_Diag_04096)*

Note: The information for UDS DTC status bit 1 (TestFailedThisOperationCycle) is non-volatile, if the PendingDTC bit is used (refer to [SWS_Dem_00006]) or if the Dem module supports operation cycles over power cycles (refer to DemOperationCycleStatusStorage).

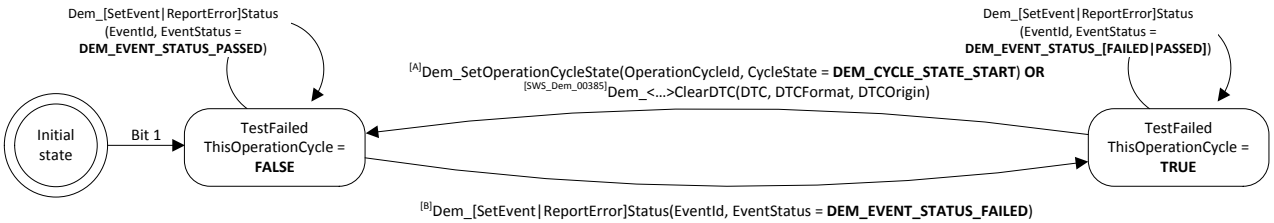

Figure 7.13: DTC status bit 1 TestFailedThisOperationCycle logic

[SWS_Dem_00390] ⌈ The Dem module shall implement the status bit transition for UDS DTC status bit 2 (PendingDTC) according to figure 7.14. ⌋*(SRS_Diag_04096)*

Note: The information for UDS DTC status bit 2 (PendingDTC) is non-volatile.

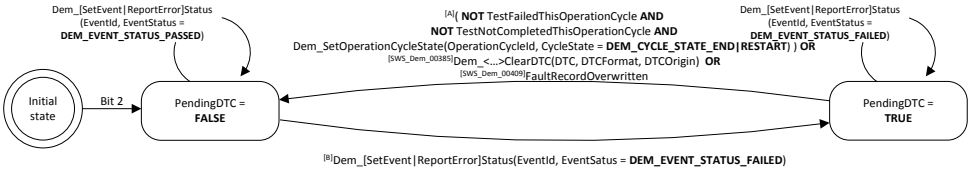

Figure 7.14: DTC status bit 2 PendingDTC logic

[SWS_Dem_00391] ⌈ The Dem module shall implement the status bit transition for UDS DTC status bit 3 (ConfirmedDTC) according to figure 7.15. ⌋*(SRS_Diag_04096)*

Note: The information for UDS DTC status bit 3 (ConfirmedDTC) is non-volatile (but it is also calculable based on the respective event memory entry).

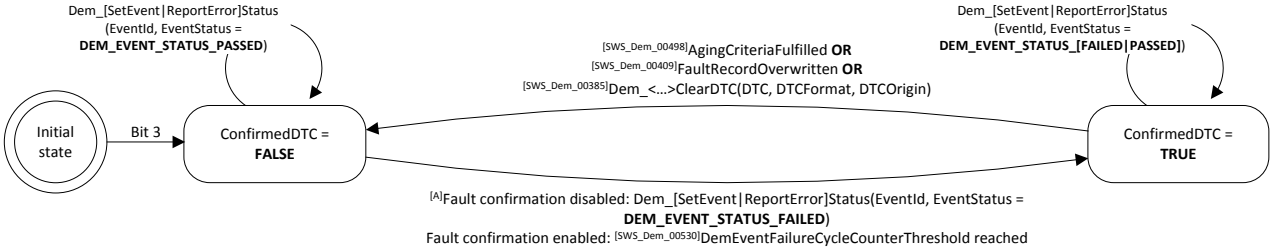

Figure 7.15: DTC status bit 3 ConfirmedDTC logic

Note: The “AgingCriteriaFulfilled” condition is specified by [SWS_Dem_00498]. The “FaultRecordOverwritten” condition is specified by [SWS_Dem_00409].

[SWS_Dem_00392] ⌈ The Dem module shall implement the status bit transition for UDS DTC status bit 4 (TestNotCompletedSinceLastClear) according to figure 7.16. ⌋*(SRS_Diag_04096)*

Note: The information for UDS DTC status bit 4 (TestNotCompletedSinceLastClear) is non-volatile.

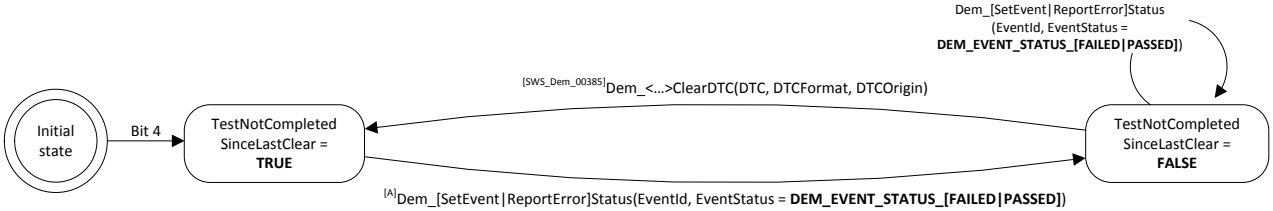

Figure 7.16: DTC status bit 4 TestNotCompletedSinceLastClear logic

[SWS_Dem_00393] ⌈ The Dem module shall implement the status bit transition for UDS DTC status bit 5 (TestFailedSinceLastClear) according to figure 7.17 . ⌋*(SRS_Diag_04096)*

Note: The information for UDS DTC status bit 5 (TestFailedSinceLastClear) is non-volatile.

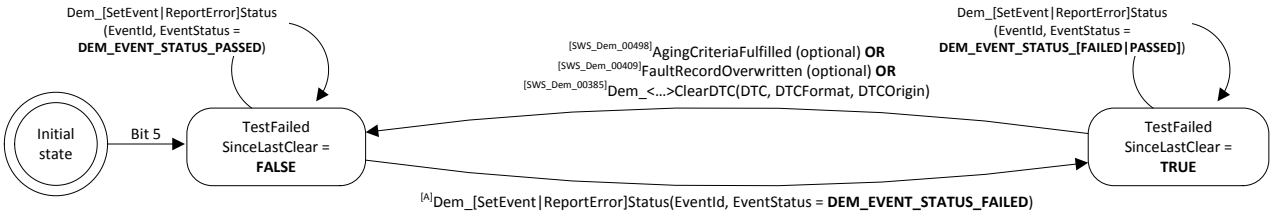

Figure 7.17: DTC status bit 5 TestFailedSinceLastClear logic

Note: The relevance of "AgingCriteriaFulfilled" condition (specified by [SWS_Dem_00498]) and "FaultRecordOverwritten" condition (specified by [SWS_Dem_00409]) depends on the configuration parameter DemStatusBitHandlingTestFailedSinceLastClear (refer to DemGeneral).

[SWS_Dem_00394] ⌈ The Dem module shall implement the status bit transition for UDS DTC status bit 6 (TestNotCompleteThisOperationCycle) according tofigure 7.18. ⌋*(SRS_Diag_04096)*

Note: The information for UDS DTC status bit 6 (TestNotCompleteThisOperationCycle) needs to be stored non-volatile, if the PendingDTC bit is used (refer to [SWS_Dem_00006]), or if the Dem module supports operation cycles over power cycles (refer to DemOperationCycleStatusStorage).

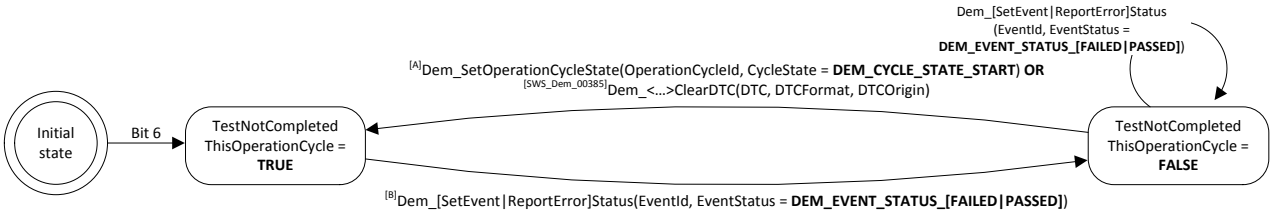

Figure 7.18: DTC status bit 6 TestNotCompleteThisOperationCycle logic

[SWS_Dem_00395] ⌈ The Dem module shall implement the status bit transition for UDS DTC status bit 7 (WarningIndicatorRequested) according to figure 7.19. ⌋*(SRS_Diag_04096)*

Note: The information for UDS DTC status bit 7 (WarningIndicatorRequested) may be volatile (because it is calculated based on the assigned warning indicator states).

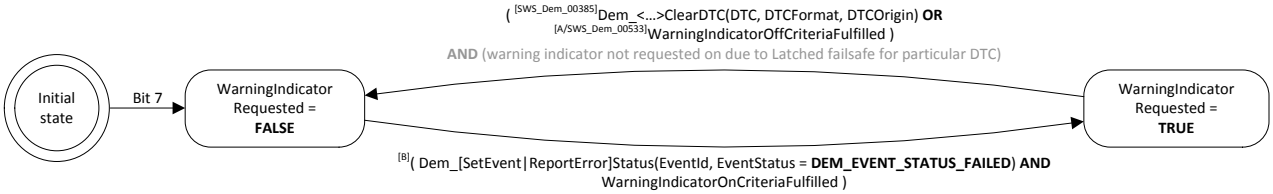

Figure 7.19: DTC status bit 7 WarningIndicatorRequested logic

Note: The “WarningIndicatorOffCriteriaFulfilled” and the “WarningIndicatorOnCriteriaFulfilled” condition are specified in chapter 7.3.11.

Note: ISO 14229-1 additionally specifies “warning indicator not requested on due to latched failsafe for particular DTC” as condition. This has to be ensured by the monitor.

7.3.1.4 Active/Passive status

If an event gets qualified as failed, it becomes active. If the event gets qualified as passed, it becomes passive. This status can be derived from the UDS DTC status byte.

As the TestFailed bit (UDS DTC status bit 0) is configurable in persistent storage ability (refer to configuration parameter DemStatusBitStorageTestFailed), also the meaning of active/passive is influenced:

- If the TestFailed bit is stored non-volatile, “event active” equals to TestFailed = 1 and “event passive” equals to TestFailed = 0.
- If the TestFailed bit is only stored volatile, additionally the information, if the event was already tested/reported this power cycle, is required. As long, as this information is not present, the active/passive status is undefined.

Note: For events, which are assigned to the operation cycle type DEM_OPCYC_POWER, the inverted TestNotCompletedThisOperationCycle bit represents the information about “already tested/reported this power cycle”.

Note: There are also ECUs, where all monitors run during startup phase. In this case, it is also sufficient, to map the active/passive status directly to the TestFailed bit.

7.3.1.5 Notification of status bit changes

The Dem module can inform the monitor and/or other components about the status change of the event or DTC.

Note: For asynchronous event processing (refer to [SWS_Dem_00379]), the Dem module will trigger the respective notification functions twice per event/DTC qualification.

[SWS_Dem_00016] ⌈ The Dem module shall trigger the event-specific callback-function EventStatusChanged (refer to chapter 8.4.3.2) on each event status change. ⌋*(SRS_Diag_04067)*

Note: The Dem module does not evaluate the return value (e.g. if other than E_OK) of this callback function.

Note: The configuration container DemCallbackEventStatusChanged (in DemEventParameter) is used to specify one or more ports/c-callbacks per event.

Note: The respective callback-functions for FiM and Dlt are specified in chapter 7.8.

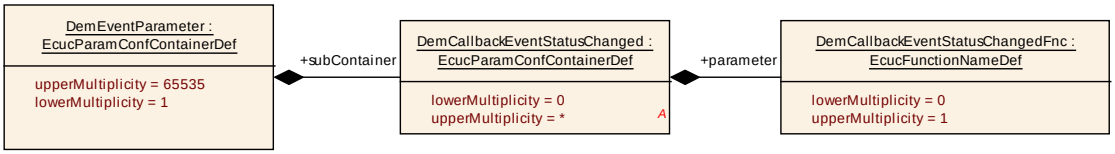

Figure 7.20: EventStatusChanged callback configuration

[SWS_Dem_00284] ⌈ The Dem module shall trigger the callback function configured in DemCallbackDTCStatusChanged on every DemDTC status change. ⌋

[SWS_Dem_00986] ⌈ The Dem module shall trigger the callback function configured in DemCallbackOBDDTCStatusChanged on every DemObdDTC status change. ⌋

[SWS_Dem_00987] ⌈ The Dem module shall trigger the callback function configured in DemCallbackJ1939DTCStatusChanged on every DemJ1939DTC status change. ⌋

Note: The Dem module does not evaluate the return value (e.g. if other than E_OK) of this callback function.

Note: The result of EventStatusChanged may only be different to the result of DTCStatusChanged in case of event combination(refer to chapter 7.3.5).

Note: The configuration containers DemCallbackDTCStatusChanged (in DemGeneral), DemCallbackOBDDTCStatusChanged (in DemGeneralOBD), and DemCallbackJ1939DTCStatusChanged (in DemGeneralJ1939) are used to specify one or more ports/c-callbacks for the Dem module globally.

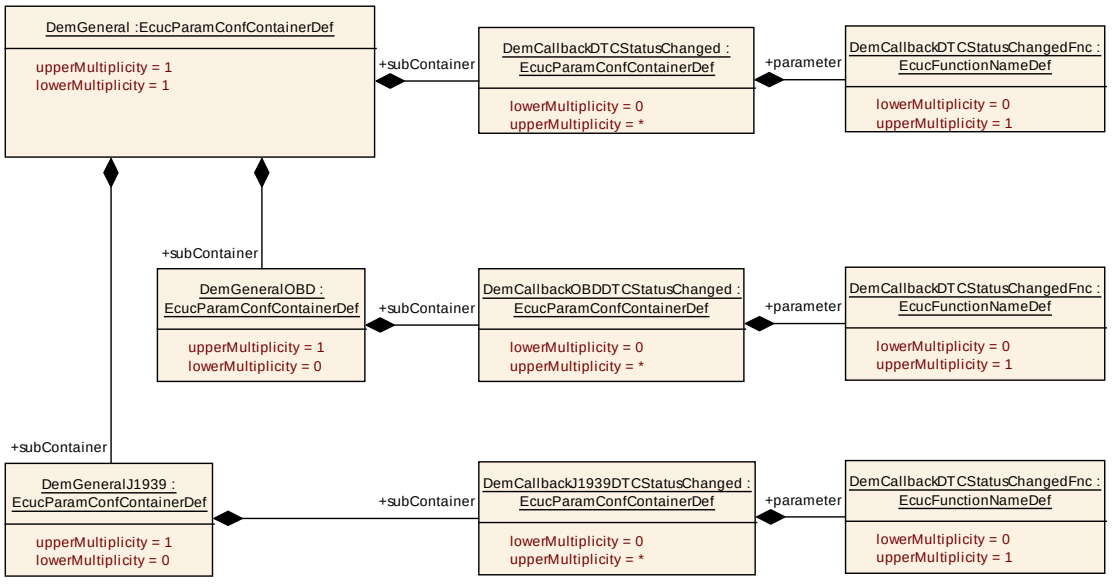

Figure 7.21: DTCStatusChanged callback configuration

7.3.2 Event memory management

The event memory management defines as the process of adding, updating and removing event memory entries in and out of the Dem module. The Dem module determines if the event memory entry is new or currently exists in the event memory.

Note: The requirements in this section do not determine the software implementation but describe the behavior of the ECU towards requests from the test tool.

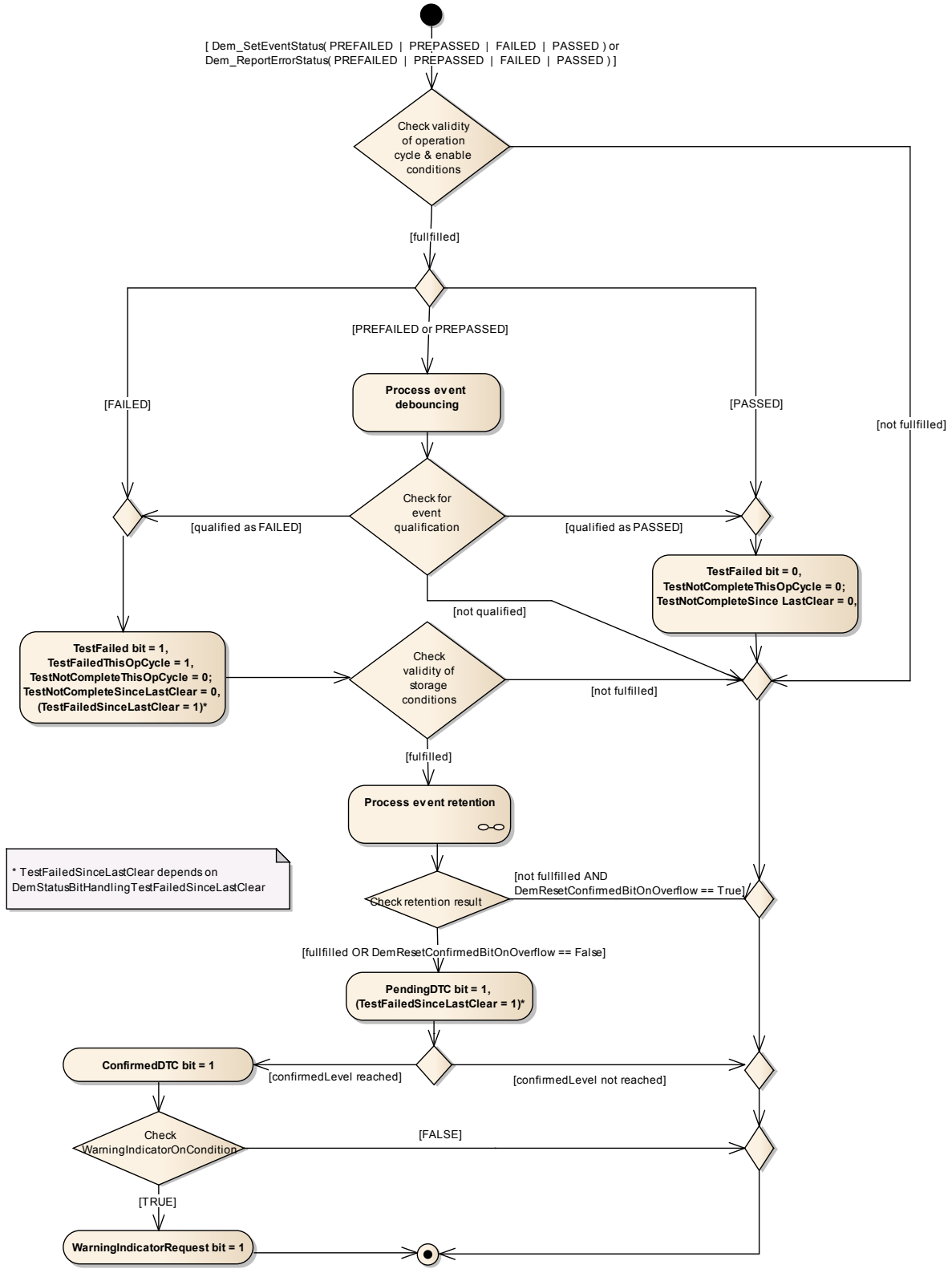

Figure 7.22: General diagnostic event storage processing

7.3.2.1 Event retention

Event retention defines the ability of the Dem module to record and handle events (DTCs), DTC status information and event related data (e.g. freeze frames, extended data).

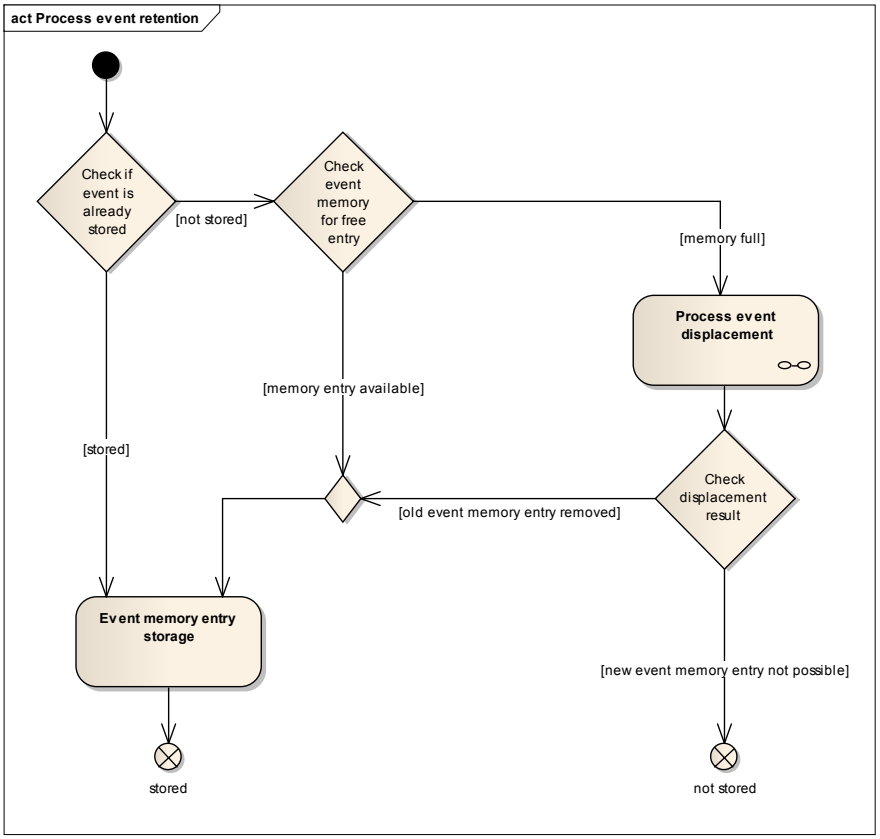

Figure 7.23: General diagnostic event retention processing

There are several different strategies in the market when to allocate an event memory entry. Therefore, the Dem module has the capability to configure the primary trigger (refer DemEventMemoryEntryStorageTrigger) to allocate an event memory entry. Beside the primary trigger there might be secondary triggers (refer DemFreezeFrameRecordTrigger) mainly to update content of event memory entry (e.g. to have most recent update FreezeFrames).

[SWS_Dem_00783] ⌈ If an event

1. gets qualified as failed (UDS DTC status bit 0 changes from 0 to 1) and
2. the configuration parameter DemEventMemoryEntryStorageTrigger is set to DEM_TRIGGER_ON_TEST_FAILED and
3. no event memory entry exist,

the Dem module shall try to allocate according Figure 7.23 an event memory entry in its configured event memory (refer to DemMemoryDestination). ⌋*(SRS_Diag_04105)*

[SWS_Dem_00784] ⌈ If an event

1. gets pending (UDS DTC status bit 2 changes from 0 to 1) and
2. the configuration parameter DemEventMemoryEntryStorageTrigger is set to DEM_TRIGGER_ON_PENDING and
3. no event memory entry is existing,

the Dem module shall try to allocate according Figure 7.23 an event memory entry in its configured event memory (refer to DemMemoryDestination). ⌋*(SRS_Diag_04105)*

[SWS_Dem_00922] ⌈ If an event

1. is pending (UDS DTC status bit 2 is set to 1) and gets re-qualified as failed (UDS DTC status bit 0 changes from 0 to 1) and
2. the configuration parameter DemEventMemoryEntryStorageTrigger is set to DEM_TRIGGER_ON_PENDING and
3. no event memory entry is existing,

the Dem module shall try to allocate according Figure 7.23 an event memory entry in its configured event memory (refer to DemMemoryDestination). ⌋*(SRS_Diag_04105)*

[SWS_Dem_00785] ⌈ If an event

1. gets confirmed (UDS DTC status bit 3 changes from 0 to 1) and
2. the configuration parameter DemEventMemoryEntryStorageTrigger is set to DEM_TRIGGER_ON_CONFIRMED and
3. no event memory entry is existing,

the Dem module shall try to allocate according Figure 7.23 an event memory entry in its configured event memory (refer to DemMemoryDestination). ⌋*(SRS_Diag_04105)*

[SWS_Dem_00923] ⌈ If an event

1. is confirmed (UDS DTC status bit 3 is set to 1) and gets re-qualified as failed (UDS DTC status bit 0 changes from 0 to 1) and
2. the configuration parameter DemEventMemoryEntryStorageTrigger is set to DEM_TRIGGER_ON_CONFIRMED and
3. no event memory entry is existing,

the Dem module shall try to allocate according Figure 7.23 an event memory entry in its configured event memory (refer to DemMemoryDestination). ⌋*(SRS_Diag_04105)*

Note: This requirement is only applicable if DemResetConfirmedBitOnOverflow is set to false.

[SWS_Dem_00786] ⌈ If an event

1. has DemDebounceAlgorithmClass set to DemDebounceCounterBased and

2. the internal debounce counter reaches the DemEventMemoryEntryFdcThresholdStorageValue by a positive increment via EventStatus set to preFailed and
3. the configuration parameter DemEventMemoryEntryStorageTrigger is set to DEM_TRIGGER_ON_CONFIRMED and
4. no event memory entry is existing,

the Dem module shall try to allocate according Figure 7.23 an event memory entry in its configured event memory (refer to DemMemoryDestination). ⌋*(SRS_Diag_04105)*

Note: In case the event memory entry already exists, there might be other triggers to update content of event memory entry(refer chapter 7.3.7 “Event related data”).

7.3.2.2 Clearing event memory entries

The event memory entries of the Dem module can be cleared DTC-based via various diagnostic services (e.g. service 0x14 ClearDiagnosticInformation or service $04 ClearErrorMemory). Therefore, the Dem offeres separate APIs for different users:

- Dem_ClearDTC (to CDDs, refer below)
- Dem_DcmClearDTC (for UDS to the Dcm, refer to chapter 7.9.2.3)
- Dem_SetClearDTC (for OBD to SW-Cs, refer to chapter 7.7.1.4)
- Dem_J1939DcmClearDTC (for J1939 to the J1939Dcm [4], refer to chapter 7.8.2).

[SWS_Dem_01057] ⌈ If the Dem module is requested to clear diagnostic information and the configuration parameter DemClearDTCBehavior is set to DEM_CLRRESP_NONVOLATILE_FINISH, the Dem module shall return DEM_CLEAR_MEMORY_ERROR (refer to Dem_ReturnClearDTCType) if the clearing of the non-volatile memory fails. ⌋*(SRS_Diag_04065)*

[SWS_Dem_00659] ⌈ The Dem module shall provide the API Dem_ClearDTC (refer to chapter 8.3.3.27) to complex device drivers for deleting single DTCs, DTC groups, and all DTCs from the event memory. This shall trigger also initializing of related SW-Cs / BSW modules according to [SWS_Dem_00003]. ⌋*(SRS_Diag_04122)*

Note: Dem_ClearDTC typically triggers further callbacks through the RTE. To indicate the respective call-tree for these runnables, a work-around is used: The respective CDD must trigger the DTC deletion using the Dem interface CddIf (operation ClearDTC, refer to chapter 8.6.2.11) instead of a direct C call.

[SWS_Dem_00660] ⌈ The API Dem_ClearDTC shall clear the status of all event(s) related to the specified DTC(s), as well as all associated event memory entries for these event(s). ⌋*(SRS_Diag_04122)*

Note: Exceptions may apply due to [SWS_Dem_00514].

The clearing process can be started by the different users using their related Dem_<...>ClearDTC interface. Whenever one of these interfaces s the clearing process, the Dem-internal clearing mechanism is locked until the clear request is finished and the result is returned to the requesting user.

[SWS_Dem_00661] ⌈ The first call of any Dem_<...>ClearDTC API (e.g. Dem_ClearDTC) while the clearing process is not locked shall lock the clearing process for all other callers. ⌋*(SRS_Diag_04065, SRS_Diag_04122)*

[SWS_Dem_00662] ⌈ Any call of a Dem_<...>ClearDTC API (e.g. Dem_ClearDTC) while the clearing process is locked by another instance (e.g. Dem_DcmClearDTC) shall return DEM_CLEAR_BUSY. ⌋*(SRS_Diag_04065, SRS_Diag_04122)*

[SWS_Dem_00663] ⌈ If the clearing process is locked, any subsequent call of the same Dem_<...>ClearDTC API with the same set of parameters shall either return DEM_CLEAR_PENDING while DTC clearing is in progress, or return DEM_CLEAR_OK in case DTC clearing has finished with no errors. ⌋*(SRS_Diag_04065, SRS_Diag_04122)*

Note: Also the interface Dem_ClearDTC is designed to have only one caller. If for example two users are using the same set of clear parameters, the clearing may be performed twice and it is not defined which clear requester gets DEM_CLEAR_OK first.

[SWS_Dem_00664] ⌈ If the clearing process is locked, any subsequent call of any Dem_<...>ClearDTC API with a different set of parameters shall return DEM_CLEAR_BUSY. ⌋*(SRS_Diag_04065, SRS_Diag_04122)*

[SWS_Dem_00670] ⌈ If the configuration parameter DemClearDTCLimitation (refer to DemGeneral) is set to DEM_ONLY_CLEAR_ALL_DTCS, the APIs Dem_<...>ClearDTC shall only accept the DTC group DEM_DTC_GROUP_ALL_DTCS for UDS (DTCFormat equals DEM_DTC_FORMAT_UDS) and shall return DEM_CLEAR_WRONG_DTC in case any other DTC group or DTC value is requested. ⌋*(SRS_Diag_04117)*

[SWS_Dem_00569] ⌈ The Dem module shall provide a configuration parameter DemClearDTCBehavior (refer to DemGeneral) to define the clearing process of diagnostic information concerning volatile and non-volatile memory and the positive response handling for the Dcm/Cdd module. ⌋*(SRS_Diag_04065, SRS_Diag_04122)*

Note: The interaction with the NvRAM Manager is described in chapter 7.9.5. Due to the hardware design in some cases, the direct access to the non-volatile memory is only possible during shutdown phase.

[SWS_Dem_00570] ⌈ If the Dem module is requested to clear diagnostic information and the configuration parameter DemClearDTCBehavior is set to DEM_CLRRESP_VOLATILE, the Dem module shall return DEM_CLEAR_OK (refer to chapter 8.2.2.12) after the volatile memory is cleared. ⌋*(SRS_Diag_04065, SRS_Diag_04122)*

[SWS_Dem_00571] ⌈ If the Dem module is requested to clear diagnostic information and the configuration parameter DemClearDTCBehavior is set

to DEM_CLRRESP_NONVOLATILE_TRIGGER, the Dem module shall return DEM_CLEAR_OK (refer to chapter 8.2.2.128) after the volatile memory is cleared and clearing of the non-volatile memory is triggered. ⌋(*SRS_Diag_04065, SRS_Diag_04122*)

[SWS_Dem_00572] ⌈ If the Dem module is requested to clear diagnostic information and the configuration parameter DemClearDTCBehavior is set to DEM_CLRRESP_NONVOLATILE_TRIGGER, the Dem module shall return DEM_CLEAR_OK (refer to chapter 8.2.2.12) after the volatile memory is cleared and clearing of the non-volatile memory is triggered. ⌋(*SRS_Diag_04065, SRS_Diag_04122*)

Note: If the Dcm module receives the return type DEM_CLEAR_OK of the API Dem_DcmClearDTC, the Dcm module sends the positive response (refer to citeSWS-DiagnosticCommunicationManager).

[SWS_Dem_00573] ⌈ The Dem module shall provide the configuration parameter DemTriggerMonitorInitBeforeClearOk (refer to DemGeneral) to configure, if the callback function InitMonitorForEvent (refer to chapter 7.5) has to be called before or after Dem_<...>ClearDTC returns DEM_CLEAR_OK. ⌋(*SRS_Diag_04065, SRS_Diag_04122*)

[SWS_Dem_00514] ⌈ The Dem module shall be able to suppress the DTC-deletion in Dem_<...>ClearDTC. The callback ClearEventAllowed (refer to sub-sub:DemClearEventAllowed), which is configurable/optional per event, shall realize this suppression handling. ⌋(*SRS_Diag_04092, SRS_Diag_04122*)

Note: Event displacement (mainly controlled by the event priority) and aging are not influenced by this callback.

This functionality is intended for some special events, which must never be cleared from event memory within specific states, like for example special operation modes of the ECU (e.g. assembly-, transport-, or flash-mode) or Dcm sessions (which can be determined, using API Dcm_GetSesCtrType).

It is configurable per event (refer to DemCallbackClearEventAllowed), whether a SW-C / BSW module shall be requested about the event-deletion allowance from this callback function. One callback (representing a specific condition) can be assigned per event. If multiple conditions apply, they need to be summarized by one callback individually.

[SWS_Dem_00515] ⌈ The Dem module shall call the callback ClearEventAllowed for each event it is configured for, before clearing this event (via Dem_<...>ClearDTC). ⌋(*SRS_Diag_04092, SRS_Diag_04122*)

[SWS_Dem_00667] ⌈ If the out-parameter 'Allowed' of the callback ClearEventAllowed returns "false" (and the return value is equal to E_OK) after the execution of this callback function, the respective event memory entry and Dem-internal data values must not be cleared. ⌋(*SRS_Diag_04092, SRS_Diag_04122*)

[SWS_Dem_00668] ⌈ If the out-parameter 'Allowed' of the callback ClearEventAllowed returns "false" and the configuration parameter DemClearEventAllowedBehavior (re-

fer to DemCallbackClearEventAllowed) is set to DEM_NO_STATUS_BYTE_CHANGE, the related event status byte shall not be modified. ⌋(*SRS_Diag_04092, SRS_Diag_04122*)

[SWS_Dem_00669] ⌈ If the out-parameter 'Allowed' of the callback ClearEventAllowed returns "false" and the configuration parameter DemClearEventAllowedBehavior (refer to DemCallbackClearEventAllowed) is set to DEM_ONLY_THIS_CYCLE_AND_READINESS, the related event status bits 1 (TestFailedThisOperationCycle), 4 (TestNotCompletedSinceLastClear), 5 (TestFailedSinceLastClear), and 6 (TestNotCompletedThisOperationCycle) shall be reset. ⌋(*SRS_Diag_04092, SRS_Diag_04122*)

[SWS_Dem_00516] ⌈ If the return value of the callback ClearEventAllowed is not equal to E_OK, the event-deletion is allowed anyway (because of safety aspects). ⌋(*SRS_Diag_04092, SRS_Diag_04122*)

7.3.2.3 Event memory overflow indication

[SWS_Dem_00397] ⌈ The Dem module shall indicate for each event memory if the event memory is full and there was an attempt to store an additional event in this event memory. ⌋(*SRS_Diag_04093*)

This overflow indication can be used to trigger further internal behavior of the Dem module (e.g. displacement strategy). Furthermore, it can be used for additional fault analysis in workshops.

[SWS_Dem_00398] ⌈ The Dem module shall provide the API Dem_GetEventMemoryOverflow (refer to chapter 8.3.3.24) to provide access to the event memory overflow indication status of the respective event memory. ⌋(*SRS_Diag_04093*)

Note: This API can be used for vendor-specific Diagnostic Services or other vendor-specific handling, and is provided to SW-Cs, as well as complex device drivers.

[SWS_Dem_00651] ⌈ The Dem module shall provide the API Dem_GetNumberOfEventMemoryEntries (refer to chapter 8.3.3.25) to return the number of event memory entries currently stored in the event memory. ⌋(*SRS_Diag_04109*)

Note: For the reported number all events are considered independently of the fault confirmation.

[SWS_Dem_00399] ⌈ The event memory overflow indication of the respective event memory shall be reset, if all DTCs of this memory are deleted by Dem_<...>ClearDTC. ⌋(*SRS_Diag_04093*)

Note: In case of aging and deleting single DTCs, the overflow indication of the event memory is not reset to keep this status information until the ECU receives an explicit clear command of this specific event memory. The ECU itself should not change this status information during normal operation (because it is used in workshops).

7.3.2.4 Event displacement

Event displacement means, that the most insignificant, already existing event memory entry is replaced by a new event memory entry, which needs to be stored. During displacement, the most insignificant entry gets lost.

[SWS_Dem_00400] ⌈ If the event retention want to allocate a new event memory entry and there is no free event memory entry available, the Dem module shall check according [SWS_Dem_00406] for allocated event memory entries to be displaced by the new event memory entry. ⌋*(SRS_Diag_04118)*

Note: If the event memory size is configured to cover all possible events, no displacement will occur.

[SWS_Dem_00401] ⌈ The Dem module provides the configuration parameter DemEventDisplacementStrategy (refer to DemGeneral defining whether the existing event memory entry can be displaced or not. ⌋*(SRS_Diag_04118)*

[SWS_Dem_00402] ⌈ If event displacement is disabled (DemEventDisplacementStrategy selects DEM_DISPLACEMENT_NONE), the Dem module shall not displace existing event memory entries if the event memory is full. ⌋*(SRS_Diag_04118)*

[SWS_Dem_00406] ⌈ If event displacement is enabled (DemEventDisplacementStrategy selects DEM_DISPLACEMENT_FULL or DEM_DISPLACEMENT_PRIO_OCC), the Dem module shall perform the following sequence by combination of the different displacement criteria (refer to Figure 25): 1. Priority 2. Active/Passive status (configurable, only for DEM_DISPLACEMENT_FULL) 3. Occurrence ⌋*(SRS_Diag_04118)*

[SWS_Dem_00403] ⌈ If displacement strategy is DEM_DISPLACEMENT_FULL or DEM_DISPLACEMENT_PRIO_OCC, the Dem module shall displace lower prioritized events by higher prioritized events. ⌋*(SRS_Diag_04118)*

[SWS_Dem_00404] ⌈ If displacement strategy is DEM_DISPLACEMENT_FULL, the Dem module shall displace passive events preferably (refer to chapter 7.3.1.4). ⌋*(SRS_Diag_04118)*

[SWS_Dem_00405] ⌈ If displacement strategy is DEM_DISPLACEMENT_FULL or DEM_DISPLACEMENT_PRIO_OCC, the Dem module shall displace older events by newer events preferably. ⌋*(SRS_Diag_04118)*

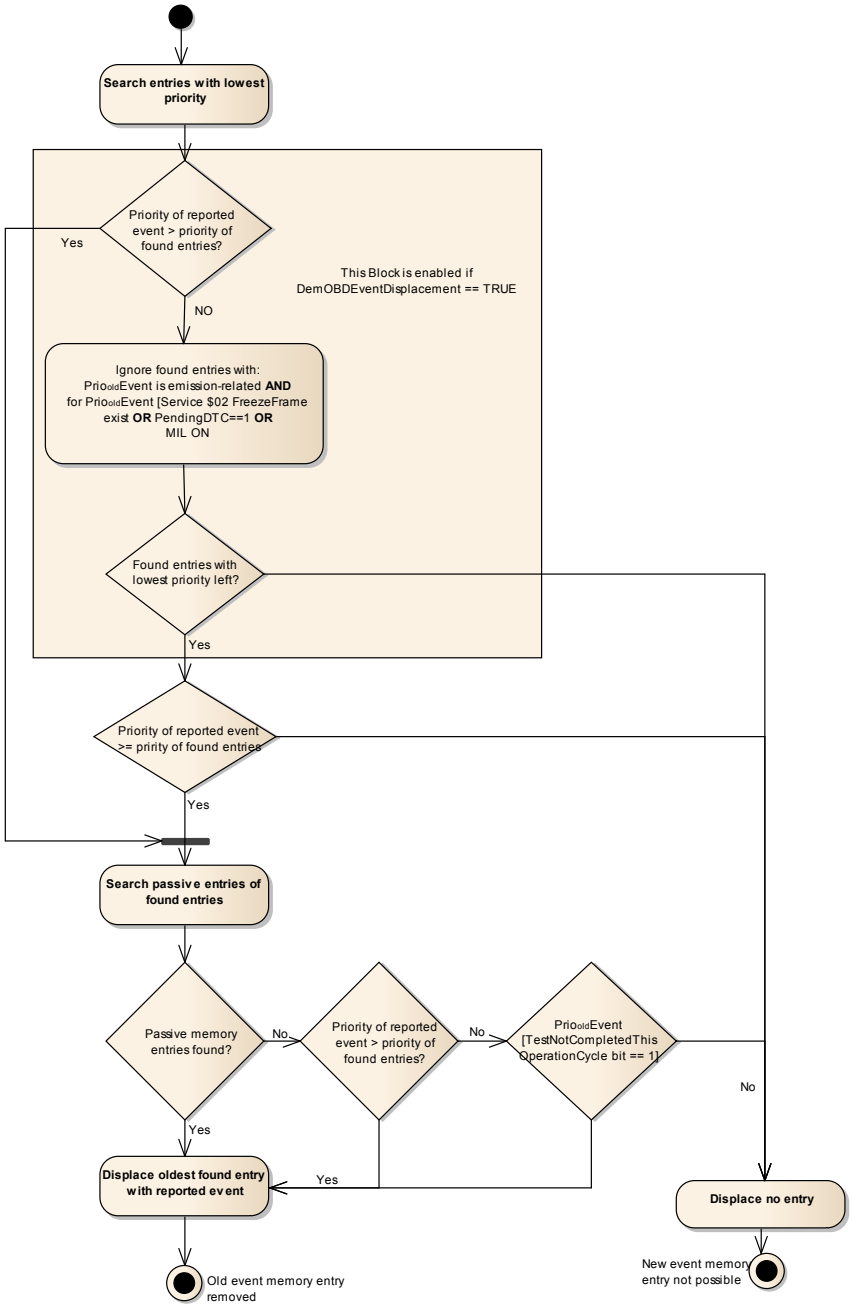

Figure 7.24: Combined displacement criteria processing

[SWS_Dem_00407] ⌈ If no event memory entry for displacement was identified, the Dem module shall discard the storage request. ⌋*(SRS_Diag_04118)*

[SWS_Dem_00692] ⌈ If an event memory entry has the same priority like the reported event and the existing event is not tested in this OperationCycle (TestNotCompletedThisOperationCycle bit == 1) the Dem module shall displace this event. ⌋*(SRS_Diag_04118, SRS_Diag_04071)*

[SWS_Dem_00693] ⌈ If an event memory entry has the same priority like the reported event and the existing event is tested in this OperationCycle (TestNotCompletedThisOperationCycle bit == 0) the Dem shall discard the reported event because the TestFailed status of the existing event is not verified. ⌋*(SRS_Diag_04076)*

[SWS_Dem_00694] ⌈ The Dem module shall provide the configuration parameter DemOBDEventDisplacement to enable a different displacement behavior for OBD events. ⌋*(SRS_Diag_04118)*

[SWS_Dem_00695] ⌈ If the configuration parameter DemOBDEventDisplacement is set to TRUE all emission related events with the following conditions shall not be considered during the displacement mechanism and shall not displaced from error memory:

PrioOldEvent is emission related AND for PrioOldEvent [Service$02 FreezeFrame exist OR Pending-DTC==1 OR MIL_ON] ⌋*(SRS_Diag_04102)*

[SWS_Dem_00696] ⌈ If the configuration parameter DemOBDEventDisplacement is set to FALSE the Dem module shall process the displacement without the OBD-behavior (refer to Figure 25). ⌋*(SRS_Diag_04102)*

[SWS_Dem_00408] ⌈ If the event memory entry for displacement was identified, the Dem module shall remove this event memory entry from the event memory. ⌋*(SRS_Diag_04118)*

Note: This removed event memory entry should now used for the pending storage trigger.

[SWS_Dem_00409] ⌈ If an event memory entry was removed during displacement, the Dem module shall reset the UDS status bit 2 (PendingDTC) and UDS status bit 3 (ConfirmedDTC) to 0 if the configuration parameter DemResetConfirmedBitOnOverflow (refer to DemGeneral) is set to true. Further the UDS status bit 5 (TestFailedSinceLastClear) shall also reset to 0 in case of DemStatusBitHandlingTestFailedSinceLastClear and DemResetConfirmed-BitOnOverflow (refer to DemGeneral) are set to True. ⌋*(SRS_Diag_04118)*

Note: All other UDS status bits are not modified by the displacement of an corresponding event memory entry.

Note: If configuration parameter DemResetConfirmedBitOnOverflow (refer toDemGeneral) is set to false the UDS status bit 3 (ConfirmedDTC) as well as the UDS status bit 5 (TestFailedSinceLastClear) will not modified.

7.3.2.5 Reporting order of event memory entries

[SWS_Dem_00410] ⌈ The Dem module shall report DTCs in the chronological order of the event storage (refer to API Dem_DcmGetNextFilteredDTC), if the DTC status mask parameter is set to “pending DTC” or “confirmed DTC” bit, or both bits (no other status bits are allowed to be set). ⌋*(SRS_Diag_04070)*

Note: The chronological order is for reporting purposes only and does not imply explicit memory structure, which is implementation specific.

Note: The reporting order may vary with the customer specific attributes used by the algorithm for sorting the DTC records in case of not confirmed or pending. If the chronological order is required for further DTC status mask parameters, additional resources may be necessary.

[SWS_Dem_00787] ⌈ If an 'stored event' gets re-qualified as failed (UDS DTC status bit 0 changes from 0 to 1) and a respective event memory entry exists, the Dem module shall update the chronological order of the event storage storage by setting the particular event as most recent event memory entry. ⌋*(SRS_Diag_04070)*

[SWS_Dem_00411] ⌈ If the Dem module is requested to report in chronological order, the most recent event memory entry shall be reported at first. ⌋*(SRS_Diag_04102)*

[SWS_Dem_00412] ⌈ The Dem module shall derive the chronological reporting order from the point in time when the event memory entry is stored. ⌋*(SRS_Diag_04070, SRS_Diag_04102)*

Note: If there are more than one event memory entries stored at the same point in time the chronological reporting order of these elements is undefined. It depends on the implementation which of these elements is reported at first or will be displaced.

7.3.3 Debouncing of diagnostic events

In general, an ECU may implement several types of debouncing improving signal quality. This section describes methods, how the Dem module shall implement basic algorithms used for fault maturation (diagnostic event debouncing).

If the Dem module is configured to implement the debounce algorithm for a specific event, one of the following debounce algorithms are to be performed Dem-internally. Otherwise, the respective monitor implemented in a SW-C or BSW will report the status of a certain event, after diagnostic event debouncing has been completed. For interaction between the Dem and SW-C or BSW please refer to the API InitMonitorForEvent (refer to chapter 7.5).

If there are any requirements to get the passed or failed status within a certain amount of time, the diagnostic monitor is responsible.

[SWS_Dem_00413] ⌈ The Dem module shall support the event-specific configuration of debounce algorithms by using the configuration container DemDebounceAlgorithmClass. ⌋*(SRS_Diag_04068)*

Note: That means for each individual event a specific algorithm can be specified.

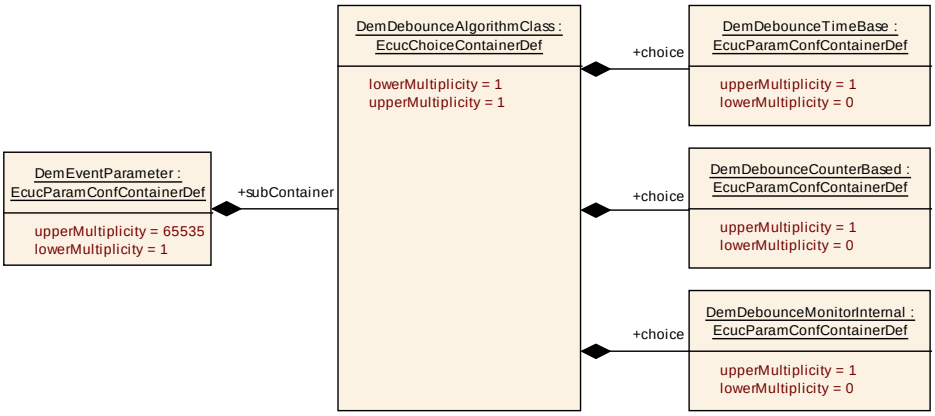

Figure 7.25: Event-specific debounce algorithms

7.3.3.1 Counter based debounce algorithm

[SWS_Dem_00526] ⌈ The Dem module shall provide a configuration parameter DemDebounceCounterBasedSupport (refer to DemGeneral) to enable or disable the Dem-internal counter based debouncing. ⌋*(SRS_Diag_04068)*

[SWS_Dem_00414] ⌈ If the configuration container DemDebounceAlgorithmClass is set to DemDebounceCounterBased, the Dem module shall provide an internal debounce counter for each individual event, to qualify the reported event. ⌋*(SRS_Diag_04068)*

For huge debounce ranges, the maximal range of the internaldebounce counter is defined as sint16. This does not imply any explicit implementation.

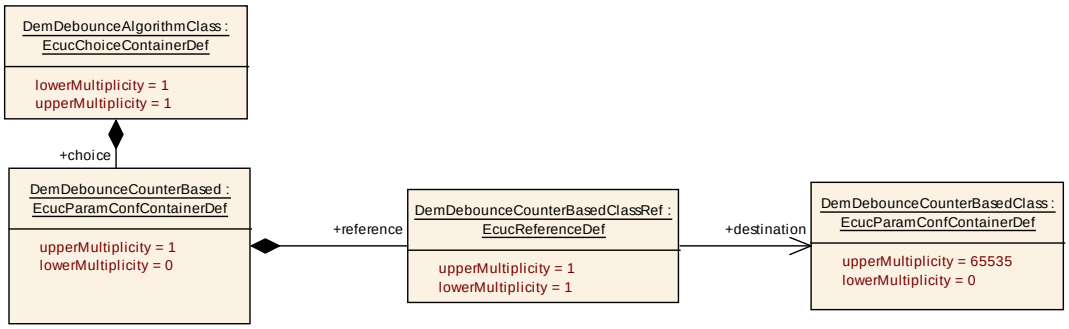

Figure 7.26: Counter based debounce algorithm

[SWS_Dem_00415] ⌈ The Dem module shall calculate the fault detection counter (-128 ... +127 according to UDS) based on the value and range of the internal debounce counter to map the internal counter values linearly to the external values. ⌋*(SRS_Diag_04010)*

[SWS_Dem_00416] ⌈ The Dem module shall provide the configuration parameter DemDebounceCounterFailedThreshold used to define the event-specific limit indicating the failed status (active). ⌋*(SRS_Diag_04068)*

[SWS_Dem_00417] ⌈ The Dem module shall provide the configuration parameter DemDebounceCounterPassedThreshold used to define the event-specific limit indicating the passed status (passive). ⌋*(SRS_Diag_04010, SRS_Diag_04068)*

[SWS_Dem_00418] ⌈ The Dem module shall increment the internal debounce counter with its configured step-size (refer to DemDebounceCounterIncrementStepSize), when the monitor reports DEM_EVENT_STATUS_PREFAILED (refer to chapter 8.2.1.3). ⌋*(SRS_Diag_04106, SRS_Diag_04068)*

[SWS_Dem_00419] ⌈ The Dem module shall decrement the internal debounce counter with its configured step-size (refer to DemDebounceCounterDecrementStepSize), when the monitor reports DEM_EVENT_STATUS_PREPASSED (refer to chapter 8.2.1.3). ⌋*(SRS_Diag_04106, SRS_Diag_04068)*

[SWS_Dem_00420] ⌈ If the monitor reports DEM_EVENT_STATUS_FAILED, the Dem module shall set the internal debounce counter value to its configured threshold being the failed criteria. ⌋*(SRS_Diag_04106, SRS_Diag_04068)*

[SWS_Dem_00421] ⌈ If the monitor reports DEM_EVENT_STATUS_PASSED, the Dem module shall set the internal debounce counter value to its configured threshold being the passed criteria. ⌋*(SRS_Diag_04096, SRS_Diag_04068, SRS_Diag_04106)*

[SWS_Dem_00422] ⌈ The Dem module shall provide the configuration parameter DemDebounceCounterJumpDown for enabling the jump-down behavior. ⌋*(SRS_Diag_04106, SRS_Diag_04068)*

[SWS_Dem_00423] ⌈ If the jump-down behavior is enabled, the Dem module shall provide the configuration parameter DemDebounceCounterJumpDownValue defining the new internal debounce counter init value. Each reporting of a pre-passed value while the current debounce counter value is greater than the DemDebounceCounterJumpDownValue shall first reset the debounce counter to DemDebounceCounterJumpDownValue before performing the pre-passed debounce event ([SWS_Dem_00419]). ⌋*(SRS_Diag_04106, SRS_Diag_04068)*

Note: This will only happen if the counting direction changes from incrementing to decrementing.

[SWS_Dem_00424] ⌈ The Dem module shall provide the configuration parameter DemDebounceCounterJumpUp for enabling the jump-up behavior. ⌋*(SRS_Diag_04106, SRS_Diag_04068)*

[SWS_Dem_00425] ⌈ If the jump-up behavior is enabled, the Dem module shall provide the configuration parameter DemDebounceCounterJumpUpValue defining the new internal debounce counter init value. Each reporting of a pre-failed value while the current debounce counter value is smaller than the DemDebounceCounterJumpUpValue shall first reset the debounce counter to DemDebounceCounterJumpUpValue before performing the pre-failed debounce event ([SWS_Dem_00418]). ⌋*(SRS_Diag_04106, SRS_Diag_04068)*

Note: This will only happen if the counting direction changes from decrementing to incrementing.

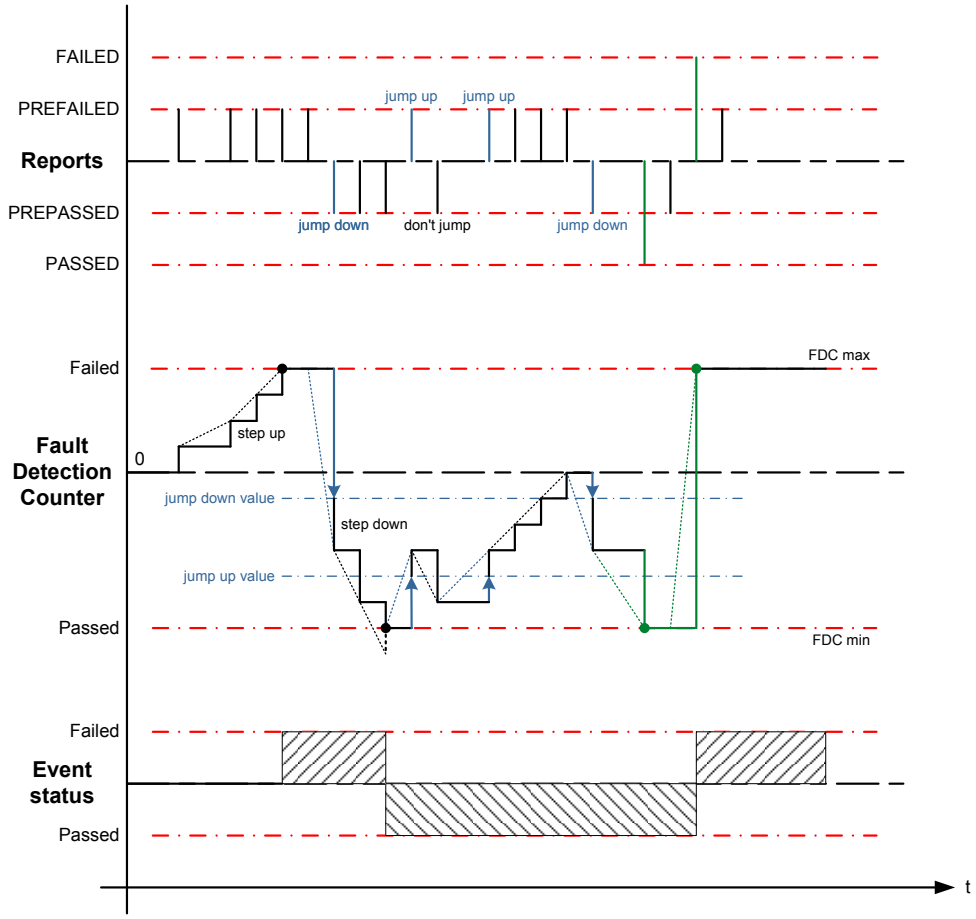

Figure 7.27: Example of counter based debouncing (including jump behaviour)

Note: In Figure 28, the value of Fault Detection Counter maps linearly to the internal debounce counter value.

The action DEM_DEBOUNCE_STATUS_FREEZE of the API Dem_ResetEventDebounceStatus is not relevant for counter-base debouncing. The action DEM_DEBOUNCE_STATUS_RESET of the Dem_ResetEventDebounceStatus is covered by [SWS_Dem_00684].

The DemDebounceBehavior DEM_DEBOUNCE_FREEZE is covered in chapter 7.3.6. The DemDebounceBehavior DEM_DEBOUNCE_RESET is covered by [SWS_Dem_00654] and [SWS_Dem_00677]. This is shown in Figure 29.

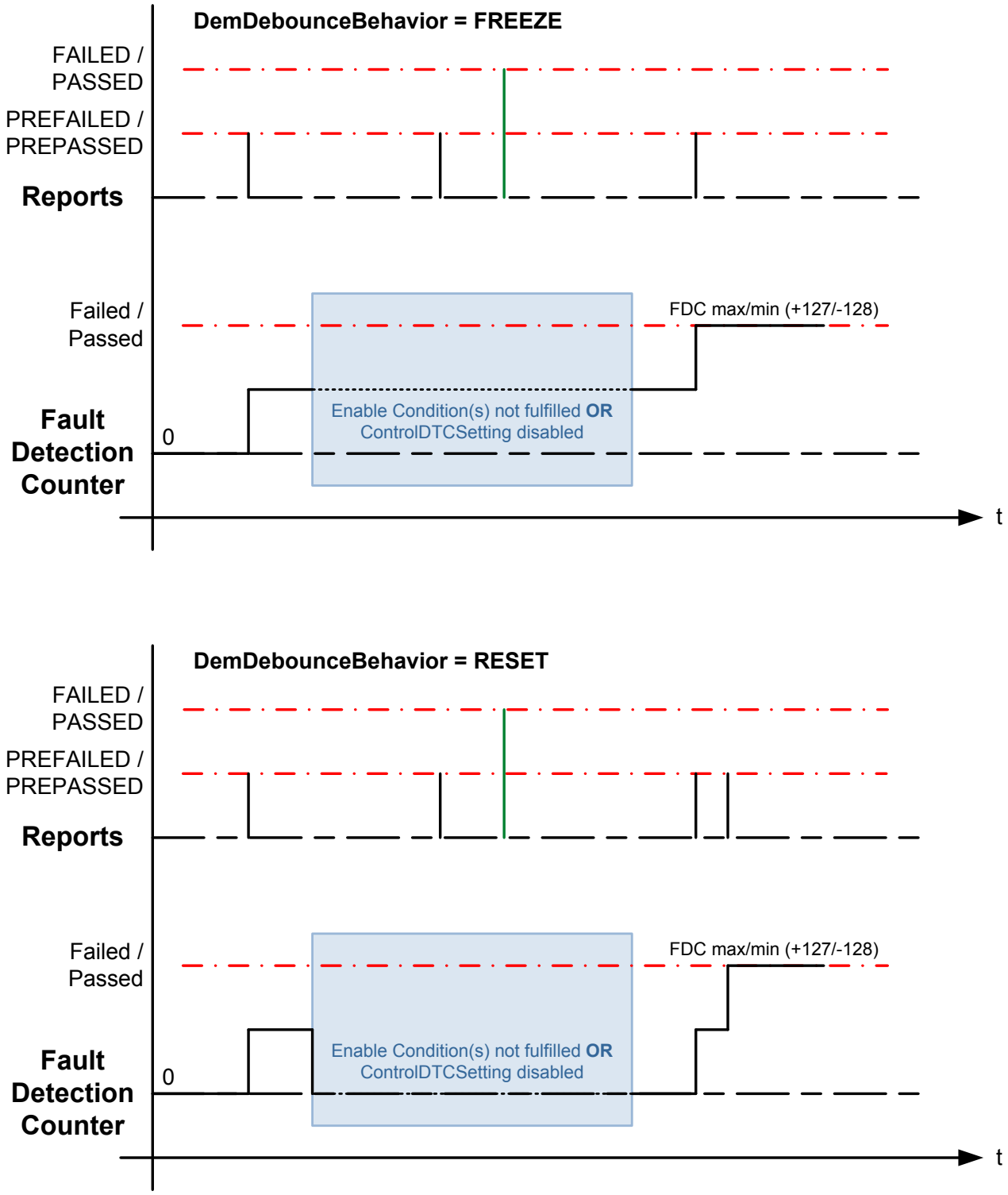

Figure 7.28: Counter based debouncing behavior on enable conditions

[SWS_Dem_00674] ⌈ If the configuration parameter DemDebounceCounterStorage is set to True, the Dem module shall store the current value of the internal debounce counter non-volatile (refer to [SWS_Dem_00341]). ⌋*(SRS_Diag_04124)*

Note: DemDebounceCounterStorage is not supported for time based debouncing.

[SWS_Dem_00675] ⌈ If the configuration parameter DemDebounceCounterStorage is set to True, the Dem module shall re-store the current value of the internal debounce counter during each startup (latest in Dem_Init). ⌋*(SRS_Diag_04124)*

Note: BSW events using non-volatile storage of the internal debounce counter cannot be reported before Dem_Init, because the value needs first to be restored.

[SWS_Dem_00676] ⌈ If development error detection is enabled and Dem_ReportErrorStatus or Dem_ResetEventDebounceStatus is called before Dem_Init or after Dem_Shutdown for events which have DemDebounceCounterStorage set to True, the Dem module shall set the error code DEM_E_WRONG_CONDITION. ⌋*(SRS_Diag_04124)*

7.3.3.2 Time based debounce algorithm

[SWS_Dem_00527] ⌈ The Dem module shall provide a configuration parameter DemDebounceTimeBasedSupport (refer to DemGeneral) to enable or disable the Dem-internal time based debouncing. ⌋*(SRS_Diag_04068)*

[SWS_Dem_00426] ⌈ If the configuration container DemDebounceAlgorithmClass is set to DemDebounceTimeBased, the Dem module shall provide an internal debounce timer for each individual event, to qualify the reported event. ⌋*(SRS_Diag_04068)*

For long debounce periods, the maximal range of the internal debounce timer is defined as sint16. This does not imply any explicit implementation.

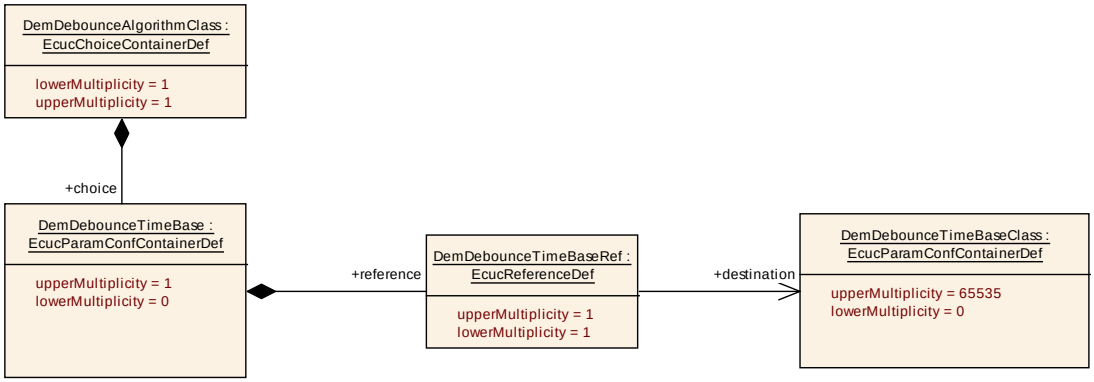

Figure 7.29: Time based debounce algorithm

[SWS_Dem_00427] ⌈ The Dem module shall calculate the fault detection counter (-128 ... +127 according to UDS) based on the value and range of the internal debounce timer, to map the internal timer values linearly to the external values (refer to Figure 31). ⌋*(SRS_Diag_04068)*

[SWS_Dem_00428] ⌈ The Dem module shall calculate the fault detection counter (-128 ... +127 according to UDS) based on the value and range of the internal debounce

timer, to map the internal timer values linearly to the external values (refer to Figure 31). ⌋(*SRS_Diag_04068*)

[SWS_Dem_00429] ⌈ If the internal debounce timer of a specific event was already triggered as pre-failed or the event is qualified as failed, and the monitor reports consecutively DEM_EVENT_STATUS_PREFAILED again, the Dem module shall not restart the internal debounce timer. ⌋(*SRS_Diag_04068*)

[SWS_Dem_00430] ⌈ If the internal debounce timer of a specific event was already triggered as pre-failed or the event is qualified as failed, and the monitor reports consecutively DEM_EVENT_STATUS_PREFAILED again, the Dem module shall not restart the internal debounce timer. ⌋(*SRS_Diag_04068*)

[SWS_Dem_00431] ⌈ If the monitor reports DEM_EVENT_STATUS_FAILED, the Dem module shall set the internal debounce timer value to its configured threshold being the failed criteria (refer to DemDebounceTimeFailedThreshold). ⌋(*SRS_Diag_04096, SRS_Diag_04068*)

[SWS_Dem_00432] ⌈ The Dem module shall start the internal debounce timer to qualify the reported event as passed when the monitor reports DEM_EVENT_STATUS_PREPASSED (refer to chapter 8.2.1.3). ⌋(*SRS_Diag_04106, SRS_Diag_04068*)

[SWS_Dem_00433] ⌈ If the internal debounce timer of a specific event was already triggered as pre-passed or the event is qualified as passed, and the monitor reports consecutively DEM_EVENT_STATUS_PREPASSED again, the Dem module shall not restart the internal debounce timer. ⌋(*SRS_Diag_04106, SRS_Diag_04068*)

[SWS_Dem_00434] ⌈ The Dem module shall provide the configuration parameter DemDebounceTimePassedThreshold used to define the event-specific delay indicating the passed status (not active). ⌋(*SRS_Diag_04106, SRS_Diag_04068*)

[SWS_Dem_00435] ⌈ If the monitor reports DEM_EVENT_STATUS_PASSED the Dem module shall set the internal debounce timer value to its configured threshold being the passed criteria (refer to DemDebounceTimePassedThreshold). ⌋(*SRS_Diag_04068*)

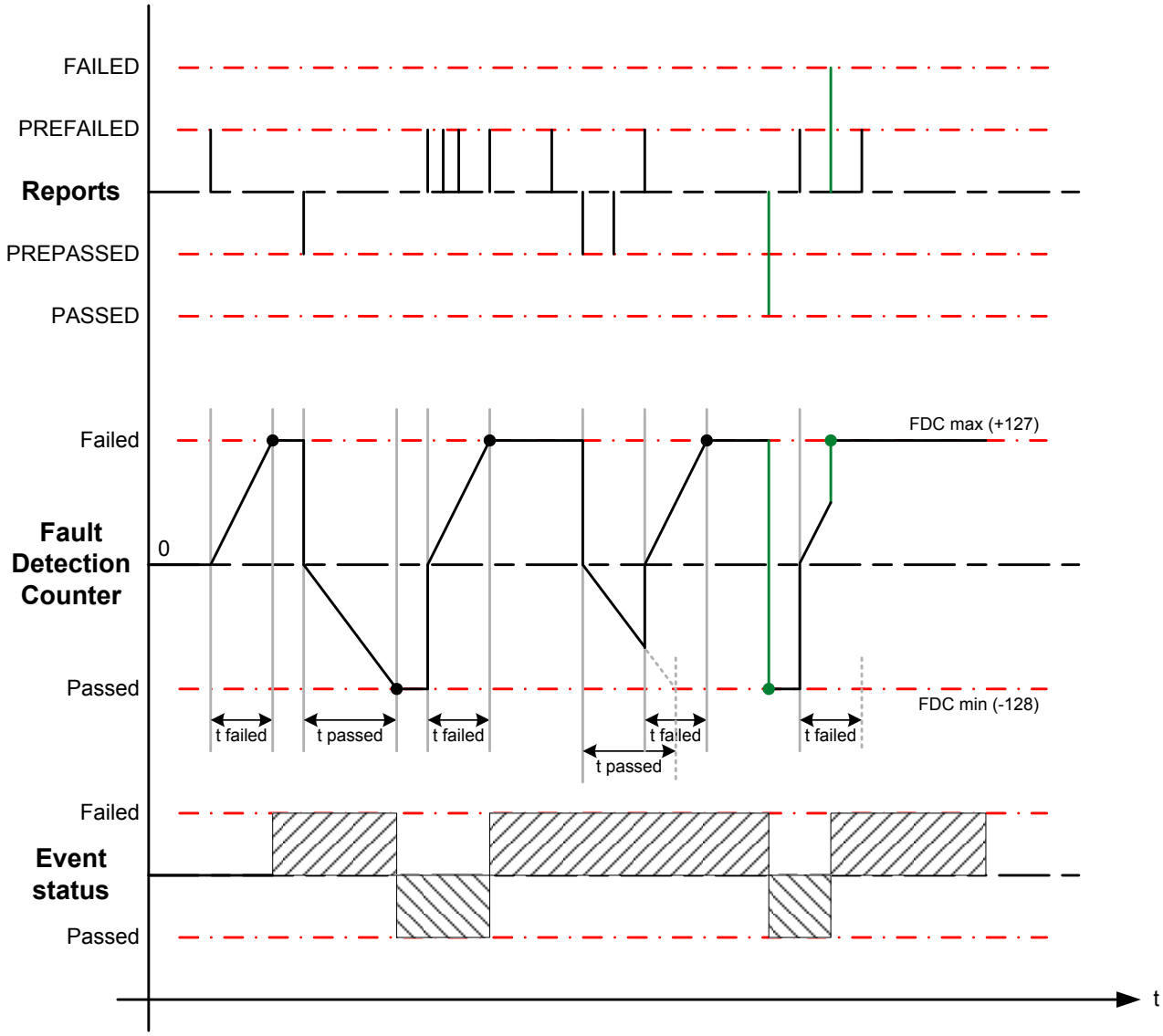

Figure 7.30: Example of time based debouncing

[SWS_Dem_00685] ⌈ If the API Dem_ResetEventDebounceStatus (refer to chapter 8.3.3.3) is called with DEM_DEBOUNCE_STATUS_FREEZE, it shall freeze the related internal debounce timer. ⌋

The action DEM_DEBOUNCE_STATUS_RESET of the API Dem_ResetEventDebounceStatus is covered by [SWS_Dem_00684].

[SWS_Dem_00655] ⌈ If the configuration parameter DemDebounceBehavior is set to DEM_DEBOUNCE_FREEZE, the Dem module shall freeze the internal debounce timer when at least one enable condition for the related event is set to not fulfilled. ⌋*(SRS_Diag_04125)*

[SWS_Dem_00678] ⌈ If the configuration parameter DemDebounceBehavior is set to DEM_DEBOUNCE_FREEZE, the Dem module shall freeze the internal de-

bounce timer when ControlDTCSetting is set to disabled for the related event. ⌋*(SRS_Diag_04125)*

[SWS_Dem_00656] ⌈ If an internal debounce timer is frozen and a new (valid) (debouncing-)result is reported for this event (as per [SWS_Dem_00428] / [SWS_Dem_00432]), the Dem module shall continue running the internal debounce timer (from current value). ⌋*(SRS_Diag_04125)*

The DemDebounceBehavior DEM_DEBOUNCE_RESET is covered by [SWS_Dem_00654] and [SWS_Dem_00677].

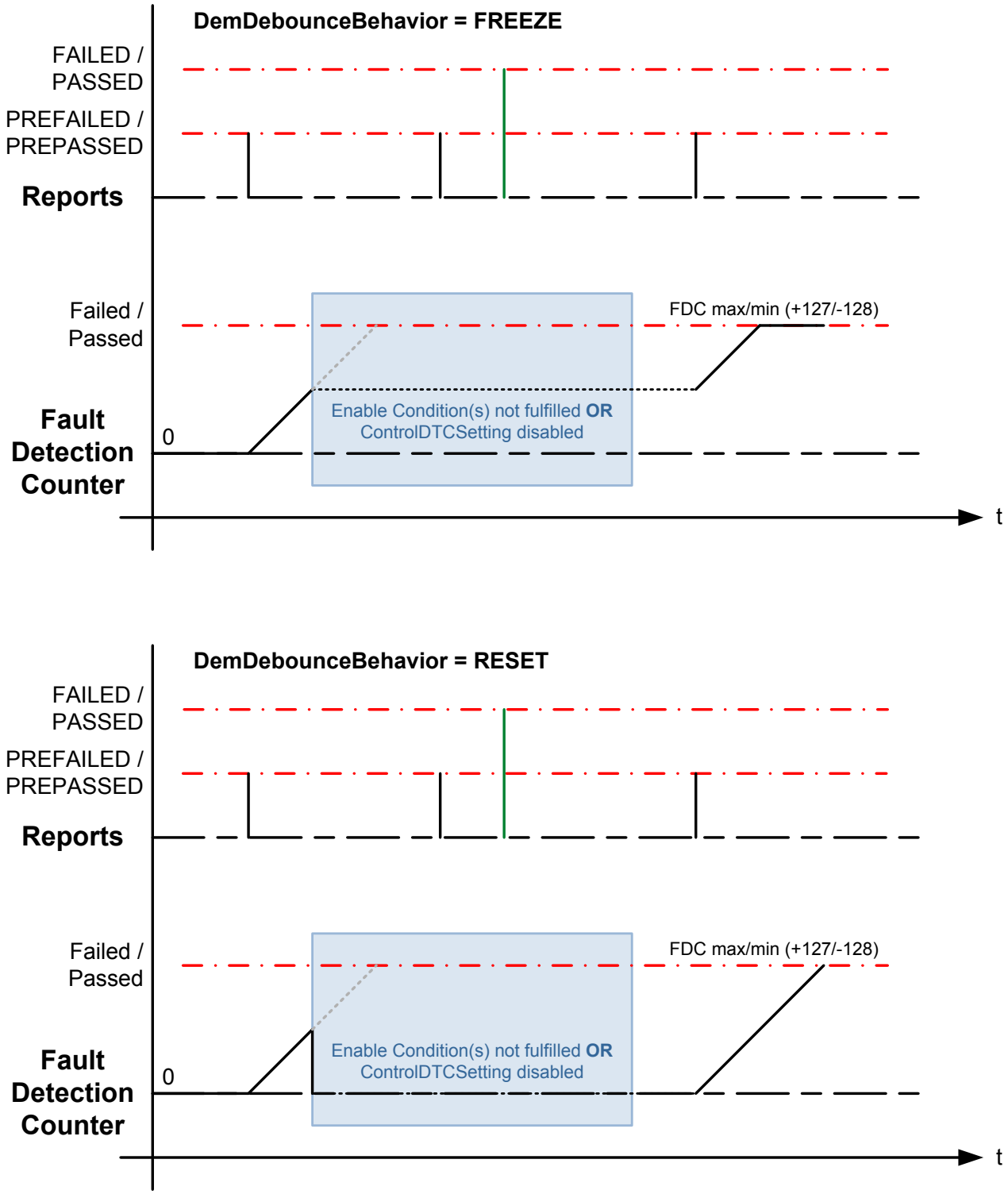

Figure 7.31: Time based debouncing behavior on enable conditions

7.3.3.3 Further specific debounce algorithms

The Dem module may be extended with other specific debounce algorithms, if an application requires particular debounce algorithms implemented Dem-internally.

[SWS_Dem_00436] ⌈ If the Dem module implements application-specific debounce algorithms, the event states DEM_EVENT_STATUS_PREFAILED and DEM_EVENT_STATUS_PREPASSED shall be used within Dem_SetEventStatus) indicating the maturation process. ⌋*(SRS_Diag_04068)*

7.3.3.4 Monitor internal debounce algorithm

[SWS_Dem_00437] ⌈ If the configuration container DemDebounceAlgorithmClass is set to DemDebounceMonitorInternal, the Dem module shall not use a Dem-internal debounce mechanism for each individual event, to qualify the reported event. ⌋*(SRS_Diag_04068)*

Note: The monitor is not allowed to report the event states DEM_EVENT_STATUS_PREFAILED and DEM_EVENT_STATUS_PREPASSED for monitor internal debouncing.

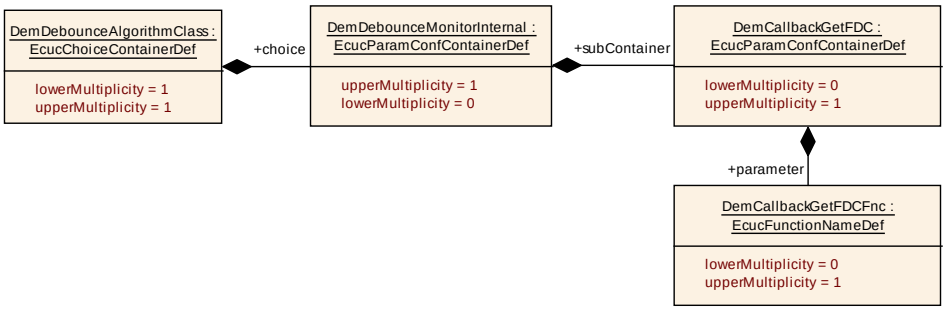

Figure 7.32: Monitor internal debounce algorithm

7.3.3.5 Debounce algorithm initialization and reset conditions

[SWS_Dem_00438] ⌈ If Dem-internal debouncing is configured, the Dem module shall reset the Dem-internal debounce algorithm when Dem_PreInit has been called. ⌋*(SRS_Diag_04106, SRS_Diag_04068)*

[SWS_Dem_00343] ⌈ After receiving a command for clearing the event memory (refer to chapter 7.3.2.2), the Dem-internal debounce algorithm shall be reset, presuming event debouncing is handled Dem-internally. ⌋*(SRS_Diag_04117, SRS_Diag_04068)*

[SWS_Dem_00684] ⌈ If the API Dem_ResetEventDebounceStatus (refer to chapter 8.3.3.3) is called with DEM_DEBOUNCE_STATUS_RESET, it shall reset the related fault detection counter. ⌋

Note: The internal debounce counter is reset to zero. The reset of an internal debounce timer will also stop this timer.

[SWS_Dem_00344] ⌈ If Dem-internal debouncing is configured, the Dem module shall reset the Dem-internal debounce algorithm upon starting a new operation cycle (refer to chapter 7.3.8). ⌋*(SRS_Diag_04106)*

Note: Resetting the debounce algorithm will also lead to a fault detection counter value of 0 (zero).

[SWS_Dem_00654] ⌈ If the configuration parameter DemDebounceBehavior is set to DEM_DEBOUNCE_RESET and the function Dem_SetEnableCondition (refer to chapter 7.3.6) sets one configured enable condition for the event to not fulfilled, it shall reset the related fault detection counter(s). ⌋*(SRS_Diag_04125)*

[SWS_Dem_00677] ⌈ If the configuration parameter DemDebounceBehavior is set to DEM_DEBOUNCE_RESET, the function Dem_DcmDisableDTCSetting (refer to chapter 7.9.2.4) shall reset the related fault detection counter(s).⌋*(SRS_Diag_04125)*

7.3.3.6 Fault detection counter retrieval

[SWS_Dem_00204] ⌈ The event-specific fault detection counter shall be accessible by using the API Dem_GetFaultDetectionCounter (refer to chapter 8.3.3.19). ⌋*(SRS_Diag_04068, SRS_Diag_04010)*

The fault detection counter can be:

1. located inside the Dem, if Dem-internal debouncing is configured (refer to chapter 7.3.3.1 and chapter 7.3.3.2).
2. located inside the monitor, if debouncing is performed by the monitor (refer to [SWS_Dem_00264], [SWS_Dem_00439], [SWS_Dem_00513]).

[SWS_Dem_00264] ⌈ If debouncing is performed by a SW-C (not handled Dem-internally), the Dem module shall use the callback-function GetFaultDetectionCounter (refer to chapter 8.4.3.4) to request the current value of the fault detection counter for a given event. ⌋*(SRS_Diag_04068, SRS_Diag_04010)*

Note: The configuration container DemCallbackGetFDC (in DemDebounceMonitorInternal) is used to specify the related port or c-callback per event.

[SWS_Dem_00439] ⌈ If the callback-function GetFaultDetectionCounter returns other than E_OK, this return value shall also be returned by the API Dem_GetFaultDetectionCounter. ⌋*(SRS_Diag_04068, SRS_Diag_04010)*

Note: For resetting the fault detection counter implemented in a monitor, the Dem module uses the callback-function InitMonitorForEvent (refer to chapter 7.5 and InitMonitorForEvent).

[SWS_Dem_00671] ⌈ If the callback-function GetFaultDetectionCounter is not configured (refer to configuration container DemCallbackGetFDC) for a given event, the API Dem_GetFaultDetectionCounter shall return DEM_E_NO_FDC_AVAILABLE. ⌋*(SRS_Diag_04010)*

7.3.3.7 Fault detection counter reporting

The maximum value the DTC Fault Detection Counter has reached during current operation cycle.

[SWS_Dem_00788] ⌈ If the maximum FDC during current operation cycle is mapped to an extended data record (DemInternalDataElement set to DEM_MAX_FDC_DURING_CURRENT_CYCLE), it shall be available for all events referencing a UDS DTC. ⌋*(SRS_Diag_04125)*

[SWS_Dem_00789] ⌈ If the UDS DTC of [[SWS_Dem_00788]] is referenced by multiple events or an combined DTC, the event with the highest maximum FDC value shall be used to report. ⌋*(SRS_Diag_04125)*

[SWS_Dem_00790] ⌈ The maximum FDC during current operation cycle shall be reset to zero with each (re-)start operation cycle (refer to DemOperationCycleRef). ⌋*(SRS_Diag_04125)*

[SWS_Dem_00791] ⌈ The maximum FDC during current operation cycle shall be update if the current fault detection counter value is greater than the current value of the maximum FDC during current operation cycle.⌋*(SRS_Diag_04125)*

The maximum value the DTC Fault Detection Counter has reached since the last time DTC information was cleared.

[SWS_Dem_00792] ⌈ If the maximum FDC since last clear is mapped to an extended data record (DemInternalDataElement set to DEM_MAX_FDC_SINCE_LAST_CLEAR), it shall be available for all events referencing a UDS DTC. ⌋*(SRS_Diag_04125)*

[SWS_Dem_00793] ⌈ If the UDS DTC of [SWS_Dem_00793] is referenced by multiple events or an combined DTC, the event with the highest maximum FDC value shall be used to report. ⌋*(SRS_Diag_04125)*

[SWS_Dem_00794] ⌈ The maximum FDC since last clear shall be reset to zero with each clear DTC command affecting this particular event. ⌋*(SRS_Diag_04125)*

[SWS_Dem_00795] ⌈ The maximum FDC since last clear shall be update if the current fault detection counter value is greater than the current value of the maximum FDC since last clear. ⌋*(SRS_Diag_04125)*

7.3.4 Fault confirmation

After reporting, a fault and entering the pending status, the fault confirmation process begins within the Dem (refer to Figure 23 General diagnostic event storage processing).

This fault confirmation process results in the confirmed state. For that purpose, respective counter thresholds and counter types are specified.

[SWS_Dem_00528] ⌈ The Dem module shall provide the configuration parameter DemEventFailureCycleCounterThreshold per event (refer to DemEventParameter) defining the maximum number of tested and failed cycles, before the event becomes “confirmed”, i.e. enters the confirmed state. ⌋*(SRS_Diag_04082)*

[SWS_Dem_00529] ⌈ The Dem module shall provide the configuration parameter DemEventFailureCycleRef per event (refer to DemEventParameter) defining the specific cycle type, which represents the trigger, that causes an update of the failure counter provided there is failed result reported for the event. ⌋

[SWS_Dem_00530] ⌈ The Dem module shall set the UDS DTC status bit 3 (ConfirmedDTC) to 1, if the respective failure counter of the event memory entry has been processed (counted further) DemEventFailureCycleCounterThreshold times. ⌋*(SRS_Diag_04067)*

Note: This cycle type could be for example equal to DEM_OPCYC_OBD_DCY or to DEM_OPCYC_POWER (refer to DemOperationCycle) and the combination of handling the counters for a specific cycle type is called “fault confirmation”.

The healing confirmation process (described in chapter 7.3.10) handles reported OK / passed results, which yields in the deactivation of a specific warning indicator. For that purpose, respective counter thresholds and counter types are specified.

Note: If the fault confirmation of an event is disabled, its UDS DTC status bit 3 (ConfirmedDTC) is set during the qualification to failed (according to [SWS_Dem_00391] and [SWS_Dem_00379]).

7.3.4.1 Method for grouping of association of events for OBD purpose

[SWS_Dem_00965] ⌈ The Dem module shall provide a configuration parameter DemOBDGroupingAssociativeEventsRef for each event to reference a representive event to one group of associate events. ⌋

Note: One event is only allowed to be reverenced to only one group of associate events. A reverenced event must refer to it self.

[SWS_Dem_00967] ⌈ For each report of a Malfunction within in a group of events DEM shall proceed cycle counter towards MIL activation, either by processing a common cycle counter or by counting the respective counter of the reported event referenceing to the event with the counter value “closest to MIL ON”. ⌋*(SRS_Diag_04069)*

[SWS_Dem_00968] ⌈ The healing / aging for each event shall to be performed individually. ⌋*(SRS_Diag_04076)*

7.3.5 Event Combination

Event combination defines the ability of the Dem module to merge several events to one DTC. It is used to adapt different monitor results to one significant fault, which is clearly evaluable in a service station. The essential part of implementing a combined DTC is the calculation of its status information. The combined DTC status byte results from a bitwise logical operation of all associated events.

[SWS_Dem_00536] ⌈ The following event combination types are supported:

1. Combination on storage: configuration parameter DemEventCombinationSupport is set to DEM_EVCOMB_ONSTORAGE. The combined DTC is stored and updated in a single event memory entry.
2. Combination on retrieval: configuration parameter DemEventCombinationSupport is set to DEM_EVCOMB_ONRETRIEVAL. Each event is stored in a separate event memory location.
3. Disabled: configuration parameter DemEventCombinationSupport is set to DEM_EVCOMB_DISABLED and event combination is not used

⌋

[SWS_Dem_00024] ⌈ In case DemEventCombinationSupport by referencing from each event to the same DTC.⌋*(SRS_Diag_04073)*

Note: For OBD it is also possible to assign multiple time the same DTC number via DemObdDTCRef. But as a difference to UDS and J1939 DTCs the OBD DTCs are reported multiple times to the DCM.

Based on the configuration of the event related data the Dem module allows two different types of event combination (assign event related data to the combine event/DTC or to the sub-events).

Note: The primary use-case of event combination is to map more than one event (represented by an EventId) to only one DTC number. If combination “on storage” is used, the Dem module uses only one event memory entry and its related data for all events combined to the DTC. If combination on retrieval is used, each event is stored in a separate event memory location with its own set of event related data. The Dem module handles the combined DTC in consideration of its related event memory entries.

Table 7.1 shows an example of a Dem configuration table including combined DTCs. Several events are mapped to the same DTC. The combination on storage is represented by the DTC[1]. The DTC[2] shows an example of the combination on retrieval. The freeze frame data is stored individual for each event.

Unique EventId	*Event status*	*DTC status*	*Assinged DTC*	*Freeze frame*	*....*
Event[1]	S1	S1|S2|S3	DTC[1]	FF[28]	
Event[2]	S2	S1|S2|S3	DTC[1]	FF[28]	
Event[3]	S3	S1|S2|S3	DTC[1]	FF[28]	
Event[4]	S4	S1|S2|S3	DTC[2]	FF[74]	
Event[5]	S5	S1|S2|S3	DTC[2]	FF[77]	
Event[6]	S6	S1|S2|S3	DTC[2]	FF[75]	
Event[7]	S7	S7	DTC[3]	FF[89]	
Event[8]	S8	S8	DTC[4]	FF[67]	
....	....	....	....	....	....

Table 7.1: Example of a Dem configuration table including combined DTCs

[SWS_Dem_00441] ⌈ The Dem module shall implement the status bit calculations for the combined DTC status byte according to Table 7.2. ⌋

DTC status bit description		*Combined DTC status information logical equation*
0	TestFailed	$CbDTC_{Bit0}$ = Event[1]$_{Bit0}$ | Event[2]$_{Bit0}$ | ... |Event[n]$_{Bit0}$
1	TestFailedThis OperationCy-cle	$CbDTC_{Bit1}$ = Event[1]$_{Bit1}$ | Event[2]$_{Bit1}$ | ... |Event[n]$_{Bit1}$
2	PendingDTC	$CbDTC_{Bit2}$ = Event[1]$_{Bit2}$ | Event[2]$_{Bit2}$ | ... |Event[n]$_{Bit2}$
3	ConfirmedDTC	$CbDTC_{Bit3}$ = Event[1]$_{Bit3}$ | Event[2]$_{Bit3}$ | ... |Event[n]$_{Bit3}$
4	TestNotCompleted SinceLastClear	$CbDTC_{Bit4}$ = Event[1]$_{Bit4}$ | Event[2]$_{Bit5}$ | ... |Event[n]$_{Bit4}$ & ! $CbDTC_{Bit4}$
5	TestFailedSince LastClear	$CbDTC_{Bit5}$ = Event[1]$_{Bit5}$ | Event[2]$_{Bit5}$ | ... |Event[n]$_{Bit5}$
6	TestNotCompleted ThisOpera-tionCycle	$CbDTC_{Bit6}$ = Event[1]$_{Bit6}$ | Event[2]$_{Bit6}$ | ... |Event[n]$_{Bit6}$ & ! $CbDTC_{Bit6}$CbDTC
7	WarningIndicator Requested	$CbDTC_{Bit7}$ = Event[1]$_{Bit7}$ | Event[2]$_{Bit7}$ | ... |Event[n]$_{Bit7}$

Table 7.2: Calculation of combined status byte

In the table above the following logical operators are used:

- ! = logical negation (NOT)
- | = logical bitwise OR-operation
- & logical bitwise AND-operation

[SWS_Dem_00440] ⌈ If the Dem module is requested to clear a combined DTC (refer to Dem_<...>ClearDTC), the Dem module shall clear all related events (refer to [SWS_Dem_01049] and [SWS_Dem_01050]) ⌋

[SWS_Dem_01049] ⌈ If the Dem module is requested to report the status of a combined DTC (refer e.g. Dem_GetStatusOfDTC or Dem_GetNextFilteredDTC), the calculation of the status byte shall be performed according [SWS_Dem_00441]. ⌋

[SWS_Dem_01050] ⌈ Each time the status of an event is updated, the combined DTC status shall be calculated. If the combined DTC status has changed the relevant callbacks shall be invoked; refer to [SWS_Dem_00284], [SWS_Dem_00986], [SWS_Dem_00987] and [SWS_Dem_00828]. ⌋

[SWS_Dem_00672] ⌈ The fault detection counter of the combined DTC shall be the maximum fault detection counter value of the sub-events. ⌋

Note: The combined fault detection counter value is required for Dem_DcmGetNextFilteredDTCAndFDC (in UDS Service 0x19 14).

[constr_6103] ⌈ In case the event combination is disabled, it is not allowed to reference from multiple events to the same dtc. ⌋

7.3.5.1 Combination On Storage

The following section describes the behavior of the Dem module in case of the combination on storage is configured.

[SWS_Dem_00163] ⌈ If the Dem module is requested to support combination on storage, the DTC status bit transitions of the combined DTC (refer [SWS_Dem_00441]) shall be used as trigger source for the allocation of the event memory entry, as well as the collection or update of its related data (freeze frames or extended data records). ⌋

[SWS_Dem_01051] ⌈ For combination on storage, the event which allocated (or reallocate) the event memory entry shall be the only event, which has the confirmed bit set. ⌋

[SWS_Dem_01052] ⌈ For combination on storage, an event that cannot set the confirmed bit due to [SWS_Dem_01051] shall still generate warningIndicatorOnCriteria according to [SWS_Dem_00501]. ⌋

[SWS_Dem_01053] ⌈ If the Dem module is requested to support combination on storage, the aging shall be calculated based on the combined DTC status (refer to [SWS_Dem_00408]) ⌋

[SWS_Dem_00442] ⌈ If a combined DTC (combination on storage) is aged, the Dem module shall remove this event memory entry and reset the status bytes of all sub-events according to [SWS_Dem_00823], [SWS_Dem_00824] and [SWS_Dem_00498]. ⌋*(SRS_Diag_04073)*

[SWS_Dem_00443] ⌈ If a combined DTC (combination on storage) is displaced, the Dem module shall remove this event memory entry and reset the status bytes of all sub-events according to [SWS_Dem_00409]. ⌋*(SRS_Diag_04118)*

[SWS_Dem_00538] ⌈ If a combined DTC memory entry (storage combination) was removed during displacement, the Dem module shall not modify the UDS status bits of the removed combined DTC, except the UDS status bits 3 (ConfirmedDTC) and 2 (PendingDTC) which are set to 0, if the configuration parameter DemResetCon-

firmedBitOnOverflow (refer to DemGeneral) is set to true. Further the UDS status bit 5 (TestFailedSinceLastClear) shall also be reset to 0 in case DemStatusBitHandlingTestFailedSinceLastClear and DemResetConfirmedBitOnOverflow (refer to DemGeneral) are set to true. ⌋*(SRS_Diag_04073)*

7.3.5.2 Combination On Retrieval

The section below describes the behavior of combination on retrieval.

[SWS_Dem_00539] ⌈ If the Dem module is requested to support combination on retrieval , the status bit transition of each event shall trigger the collection, update and storage of the related data (freeze frame / extended data). ⌋*(SRS_Diag_04073, SRS_Diag_04096)*

[SWS_Dem_00540] ⌈ If the Dem module is requested to report data of a combined DTC (combination on retrieval), the Dem module shall return the event related data of all assigned events. The Dem shall concatenate the data of each sub-event treating it like a single record. ⌋

[SWS_Dem_00541] ⌈ Aging of each event of a combined DTC (combination on retrieval) shall be processed individually according to [SWS_Dem_00493] and [SWS_Dem_00498]. ⌋*(SRS_Diag_04073)*

[SWS_Dem_00542] ⌈ For combination on retrieval, the displacement algorithm for each event of a combined DTC shall be treated individually (refer to [SWS_Dem_00408]). ⌋

7.3.6 Enable and storage conditions of diagnostic events

In certain cases, the event retention depends on parameters, which are available on operation system level. These parameters are combined to groups and define a certain number of checks (e.g. correct voltage range) before the event report is accepted or the event gets qualified. The checks are done by software components. The Dem module provides the ability to consider the reported result (a specific condition is fulfilled or not) during the event handling. There are two different types of conditions: enable conditions and storage conditions.

The strategy how to use the conditions (e.g. suppression of subsequent faults) and especially the assigning matrix of conditions to the events depends on the OEM.

The enable conditions are defined as a set of parameters, which are assigned to a specific condition. As long as this condition is not fulfilled, the event reports (refer to Dem_ReportErrorStatus and Dem_SetEventStatus) are not valid and therefore will not be accepted. It has no impact on Dem_ResetEventDebounceStatus, Dem_ResetEventStatus and Dem_<...>ClearDTC. A similar functionality is used for the function inhibition. In contrast to the mutual exclusion matrix of the FiM, which is

based on events, the enable conditions are based on system parameters (e.g. ignition status, local voltage).

The storage conditions are defined as a set of parameters, which are assigned to a specific condition. As long as this condition is not fulfilled, the Dem module does not store the event to the event memory.

The following requirements introduce the handling of the enable conditions.

[SWS_Dem_00202] ⌈ If the Dem module is requested to support enable conditions, the Dem module shall provide the API Dem_SetEnableCondition (refer to chapter 8.3.3.17) receiving the current status (condition fulfilled or not) of a specific enable condition. ⌋*(SRS_Diag_04095)*

[SWS_Dem_00446] ⌈ If the Dem module is requested to support enable conditions, the Dem module shall provide the ability to assign one enable condition group (refer to DemEnableConditionGroup), which includes one or several enable conditions (refer to DemEnableCondition) to a specific event. ⌋*(SRS_Diag_04095)*

Note: The enable condition groups are introduced to improve the configuration methodology of the Dem module. If a significant number of events depend always on the same enable conditions, it is less effort to select one of the defined enable condition groups instead of assigning several enable conditions to each of those events.

[SWS_Dem_00447] ⌈ If the Dem module is requested to support enable conditions, the Dem module shall check the assigned enable conditions after the diagnostic monitor reports an event (passed/failed or pre-passed/pre-failed, refer to APIs Dem_ReportErrorStatus and Dem_SetEventStatus)). ⌋*(SRS_Diag_04095)*

[SWS_Dem_00449] ⌈ If one enable condition is not fulfilled, all status reports from SWCs (Dem_SetEventStatus)) and BSW modules (Dem_ReportErrorStatus) for those events being assigned to this condition shall be ignored (no change of UDS DTC status byte) by the Dem. ⌋*(SRS_Diag_04095)*

Note: In case of Dem-internal debouncing the related fault detection counter will be frozen or reset (refer to chapter Figure 29 and Figure 32).

[SWS_Dem_00450] ⌈ If all event-specific enable conditions are fulfilled, all status reports from SW-Cs (Dem_SetEventStatus)) and BSW modules (Dem_ReportErrorStatus) for those events being assigned to these conditions shall be accepted by the Dem from this point in time on. ⌋*(SRS_Diag_04095)*

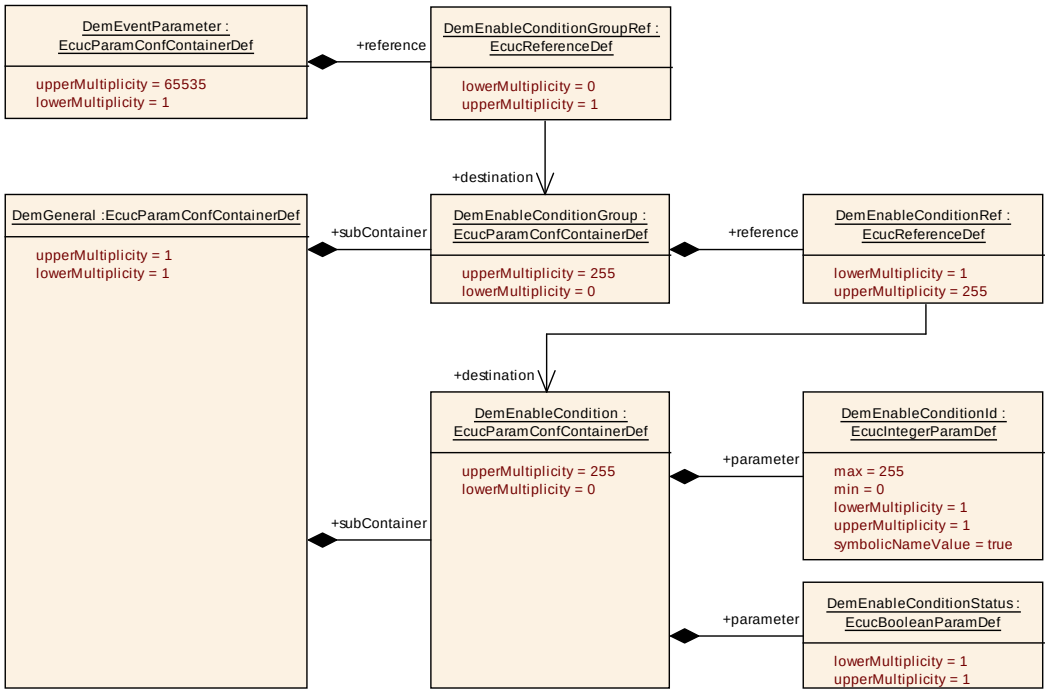

Figure 7.33: Enable condition assignment configuration

The following requirements introduce the handling of the storage conditions.

[SWS_Dem_00543] ⌈ If the Dem module is requested to support storage conditions, the Dem module shall provide the API Dem_SetStorageCondition (refer to chapter 8.3.3.18) receiving the current status (condition fulfilled or not) of a specific storage condition. ⌋*(SRS_Diag_04095)*

[SWS_Dem_00453] ⌈ If the Dem module is requested to support storage conditions, the Dem module shall provide the ability to assign one storage condition group (refer to DemStorageConditionGroup), which includes one or several storage conditions (refer to DemStorageConditionGroup) to a specific event. ⌋*(SRS_Diag_04095)*

Note: The storage condition groups are introduced to improve the configuration methodology of the Dem module. If a significant number of events depend always on the same storage conditions, it is less effort to select one of the defined storage condition groups instead of assigning several storage conditions to each of those events.

[SWS_Dem_00455] ⌈ If the Dem module is requested to support storage conditions, the Dem module shall check the assigned storage conditions after the event gets qualified as failed (UDS DTC status bit 0 changes from 0 to 1). ⌋*(SRS_Diag_04095)*

[SWS_Dem_00458] ⌈ If one storage condition is not fulfilled and no respective event memory entry exists, the Dem module shall not enter the reported event into the event memory. ⌋*(SRS_Diag_04095)*

[SWS_Dem_00591] ⌈ If one storage condition is not fulfilled and a respective event memory entry exists, the Dem module shall not update the event memory of the reported event. ⌋*(SRS_Diag_04095)*

[SWS_Dem_00459] ⌈ If all event-specific storage conditions are fulfilled, the Dem module shall permit the storage of the reported event. ⌋*(SRS_Diag_04095)*

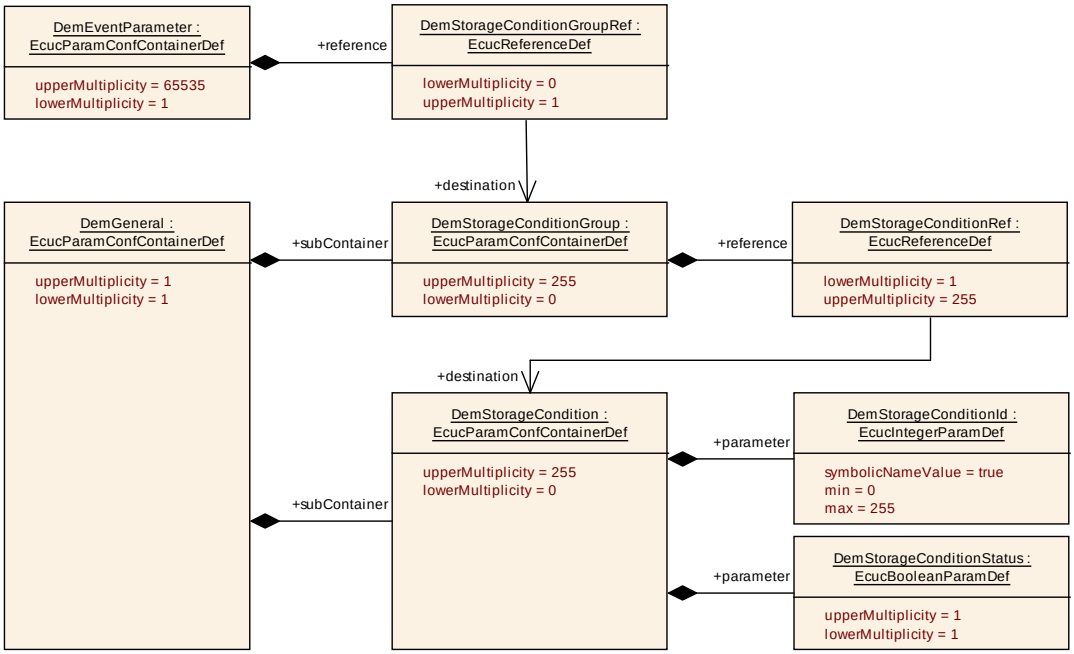

Figure 7.34: Storage condition assignment configuration

7.3.7 Event related data

'Event related data' are additional data, e.g. sensor values or time stamp/mileage, which are stored in case of an event. [2, ISO 14229-1] defines two different types of event related data: snapshot data (freeze frames) and extended data. The number or sets of stored event related data are strongly OEM / failure specific and are therefore configurable. This data is provided by SW-C or other BSW modules.

[SWS_Dem_00796] ⌈ Each event memory entry shall support the capability to store the configured 'Event related data' (freeze frame data (DTCSnapshot) or DTCExtendedData according to chapter chapter 7.3.7.4). ⌋*(SRS_Diag_04074)*

Note: The presence of a Confirmed DTC status does not necessarily mean that 'Event related data' is available as well.

The Dem module is not in charge of validity of event related data. Time consistency of event related data is depending on data source and storage time.

The Dem module provides a configuration table to combine event related data (freeze frames & extended data) with a specific DTC number, etc. (refer to chapter 7.3.7.4).

Note: This does not define a specific implementation (e.g. look-up table, matrix, etc.). Furthermore it relates to the link between the configured data. An event is characterized by its event Id, DTC value, configured freeze frames and extended data records, etc.

7.3.7.1 Storage of freeze frame data

[SWS_Dem_00039] ⌈ The Dem module shall support event-specific freeze frame storage. ⌋*(SRS_Diag_04074)*

In general, there are two options for freeze frame configuration: (1) Non-emission related freeze frames are configured specific to one particular event. The freeze frame records can be addressed by using relative numbers (calculated or configured), so the record numbers are unique to an event (not globally). (2) Emission related freeze frames are configured globally for a particular ECU (OBD legislation requires one single freeze frame class only). The freeze frame record can be addressed per event by using the value 0x00.

[SWS_Dem_00040] ⌈ The Dem module shall support the storage of one or several DIDs per freeze frame record assigned by configuration (refer to DemFreezeFrameClass). ⌋*(SRS_Diag_04074)*

Note: A freeze frame is represented as a list of DIDs (refer to DemDidClass) or PIDs (refer to DemDidClass).

Note: Due to implementation reasons, the Dem usually needs to reserve memory for the maximum freeze frame size multiplied by the number of freeze frames.

[SWS_Dem_00337] ⌈ If the Dem module uses calculated record numbers (refer to DemTypeOfFreezeFrameRecordNumeration), the Dem module shall be capable to store the configured number of freeze frames (refer to DemMaxNumberFreezeFrameRecords). ⌋*(SRS_Diag_04127)*

[SWS_Dem_00581] ⌈ If the Dem module uses calculated record numbers, the Dem module shall numerate the event-specific freeze frame records consecutively starting by 1, based on their chronological order. ⌋*(SRS_Diag_04127)*

[SWS_Dem_00582] ⌈ If the Dem module uses dedicated, configured record numbers (refer to DemTypeOfFreezeFrameRecordNumeration), the Dem module shall be capable to store per event memory entry all configured freeze frame records for this particular event (refer to DemFreezeFrameRecNumClassRef). ⌋*(SRS_Diag_04074)*

[SWS_Dem_00797] ⌈ If the FreezeFrame uses dedicated, configured record numbers (refer to DemFreezeFrameRecNumClassRef) and an event memory entry exists, the Dem module shall capture the FreezeFrame on the configured trigger (refer to DemFreezeFrameRecordTrigger) and store it to the event memory entry. ⌋*(SRS_Diag_04127)*

In case, the storage trigger was not able to allocate an event memory entry (due to event retention) there might meanwhile the possibility (due to aging) to have suitable memory entries Therefore the FreezeFrame trigger should try again to allocate an event memory entry.

[SWS_Dem_00798] ⌈ If the FreezeFrame uses dedicated, configured record numbers (refer to DemFreezeFrameRecNumClassRef) and no event memory entry exists, the Dem module shall shall first try to allocate an event memory entry

as described in [SWS_Dem_00783], [SWS_Dem_00784], [SWS_Dem_00785] and [SWS_Dem_00786]. Afterwards requirement [SWS_Dem_00797] applies. ⌋(*SRS_Diag_04127*)

[SWS_Dem_00799] ⌈ If the DemFreezeFrameRecordTrigger is set to DEM_TRIGGER_ON_FDC_THRESHOLD, the FreezeFrame shall be captured (as allowed by [SWS_Dem_00797]) each time the configured FDC threshold (refer to DemEventMemoryEntryFdcThresholdStorageValue is reached (by a positive increment), but at most once per operation cycle. ⌋(*SRS_Diag_04127*)

[constr_6102] ⌈ FreezeFrame trigger with FDC threshold is limited to counter-based debouncing. The DemFreezeFrameRecordTrigger DEM_TRIGGER_ON_FDC_THRESHOLD is only available with counter-based debouncing (DemDebounceAlgorithmClass set to DemDebounceCounterBased). ⌋

[SWS_Dem_00800] ⌈ If the DemFreezeFrameRecordTrigger is set to DEM_TRIGGER_ON_TEST_FAILED, the FreezeFrame shall be captured (as allowed by [SWS_Dem_00797]) each time the TestFailed UDS DTC status bit 0 is set (changing from 0 to 1). ⌋(*SRS_Diag_04127*)

[SWS_Dem_00801] ⌈ If the DemFreezeFrameRecordTrigger is set to DEM_TRIGGER_ON_PENDING, the FreezeFrame shall be captured (as allowed by [SWS_Dem_00797]) each time the Pending UDS DTC status bit 2 is set (changing from 0 to 1). ⌋(*SRS_Diag_04127*)

[SWS_Dem_00802] ⌈ If the DemFreezeFrameRecordTrigger is set to DEM_TRIGGER_ON_CONFIRMED, the FreezeFrame shall be captured (as allowed by [SWS_Dem_00797]) each time the Confirmed UDS DTC status bit 3 is set (changing from 0 to 1). ⌋(*SRS_Diag_04127*)

[SWS_Dem_00803] ⌈ If the FreezeFrame uses dedicated, configured record numbers (refer to DemFreezeFrameRecNumClass) and DemFreezeFrameRecordUpdate is set to DEM_UPDATE_RECORD_NO, the FreezeFrame shall be stored only if the FreezeFrame is currently not stored in this event memory entry. ⌋(*SRS_Diag_04127*)

[SWS_Dem_00804] ⌈ If the FreezeFrame uses dedicated, configured record numbers (refer to DemFreezeFrameRecNumClass) and DemFreezeFrameRecordUpdate is set to DEM_UPDATE_RECORD_YES, the FreezeFrame shall be updated with each trigger (refer to DemFreezeFrameRecordTrigger). ⌋(*SRS_Diag_04127*)

[SWS_Dem_00461] ⌈ If the Dem module uses calculated record numbers and the configuration parameter DemFreezeFrameCapture (refer to DemGeneral) is set to DEM_CAPTURE_ON_EVENT_MEMORY_STORAGE, event-specific freeze frame data shall be captured when the event memory entry is allocated. ⌋(*SRS_Diag_04127*)

[SWS_Dem_00805] ⌈ If the Dem module uses calculated record numbers and the configuration parameter DemFreezeFrameCapture (refer to DemGeneral) is set to DEM_CAPTURE_ON_TESTFAILED, event-specific freeze frame data shall be captured when the UDS DTC status bit 0 (TestedFailed) changes from 0 to 1. ⌋(*SRS_Diag_04127*)

[SWS_Dem_00261] ⌈ The Dem module shall use the C-callback ReadDataElement (refer to paragraph 8.4.3.3.4) respectively the operation ReadData of the interface DataServices_<SyncDataElement> (refer to chapter 8.6.2.13) to collect all configured data elements of the respective freeze frame. ⌋

[SWS_Dem_00806] ⌈ The Dem module may invoke the data collection according [SWS_Dem_00261] from Dem_MainFunction and not within the reporting functions. Note: To ensure a synchronous data collection the prestore freeze frame feature should be used. ⌋*(SRS_Diag_04127)*

[SWS_Dem_00463] ⌈ If the SW-C or BSW module cannot not provide the requested data (ReadDataElement returns other than E_OK), the Dem shall fill the missing data with the padding value 0xFF, reports the development error DEM_E_NODATAAVAILABLE to the Det and continues its normal operation. ⌋*(SRS_Diag_04085)*

[SWS_Dem_00585] ⌈ If the Dem module uses calculated record numbers, and if more than one freeze frame record is configured for a specific event, and this event is updated in the event memory, and all available freeze frame record slots for this event are occupied, the Dem module shall update the most recent record. ⌋*(SRS_Diag_04074)*

Note: The first freeze frame record slot will always represent the first occurrence.

7.3.7.2 Pre-storage of freeze frame data

The pre-storage of freeze frames can be used for events with highly volatile freeze frame data. With the first indication of the appearance of a specific event, even if the event is not yet de-bounced or qualified, the freeze frame data is captured (e.g. because of rapid changing of event related data during running failure monitoring phase). The pre-stored freeze frame functionality is used by monitors.

[SWS_Dem_00002] ⌈ The Dem module shall provide the configuration parameter DemFFPrestorageSupported (refer to DemEventParameter) to enable or disable pre-storage handling of freeze frames per event. ⌋

Note: If DemMaxNumberPrestoredFF (refer to DemGeneral) is set to 0, DemFFPrestorageSupported can not be enabled for any DTC.

[SWS_Dem_00334] ⌈ If any event is configured to use pre-storage of freeze frames (refer to DemFFPrestorageSupported), the Dem module shall provide the API Dem_PrestoreFreezeFrame (refer to chapter 8.3.3.5) and Dem_ClearPrestoredFreezeFrame (refer to chapter 8.3.3.6). Otherwise they are not provided. ⌋*(SRS_Diag_04127)*

[SWS_Dem_00189] ⌈ The Dem module shall provide the API Dem_PrestoreFreezeFrame (refer to chapter 8.3.3.5) to capture the pre-storage data of an event-specific freeze frame regardless of the UDS DTC status bit changes.⌋*(SRS_Diag_04074)*

[SWS_Dem_00807] ⌈ The capture (sampling) of data (using [SWS_Dem_00261]) shall be synchronous in the call of Dem_PrestoreFreezeFrame. ⌋*(SRS_Diag_04074)*

[SWS_Dem_00808] ⌈ The API Dem_PrestoreFreezeFrame shall return E_NOT_OK if no memory is available (see DemMaxNumberPrestoredFF). ⌋*(SRS_Diag_04127)*

[SWS_Dem_00464] ⌈ If a pre-stored freeze frame is available, the Dem module shall use the data of the pre-stored freeze frame instead of the current data at the point in time when the event related date is captured (refer to [SWS_Dem_00461]. ⌋*(SRS_Diag_04074)*

[SWS_Dem_00969] ⌈ The pre-stored freeze frame shall be released after the data event has been stored (event retention was successfull) or discarded (event retention failed). ⌋*(SRS_Diag_04074)*

[SWS_Dem_00191] ⌈ If no pre-stored freeze frame is available, the Dem module shall behave according to chapter 7.3.7.1 Storage of freeze frame data. ⌋

Note: The captured data while using pre-stored freeze frames can differ from the data, which is collected using the UDS DTC status bit transitions as a trigger.

Note: To ensure absence of reaction to stored freeze frames of qualified events an additional freeze frame buffer should be used. Due to restrictions in hardware usage, the amount of possible entries can be restricted. Therefore, a replacement strategy could be required.

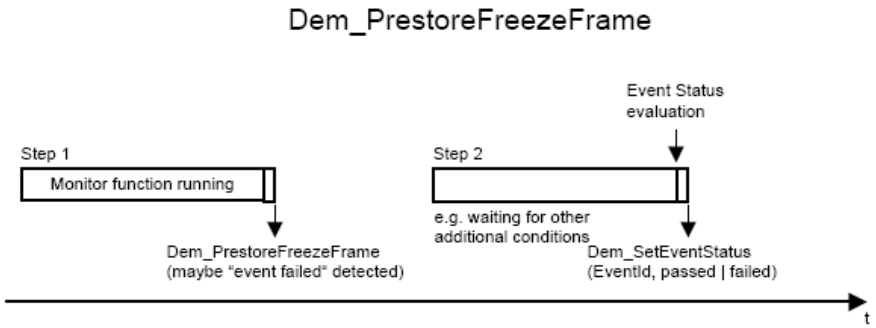

Figure 7.35: Example to use Dem_PrestoreFreezeFrame to prestore freeze frame data

[SWS_Dem_00050] ⌈ The Dem module shall provide the API Dem_ClearPrestoredFreezeFrame (refer to chapter 8.3.3.6) to release the pre-stored freeze frame for the specific event. ⌋

[SWS_Dem_00465] ⌈ If an event gets qualified as passed (UDS DTC status bit 0 changes from 1 to 0) the Dem module shall release the pre-stored freeze frame for the specific event. ⌋*(SRS_Diag_04074)*

7.3.7.3 Storage of extended data

An extended data record contains additional information associated to a specific event that is not contained in a freeze frame (extended data, e.g. frequency counters, aging counters, etc.). According to the DID- or PID-based configuration of freeze frame data, extended data are divided in extended data records defined by its record numbers.

[SWS_Dem_00809] ⌈ If an event memory entry exists, the Dem module shall capture the ExtendedDataRecord on the configured trigger (refer to DemExtendedDataRecordTrigger) and store it to the event memory entry. ⌋*(SRS_Diag_04127)*

In case, the storage trigger was not able to allocate an event memory entry (due to event retention) there might meanwhile the possibility (due to aging) to have suitable memory entries Therefore the ExtendedDataRecord trigger should try again to allocate an event memory entry.

[SWS_Dem_00810] ⌈ If no event memory entry exists, the Dem module shall first try to allocate an event memory entry as described in [SWS_Dem_00783], [SWS_Dem_00784], [SWS_Dem_00785] and [SWS_Dem_00786]. Afterwards requirement [SWS_Dem_00809] applies. ⌋*(SRS_Diag_04127)*

[SWS_Dem_00811] ⌈ If the DemExtendedDataRecordTrigger is set to DEM_TRIGGER_ON_FDC_THRESHOLD, the ExtendedDataRecord shall be captured (as allowed by [SWS_Dem_00810]) each time the configured FDC threshold (refer to DemEventMemoryEntryFdcThresholdStorageValue is reached (by a positive increment), but at most once per operation cycle. ⌋*(SRS_Diag_04127)*

[constr_6100] ⌈ ExtendedDataRecord trigger with FDC threshold is limited to counter-based debouncing. The DemExtendedDataRecordTrigger DEM_TRIGGER_ON_FDC_THRESHOLD is only available with counter-based debouncing (DemDebounceAlgorithmClass set to DemDebounceCounterBased). ⌋

[SWS_Dem_00812] ⌈ If the DemExtendedDataRecordTrigger is set to DEM_TRIGGER_ON_TEST_FAILED, the ExtendedDataRecord shall be captured (as allowed by [SWS_Dem_00810]) each time the TestFailed UDS DTC status bit 0 is set (changing from 0 to 1). ⌋*(SRS_Diag_04127)*

[SWS_Dem_00813] ⌈ If the DemExtendedDataRecordTrigger is set to DEM_TRIGGER_ON_PENDING, the ExtendedDataRecord shall be captured (as allowed by [SWS_Dem_00810]) each time the Pending UDS DTC status bit 2 is set (changing from 0 to 1). ⌋*(SRS_Diag_04127)*

[SWS_Dem_00814] ⌈ If the DemExtendedDataRecordTrigger is set to DEM_TRIGGER_ON_CONFIRMED , the ExtendedDataRecord shall be captured (as allowed by [SWS_Dem_00810]) each time the Confirmed UDS DTC status bit 3 is set (changing from 0 to 1). ⌋*(SRS_Diag_04127)*

[constr_6101] ⌈ DemExtendedDataRecordTrigger needs to be configured. DemExtendedDataRecordTrigger shall always be configured, except for internal data elements like occurrence counters. ⌋

[SWS_Dem_00815] ⌈ If the configuration parameter DemExtendedDataRecordUpdate is set to DEM_UPDATE_RECORD_NO, the ExtendedDataRecord shall be stored only if the ExtendedDataRecord is currently not stored in this event memory entry. ⌋*(SRS_Diag_04127)*

[SWS_Dem_00816] ⌈ If the configuration parameter DemExtendedDataRecordUpdate is set to DEM_UPDATE_RECORD_YES, the ExtendedDataRecord shall be updated with each trigger (refer to DemExtendedDataRecordTrigger). ⌋*(SRS_Diag_04127)*

[SWS_Dem_00282] ⌈ The Dem module shall use the C-callback ReadDataElement respectively the operation ReadData of the interface DataServices_<SyncDataElement> to collect all configured data elements which are not typed as internal data elements (refer to DemInternalDataElementClass) of the respective extended data record. ⌋

[SWS_Dem_00468] ⌈ If an event is stored or updated in the event memory, the Dem module shall store the collected extended data into the event memory entry. ⌋

[SWS_Dem_00817] ⌈ Internal data elements (refer to DemInternalDataElementClass) shall not be stored, but the current value shall be used instead. ⌋*(SRS_Diag_04127)*

7.3.7.4 Configuration of Event related data

This section describes the configuration of event related data and the access of event related data from the SW-Cs/BSW modules.

Note: The configuration model follows a flexible configuration process, but does not imply any explicit implementation.

The event related data of diagnostic events contain none or one set of extended data records (refer to DemExtendedDataClass), and none or one set of freeze frame records (refer to DemFreezeFrameClass) with its calculated or configured record numbers. Therefore a class-concept is used (refer to Figure 37).

An extended data record, a DID, or a PID can contain one ore more data elements (refer to DemDataElementClass).

Note: Asynchronous DIDs, as well as DIDs with a variable length are not supported by the Dem module, and shall therefore not be connected to the Dem.

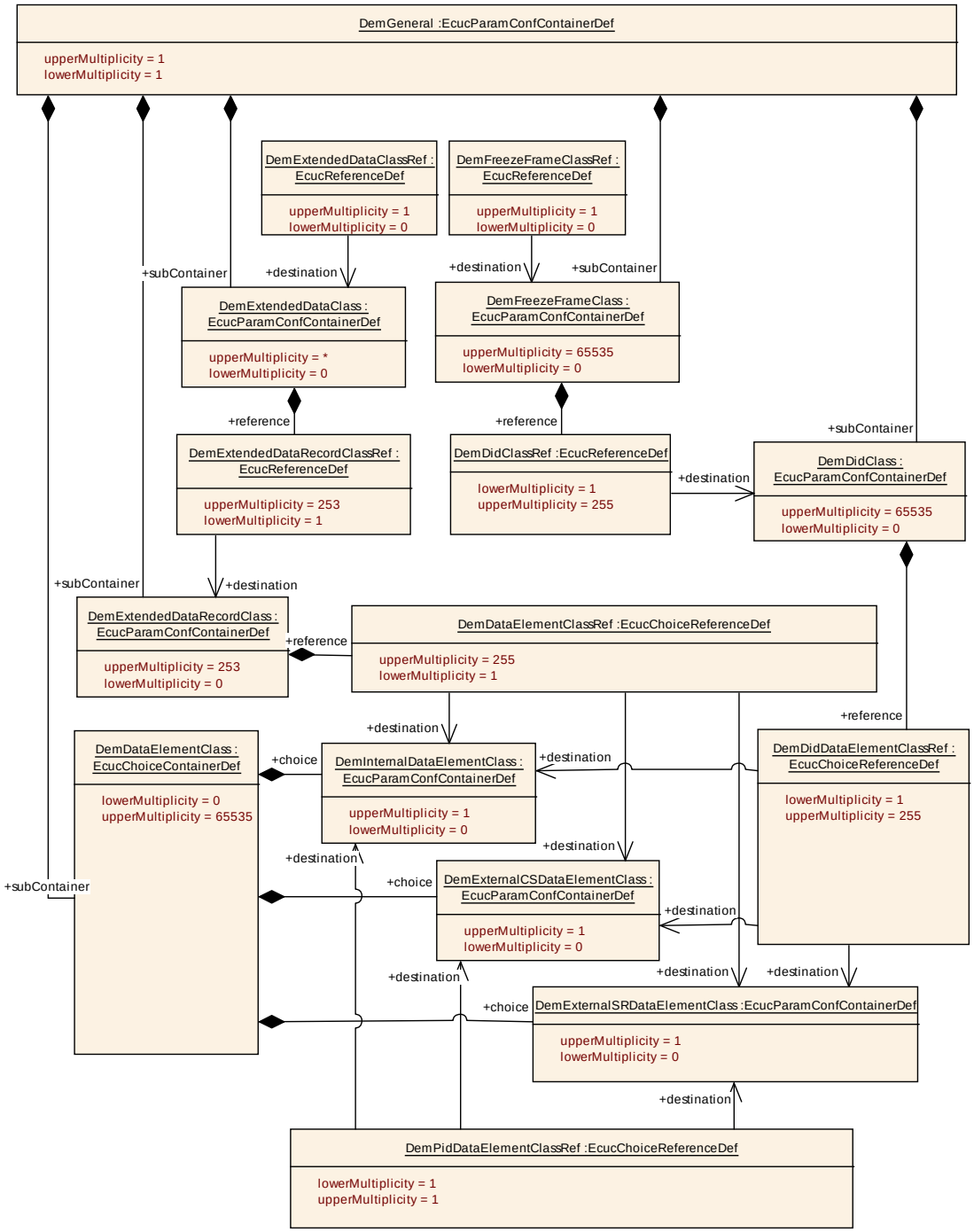

Figure 7.36: Event related data configuration

A data element is provided from a SW-C or BSW module, or is computed Dem-internally.

For each external data element a respective require-port (refer to Service Interface DataServices_<SyncDataElement>) or C-callback (refer to ReadDataElement) is generated based on the configuration DemExternalCSDataElementClass and DemExternalSRDataElementClass. For each internal data element, the respective Dem-internal value is mapped.

Note: These data elements are typically specified in a Diagnostic Data Template.

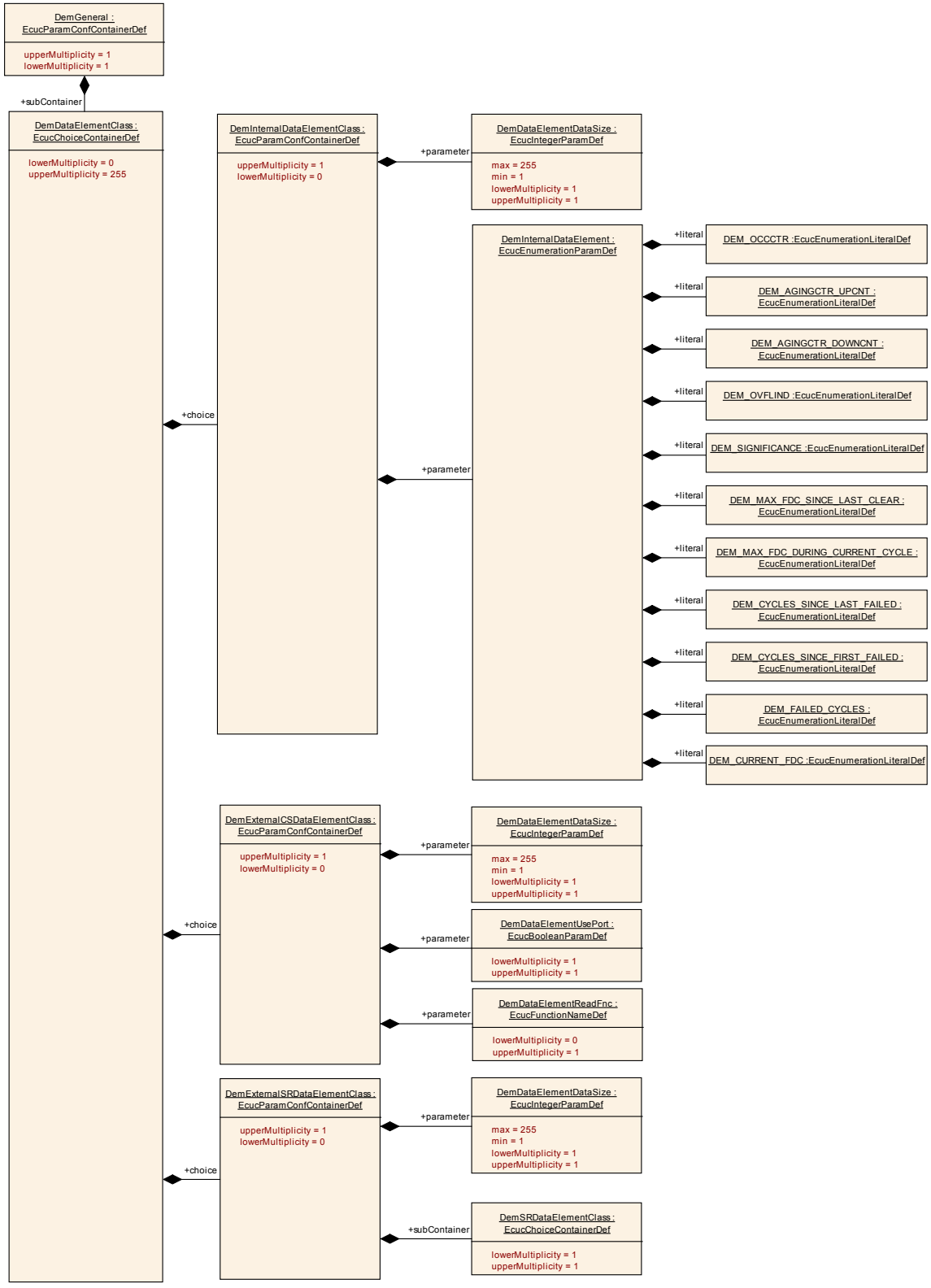

Figure 7.37: Data element configuration

[SWS_Dem_00469] ⌈ The Dem module shall provide the ability to map Dem-internal data values (e.g. aging counter, occurrence counter) to specific data element (refer to DemInternalDataElement in DemDataElementClass)contained in an extended data records. ⌋*(SRS_Diag_04104)*

Note: If a Dem-internal data element is mapped to e.g. an extended data record (by configuration), this information can simply be requested by UDS Service ReadDTCInformation - Sub-Service reportDTCExtendedDataRecordByDTCNumber (0x19, 06).

[SWS_Dem_00471] ⌈ If the configuration parameter DemInternalDataElement is set to DEM_OCCCTR, then the Dem-internal value of the occurrence counter (refer to chapter 7.1.2) shall be mapped to the respective data element. ⌋*(SRS_Diag_04127)*

[SWS_Dem_00472] ⌈ If the configuration parameter DemInternalDataElement is set to DEM_AGINGCTR_UPCNT or to DEM_AGINGCTR_DOWNCNT, then the Dem-internal value of the aging counter (refer to chapter 7.3.9) shall be mapped to the respective data element (based on [SWS_Dem_00643], [SWS_Dem_00673], [SWS_Dem_00646], [SWS_Dem_00644], or [SWS_Dem_00647]). ⌋

[SWS_Dem_00643] ⌈ If DemAgingCycleCounterProcessing is set to DEM_PROCESS_AGINGCTR_INTERN and the data element DEM_AGINGCTR_UPCNT is configured, the aging counter mapping shall be based on a count-up mechanism from 0 to DemAgingCycleCounterThreshold (refer to ISO 14229-1, Annex D). ⌋*(SRS_Diag_04106)*

[SWS_Dem_00673] ⌈ If DemAgingCycleCounterProcessing is set to DEM_PROCESS_AGINGCTR_INTERN and the data element DEM_AGINGCTR_DOWNCNT is configured, the aging counter mapping shall be based on a count-down mechanism from DemAgingCycleCounterThreshold to 0 (refer to ISO 14229-1, Annex D). ⌋*(SRS_Diag_04076)*

[SWS_Dem_00646] ⌈ If no aging is supported for an event and DemAgingCycleCounterProcessing is set to DEM_PROCESS_AGINGCTR_INTERN, the reported data element shall be 0. ⌋

[SWS_Dem_00644] ⌈ If DemAgingCycleCounterProcessing is set to DEM_PROCESS_AGINGCTR_EXTERN, the aging counter mapping shall be based on the event-specific aging counter value (refer to [SWS_Dem_00640], [SWS_Dem_00641], [SWS_Dem_00642]). ⌋*(SRS_Diag_04076)*

[SWS_Dem_00647] ⌈ If no aging is supported for an event and DemAgingCycleCounterProcessing is set to DEM_PROCESS_AGINGCTR_EXTERN, the reported data element shall be the unavailable value (255).⌋

[SWS_Dem_01043] ⌈ If DemAgingAllowed is set to ‘false’ the internal data element DEM_AGINGCTR_DOWNCNT shall be DemAgingCycleCounterThreshold if configured or ’255’ if not configured. ⌋

[SWS_Dem_01044] ⌈ If Dem DemAgingAllowed is set to ‘false’ the internal data element DEM_AGINGCTR_UPCNT shall be ‘0’. ⌋

[SWS_Dem_00473] ⌈ If the configuration parameter DemInternalDataElement is set to DEM_OVFLIND, then the Dem-internal value of the overflow indication (refer to chapter 7.3.2.3) shall be mapped to the respective data element as boolean (0 = False, 1 = True). ⌋

[SWS_Dem_00592] ⌈ If the configuration parameter DemInternalDataElement is set to DEM_SIGNIFICANCE, then the (static) Dem-internal value of the event significance (refer to chapter 7.1.4) shall be mapped to the respective data element with 0 = OCCURRENCE and 1 = FAULT. ⌋

[SWS_Dem_00818] ⌈ If the configuration parameter DemInternalDataElement is set to DEM_MAX_FDC_SINCE_LAST_CLEAR, then the Dem-internal value of the maximum Fault Detection Counter sincle last clear (refer to chapter 7.3.3.7) shall be mapped to the respective data element. ⌋*(SRS_Diag_04106)*

[SWS_Dem_00819] ⌈ If the configuration parameter DemInternalDataElement is set to DEM_MAX_FDC_DURING_CURRENT_CYCLE, then the Dem-internal value of the maximum Fault Detection Counter during current operation cycle (refer to chapter 7.3.3.7) shall be mapped to the respective data element. ⌋*(SRS_Diag_04127)*

[SWS_Dem_00820] ⌈ If the configuration parameter DemInternalDataElement is set to DEM_CYCLES_SINCE_LAST_FAILED, then the Dem-internal value of the operation cycle counter since last failed (refer to chapter 7.3.9.1) shall be mapped to the respective data element. ⌋*(SRS_Diag_04127)*

[SWS_Dem_00821] ⌈ If the configuration parameter DemInternalDataElement is set to DEM_CYCLES_SINCE_FIRST_FAILED, then the Dem-internal value of the operation cycle counter since first failed (refer to chapter 7.3.9.2) shall be mapped to the respective data element. ⌋*(SRS_Diag_04104)*

[SWS_Dem_00822] ⌈ If the configuration parameter DemInternalDataElement is set to DEM_FAILED_CYCLES, then the Dem-internal value of the failed operation cycle counter (refer to chapter 7.3.9.3) shall be mapped to the respective data element. ⌋*(SRS_Diag_04079)*

The Dem module may be extended with further specific Dem-internal data elements, if a specific configuration requires particular data values, which are computed Dem-internally.

[SWS_Dem_00470] ⌈ If the Dem module implements customer-specific Dem-internal data elements, the configuration parameter DemInternalDataElement shall be extended with the respective enumeration values. ⌋

Note: The computation of any Dem-internal data value, which is not configured as data element, can be discarded (if it is not necessary for other internal behavior handling).

[SWS_Dem_01045] ⌈ If the configuration parameter DemInternalDataElement is set to DEM_CURRENT_FDC, then the fault detection counter (refer to chapter 7.3.3) shall be mapped to the respective data element. ⌋

[SWS_Dem_00918] ⌈ The AUTOSAR DEM module shall treat the non-integer data type uint8[n] either like an integer data type of the matching size or leave the contents uninterpreted in case the Dem_DataEndianness is configured to OPAQUE. ⌋

[SWS_Dem_00919] ⌈ The AUTOSAR DEM module shall interpret opaque data as uint8[n] and shall always map it to an n-bytes sized signal. ⌋

Note: For opaque data endianness, conversion has to be configured to OPAQUE.

[SWS_Dem_00920] ⌈ The AUTOSAR DEM module shall extend the endianness conversion defined in [12] Chapter 2.4 to signed data types. ⌋

Note: In [12] Chapter 2.4 defines the endianness conversion for unsigned data types.

7.3.7.5 Notification of data changes

The Dem module shall notify other SW-Cs / BSW modules about updates of the event related data in the event memory (refer to 8.4.3.1.6 EventDataChanged).

An update of the event related data occurs every time, a new event memory entry is done or an existing is updated.

Note: The Dem module does not evaluate the return value (e.g. if other than E_OK) of this callback function.

Note: The configuration container DemCallbackEventDataChanged (in DemEventParameter) is used to specify the related port or c-callback per event.

[SWS_Dem_00475] ⌈ If 'event related data' of an 'event memory entry' is added or updated (triggering the collection of new 'event related data') AND notifications on data changes are configured via 'DemCallbackEventDataChanged' AND the DemDataElementClass of the 'event related data' is of type 'DemExternalCSDataElementClass' or 'DemExternalSRDataElementClass', the DEM shall trigger these configured event-specific notifications as well as the general notification 'GeneralCallbackEventDataChanged'. ⌋

Note: Deleted, displaced or aged 'event memory entries' do not trigger EventDataChanged callbacks.

Note: 'event related data' of type DemInternalDataElementClass do not trigger EventDataChanged callbacks.

[SWS_Dem_00479] ⌈ The function Dem_GetEventFreezeFrameData shall report the data of the DID (defined by parameter DataId) in the requested freeze frame record (defined by parameter RecordNumber, except RecordNumber equal 0xFF) of the requested diagnostic event (EventId). If the RecordNumber is equal to 0xFF and parameter DemTypeOfFreezeFrameRecordNumeration is set to DEM_FF_RECNUM_CALCULATED the most recent record shall be used, otherwise E_NOT_OK shall be returned. ⌋*(SRS_Diag_04024)*

[SWS_Dem_00991] ⌈ The format of the data in the destination buffer (DestBuffer) of the function Dem_GetEventFreezeFrameData is raw hexadecimal values and contains no header-information like RecordNumber or DataId. The size of the buffer equals to the configuration settings of all respective data elements. ⌋*(SRS_Diag_04127)*

Note: The Dcm uses the functions Dem_DcmGetOBDFreezeFrameData, Dem_DcmGetFreezeFrameDataByDTC and Dem_DcmReadDataOfOBDFreezeFrame instead of the function Dem_GetEventFreezeFrameData. The Dlt uses the function Dem_DltGetMostRecentFreezeFrameRecordData instead.

[SWS_Dem_00477] ⌈ The function Dem_GetEventExtendedDataRecord shall report the data of the extended data record of the requested diagnostic event. ⌋*(SRS_Diag_04102)*

[SWS_Dem_00989] ⌈ The format of the data in the destination buffer (DestBuffer) of the function Dem_GetEventExtendedDataRecord is raw hexadecimal values and contains no header-information like RecordNumber. The size of the buffer equals to the configuration settings of all respective data elements. ⌋

Note: The Dcm uses the function Dem_DcmGetExtendedDataRecordByDTC instead of the function Dem_GetEventExtendedDataRecord. The Dlt uses the function Dem_DltGetAllExtendedDataRecords instead.

[SWS_Dem_00995] ⌈ If the interfaces Dem_GetEventFreezeFrameData and Dem_GetEventExtendedDataRecord are called in the context of GeneralCallbackEventDataChanged or CallbackEventDataChanged, the data of the triggering event shall be retrievable (return value E_NOT_OK is not allowed). ⌋*(SRS_Diag_04074, SRS_Diag_04104)*

[SWS_Dem_00996] ⌈ Dem_GetEventFreezeFrameData may return E_NOT_OK if the requested FreezeFrame data is currently not accessible due to asynchronous processing (e.g. preempted data retrieval from application). ⌋

[SWS_Dem_00997] ⌈ Dem_GetEventExtendedDataRecord may return E_NOT_OK if the requested data is currently not accessible due to asynchronous processing (e.g. preempted data retrieval from application). ⌋*(SRS_Diag_04074)*

7.3.8 Operation cycle management

The Dem module uses different operation cycles (ref. to ISO 14229-1). Those cycles could either be provided by other BSW modules and SW-C or generated by the Dem module itself.

Examples of operation cycles are: -ignition on/off cycle - power up/power down cycle - OBD driving cycle - engine warm up cycle - operation active/passive cycle - accumulated operating time

The Dem operation cycle management processes these different types of operation cycle definitions to create Dem-specific operation cycle state information, or accepts cy-

cle state changes via the API Dem_SetOperationCycleState (refer to chapter 8.3.3.7). The OBD driving cycle (DEM_OPCYC_OBD_DCY) behaves differently than other driving cycles. Some requirements differ for “OBD” and “non-OBD” cycle.

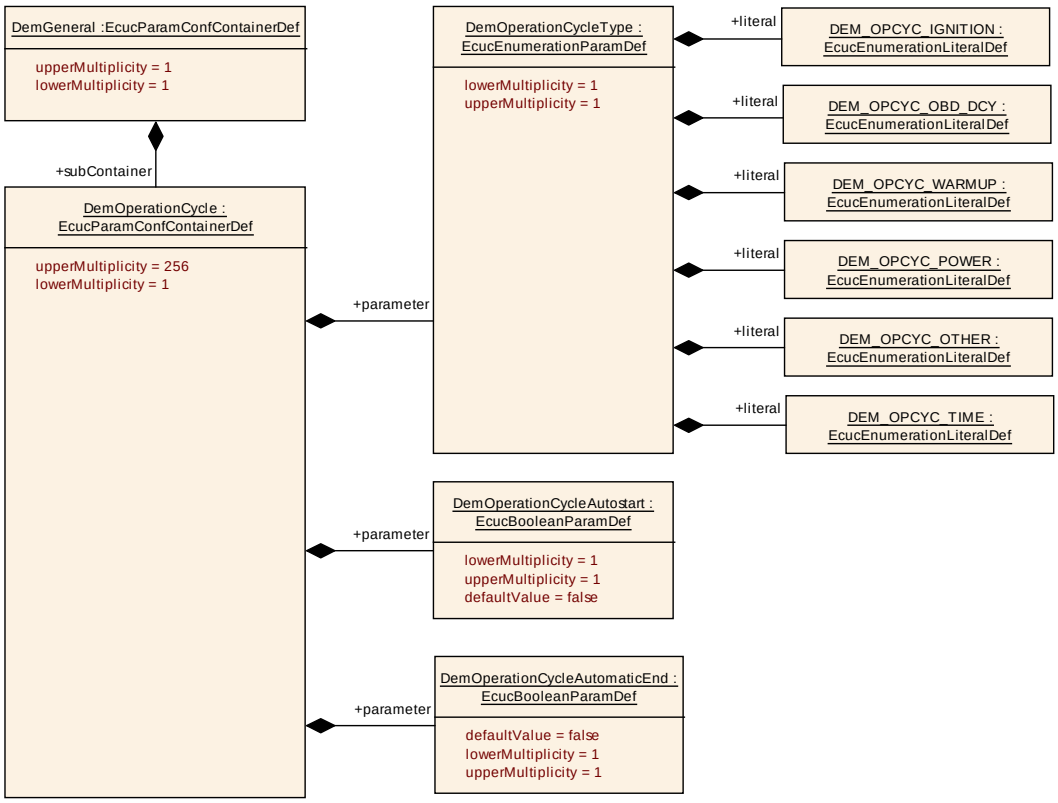

Figure 7.38: Operation cycle configuration

[SWS_Dem_00577] ⌈ The Dem module shall provide the configuration parameter DemOperationCycleStatusStorage (refer to DemGeneral) to define if the operation cycle state shall be available over the power cycle (stored non-volatile) or not. ⌋*(SRS_Diag_04076)*

[SWS_Dem_00480] ⌈ The operation cycle management of the Dem module shall handle different types of operation cycle definitions, which are defined by the configuration parameter DemOperationCycleType (refer to DemOperationCycle). ⌋*(SRS_Diag_04076)*

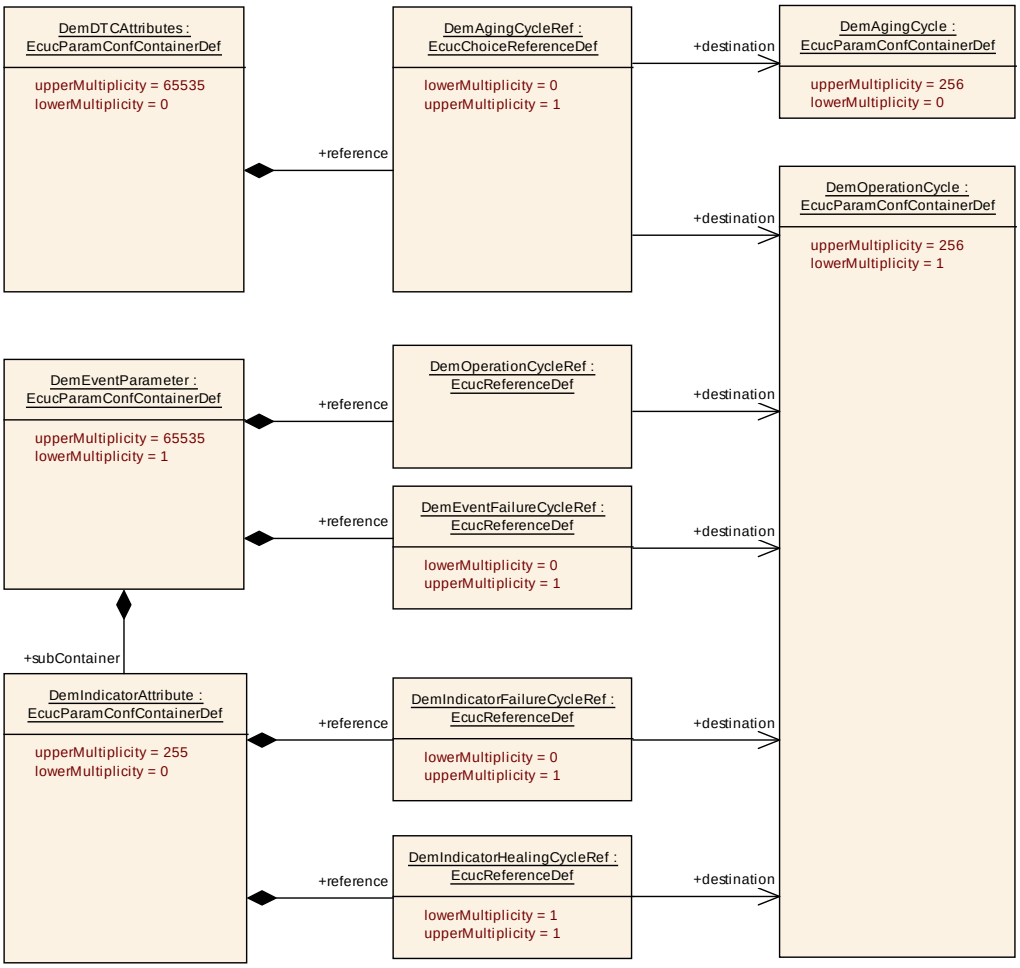

Figure 7.39: Operation and aging cycle assignment

[SWS_Dem_00853] ⌈ For all operation cycles with DemOperationCycleAutostart set to true Dem_Init shall start the operation cycle (identical to Dem_SetOperationCycleState is called with parameter DEM_CYCLE_STATE_START) with the exception of executing all callbacks resulting from UDS status byte changes. ⌋

Note: Each call Dem_SetOperationCycleState with parameter DEM_CYCLE_STATE_START would re-start the operation cycle, wherefore the start-up sequence should not explicitly start auto-started cycles.

Note: The DemOperationCycleType DEM_OPCYC_POWER should use the DemOperationCycleAutostart feature.

[SWS_Dem_00481] ⌈ If an operation cycle has started, all status reports from SW-Cs (Dem_SetEventStatus)) and BSW modules (Dem_ReportErrorStatus) for those events, being assigned to this cycle, shall be accepted by the Dem from this point in time on. ⌋*(SRS_Diag_04076)*

[SWS_Dem_00699] ⌈ At the moment the OBD driving cycle is started, the Dem shall execute computations required to reach the confirmed states of Events collected during the “ended” phase of the cycle. ⌋*(SRS_Diag_04123)*

[SWS_Dem_00482] ⌈ If an non-OBD operation cycle has ended, all status reports from SW-Cs (Dem_SetEventStatus)) and BSW modules (Dem_ReportErrorStatus) for those events being assigned to this cycle shall be ignored (no change of UDS DTC status byte) by the Dem. ⌋*(SRS_Diag_04076)*

The operation cycle status has no impact on Dem_ResetEventDebounceStatus and Dem_ResetEventStatus.

[SWS_Dem_00700] ⌈ If the OBD driving cycle has ended, status reports shall be processed, except update of the confirmed status. ⌋*(SRS_Diag_04067, SRS_Diag_04129)*

OBD legislation requires specific implementation of operation cycles. For emission related ECUs it is mandatory to implement these accordingly.

[SWS_Dem_00338] ⌈ The operation cycle management of the Dem module shall use the reported state (DEM_CYCLE_STATE_START / DEM_CYCLE_STATE_END) of the API Dem_SetOperationCycleState (refer to chapter 8.3.3.7) to set the Dem specific operation cycle state (started / ended). ⌋*(SRS_Diag_04076)*

Note: This API is called by the SW-Cs / BSW modules, as soon as those detect the status change of the monitored operation cycles. The part of this API, which will be handled asynchronously, is implementation-specific.

The operation cycle reporting can be Dem-internal for Dem module self-calculated operation cycles.

[SWS_Dem_00697] ⌈ Dem shall provide a configuration parameter DemOperationCycleAutomaticEnd (refer to ECUC_Dem_00837) to automatically END the OBD driving cycle at initialization time. The implementation may either END the cycle either at the beginning of Dem_Shutdown() or during Dem_Init(). If the cycle is ENDed at Dem_Init(), any event status change received before (after Dem_PreInit()) shall be processed after the cycle is ENDed, i.e., be considered for the new cycle. ⌋*(SRS_Diag_04001)*

Note: If the control unit is restarted (e.g., by turning the ignition key from the OFF to the ON position) before Dem_Shutdown() is processed, the Dem will usually receive a END from the OBD driving cycle defining SW-C.

[SWS_Dem_00483] ⌈ If the API Dem_SetOperationCycleState is called with DEM_CYCLE_STATE_START and the respective operation cycle was already started, the operation cycle shall be restarted (started again). ⌋*(SRS_Diag_04076)*

Note: This will be the case, if the end- and start-criteria of an operation cycle is fullfilled at exactly the same point in time. Therefore, the caller of Dem_SetOperationCycleState needs only to report “start”.

[SWS_Dem_00484] ⌈ If the API Dem_SetOperationCycleState is called with DEM_CYCLE_STATE_END and the respective operation cycle was already ended, the API shall perform no further action. ⌋*(SRS_Diag_04076)*

Note: The operation cycle state may not be started during Dem_PreInit via interface Dem_SetOperationCycleState. Also persistently stored cycle states cannot be ensured to be restored until end of initialization.

[SWS_Dem_00988] ⌈ If any as volatile configured operation cycle is running (configuration parameter DemOperationCycleStatusStorage is set to false) while Dem_Shutdown is called, the Dem shall trigger the Det error DEM_E_OPERATION_CYCLE_STARTED.⌋*(SRS_Diag_04057)*

7.3.9 Operation Cycle Counters

7.3.9.1 Cycles since last failed

The counter Cycles since last failed is representing the number of operation cycles since the DTC fault detection counter last reached its maximum value +127 (since DTC information was last cleared). All operation cycles, including those during which the test was not completed shall be included.

[SWS_Dem_00984] ⌈ If the counter 'Cycles since last failed' is mapped to an extended data record (DemInternalDataElement set to DEM_CYCLES_SINCE_LAST_FAILED), it shall be available per 'event related data' record. ⌋

[SWS_Dem_00771] ⌈ If internal debounce counter reach DemDebounceCounterFailedThreshold (latest UDS DTC status bit 0 changes from 0 to 1) and this counter is not stored in event memory and there are available event memory entry, new entry shall be allocated and the counter shall be started and initialized to zero. ⌋

[SWS_Dem_00772] ⌈ If internal debounce counter reach DemDebounceCounterFailedThreshold (latest UDS DTC status bit 0 changes from 0 to 1) and this counter is stored in event memory the counter shall initialized to zero. ⌋*(SRS_Diag_04068)*

[SWS_Dem_00773] ⌈ In case the counter is available and started, it shall be incremented at the end of the referenced operation cycle (refer to DemOperationCycleRef). ⌋*(SRS_Diag_04076)*

[SWS_Dem_00774] ⌈ The counter shall be implemented as one byte. If any count operation occurs which would cause a counter to roll over past 0xFF then the count value shall instead be maintained at 0xFF. ⌋*(SRS_Diag_04068)*

7.3.9.2 Cyles since first failed

The counter Cyles since first failed is representing the number of operation cycles since the DTC fault detection counter first reached its maximum value of +127 (since DTC information was last cleared). All operation cycles, including those in which the test has not been completed shall be included.

[SWS_Dem_00775] ⌈ If the counter 'Cyles since first failed' is mapped to an extended data record (DemInternalDataElement set to DEM_CYCLES_SINCE_FIRST_FAILED), it shall be available per 'event related data' record. ⌋*(SRS_Diag_04074)*

[SWS_Dem_00776] ⌈ If internal debounce counter reach DemDebounceCounterFailedThreshold (latest UDS DTC status bit 0 changes from 0 to 1) and this counter is not stored in event memory and there are available event memory entry, new entry shall be allocated and the counter shall be started and initialized to zero. ⌋*(SRS_Diag_04125)*

[SWS_Dem_00777] ⌈ In case the counter is available and started, it shall be incremented at the end of the referenced operation cycle (refer to DemOperationCycleRef). ⌋

[SWS_Dem_00778] ⌈ The counter shall be implemented as one byte. If any count operation occurs which would cause a counter to roll over past 0xFF then the count value shall instead be maintained at 0xFF. ⌋*(SRS_Diag_04068)*

7.3.9.3 Failed cycles

The counter Failed cycles is representing the number of operation cycles during which the DTC fault detection counter reached its maximum value of +127 (since DTC information was last cleared)

[SWS_Dem_00779] ⌈ If the counter 'Failed cycles' is mapped to an extended data record (DemInternalDataElement set to DEM_FAILED_CYCLES), it shall be available per 'event related data' record. ⌋*(SRS_Diag_04068, SRS_Diag_04104)*

[SWS_Dem_00780] ⌈ If internal debounce counter reach DemDebounceCounterFailedThreshold (latest UDS DTC status bit 0 changes from 0 to 1) and this counter is not stored in event memory and there are available event memory entry, new entry shall be allocated and the counter shall be started and initialized to zero. ⌋

[SWS_Dem_00781] ⌈ In case the counter is available and started, it shall be incremented at the end of the referenced operation cycle (refer to DemOperationCycleRef) in case the UDS DTC status bit 1 (TestFailedThisOperationCycle) is set to 1. ⌋

[SWS_Dem_00782] ⌈ The counter shall be implemented as one byte. If any count operation occurs which would cause a counter to roll over past 0xFF then the count value shall instead be maintained at 0xFF. ⌋*(SRS_Diag_04124)*

7.3.10 Aging of diagnostic events

The Dem module provides the ability to remove a specific event from the event memory, if its fault conditions are not fulfilled for a certain period of time (operation cycles). This process is called as "aging" or "unlearning".

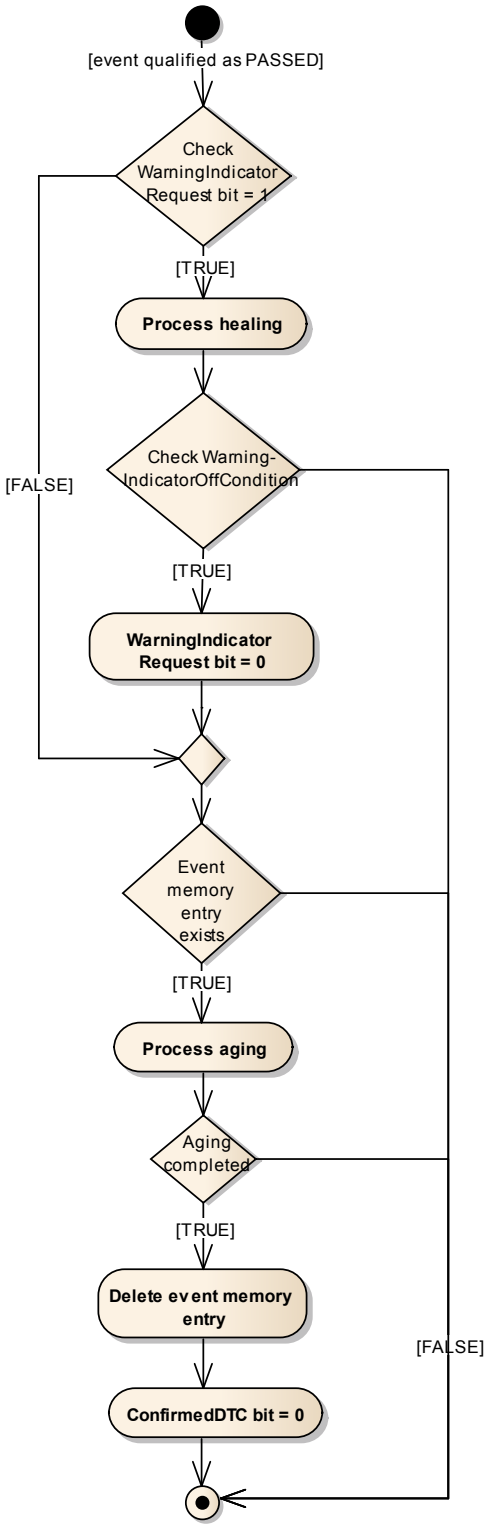

Figure 7.40: General diagnostic event deletion processing

[SWS_Dem_00698] ⌈ The process of aging (counting of aging counter) starts when healing is completed (MIL is off and WarningIndicatorRequested bit == 0 , refer to Figure 7.40). ⌋*(SRS_Diag_04076)*

[SWS_Dem_00019] ⌈ The Dem module shall support at least for each event memory entry aging counter. ⌋*(SRS_Diag_04068, SRS_Diag_04106, SRS_Diag_04076)*

[SWS_Dem_00985] ⌈ The aging counter shall be calculated based on the referenced aging/operation cycle (refer to configuration parameter DemAgingCycleRef) or on the external aging cycle, if aging is enabled (refer to DemAgingAllowed) for this event. ⌋*(SRS_Diag_04068, SRS_Diag_04106, SRS_Diag_04076)*

[SWS_Dem_00488] ⌈ The Dem module shall provide the configuration parameter DemAgingCycleCounterProcessing (refer to DemGeneral) defining the handling of the aging counter value, which is either internally derived by a status change of the defined aging/operation cycle or provided externally by Dem_SetAgingCycleCounterValue (refer to [SWS_Dem_00491]) ⌋

[SWS_Dem_00492] ⌈ The Dem module shall be able to cover the current value of the aging counter of each individual event memory entry, to support an output.⌋

Note: For extended fault analysis, it is possible to map the current value of the aging counter to a specific data element of an extended data record (refer to chapter 7.3.7.3 Storage of extended data).

[SWS_Dem_00493] ⌈ The Dem module shall provide the configuration parameter DemAgingCycleCounterThreshold (refer to DemDTCAttributes) defining the number of passed aging cycles, after which the event memory entry shall be deleted (aged) from the event memory. ⌋

[SWS_Dem_00823] ⌈ If configuration parameter DemResetConfirmedBitOnOverflow is set to false and in case an event has UDS status bit 3 (ConfirmedDTC) set and gets qualified as passed and is not stored in an event memory entry the Dem module shall shall try to allocate an event memory entry to get an aging counter. ⌋*(SRS_Diag_04096)*

Note: If it is not possible to allocate an event memory entry, the aging delays until an event memory entry becomes available (either by [SWS_Dem_00742] or by [SWS_Dem_00743]).

[SWS_Dem_00824] ⌈ If configuration parameter DemResetConfirmedBitOnOverflow is set to false and an event memory entry aging occures the Dem module shall check for other events having UDS status bit 3 (ConfirmedDTC) set to 1 and UDS status bit 0 (TestFailed) set to 0. ⌋*(SRS_Diag_04096)*

Note: The priorizitation which event is choosen by [SWS_Dem_00743] is implementation specific.

[SWS_Dem_00825] ⌈ If an event is found in [SWS_Dem_00743] the Dem module shall allocate this event memory entry (of [SWS_Dem_00493]) for this found event to process the aging for this event. ⌋*(SRS_Diag_04076)*

Note: In case of OBD-relevant events, the aging cycle will be based on cycles defined by OBD legislation.

[SWS_Dem_00498] ⌈ Upon event aging, the UDS status bit 3 (ConfirmedDTC) shall be set to 0. ⌋*(SRS_Diag_04096)*

Note: All other UDS status bits are not modified by aging of corresponding event memory entry.

[SWS_Dem_00161] ⌈ The Dem module shall handle the reoccurrence of unlearned events like new events, since they were previously deleted from the event memory by aging. ⌋*(SRS_Diag_04070)*

[SWS_Dem_00489] ⌈ The Dem module shall only allow processing (counting further) the value of the aging counter, if the related event is stored in the event memory and is qualified as passed. ⌋

Note: Aging is independent of the UDS status bit 3 (ConfirmedDTC) and therefore independent of the fault confirmation (refer to chapter 7.3.4).

[SWS_Dem_01054] ⌈ Upon event aging, the UDS status bit 5 (TestFailedSinceLastClear) shall be set to 0 if DemStatusBitHandlingTestFailedSinceLastClear (refer to DemGeneral) is set to DEM_STATUS_BIT_AGING_AND_DISPLACEMENT. ⌋

Note : The other UDS status bits are not modified by aging.

7.3.10.1 Internal aging

[SWS_Dem_00494] ⌈ The Dem module shall provide the configuration parameter DemAgingCycleRef (refer to DemDTCAttributes) defining the event-specific operation/aging cycle, whose status change triggers the processing (counting further) of the aging counter value. ⌋

Note: Refer to chapter 7.3.8 for the handling of the operation cycle.

[SWS_Dem_00490] ⌈ If aging is processed Dem-internally (DemAgingCycleCounterProcessing is set to DEM_PROCESS_AGINGCTR_INTERN) and configuration parameter DemAgingRequieresTestedCycle (refer to DemGeneral) set to False, the Dem module shall process (count further) the aging counter value, if the respective aging cycle ends/restarts. ⌋

Note: The aging counter in [SWS_Dem_00490] is processed also if no new test result is available in the respective aging cycle.

[SWS_Dem_00826] ⌈ If aging is processed Dem-internally (DemAgingCycleCounterProcessing is set to DEM_PROCESS_AGINGCTR_INTERN) and configuration parameter DemAgingRequieresTestedCycle (refer DemGeneral) set to True, the Dem module shall process (count further) the aging cycle counter value, if the respective aging cycle ends/restarts and the UDS status bit 6 (TestNotCompleteThisOperationCycle) is set to zero. ⌋*(SRS_Diag_04076)*

Note: It is possible to define an event-specific aging cycle (refer to DemAgingCycle), which does not correspond to the event-specific operation cycle. In this case, the Dem module is not able to trigger the event-specific aging counter by using the API Dem_SetOperationCycleState (refer to chapter 8.3.3.7).

[SWS_Dem_00496] ⌈ If any configured event-specific aging cycle differs from the event-specific operation cycle, the Dem module shall provide the API Dem_SetAgingCycleState (refer to chapter 8.3.3.10 and chapter 8.3.3.9) to indicate the status change of the defined aging cycle. ⌋

7.3.10.2 External aging

[SWS_Dem_00491] ⌈ If an external aging counter value is configured (DemAgingCycleCounterProcessing is set to DEM_PROCESS_AGINGCTR_EXTERN), the Dem module shall provide the API Dem_SetAgingCycleCounterValue (refer to chapter 8.3.3.9) to get the current value of the aging counter reported centrally. ⌋

[SWS_Dem_00639] ⌈ If Dem_SetAgingCycleCounterValue is called the Dem module shall process aging with the handed parameter AgingCycleCounterValue refer to [SWS_Dem_00493] for all events, fulfilling [SWS_Dem_00489]. ⌋*(SRS_Diag_04076)*

Note: If DemAgingCycleCounterProcessing is set to DEM_PROCESS_AGINGCTR_EXTERN, no event-specific aging cycle (DemAgingCycleRef) and no usage of the API Dem_SetAgingCycleState is available.

[SWS_Dem_00640] ⌈ If DemAgingCycleCounterProcessing is set to DEM_PROCESS_AGINGCTR_EXTERN and Dem_SetAgingCycleCounterValue was never invoked since the first start of the Dem module, the aging counter shall be set to be unavailable (255). ⌋

[SWS_Dem_00641] ⌈ If DemAgingCycleCounterProcessing is set to DEM_PROCESS_AGINGCTR_EXTERN and the UDS DTC Status Bit 0 (TestFailed) changes from 0 to 1, the event-specific aging counter shall be set to the latest reported external aging counter value. ⌋*(SRS_Diag_04076)*

[SWS_Dem_00642] ⌈ If DemAgingCycleCounterProcessing is set to DEM_PROCESS_AGINGCTR_EXTERN and the UDS DTC Status Bit 0 (TestFailed) changes from 1 to 0, the event-specific aging counter shall be set according to the following calculation: event-specific aging counter = ("latest reported external aging counter value" + DemAgingCycleCounterThreshold) % ("maximum value (254)"' + 1). ⌋*(SRS_Diag_04076)*

Note: For external aging, the value 256 cannot be used for DemAgingCycleCounterThreshold.

7.3.11 Healing of diagnostic events

The Dem module provides the ability to activate and deactivate indicators per event stored in the event memory. The process of deactivation is defined as healing of a diagnostic event (refer to Figure 7.40).

[SWS_Dem_01056] ⌈ The Dem shall process Healing only on passed events using the Failure cycle reference DemEventFailureCycleRef. ⌋

7.3.11.1 Warning indicator handling

The detailed configuration of the warning indicator handling within the Dem is shown in Figure 7.41.

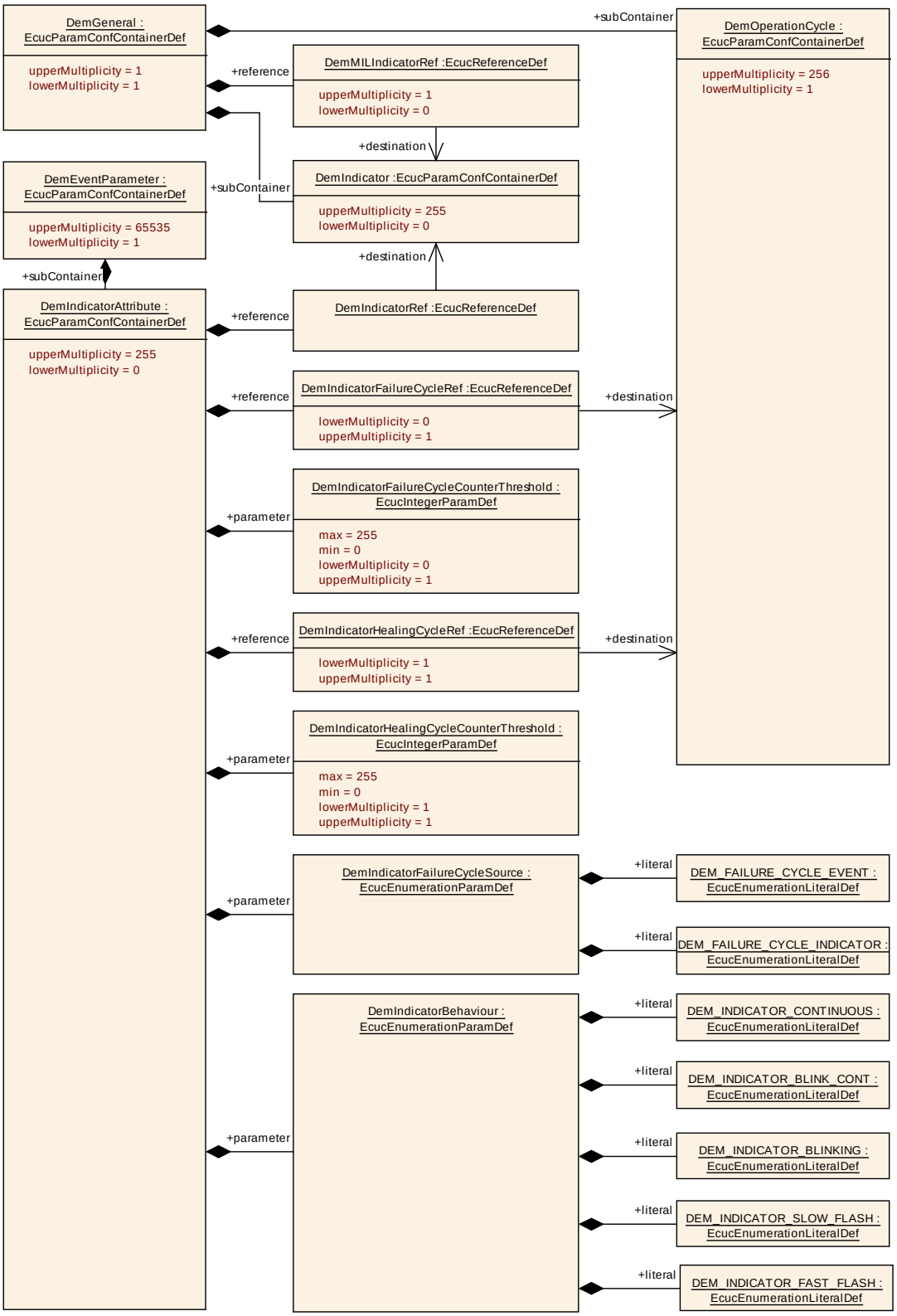

Figure 7.41: Warning indicator configuration

[SWS_Dem_00499] ⌈ The Dem module shall support event specific counters to activate and deactivate indicators, which are calculated based on the configured failure

and healing cycles (e.g. to turn on the MIL upon fault confirmation, and turn off the MIL after subsequent healing over three OBD-driving cycles). ⌋

[SWS_Dem_00509] ⌈ The Dem module shall provide the configuration parameter DemIndicatorId (refer to DemIndicator) to cover the identifier of a specific indicator. ⌋

[SWS_Dem_00510] ⌈ The Dem module shall be able to assign no indicator, one indicator, or several indicators to a specific event (refer to DemIndicatorAttribute). ⌋*(SRS_Diag_04069)*

[SWS_Dem_00511] ⌈ The Dem module shall provide the configuration parameter DemIndicatorBehaviour (refer to DemIndicatorAttribute) to assign a specific behavior (e.g. illuminate continuously or blinking) to each indicator and per event. ⌋*(SRS_Diag_04069)*

Note: During the integration process of the Dem module, different indicators and behaviors (e.g. indicator lamps, text messages or icons) can be assigned to an event.

[SWS_Dem_00566] ⌈ If more than one indicator is configured for a specific event, the Dem module shall use a logical OR operation of all combined warning indicators assigned to this event to calculate the UDS DTC status bit 7 (WarningIndicator). ⌋*(SRS_Diag_04067)*

[SWS_Dem_00500] ⌈ The Dem module shall provide the configuration parameter DemIndicatorFailureCycleCounterThreshold per indicator per event (refer to DemIndicatorAttribute) to define the maximum number of tested and failed cycles, before the respective indicator is activated (i.e. generates a specific condition WarningIndicatorOnCriteriaFulfilled, refer to Figure 20). ⌋*(SRS_Diag_04069)*

[SWS_Dem_00504] ⌈ The Dem module shall provide the configuration parameter DemIndicatorFailureCycleRef per indicator and event (refer to DemIndicatorAttribute) defining the indicator specific cycle, which represents the event or trigger, that causes an update of the failure counter provided there is failed result reported for the event. ⌋*(SRS_Diag_04069)*

[SWS_Dem_00501] ⌈ The Dem module shall generate the condition WarningIndicatorOnCriteriaFulfilled specific for the assigned warning indicator, if the respective indicator failure counter of the event entry has been processed (counted further) DemIndicatorFailureCycleCounterThreshold times. ⌋

[SWS_Dem_00568] ⌈ The Dem module shall provide the configuration parameter DemIndicatorFailureCycleSource (refer to DemIndicatorAttribute) to define which failure cycle (event or indicator based) is used for the WarningIndicatorOnCriteria handling. ⌋*(SRS_Diag_04069)*

Note: For emission related events the fault confirmation given by the DemDTCAttributes also leads to the MIL on i.e. in that case, the IndicatorOnCriteria bases on the same failure cycle settings (trigger and threshold) of the event. Then the source would be DEM_FAILURE_CYCLE_EVENT and the indicator specific settings would be ignored (grayed out). If an additional indicator needs to be activated e.g. via time for this particular event, then the DemIndicatorSource would be

DEM_FAILURE_CYCLE_INDICATOR. That means the trigger and the threshold are defined for this indicator.

[SWS_Dem_00502] ⌈ The Dem module shall provide the configuration parameter DemIndicatorHealingCycleCounterThreshold per indicator per event (refer to DemIndicatorAttribute) to define the maximum number of tested and passed healing cycles, before the respective indicator is deactivated (i.e. generates a specific condition WarningIndicatorOffCriteriaFulfilled, refer to Figure 7.19). ⌋

[SWS_Dem_00505] ⌈ The Dem module shall provide the configuration parameter DemIndicatorHealingCycleRef per indicator per event (refer to DemIndicatorAttribute) defining the indicator specific cycle, which represents the event or trigger, that causes an update of the healing counter provided there is OK / passed result reported for the EventId. ⌋

[SWS_Dem_00503] ⌈ The Dem module shall generate the condition WarningIndicatorOffCriteriaFulfilled specific for the assigned warning indicator, if the respective indicator healing counter of the event entry has been processed (counted further) DemIndicatorHealingCycleCounterThreshold times. ⌋

[SWS_Dem_00533] ⌈ If more than one indicator is configured for a specific event and each assigned indicator healing counter has been processed (counted further) DemIndicatorHealingCycleCounterThreshold times, the Dem module shall reset the UDS DTC status bit 7 (WarningIndicatorRequested). ⌋*(SRS_Diag_04069)*

[SWS_Dem_00506] ⌈ The Dem module shall process all indicator failure or indicator healing counters, if the cycle is started (refer to chapter 7.3.8) according to the current status of the stored events. ⌋*(SRS_Diag_04069)*

Note: This cycle type could be for example equal to DEM_OPCYC_OBD_DCY or to DEM_OPCYC_POWER (refer to DemOperationCycle).

It is possible to define a failure cycle or healing cycle, which does not correspond to an operation cycle in the sense, e.g. of a period of time with a beginning and an end. For that reason, the operation cycles can be extended with new types, e.g. DEM_OPCYC_TIME_200MS interpreted as triggers. In this example, the trigger is set every 200ms, i.e. is started every 200ms, to initiate the indicator counters to be processed.

7.3.11.2 User controlled WarningIndicatorRequested-bit

In some cases (e.g. controlling a failsafe reaction in application) the WIR-bit of a corresponding event in Dem shall be set/reset by an especial “failsafe SW-C”.

The failsafe SW-C has to ensure a proper status of the WIR-bit (e.g. regarding to ISO14229-1 or manufacture specific requirements). Therefore, the failsafe SW-C can use existing Dem mechanism to get the information about status changes of events in Dem (e.g. Callback EventStatusChanged).

The failsafe SW-C shall report the required WIR-status to Dem (via Dem_SetWIRStatus and has to ensure that the current WIR-status of an event (in Dem) fits to the current failsafe-status in application:

- failsafe running: WIR-bit shall be set to "1"
- failsafe not running: WIR-bit shall be set to "0"

The failsafe SW-C has to report the status after every change of failsafe state.

Each invocation of Dem_SetWIRStatus updated the WIR-bit for the corresponding event (see parameter EventId)

Due to not storing the StatusOfDTC-bit 7 (WIR-bit) on ECU shutdown the failsafe SW-C has to ensure that the WIR-bit of an event fit to the current failsafe status after Dem_Init.

[SWS_Dem_00831] ⌈ Dem shall provide a function to control (set/reset) the WarningIndicatorRequested-bit of a configured event (in Dem) regarding to e.g. failsafe state. ⌋*(SRS_Diag_04128)*

[SWS_Dem_00832] ⌈ Setting of the WIR-bit of an event can be controlled via Dem_SetWIRStatus OR by the Dem internal WIR-bit handling. (OR-Operation). ⌋*(SRS_Diag_04128)*

Note: The parallel use of Dem_SetWIRStatus and the Dem internal WIR-Bit handling (at the same time) is needed for example for OBD systems, where the WIR-Bit needs to be activated according to the legislation (Dem-Internal) and a system reaction is controlled by the same event. Therefore the WIR-Bit needs to stay active until the system reaction is deactivated (Dem_SetWIRStatus).

[SWS_Dem_00833] ⌈ The WIR-bit of the corresponding event shall be set to "1" if Dem_SetWIRStatus is called with parameter WIRStatus = TRUE . ⌋*(SRS_Diag_04128)*

[SWS_Dem_00834] ⌈ The WIR-bit of the corresponding event shall be set to "0" if Dem_SetWIRStatus is called with WIRStatus = FALSE and no referenced Dem Indicator(s) are set. ⌋*(SRS_Diag_04128)*

[SWS_Dem_00835] ⌈ The WIR-bit of the corresponding event shall be set to "0" if all referenced Dem Indicator(s) are not active and WIRbit is not controlled by the API Dem_SetWIRStatus. ⌋*(SRS_Diag_04128)*

[SWS_Dem_00836] ⌈ During disabled ControlDTCSettings the WIR-bit of an event shall not be changed via Dem_SetWIRStatus and the function shall return E_NOT_OK. ⌋*(SRS_Diag_04128)*

Note: In case the failsafe application is not able to set the WIR bit (Dem_SetWIRStatus returned E_NOT_OK), the failsafe application needs to observe the general status of the event and coordinate the retry of Dem_SetWIRStatus itself e.g. by using the callback function InitMonitorForEvent with parameter DEM_INIT_MONITOR_REENABLED.

7.3.11.3 Handling of the warning indicator lamp (MIL)

[SWS_Dem_00546] ⌈ For OBD-relevant ECUs, the Dem module shall provide the configuration parameter DemMILIndicatorRef (refer to DemGeneral) to indicate that the configured indicator controls the MIL activation and deactivation. ⌋

[SWS_Dem_00567] ⌈ If an indicator is configured for controlling the MIL of an OBD-relevant ECU, the Dem module shall use the configured event failure cycle counter of this event (refer to 0 Fault confirmation) to define the maximum number of tested and failed cycles, before the stored event activates the respective indicator. ⌋

Note: For OBD systems, the activation of the Malfunction Indicator Lamp (MIL) is linked with the entry to confirmed state. Therefore the event specific fault confirmation counter (refer to the configuration parameter DemEventFailureCycleCounterThreshold and DemEventFailureCycleRef) has to be consitant with the indicator failure cycle counter.

Note: Leaving Pending state and the deactivation of the MIL is controlled by the configuration of the indicator healing cycle counter.

[SWS_Dem_00701] ⌈ If the MIL is deactivated and the event is confirmed the MIL shall be reactivated according to [[SWS_Dem_00567]] i.e. again according to the indicator failure cycle counter is reaching it's threshold. ⌋*(SRS_Diag_04069)*

[SWS_Dem_00535] ⌈ In case of OBD-relevant events the indicator cycles shall be based on cycles defined by OBD legislation. ⌋*(SRS_Diag_04069)*

7.3.11.4 Notification and Set of the warning indicator status

[SWS_Dem_00046] ⌈ The Dem module shall provide the API Dem_GetIndicatorStatus (refer to chapter 8.3.3.20) that a software component can get information about the calculated indicator status ⌋*(SRS_Diag_04069)*

Note: The Dem module derives the indicator status internally based on the event status as a summary of all assigned events. One indicator can be activated by several events and one event can also have more than one indicator assigned to it (refer to DemIndicatorAttribute).

[SWS_Dem_00749] ⌈ The Dem module shall provide the API Dem_SetIndicatorStatus (refer to chapter 8.3.3.21) that a software component can set the indicator status. ⌋*(SRS_Diag_04128)*

7.4 Startup behavior

[SWS_Dem_00169] ⌈ The Dem module shall distinguish between a pre-initialization mode and a full-initialized mode (operation mode). ⌋*(SRS_BSW_00406)*

[SWS_Dem_00180] ⌈ The function Dem_PreInit (refer to chapter 8.3.2.1) shall initialize the internal states of the Dem module necessary to process events and reset debounce counters reported by BSW modules by using Dem_ReportErrorStatus (refer to chapter 7.6) and Dem_ResetEventDebounceStatus (refer to chapter 7.3.3). ⌋

The function Dem_PreInit is called by the ECU State Manager during the startup phase of the ECU before the NVRAM Manager is initialized. For all operation cycles having a autostart functionality by configuration parameter DemOperationCycleAutostart, the BSW modules can start reporting of related events via Dem_ReportErrorStatus (refer to [SWS_Dem_00854], [SWS_Dem_00851], [SWS_Dem_00167]).

The function Dem_Init (refer to [SWS_Dem_00340]and chapter 8.3.2.2) is called during the startup phase of the ECU, after the NVRAM Manager has finished the restore of NVRAM data. SW-Components including monitors are initialized afterwards. The function Dem_Init is also used to reinitialize the Dem module after the Dem_Shutdown was called.

Caveats of Dem_Init: The Dem module is not functional until the Dem module's environment has called the function Dem_Init.

7.5 Monitor re-initialization

The primary initialization of the monitors in the application is done via Rte_Start. The initialization of the event-specific part of the monitor can be triggered by the Dem.

[SWS_Dem_00003] ⌈ The Dem module shall provide the interface InitMonitorForEvent (refer to InitMonitorForEvent) to trigger the initalization of a diagnostic monitor (refer also to chapter 7.3.3 and [SWS_Dem_00573]). ⌋

The function parameter InitMonitorReason indicates the reason / trigger of initialization.

Note: The Dem module does not evaluate the return value (e.g. if other than E_OK) of this callback function.

Note: The configuration container DemCallbackInitMForE (in DemEventParameter) is used to specify the related port or c-callback per event.

[SWS_Dem_00679] ⌈ The API Dem_SetOperationCycleState (refer to chapter 7.3.8) shall trigger the callback function InitMonitorForEvent of the related event(s) in case of starting or restarting the operation cycle of the event(s). The InitMonitorReason parameter shall be set to DEM_INIT_MONITOR_RESTART. ⌋*(SRS_Diag_04117)*

[SWS_Dem_00680] ⌈ Any Dem_<...>ClearDTC API (refer to chapter 7.3.2.2) shall trigger the callback function InitMonitorForEvent of the related event(s) in case of clearing the event(s). The InitMonitorReason parameter shall be set to DEM_INIT_MONITOR_CLEAR. ⌋*(SRS_Diag_04117)*

[SWS_Dem_00681] ⌈ The API Dem_SetEnableCondition (refer to chapter 7.3.6) shall trigger the callback function InitMonitorForEvent of the related event(s) in case an

enable condition of the event(s) is changed to fulfilled and thus all enable conditions of the event(s) are fulfilled. The InitMonitorReason parameter shall be set to DEM_INIT_MONITOR_REENABLED. ⌋(*SRS_Diag_04125*)

[SWS_Dem_00682] ⌈ The API Dem_DcmEnableDTCSetting (refer to chapter 7.9.2.4) shall trigger the callback function InitMonitorForEvent of the related event(s) in case ControlDTCSetting of the event(s) is re-enabled. The InitMonitorReason parameter shall be set to DEM_INIT_MONITOR_REENABLED. ⌋(*SRS_Diag_04125*)

The following figures show two examples:

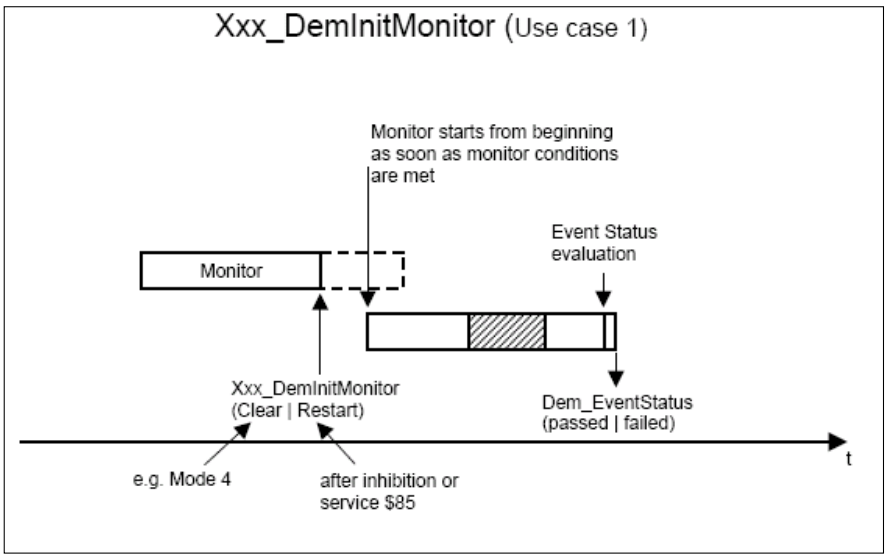

Figure 7.42: Use-case of the interface InitMonitorForEvent for a specific event

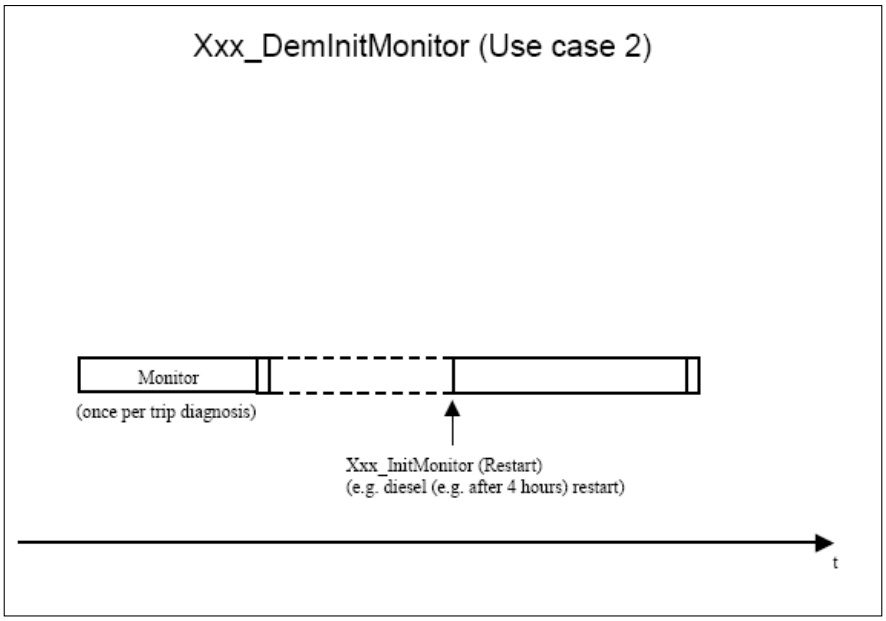

Figure 7.43: Use-case of the interface InitMonitorForEvent for a specific event

The initialization of any function (which may relate to the monitor) can also be triggered by the Dem.

[SWS_Dem_00335] ⌈ The Dem module shall provide the interface InitMonitorForFunction (refer to paragraph 8.4.3.1.2) to trigger the initialization of an assigned functionality, which is not a monitor. ⌋

[SWS_Dem_00049] ⌈ The Dem module shall trigger the callback function InitMonitorForFunction to initialize the respective functionality of a specific SW-C or BSW module. ⌋

Note: The Dem module does not evaluate the return value (e.g. if other than E_OK) of this callback function.

Example: Adaptations may be initialized in case of clearing the Dem module (on service 04/ISO 15031-5 request).

Configuration of InitMonitorForFunction: During the configuration of a system, one list has to be created to assign functions to be initialized. If different clearing processes have to be distinguished (only powertrain, wiper system, etc.) then several task lists have to be created.

[SWS_Dem_01046] ⌈ If both events CLEAR/RESTART occur simultaneously, the DemInitMonitorForEvent shall be called with "CLEAR". ⌋

7.6 BSW Error Handling

Beside application software components also the basic software (BSW) can detect errors (e.g. hardware driver faults), especially during startup (ref. to document [10] for further details). For these errors (only a small number compared to application specific events), some additional aspects apply:

- Errors can be detected before Dem is fully initialized
- Errors can be reported during startup, information needs to be buffered until Dem is fully available
- Errors can be reported between startup and shutdown, information needs to be buffered and need to be processed by the Dem main function (RTE related call tree requirement)
- Entries in the event memory can have a different configuration (e.g. no emphasis on freeze frame data for the workshop)

Since BSW events are treated as normal SW-C events in the event memory, they can also be classified (availability in workshop tester), aged (refer to chapter 7.3.9) and prioritized (displacement handling).

[SWS_Dem_00107] ⌈ The Dem module shall provide the interface Dem_ReportErrorStatus (refer to chapter 8.3.3.1) to the BSW modules, to re-

port BSW events which are processed like event reports by Dem_SetEventStatus). ⌋(*SRS_BSW_00417, SRS_BSW_00420, SRS_BSW_00421*)

[SWS_Dem_00167] ⌈ The Dem module shall provide a buffer mechanism (FIFO) to queue events which are reported before Dem_Init via Dem_ReportErrorStatus as qualified (Failed/Passed) or reaching the qualification (the Dem module cannot access the event memory). ⌋

[SWS_Dem_00852] ⌈ The Dem module shall provide a buffer mechanism (FIFO) to queue events which are reported via Dem_ReportErrorStatus during normal operation (after the Dem module has been already initialized). ⌋(*SRS_Diag_04070*)

Dem_ReportErrorStatus is used by BSW modules to report errors from the point in time when the Dem module is pre-initialized. Within Dem_Init, the queued events are processed. During normal operation (after full initialization), the queuing mechanism of the API Dem_ReportErrorStatus is necessary to process the reported fault within the main function of the Dem module.

During initialization of the Dem module, the API Dem_ReportErrorStatus supports debouncing (pre-failed and pre-passed). The corresponding internal debounce counters/timers will be initialized by calling Dem_PreInit and can be reset by Dem_ResetEventDebounceStatus.

[SWS_Dem_00207] ⌈ The size of the queue of the function Dem_ReportErrorStatus is configurable by the configuration parameter DemBswErrorBufferSize (refer to DemGeneral). ⌋

Note: During startup phase, not all event-related data might be available. SW-C events cannot be stored before complete initialization of the Dem.

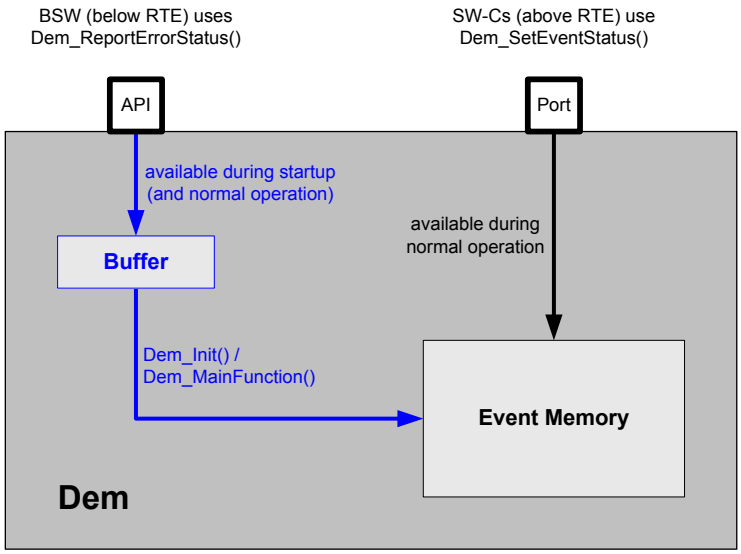

Figure 7.44: Dem_ReportErrorStatus buffering behavior

[SWS_Dem_00851] ⌈ All events which are reported via Dem_ReportErrorStatus before Dem_Init shall not consider the state of the operation cycles (refer to [SWS_Dem_00481]). ⌋

[SWS_Dem_00854] ⌈ The operation cycle state shall be evaluated, when processing the events from the queue. If the operation cycle related to that event is started, the event shall be processed regularly (similar to Dem_SetEventStatus)); otherwise in case the operation cycle is not started the event shall be dropped and the debounce counter shall be reset to zero. ⌋

Note: All events which are reported via Dem_ReportErrorStatus before Dem_Init should use a operation cycle which is auto started (DemOperationCycleAutostart set to true) or persistently stored (DemOperationCycleStatusStorage set to true).

7.7 OBD-specific functionality

7.7.1 General overview and restrictions

In the following, a specification of the OBD handling in AUTOSAR is introduced. Herein, “OBD“ is used for automotive OBD with respect to different target markets. For SW-sharing and distributed development reasons as well as aspects of packaging and responsibility of releases, the OBD-relevant information / data structures need to be reported via Standardized AUTOSAR interfaces.

In a vehicle there can be 3 different kinds of OBD ECUs:

- Master ECU (one per vehicle)
- Primary ECU (several per vehicle)
- Dependend / Secondary ECUs (several per vehicle)

From the Basic Software point of view Dependend / Secondary ECUs doesn’t need any specific OBD functionality. In Dependend / Secondary ECUs are always related to a Master or a Primary ECU. In Dependend / Secondary ECUs OBD-relevant information will not be stored in the Basic Software (e.g. OBD events will be forwarded to the respective Master or Primary ECU via the Bussystem). In Dependend / Secondary ECUs this "reported errors" and other OBD functionality might be handeled by a SW component.

The following table will give an overview about which OBD functionality must be supported in a Master ECU, Primary ECU or Dependend / Secondary ECU:

Functionality	***Master ECU***	***Primary ECU***	***Dependend / Secondary ECU***
OBD Event Memory	Own and “reported errors” of Dep. / Sec ECUs	Own and “reported errors” of Dep. / Sec ECUs	No

MIL Master	Yes	No	No
Calulation and provision of General Information (DYC, General Nominator, PFC cycle,..)	Yes	No	No
Reception and execution of General Information (DYC, General Nominator, PFC cycle,..)	(Yes)	Yes	No
Calibration Identification (CAL-ID)	Not in BSW	Not in BSW	Not in BSW
Calibration Verification Number (CVN)	Not in BSW	Not in BSW	Not in BSW

Table 7.3: Overview about OBD functionality in different OBD ECUs

The following OBD requierments are only valid for Master and Primary ECUs. If necessary the OBD requierments differenciate between Master and Primary Requierment. Master and Primary ECUs should have the same interface to the SWComponents. To build a sufficant and lean Master ECU there is no compulsion to use this interfaces.

Together with the Dcm, this Dem SWS provides standardized AUTOSAR interfaces to support the OBD services $01 - $0A defined in SAE J1979 Rev May 2007 [13]. With these services, Autosar OBD functionality shall be capable of meeting all light duty OBD regulations worldwide (California OBDII, EOBD, Japan OBD, and all others.)

Some details on the interaction between Dem and specific SW-C might remain open, since they are dependent on the Dem and SW-C implementation. The following functionality is not defined:

- Malfunction Indicator Lamp (MIL)-activation (interface Dem to MIL handler, MIL-bulb check, readiness blinking, blinking in case of catalyst damaging misfire...)
- Misfire fault handling (common debouncing, filtering single / multiple misfire faults).

However, this Dem SWS does not prescribe implementation details on how OBD compliance can be achieved within the Dem module, e.g. concerning state handling. Furthermore, the Dem SWS does not prescribe implementation details on the diagnostic algorithms of the SW-C necessary to achieve OBD compliance (how to detect a fault, when to trigger incrementation of IUMPR-numerator...).

In the following chapters, OBD relevant functionality and interfaces are described. It is important to note, that independent of standard Autosar mechanisms (e.g. communication via RTE), response codes and timing constraints need to fulfill OBD requirements (refer to [14] and [13]).

[SWS_Dem_00584] ⌈ While applying standard mechanisms (like Std_ReturnType), the Dem module shall only return values ensuring OBD compliant behavior with regard to permitted response codes and timing constraints. ⌋*(SRS_Diag_04010)*

Calulation and provision of General Information Data: The Master ECU shall calculate and provide the following General Information Data via the Bussystem to the Primary ECUs:

- OBD Driving cycle information (DYC)
- General Nominator / Rate-based monitoring - driving cycle (RBM cycle)
- Warm up cycle (WUC)
- Ignition cycle
- Permanent fault code - driving cycle (PFC cycle)

The informations about the actual state (started / ended) can be accessed via the API Dem_GetOperationCycleState() (refer to chapter 8.3.3.8).

OBD Driving cycle information (DYC)
The Master ECU will provide the driving cycle information (DYC) via the Busystem. The driving cycle information (DYC) shall not be computed internally in the Primary ECUs.

General Nominator / Rate-based monitoring - driving cycle (RBM cycle)
Included in the IUMPR-Cycle Flag

Warm up cycle (WUC)
The warm up cycle (WUC) is a legally prescribed cycle and is computed by the Master ECU. The Master ECU will provide the warm-up cycle information (WUC) via the Busystem.

Ignition cycle
An ignition cycle describes the cycle between the “Terminal 15 on“ status and “Terminal 15 off“, including the shut down/after-run phase of the electronic control unit if the engine start condition was met for at least 2 seconds after “Terminal 15 on“. The Master ECU will provide the Ignition cycle information via the Busystem.

Permanent fault code - driving cycle (PFC cycle)
(refer to chapter 7.7.1.4)

Note: Calibration Identification (CAL-ID) and Calibration Verification Number (CVN)
The Calulation of the Calibration Identification (CAL-ID) and Calibration Verification Number (CVN) is not a BSW Task and will not handled within the Dem.

Destination of OBD Events

[SWS_Dem_00702] ⌈ The Dem module shall support the configuration parameter DemOBDDestinationOfEvents DemGeneralOBD to allow the handling of OBD related events (which have been forwarded from Dependend / Secondary ECUs) ether in the primary event memory or the secondary event memory of a Master or Primary ECU. ⌋*(SRS_Diag_04129)*

7.7.1.1 Service $01, Service $02 and OBD PID data

In order to retrieve relevant data upon service $01 request or a fault entry, the Dem needs access to current data, addressed via PIDs. For that purpose, a Client (=Dcm/-Dem)/Server (=SW-C) interface is assigned based on configuration items.

[SWS_Dem_00291] ⌈ The Dem module shall support only the legislative freeze frame (record number 0). This will be a single list of PIDs assigned to this freeze frame (refer to DemPIDClass). ⌋*(SRS_Diag_04082)*

This means a request by a generic scan tool for record number one and above will be ignored.

Upon the entry of a fault in the memory, the values of these PIDs/DIDs are requested through the RTE via a client server interface.

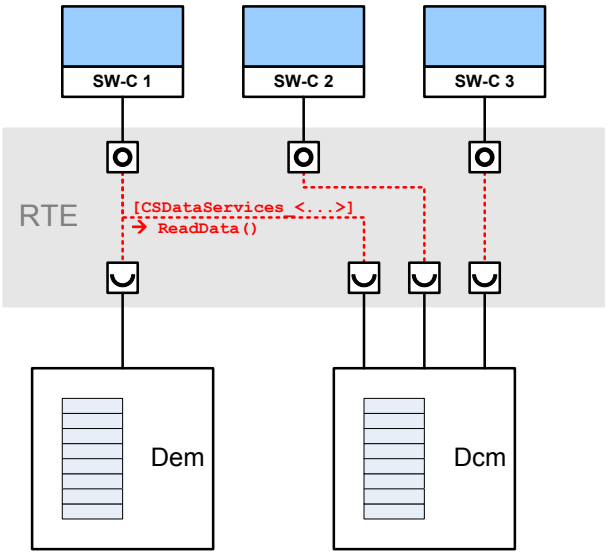

Figure 7.45: Dem and Dcm module requests PID data elements of SW-C via ReadData operation

[SWS_Dem_00596] ⌈ The Dem module shall provide access on PID data elements of the most important freeze frame being selected for the output of service $02 (OBD freeze frame of the event which caused MIL on) to the Dcm module (refer to 8.3.4.4.12 Dem_DcmReadDataOfOBDFreezeFrame). ⌋*(SRS_Diag_04082)*

Note: The Dem processes and stores only the raw data for service $02. Any PID header-information and fill bytes are added by the Dcm. The Dcm configuration defines the individual PID layout, which also defines an order of the contained data elements. Each data element references to its linked element in the Dem configuration (refer to Figure 47).

The individual data elements of a PID are selected via an index by the Dcm.

[SWS_Dem_00597] ⌈ The index values (refer to API parameter DataElementIndex-OfPID) of Dem_DcmReadDataOfOBDFreezeFrame shall be assigned zerobased and consecutive. Their order shall be derived from the position of the data elements (refer

to DcmDspPidDataPos, of the referencing DcmDspPidData containers) in the Dcm's PID layout. ⌋(*SRS_Diag_04082*)

Note: This index is not configured explicitly (to be able to avoid resource overhead for e.g. mapping tables in the implementation).

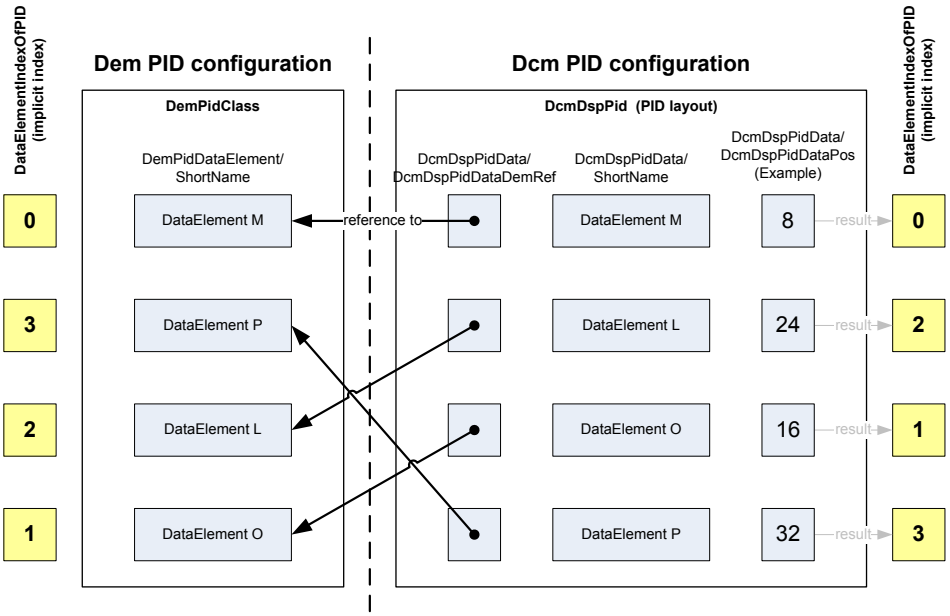

Figure 7.46: PID layout configuration within Dcm (master) and Dem

[SWS_Dem_00623] ⌈ The function Dem_DcmGetDTCOfOBDFreezeFrame (refer to chapter 8.3.4.4.13) shall return the DTC associated with the most important freeze frame being selected for the output of service $02 (PID $02). ⌋(*SRS_Diag_04082*)

7.7.1.2 PIDs for service $01 computed by Dem

[SWS_Dem_00293] ⌈ There are some PIDs, which shall be computed di6rectly within the Dem.

- PID $01 Monitor status since DTCs cleared (4 byte)
- PID $02(1) Freeze frame DTC (2 byte)
- PID $1C OBD requirements to which vehicle or engine is certified (1 byte)
- PID $21 km with MIL On (2 byte)
- PID $30 number of Warm Up cycles (WUC) since last fault clear (1 byte)
- PID $31 km since last fault clear (2 byte)
- PID $41 Monitor status this driving cycle (4 byte)
- PID $4D time with MIL On (2 byte)

- PID $4E time since last fault clear (2 byte)

⌋(*SRS_Diag_04082*)

(1) PID $02 is only required for service $02 and therefore no interface (like Dem_DcmReadDataOfPID02) is necessary. Instead the API Dem_DcmGetDTCOfOBDFreezeFrame is used (refer to[SWS_Dem_00623])

[SWS_Dem_00748] ⌈ The function Dem_DcmReadDataOfPID1C shall return the appropriate value “OBD requirements to which vehicle or engine is certified.“ according to the respective standards [14], e.g. OBD, OBDII, JOBD etc. The value PID1Cvalue to return is in configuration parameter DemOBDCompliancy defined. ⌋(*SRS_Diag_04082*)

7.7.1.3 Centralized PID $21 / $31 / $4D / $4E handling

[SWS_Dem_00703] ⌈ The Dem of the Master ECU shall calculate informations for PID $31 / $4D / $4E vehicle wide. ⌋(*SRS_Diag_04082*)

The Primary ECUs are not allowed to calculate informations for PID $31 / $4D / $4E.

Note: Therefore a Software component of the Master ECU will use API Dem_DcmReadDataOfPID<NN> (where NN is 31 / 4D / 4E) to read out the informations and will provide informations about mileage / time for PID $31 / $4D / $4E to the Primary ECUs via the bussystem.

On receiving the informations for PID $31 / $4D / $4E in a software component in the Primary ECU via the bussystem, the software component will set the PID $31 / $4D / $4E in calling the API Dem_SetDataOfPID<NN> (where NN is 31 / 4D / 4E) of the Dem of the Primary ECU.

[SWS_Dem_00704] ⌈ Only the Master ECU is allowed to report informations for PID $31 / $4D / $4E to the scan tool. ⌋(*SRS_Diag_04082*) The Primary ECUs are not allowed report informations for PID $31 / $4D / $4E to the scan tool.

[SWS_Dem_00705] ⌈ The Dem module shall support the configuration parameter DemOBDCentralizedPID21Handling DemGeneralOBD to enable or disable the centralized handling of PID $21. ⌋(*SRS_Diag_04082*)

Note: If enabled via configuration switch DemOBDCentralizedPID21Handling: A Software component of the Master ECU will use API Dem_DcmReadDataOfPID21 to read out the informations and will provide informations about mileage / time for PID $21 to the Primary ECUs via the bussystem.

On receiving the informations for PID $21 in a software component in the Primary ECU via the bussystem, the software component will set the PID $21 in calling the API Dem_SetDataOfPID21 of the Dem of the primary ECU.

[SWS_Dem_00706] ⌈ Only the Master ECU is allowed to report informations for PID $21 to the scan tool (if enabled via configuration switch DemOBDCentralizedPID21Handling). ⌋*(SRS_Diag_04082)*

The Primary ECUs are not allowed report informations for PID $21 to the scan tool (if enabled via configuration switch DemOBDCentralizedPID21Handling). If configuration switch DemOBDCentralizedPID21Handling disables the centralized PID $21handling, the Primary ECUs will compute PID $21 information internally and will report informations for PID $21 to the scan tool.

Note: Any ECU supporting PID $21 and PID $31 is required to support PID $0D (vehicle speed). PID $21 and PID $31 are required for OBD ECUs.

[SWS_Dem_00346] ⌈ The Dem module shall use PID $0D (refer to chapter 7.9.8) to calculate PID $21 and PID $31. ⌋*(SRS_Diag_04082)*

[SWS_Dem_00304] ⌈ A Dem delivery shall provide function call interfaces to the Dcm and the respective ServiceNeeds to declare to the Dcm that these PIDs are supported. ⌋*(SRS_Diag_04082)*

[SWS_Dem_00347] ⌈ If PID $1E (auxiliary input status) is supported the PTO (Power Take Off) related event status handling shall implemented inside the Dem module (refer to [15]). ⌋*(SRS_Diag_04082)*

[SWS_Dem_00377] ⌈ The Dem module shall provide the interface Dem_SetPtoStatus (refer to chapter 8.3.8.7) allowing a SWC implementing the PTO functionality to notify the Dem module if PTO is active or inactive (refer to section 8). ⌋*(SRS_Diag_04082)*

[SWS_Dem_00378] ⌈ The Dem module shall support the configuration parameter DemConsiderPtoStatus (refer to DemObdDTC) indicating that a certain event is affected by the Dem PTO handling. ⌋*(SRS_Diag_04082)*

The Dem module provides the configuration switch DemPTOSupport (refer to ECUC_Dem_00704) to enable or disable the usage of PID $1E.

A special configuration is applied for the computation of PID $01 and $41:

[SWS_Dem_00349] ⌈ The Dem module shall support the configuration parameter DemEventOBDReadinessGroup (refer to DemObdDTC) for OBD systems, to assign individual events to one specific readiness group. ⌋*(SRS_Diag_04082)*

According to SAEJ1979 [13], the group AirCondition Component shall not be supported anymore. However, it is still included in ISO 15031-5 [14]. The groups distinguish between spark ignition engines (spark) and compression ignition engines (compr.). However, it is necessary to configure per event, to which readiness group the monitor contributes (if at all).

[SWS_Dem_00351] ⌈ The Dem module shall compute and provide the number of confirmed faults (PID $01, Byte A). ⌋*(SRS_Diag_04082)*

7.7.1.4 MIL Handling

[SWS_Dem_00352] ⌈ The Dem module shall provide the MIL status. ⌋(*SRS_Diag_04082*)

Typically, the “MIL Master functionality“ is located in the OBD Master ECU. Only the MIL Master functionality is allowed to send the MIL request to the instrument cluster. It is open to the OBD Master ECU implementation if the MIL Master functionality is solved within the Dem or not.

Primary ECUs must report their Indicator (MIL) request (when the criterion for triggering the MIL is fulfilled) via the MIL Master functionality.

[SWS_Dem_00707] ⌈ In the Master and Primary ECUs a software component shall use the function Dem_GetIndicatorStatus to poll the Dem to get information about an event with a related Indicator (MIL). ⌋(*SRS_Diag_04069*)

[SWS_Dem_00708] ⌈ This software component shall inform the MIL Master functionality (via the bussystem / in the case of Primary ECU). This applies for Master / Primary ECU as well for socalled “reported errors“ via assigned Dependent / Secondary control units. ⌋(*SRS_Diag_04082*)

In a Master ECU a software component might use the function Dem_SetIndicatorStatus to set the status for a Indicator (MIL) of an event (if the MIL Master functionality is solved within the Dem) in the Dem.

7.7.1.5 Readiness status

[SWS_Dem_00354] ⌈ The Dem module shall compute the readiness status (if all events of a DemEventOBDReadinessGroup are reported as OK tested since last clear, or the event has caused MIL on). Suppressed Events (refer to chapter 7.2.7) shall be ignored for this computation. ⌋(*SRS_Diag_04082*)

Note: DemEventOBDReadinessGroup has a regulation backround and special computions based on ModelYear (MY) and Market.

[SWS_Dem_00355] ⌈ The Dem module shall compute the readiness group complete for current driving cycle (if all events of a group are tested in the current driving cycle). OBD Events Suppression (refer to chapter 7.2.7) shall be ignored for this computation. ⌋(*SRS_Diag_04082*)

Note: For calculation of the group readiness, the Dem module considers all events assigned to the specific readiness group.

[SWS_Dem_00356] ⌈ The Dem module shall compute the readiness group disabled (if the disabled status is reported by the monitor for any event of a group). OBD Events Suppression (refer to chapter 7.2.7) shall be ignored for this computation. ⌋(*SRS_Diag_04082*)

[SWS_Dem_00348] ⌈ The Dem module shall provide the disabling of events (refer to chapter 8.3.8.1). OBD Events Suppression (refer to chapter 7.2.7) shall be ignored for this computation. ⌋*(SRS_Diag_04082)*

[SWS_Dem_00294] ⌈ In order to allow a monitor to report that the event cannot be computed in the driving cycle (aborted e.g. due to physical reasons), the Dem shall provide the API Dem_SetEventDisabled. ⌋*(SRS_Diag_04082)*

Note: For the computation of PID $41, the monitor has to report its event as disabled, if the test cannot be carried out anymore until the end of this driving cycle.

Note: SetEventDisabled does report an Event as “uncompletable“ during the current driving cycle. It does not turn the Event into a Events Suppression according to chapter 7.2.7

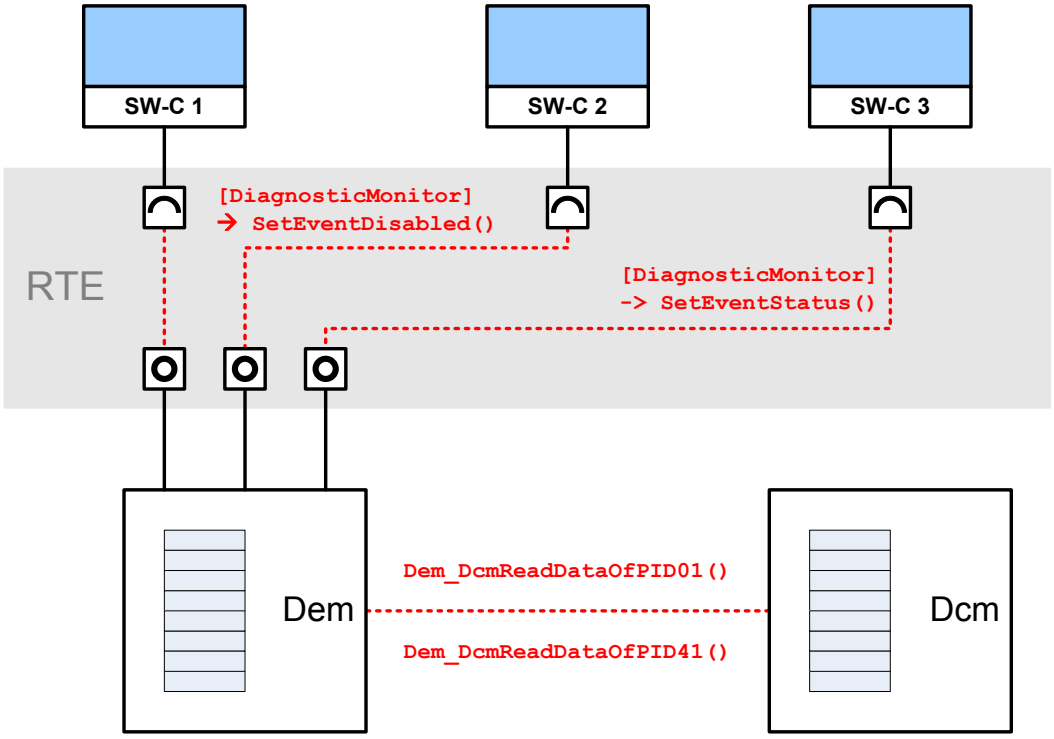

Figure 7.47: Dem calculates PID$01 and PID$41 data based on specific port operations

7.7.1.6 In-Use-Monitor Performance Ratio (IUMPR) Support

The In-Use-Monitor Performance Ratio (IUMPR) indicates how often the OBD system monitors particular components, compared to the amount of the vehicle operation. It is defined as the number of times a fault could have been found (=numerator), divided by the number of times the vehicle operation conditions have been fulfilled (=denominator) as defined in the respective OBD regulations.

The IUMPR will be calculated within the ECU using the following formula: In-Use-Monitor Performance Ratio = NUMERATOR / DENOMINATOR

[SWS_Dem_00709] ⌈ Ratios which refer to a Suppressed Event shall not be computed. ⌋*(SRS_Diag_04082)*

[SWS_Dem_00710] ⌈ The numerator shall be calculated in the ECU (Master and Primary). ⌋*(SRS_Diag_04082)*

[SWS_Dem_00711] ⌈ The Dem of the Master ECU shall calulate the IUMPR-Cycle Flag. ⌋

[SWS_Dem_00966] ⌈ The Dem shall provide an API Dem_GetIUMPRDenCondition (refer to chapter 8.3.8.4) give a software component the possibility to get the General Denominator status information. ⌋*(SRS_Diag_04082)*

The Master ECU (software component) will provide the IUMPR-Cycle Flag (included in the General Denominator signal) via the bus system.

[SWS_Dem_00712] ⌈ The ECU internal denominator incrementation (in a Master / Primary ECU) shall be done depended on the IUMPR-Cycle Flag (included in the General Denominator signal). ⌋*(SRS_Diag_04082)*

[SWS_Dem_00714] ⌈ The Dem shall provide an API Dem_SetIUMPRDenCondition (refer to chapter 8.3.8.3) to get informed about the IUMPR-Cycle Flag status by a software component. ⌋

Note: APIs to lock and unlock the denominator under special conditions are definded in [SWS_Dem_00362].

Note: A Timeout of the IUMPR-Cycle Flag / General Denominator signal of the Master will lead to an Event memory entry in the COM Stack (timeout of signal) in the Primary ECU.

Further details concerning In-Use-Monitor Performance Ratio (IUMPR) can be found in regulation documention.

Differencation InfoType $08 / InfoType $0B

The IUMPR-data collected need to be provided upon a service $09 request.

[SWS_Dem_01055] ⌈ The Dem shall provide the Info Type $08 if the DemOBDEngineType is set to DEM_IGNITION_SPARK and $0B if the DemOBDEngineType is set to DEM_IGNITION_COMPRESSION. ⌋

[SWS_Dem_00298] ⌈ In order to support the data requests in service $09 as described above, the Dem shall provide the API Dem_DcmGetInfoTypeValue08 or Dem_DcmGetInfoTypeValue0B to the Dcm. ⌋*(SRS_Diag_04082)*

[SWS_Dem_00357] ⌈ If the OBD system is a spark ignition system the Dem module shall provide the API Dem_DcmGetInfoTypeValue08 (refer to chapter 8.3.5.1) for Info Type $08 IUMPR data. ⌋*(SRS_Diag_04082)*

The service Dem_DcmGetInfoTypeValue08 shall be used by the Dcm, to request the IUMPR-data according to the InfoType $08 data format used for the output to service $09.

[SWS_Dem_00358] ⌈ If the OBD system is a compression ignition system the Dem module shall provide the API Dem_DcmGetInfoTypeValue0B (refer to chapter 8.3.5.2) for Info Type $0B IUMPR data. ⌋*(SRS_Diag_04082)*

The service Dem_DcmGetInfoTypeValue0B shall be used by the Dcm, to request the IUMPRdata according to the InfoType $0B data format used for the output to service $09.

The type of OBD system is defined by using the configuration switch DemOBDSupport (refer to DemGeneral).

In-Use-Monitor Performace Ratio (IUMPR) groups or components
The relevant In-Use-Monitor Performace Ratio (IUMPR) data recording is allocated in the Dem based on FIDs and events. The IUMPR data are recorded for different groups or components respectively.

Typically, one or more events serve for the monitoring of these components, e.g. the oxygen sensor. Hence, in order to record the in-use performance of the OBD-system, the test performance of all the relevant events for all the IUMPR-groups needs to be recorded. For that purpose, certain data structures need to be configured by the Dem.

However, in order to stop the incrementing of numerator and denominator in the case a monitor is stopped, due to the occurrence of another event preventing the monitor from running, it is necessary to list all events that can affect the computation of a particular IUMPRrelevant event. However, this information is already embodied in a FID (refer to FiM SWS) representing a function which serves for the computation e.g. of a particular even. A FID is inhibited in case of certain events and thus, this relation can also be used to stop the IUMPR record.

This leads to a combination of an event to be recorded and a primary FID (and optionally multiple secondary FIDs) representing all the events stopping the computation of the IUMPR-event. For the purpose of classifying the events into groups, an IUMPR-group is also part of this combination.

For the configuration of data structure per EventId / FID(s) / IUMPR-group combination, a new data object is introduced, namely, the RatioId (refer to DemRatio). Thus, the container DemRatio contains the EventId, primary FID, secondary FIDs, IUMPR-group and an interface option to configure “APIuse“ vs. “observer“, whereas the interface option is explained in more detail in the following.

Also for the purpose of the port configuration, an ObdRatioServiceNeeds is offered to the SW-C. If a SW-C is IUMPRrelevant, this ServiceNeed is filled out.

If the monitor is “symmetric“, i.e. upon completion of the test a fault could have been found, even if there is currently no fault in the system, then the numerator can be incremented just by observing the TESTED-status of the assigned event.

[SWS_Dem_00359] ⌈ Only for monitors being configured with the option “observer“, the Dem module shall increment the numerator of the corresponding monitor, if the assigned event gets tested/qualified (as passed or failed). ⌋*(SRS_Diag_04001)*

If the diagnostic is asymmetric and it takes more efforts to detect a malfunction than to detect an OKstatus, the monitor needs to call an API in order to report that a malfunction could have been found, because this may require some simulation within the monitor and can therefore not be purely derived from the TESTED status.

[SWS_Dem_00360] ⌈ For OBD relevant systems the Dem module shall provide the API Dem_RepIUMPRFaultDetect (refer to chapter 8.3.8.2). ⌋*(SRS_Diag_04082)*

[SWS_Dem_00296] ⌈ The Dem module shall provide the API Dem_RepIUMPRFaultDetect (refer to chapter 8.3.8.2) for the asymmetric monitor to report that a malfunction could have been found. ⌋*(SRS_Diag_04082)*

Note: This service shall be used by the monitor to report that a fault could have been found even if there is currently no fault at all according to the IUMPR regulations that all conditions are met for the detection of a malfunction.

[SWS_Dem_00361] ⌈ The Dem module shall provide the configuration parameter DemRatioKind (refer toDemRatio), to indicate per RatioId if the numerator is calculated based on the TESTED-status or the API call. ⌋*(SRS_Diag_04082)*

Additional denominator conditions:
For some particular monitors (e.g. for secondary air system, comprehensive components), there are additional conditions defined on the denominator: Their denominator will only be incremented if certain temperature or activity conditions are met. Then, the monitor needs to lock the denominator per API at the beginning of the driving cycle and will unlock it when the additional conditions on the denominator are met. In case of a fault being detected that prevents further computations of these additional denominator conditions, the numerator is required to be frozen as well. For that reason it is necessary to assign particular denominator conditions if applicable. Upon the report of an inhibited computation of a particular denominator condition, the affected ratio are set to frozen,

For this handling of commonly used denominator conditions within the Dem, further interfaces and configuration items are introduced.

[SWS_Dem_00715] ⌈ The Dem module shall provide configuration parameter DemIUMPRDenGroup (refer to ECUC_Dem_00838 :) to offer several conditions to be applied to the denominator per RatioId. ⌋*(SRS_Diag_04082)*

[SWS_Dem_00362] ⌈ The Dem module shall provide the APIs for locking (refer to chapter 8.3.8.5) and unlocking (refer to chapter 8.3.8.6) of the denominator under special conditions if the DemIUMPRDenGroup is configured as DEM_IUMPR_DEN_PHYS_API. ⌋*(SRS_Diag_04082)*

[SWS_Dem_00297] ⌈ The Dem shall provide the API Dem_RepIUMPRDenLock (refer to chapter 8.3.8.5) to IUMPR-relevant SWC, to control the denominator specific to the respective RatioId.⌋*(SRS_Diag_04082)*

Note: This service shall be used by the monitor to report, that a denominator of a specific monitor (represented by FID and EventId) is locked for physical reasons.

[SWS_Dem_00308] ⌈ The Dem shall provide the API Dem_RepIUMPRDenRelease (refer to chapter 8.3.8.6) to IUMPRrelevant SW-C, to control the denominator specific to the respective RatioId.⌋*(SRS_Diag_04082)*

Note: This service shall be used by the monitor to report, that a denominator of a specific monitor (represented by FID and EventId) is released for physical reasons.

In systems with multiple ECUs being OBD relevant and even IUMPR relevant, the status information on the additional denominator conditions needs to be communicated to synchronize the behavior of the counters. For that reason, an additional interface is introduced to read out and to report the status of the individual denominator conditions.

[SWS_Dem_00716] ⌈ The Dem shall provide the API Dem_GetIUMPRDenCondition (refer to chapter 8.3.8.4) to read out the status of a particular condition. ⌋

[SWS_Dem_00717] ⌈ The Dem shall provide the API Dem_SetIUMPRDenCondition (refer to chapter 8.3.8.3) to set the status of a particular condition (mainly in depending ECUs). ⌋*(SRS_Diag_04082)*

Note: To comuincate the received general denominator status, the Primary ECU should make use existing API as offered for SWCs.

Legislation requires that IUMPR tracking shall be stopped for a specific monitor, if it is inhibited by another service $07 visible fault.

[SWS_Dem_00299] ⌈ As long as an event has Pending status, the Dem module shall stop increasing the numerator and denominator. ⌋*(SRS_Diag_04001)*

Based on the related FID (FiM access) and RatioId, the Dem module can determine which numerator and denominator has to be stopped.

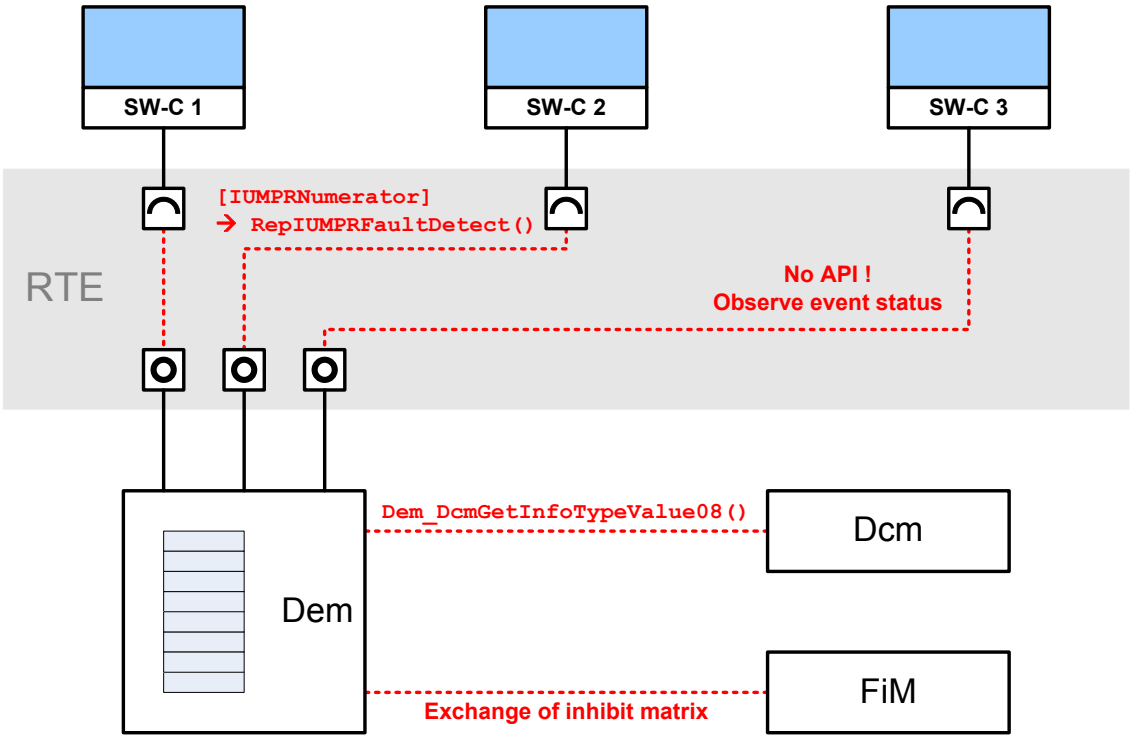

Figure 7.48: Dem calculates IUMPR data based on specific interface operations

7.7.1.7 Service $04 - Clear error memory

[SWS_Dem_00718] ⌈ In Master, Primary and Dependend / Secondary ECUs executing service $04 shall clear all emission-related diagnostic information in primary and secondary (event) memory (if configured) as defined in [13]. ⌋*(SRS_Diag_04082)*

[SWS_Dem_00719] ⌈ Executing service $04 shall include non emissionrelated diagnostic information (e.g. Debouncing counter, Event/ DTC related status information as TestFailedSinceLastClear) of primary and secondary (event) memory (if configured) in Master, Primary and Dependend / Secondary ECUs. ⌋*(SRS_Diag_04082)*

[SWS_Dem_00720] ⌈ The Master ECU shall inform the Dependend / Secondary ECUs about the execution of service $04 via the bussystem. ⌋

[SWS_Dem_00721] ⌈ The Dem (of the Master ECU) shall execute the software component API <Module>_SetClearDTC (refer to 8.4.3.1.4) to inform a software component about service $04 execution. The software component shall send this information via the bussystem. ⌋*(SRS_Diag_04082)*

[SWS_Dem_00722] ⌈ The Dem (of a Dependend / Secondary ECU) shall provide an API Dem_SetClearDTC (refer to chapter 8.3.8.14) to get informed about the reception of service $04 execution by a software component. ⌋*(SRS_Diag_04082)*

[SWS_Dem_00723] ⌈ In Dependend / Secondary ECU service $04 execution shall lead to reset / erase to all (non emissionrelated and emission-related) DTCs in primary and secondary (event) memory (if configured). ⌋*(SRS_Diag_04082)*

7.7.1.8 Service $06 - Support of central DTR handling

With Service $06, an external test tool can request an emissionrelated ECU to return the supported OBDMIDs (“OBD Monitor Identifier”) and / or the standardized diagnostic test results associated with the OBDMID.

The OBD related standards (refer to [14] and [13]) reserve certain OBDMIDs for the special purpose of obtaining the list of supported OBDMIDs within the vehicle. These OBDMIDs are called “availability OBDMIDs” and are numbered with $00, $20, $40, $60, $80, $A0, $C0 and $E0.

In contrast to earlier AR releases, the Dem additionally provides interfaces to report latest test results handled per particular identifier (DTR “Diagnostic Test Result“) introduced for that purpose.

Within the Service $06, the response (apart from the support information) consists of a data triple with test result, lower and upper limit representing the latest result of a conducted and qualified monitoring check. In order to refer to a particular component and a particular certified test, the response also contains the OBDMID and the TID (“Test Identifier“). For the external representation of the data, a socalled UnitandScalingID (UaSID) is passed along with the response onto the tester. From OBD regulation point of view it is required to have consistent data in Service$06 and Service$07. And in this scenario, a test has failed if the test result is either less than the lower limit or higher than the upper limit. Along with this violation of the thresholds, a corresponding Event has to be reported and vice versa. Note that this consistency is valid strictly only upon first detection of the malfunction, since a healing from Service$07 takes one entire cycle without a failure detected while the Service$06 contents simply represent the latest test result.

In order to support this consistency requirement, the Dem provides a specific interface to report the results per DTR (DTR “Diagnostic Test Result“) whereas an Event is associated to a DTR on configuration level to refer to.

Note that the Event serves as master and within its status additional effects such as enable / storage conditions are encountered. Given a specific DTR update type (by configuration), the Event status (Tested, Failed, debounce status etc.) decides whether a DTR result being reported is further processed or suppressed/ignored.

The Dem offers a service of conversion for the SWC to support a report of test results and thresholds within the ECU-internal resolution. The conversion onto the target format (standardized by [17, 19]: uint16, formula according to UaSID) including a correction in case of rounding effects is done within the Dem. For that reason the configuration includes a conversion formula similar to the one used by the RTE to describe physical data (but simpler) where only a linear mapping is supported. Note that

the conversion does not respect the UaSId. The service Need for the DTR provides a reference to a physical unit (see SWC Template chapter “Physical Units“) and a linear conversion (see SWC Template chapter “Computation Methods“) of CompuRational-Coeffs with a maximum of two numerator coefficients and one denominator coefficient, as well as the desired UaSId During the configuration process, this information is processed into a single linear conversion, which is then stored inside the ECUC of the Dem.

By this central data handling, the aspect of data consistency for Service$07 / $06 is supported as well as reset in case of fault clear.

[SWS_Dem_00750] ⌈ To ensure consistency between the state of its Event and the related DTR, Monitors providing DTRs shall not use EnableConditions. ⌋

[SWS_Dem_00751] ⌈ The Dem shall provide data structures and functionality to receive, store and report Service$06 test results persistently. ⌋

[SWS_Dem_00752] ⌈ The DTRId is assigned by the Dem during the BSW configuration step. ⌋*(SRS_Diag_04129)*

Note: The Dem refers to a BSW configuration DemDTRClass which associates

- a DTRId to an EventId
- an indication whether the Monitor will provide data for the minimum limit, maximum limit, or both
- OBDMID
- TID
- UaSID
- DemDtrUpdateKind
- Coefficients for linear conversion analogous to the SW-C Description (chapter 5.5.)

[SWS_Dem_00753] ⌈ For ensuring data consistency, not all EventId DTR mappings can be supported. That is, the Dem shall encounter (reference) only one single EventId. For two sided DTRs, the EventId has to be same for both limits. ⌋*(SRS_Diag_04001)*

[SWS_Dem_00754] ⌈ The Dem shall use the DemDtrUpdateKind for the evaluation and processing of reported DTR-values, i.e. DTRs are only forwarded to the Dcm, if the related event (if it is configured) has a certain state.

- ALWAYS: State of the Event is not considered (default, and used if DemDtrEventRef is not configured)
- STEADY: Only when the Events pre-debouncing within Dem is “stuck“ at the FAIL or PASS limit, and the latest result matches the debouncing direction

⌋*(SRS_Diag_04082)*

[SWS_Dem_00755] ⌈ The Dem shall only support a linear mapping, i.e. only support values for the coefficients b, c, and f. ⌋*(SRS_Diag_04082)*

Note: ASAM formula coefficients: (ax2 + bx + c) / (dx2 + ex + f) supporting on b, c and f.

[SWS_Dem_00756] ⌈ The Dem shall provide an API Dem_SetDTR within an interface DTRCentralReport to provide a report mechanism to the SWC for their test results. Arguments shall be the DTRId, the test value, the minimum and maximum limit, and a state indicating whether the monitor provides a valid minium limit, maximum limit, or both, or whether the reported values shall be reset to zero. ⌋

[SWS_Dem_00757] ⌈ Based on DemDtrUpdateKind and the status of the associated event (taking enable/storage conditions into account), the Dem shall either process or ignore the reported DTR values. ⌋*(SRS_Diag_04082)*

Note the example with STEADY: If an event has reached that status TESTED starting from an initialized state but has not detected a FAILED, the received DTR values within the limits are processed (c̄onverted and stored). If the test result violates its associated limit (cf. control value) while the eventis indeed TESTED and FAILED, the DTR report is processed as well. But if the test result violates its associated limit (cf. control value) while the event is TESTED and PASSED, the DTR report is ignored assuming that the EventId is also not updated for good reasons (cf. enable / storage conditions).

Applying this validation check within the DTR report, the data consistency in Service$06 and $07 is ensured.

[SWS_Dem_00758] ⌈ The Dem shall support a conversion of the ECUinternal resolution into the standardized external uint16 size and resolution according to the configured and standardized UaSID. For that reason, the conversion formula is configured per DTR. It is assumed that the conveyed parameter of the API call contain the values in the internal size and resolution. The Dem does offer sint32 as parameter for that reason to cover uint16 as well as sint16 (or even sint32) internal data types. ⌋

[SWS_Dem_00759] ⌈ The Dem shall ensure the validity of the test results vs. threshold(s). That is, if the threshold are (not) violated prior to the conversion, they shall (not) be violated after the conversion consistently. If rounding effects lead to an unintended change of the meaning, the thresholds shall be shifted by one increment accordingly to restore the result. ⌋

[SWS_Dem_00760] ⌈ Upon request by the Dcm, the Dem shall compute “SupportedOBDMID” information, i.e. the “availability OBDMIDs”. Since the Dem_DcmGetAvailableOBDMIDs() reports only value per standardized OBDMID ($00, $20, $40...), the DCM has to call this API iteratively until no further availability OBDMID is supported. ⌋

[SWS_Dem_00761] ⌈ Upon request by the Dcm, the Dem shall repond with the number of TIDs per requested OBDMID using the API Dem_DcmGetNumTIDsOfOBDMID(). This value can be used by the Dcm to iteratively request for the DTR data per OBDMID / TIDindex whereas the TIDindex loops from 0 to number-of-TIDs minus one. ⌋

[SWS_Dem_00762] ⌈ Upon request by the Dcm, the Dem shall respond with the data available for a particular OBDMID / TIDindex per requested OBDMID using the API Dem_DcmGetDTRData(). This value can be used by the Dcm to iteratively request for the DTR data per OBDMID / TIDindex starting from 0 to numberofTIDs minus one. ⌋*(SRS_Diag_04082)*

[SWS_Dem_00763] ⌈ The Dem shall reset the DTR data using the reference to the Event per DTR, whenever the Event is affected by a fault clear command. ⌋*(SRS_Diag_04001)*

[SWS_Dem_00764] ⌈ If no Event is assigned per DTR, the Dem shall reset the DTR data upon a Service$04 clear command by the Dcm, i.e. a Dem_DcmClearDTC () with:

DTC = DEM_DTC_GROUP_ALL_DTCS
DTCFormat = DEM_DTC_FORMAT_OBD
DTCOrigin = DEM_DTC_ORIGIN_PRIMARY_MEMORY

⌋*(SRS_Diag_04082)*

7.7.1.9 Service $0A - Permanent DTC

Permanent DTCs are stored in the permanent fault memory. An OBD related DTC entering the permanent fault memory with illuminating the MIL and leaving it with deactivating the MIL. The permanent fault memory is robust against ClearDTC and Power Fail. After a ClearDTC an event can only be removed from the permanent fault memory under special conditions. This special conditions are defined by the OBD regulation and correspond to the IUMPR general denominator conditions.

[SWS_Dem_00300] ⌈ The Dem shall be able to handle Permanent DTCs according to regulations (refer to [16]) within the specific event memory type “permanent” (refer to chapter 7.1.5). ⌋*(SRS_Diag_04082)*

[SWS_Dem_00590] ⌈ Events that have been confirmed and activate the MIL (refer to OBDrelevant DTCs, chapter 7.2.1), shall be stored robustly against ClearDTC or powerfail (for details confer [16]). ⌋*(SRS_Diag_04082)*

[SWS_Dem_00301] ⌈ The Dem shall provide the ability to access the stored Permanent DTC by filtering for permanent DTCs to the Dcm (refer to paragraph 8.3.4.1.6 Dem_DcmSetDTCFilter and paragraph 8.3.4.1.8 Dem_DcmGetNextFilteredDTC). ⌋*(SRS_Diag_04082)*

Note: For service $0A, the Dcm uses the DTCOrigin DEM_DTC_ORIGIN_PERMANENT_MEMORY.

Permanent DTCs must only be cleared once they have been self-healed if the last diagnosis result is detected as a PASS and the MIL is no longer being triggered for this event.

[SWS_Dem_00724] ⌈ Permanent DTCs from service $0A shall be deleted synchronously to MIL deactivation with the subsequent END of the OBD driving cycle. The Master ECU provides the “Minimum trip conditions satisfied” signal as socalled “PFC cycle” to the Primary ECUs via the Bussystem. ⌋

[SWS_Dem_00725] ⌈ Dem of the Master ECU shall provide the interface Dem_GetPfcCycleQualified (returning a boolean) (refer to chapter 8.3.8.13) to read the current state of the PFC cycle. TRUE indicates that during the current OBD driving cycle the conditions for the PFC cycle have been met. ⌋

[SWS_Dem_00726] ⌈ A software component in the Master ECU shall poll the API Dem_GetPfcCycleQualified(). ⌋

[SWS_Dem_00727] ⌈ This software component shall send this information via the bussystem. ⌋

[SWS_Dem_00728] ⌈ The Dem (of a Primary ECU) shall provide the interface Dem_SetPfcCycleQualified (refer to chapter 8.3.8.12) to get informed that the current OBD driving cycle has met the criteria for the PFC cycle by a software component. ⌋*(SRS_Diag_04082)*

7.8 J1939 specific functionality

The Dem module is capable to support J1939 networks. On these, it uses a J1939 specific DTC.

[SWS_Dem_00845] ⌈ The J1939 support shall be available in case the configuration container DemGeneralJ1939 (see DemGeneral) is present. ⌋*(SRS_Diag_04112)*

7.8.1 Read DTC

[SWS_Dem_00855] ⌈ The function Dem_J1939DcmSetDTCFilter (refer to section 8.3.5.1.1) shall set the filter mask attributes to be used for the subsequent calls of Dem_J1939DcmGetNumberOfFilteredDTC and Dem_J1939DcmGetNextFilteredDTC, and reset an internal counter to the first event. ⌋

[SWS_Dem_00856] ⌈ The filter mask attributes set via Dem_J1939DcmSetDTCFilter shall be used until the next call of Dem_J1939DcmSetDTCFilter or Dem initialization. ⌋*(SRS_Diag_04112)*

[SWS_Dem_00857] ⌈ The function Dem_J1939DcmSetDTCFilter shall return the current composite status of the J1939 lamps. The matching filter citreria are defined in Table 7.4. ⌋

SetFilter Parameter	*DTC Status filter*	*further filter criteria*	*DM representation*	
			AllDTCs	*EmissionDTCs*

DEM_J1939DTC_ACTIVE	(ConfirmedDTC == 1 AND TestFailed == 1) OR MIL_ON	n/a	DM1	DM12
DEM_J1939DTC_ PREVIOUSLY_ACTIVE	ConfirmedDTC == 1 AND TestFailed == 0 AND MIL_OFF	n/a	DM2	DM23
DEM_J1939DTC_PENDING	PendingDTC == 1	n/a	DM27	DM6
DEM_J1939DTC_ PERMANENT	n/a	permanent memory entry available	n/a	DM28
DEM_J1939DTC_ CURRENTLY_ACTIVE	TestFailed == 1	n/a	DM35	n/a

Table 7.4: Types of errors which can be detected by the Dem module

Each reading of DTC returns a compsite lamp status of the following lamps:

a) Malfunction Indicator Lamp

b) Red Stop Lamp

c) Amber Warning Lamp

d) Protect Lamp

7.8.1.1 Composite Malfunction Indicator Lamp Status

[SWS_Dem_00858] ⌈ The composite “Malfunction Indicator Lamp” shall be set to “Lamp Off” in case the Indicator referenced by DemMILIndicatorRef has an IndicatorStatus DEM_INDICATOR_OFF. All other IndicatorStatus states than “DEM_INDICATOR_OFF” shall set the composite Malfunction Indicator Lamp to “Lamp On”. ⌋*(SRS_Diag_04110)*

[SWS_Dem_00859] ⌈ The composite “Flash Malfunction Indicator Lamp” shall be set to “Unavailable / Do Not Flash” (0x03) in case the Indicator referenced by DemMILIndicatorRef has an IndicatorStatus “DEM_INDICATOR_OFF“ or “DEM_INDICATOR_CONTINUOUS”. ⌋*(SRS_Diag_04110)*

[SWS_Dem_00860] ⌈ The composite “Flash Malfunction Indicator Lamp” shall be set to “Slow Flash” (0x00) in case the Indicator referenced by DemMILIndicatorRef has an IndicatorStatus DEM_INDICATOR_SLOW_FLASH. ⌋*(SRS_Diag_04110)*

[SWS_Dem_00861] ⌈ The composite “Flash Malfunction Indicator Lamp” shall be set to “Fast Flash” (0x01) in case the Indicator referenced by DemMILIndicatorRef has an IndicatorStatus DEM_INDICATOR_FAST_FLASH. ⌋*(SRS_Diag_04110)*

7.8.1.2 Composite Red Stop Lamp Status

[SWS_Dem_00862] ⌈ The composite “Red Stop Lamp” shall be set to “Lamp Off” in case the Indicator referenced by DemRedStopLampIndicatorRef has an IndicatorStatus DEM_INDICATOR_OFF. All other IndicatorStatus states than “DEM_INDICATOR_OFF” shall set the composite Red Stop Lamp to “Lamp On”. ⌋*(SRS_Diag_04110)*

[SWS_Dem_00863] ⌈ The composite “Flash Red Stop Lamp” shall be set to “Unavailable / Do Not Flash” (0x03) in case the Indicator referenced by DemRedStopLampIndicatorRef has an IndicatorStatus “DEM_INDICATOR_OFF“ or “DEM_INDICATOR_CONTINUOUS”. ⌋*(SRS_Diag_04110)*

[SWS_Dem_00864] ⌈ The composite “Flash Red Stop Lamp” shall be set to “Slow Flash” (0x00) in case the Indicator referenced by DemRedStopLampIndicatorRef has an IndicatorStatus DEM_INDICATOR_SLOW_FLASH. ⌋*(SRS_Diag_04112)*

[SWS_Dem_00865] ⌈ The composite “Flash Red Stop Lamp” shall be set to “Fast Flash” (0x01) in case the Indicator referenced by DemRedStopLampIndicatorRef has an IndicatorStatus DEM_INDICATOR_FAST_FLASH. ⌋*(SRS_Diag_04069)*

7.8.1.3 Composite Amber Warning Lamp Status

[SWS_Dem_00866] ⌈ The composite “Amber Warning Lamp” shall be set to “Lamp Off” in case the Indicator referenced by DemAmberWarningLampIndicatorRef has an IndicatorStatus DEM_INDICATOR_OFF. All other IndicatorStatus states than “DEM_INDICATOR_OFF” shall set the composite Amber Warning Lamp to “Lamp On”. ⌋*(SRS_Diag_04110)*

[SWS_Dem_00867] ⌈ The composite “Amber Warning Lamp” shall be set to “Unavailable / Do Not Flash” (0x03) in case the Indicator referenced by DemAmberWarningLampIndicatorRef has an IndicatorStatus “DEM_INDICATOR_OFF” or “DEM_INDICATOR_CONTINUOUS”. ⌋

[SWS_Dem_00868] ⌈ The composite “Flash Amber Warning Lamp” shall be set to “Slow Flash” (0x00) in case the Indicator referenced by DemAmberWarningLampIndicatorRef has an IndicatorStatus DEM_INDICATOR_SLOW_FLASH. ⌋*(SRS_Diag_04110)*

[SWS_Dem_00869] ⌈ The composite “Flash Amber Warning Lamp” shall be set to “Fast Flash” (0x01) in case the Indicator referenced by DemAmberWarningLampIndicatorRef has an IndicatorStatus DEM_INDICATOR_FAST_FLASH. ⌋

7.8.1.4 Composite Protect Lamp Status

[SWS_Dem_00870] ⌈ The composite “Protect Lamp” shall be set to “Lamp Off” in case the Indicator referenced by DemProtectIndicatorRef has an

IndicatorStatus DEM_INDICATOR_OFF. All other IndicatorStatus states than “DEM_INDICATOR_OFF” shall set the composite Protect Lamp to “Lamp On”. ⌋

[SWS_Dem_00871] ⌈ The composite “Flash Protect Lamp” shall be set to “Unavailable / Do Not Flash” (0x03) in case the Indicator referenced by DemProtectIndicatorRef has an IndicatorStatus “DEM_INDICATOR_OFF” or “DEM_INDICATOR_CONTINUOUS”. ⌋

[SWS_Dem_00872] ⌈ The composite “Flash Protect Lamp” shall be set to “Slow Flash” (0x00) in case the Indicator referenced by DemProtectIndicatorRef has an IndicatorStatus DEM_INDICATOR_SLOW_FLASH. ⌋*(SRS_Diag_04110)*

[SWS_Dem_00873] ⌈ The composite “Flash Protect Lamp” shall be set to “Fast Flash” (0x01) in case the Indicator referenced by DemProtectIndicatorRef has an IndicatorStatus DEM_INDICATOR_FAST_FLASH. ⌋*(SRS_Diag_04110)*

7.8.1.5 DTC data acqusition

[SWS_Dem_00874] ⌈ The function Dem_J1939DcmGetNumberOfFilteredDTC shall return the number of J1939 DTCs matching the defined filter criteria of Table 4 for a specific NodeAddress defined by the function call of Dem_J1939DcmSetDTCFilter.

- In case the function has calculated the total number of matching DTCs, the return value shall be DEM_NUMBER_OK.
- In case the search needs to be interrupted due to internal implementations or limitations, the return value can be set to DEM_NUMBER_PENDING. The out parameter needs not to be valid in this case. The next call of Dem_J1939DcmGetNumberOfFilteredDTC should continue after the interrupted element.

⌋*(SRS_Diag_04112)*

[SWS_Dem_00875] ⌈ Each call of the function Dem_J1939DcmGetNextFilteredDTC shall search for the next event having a J1939DTC assigned and matching the filter criteria of Table 4 for the NodeAddress (see DemJ1939DTC_NodeAddressRef) provided by the function call of Dem_J1939DcmSetDTCFilter. In case no more events are matching the filter criteria, the function return value shall be DEM_FILTERED_NO_MATCHING_ELEMENT. The out parameters need not to be valid in this case. ⌋

[SWS_Dem_00877] ⌈ In case the function Dem_J1939DcmGetNextFilteredDTC has found an event matching the filter criteria it shall return the corresponding J1939DTC (refer to DemDTCAttributes for details) and the corresponding occurrence counter. The return value shall be DEM_FILTERED_OK. In case the search needs to be interrupted due to internal implementations or limitations, the return value can be set to DEM_FILTERED_PENDING. The out parameter needs not to be valid in this case. The next call of Dem_J1939DcmGetNextFilteredDTC should continue after the interrupted

element. In case the occurrence counter is above +126 (0x7F), the returned values shall be set to +126 (0x7F). ⌋*(SRS_Diag_04111)*

7.8.2 Clear DTCs

For definition of locking the clearing process, refer to chapter 7.3.2.2.

[SWS_Dem_00878] ⌈ The Dem module shall provide the Dem_J1939DcmClearDTC (refer to paragraph 8.3.6.2.1) to the J1939Dcm [4] for deleting all active or previously active DTCs from the event memory. This shall trigger also initializing of related SW-Cs / BSW modules according to [SWS_Dem_00003]. ⌋

When one of the diagnostic messages DM3 and DM11 of SAE J1939-73 [11] are requested, all active (DM11) or previously active (DM3) DTCs have to be cleared, along with additional information like freeze frames.

[SWS_Dem_00879] ⌈ The function Dem_J1939DcmClearDTC shall clear the status of all event(s) related to the specified DTC(s), as well as all associated event memory entries for these event(s). ⌋*(SRS_Diag_04117)*

Note: Exceptions may apply due to [SWS_Dem_00514].

7.8.3 DM31

The DM31 provides the applicable lamp(s) and their status for each individual DTC, wherefore the DTC-specific lamp status needs to be returned by the function Dem_J1939DcmGetNextDTCwithLampStatus.

[SWS_Dem_00880] ⌈ Each call to Dem_J1939DcmFirstDTCwithLampStatus shall set the internal counter to the first event having a J1939DTC for this particular “NodeAddress” assiged. ⌋*(SRS_Diag_04113)*

[SWS_Dem_00881] ⌈ Each call to Dem_J1939DcmGetNextDTCwithLampStatus shall search for the next event having a J1939DTC assigned for “NodeAddress” set by the function Dem_J1939DcmFirstDTCwithLampStatus. In case no more events are available that have a J1939DTC assigend, the function return value shall be DEM_FILTERED_NO_MATCHING_ELEMENT. The out parameter needs not to be valid in this case. ⌋*(SRS_Diag_04113)*

[SWS_Dem_00882] ⌈ In case Dem_J1939DcmGetNextDTCwithLampStatus has found an event matching the filter criteria it shall return the J1939DTC specific status of the lamps (refer to following sections for details), the corresponding J1939DTC (refer DemDTCAttributes for details) and the corresponding occurrence counter. The return value shall be DEM_FILTERED_OK. In case the search needs to be interrupted due to internal implementations or limitations, the return value can be set to DEM_FILTERED_PENDING. The out parameter needs not to be valid in this case. The next call of Dem_J1939DcmGetNextDTCwithLampStatus should continue after the in-

terrupted element. In case the occurrence counter is above +126 (0x7F), the returned values shall be set to +126 (0x7F). ⌋(*SRS_Diag_04113*)

7.8.3.1 Malfunction Indicator Lamp

[SWS_Dem_00883] ⌈ The DTCspecific “Malfunction Indicator Lamp” returned in the function Dem_J1939DcmGetNextDTCwithLampStatus shall be set to “Lamp On” in case the corresponding event has assigned the Indicator referenced by DemMILIndicatorRef and the UDS DTC status bit 7 (WarningIndicator) is active (set to 1), otherwise it shall be set to ’Lamp Off’. ⌋(*SRS_Diag_04110*)

[SWS_Dem_00884] ⌈ The DTC-specific “Flash Malfunction Indicator Lamp” shall be set to “Unavailable / Do Not Flash” (0x03) in case the corresponding event has set “Lamp On” and the DemIndicatorBehaviour of the Indicator referenced by DemMILIndicatorRef is set to “DEM_INDICATOR_CONTINUOUS”. ⌋(*SRS_Diag_04110*)

[SWS_Dem_00885] ⌈ The DTC-specific “Flash Malfunction Indicator Lamp” shall be set to “Slow Flash” (0x00) in case the corresponding event has set “Lamp On” and the DemIndicatorBehaviour of the Indicator referenced by DemMILIndicatorRef is set to “DEM_INDICATOR_SLOW_FLASH”. ⌋(*SRS_Diag_04110*)

[SWS_Dem_00886] ⌈ The DTC-specific “Flash Malfunction Indicator Lamp” shall be set to “Fast Flash” (0x01) in case the corresponding event has set “Lamp On” and the DemIndicatorBehaviour of the Indicator referenced by DemMILIndicatorRef is set to “DEM_INDICATOR_FAST_FLASH”. ⌋(*SRS_Diag_04069*)

7.8.3.2 Red Stop Lamp

[SWS_Dem_00887] ⌈ The DTC-specific “Red Stop Lamp” returned in the function Dem_J1939DcmGetNextDTCwithLampStatus shall be set to “Lamp On” in case the corresponding event has assigned the Indicator referenced by DemRedStopIndicatorRef and the UDS DTC status bit 7 (WarningIndicator) is active (set to 1), otherwise it shall be set to “Lamp Off”. ⌋(*SRS_Diag_04110*)

[SWS_Dem_00888] ⌈ The DTC-specific “Flash Red Stop Lamp” shall be set to “Unavailable / Do Not Flash” (0x03) in case the corresponding event has set “Lamp On” and the DemIndicatorBehaviour of the Indicator referenced by DemRedStopIndicatorRef is set to “DEM_INDICATOR_CONTINUOUS”. ⌋

[SWS_Dem_00889] ⌈ The DTC-specific “Flash Red Stop Lamp” shall be set to “Slow Flash” (0x00) in case the corresponding event has set “Lamp On” and the DemIndicatorBehaviour of the Indicator referenced by DemRedStopIndicatorRef is set to “DEM_INDICATOR_SLOW_FLASH”. ⌋(*SRS_Diag_04110*)

[SWS_Dem_00890] ⌈ The DTC-specific “Flash Red Stop Lamp” shall be set to “Fast Flash” (0x01) in case the corresponding event has set “Lamp On” and the De-

mIndicatorBehaviour of the Indicator referenced by DemRedStopIndicatorRef is set to “DEM_INDICATOR_FAST_FLASH”. ⌋

7.8.3.3 Amber Warning Lamp

[SWS_Dem_00891] ⌈ The DTC-specific “Amber Warning Lamp” returned in the function Dem_J1939DcmGetNextDTCwithLampStatus shall be set to “Lamp On” in case the corresponding event has assigned the Indicator referenced by DemAmberWarningIndicatorRef and the UDS DTC status bit 7 (WarningIndicator) is active (set to 1), otherwise it shall be set to “Lamp Off”. ⌋

[SWS_Dem_00892] ⌈ The DTC-specific “Flash Amber Warning Lamp” shall be set to “Unavailable / Do Not Flash” (0x03) in case the corresponding event has set “Lamp On” and the DemIndicatorBehaviour of the Indicator referenced by DemAmberWarningIndicatorRef is set to “DEM_INDICATOR_CONTINUOUS”. ⌋

[SWS_Dem_00893] ⌈ The DTC-specific “Flash Amber Warning Lamp” shall be set to “Slow Flash” (0x00) in case the corresponding event has set “Lamp On” and the DemIndicatorBehaviour of the Indicator referenced by DemAmberWarningIndicatorRef is set to “DEM_INDICATOR_SLOW_FLASH”. ⌋*(SRS_Diag_04110)*

[SWS_Dem_00894] ⌈ The DTC-specific “Flash Amber Warning Lamp” shall be set to “Fast Flash” (0x01) in case the corresponding event has set “Lamp On” and the DemIndicatorBehaviour of the Indicator referenced by DemAmberWarningIndicatorRef is set to “DEM_INDICATOR_FAST_FLASH”. ⌋*(SRS_Diag_04110)*

7.8.3.4 Protect Lamp

[SWS_Dem_00895] ⌈ The DTC-specific “Protect Lamp” returned in the function Dem_J1939DcmGetNextDTCwithLampStatus shall be set to “Lamp On” in case the corresponding event has assigned the Indicator referenced by DemProtectIndicatorRef and the UDS DTC status bit 7 (WarningIndicator) is active (set to 1), otherwise it shall be set to “Lamp Off”. ⌋*(SRS_Diag_04110)*

[SWS_Dem_00896] ⌈ The DTC-specifc “Flash Protect Lamp” shall be set to “Unavailable / Do Not Flash” (0x03) in case the corresponding event has set “Lamp On” and the DemIndicatorBehaviour of the Indicator referenced by DemProtectIndicatorRef is set to “DEM_INDICATOR_CONTINUOUS”. ⌋*(SRS_Diag_04069)*

[SWS_Dem_00897] ⌈ The DTC-specifc “Flash Protect Lamp” shall be set to “Slow Flash” (0x00) in case the corresponding event has set “Lamp On” and the DemIndicatorBehaviour of the Indicator referenced by DemProtectIndicatorRef is set to “DEM_INDICATOR_SLOW_FLASH”. ⌋*(SRS_Diag_04110)*

[SWS_Dem_00898] ⌈ The DTC-specifc “Flash Protect Lamp” shall be set to “Fast Flash” (0x01) in case the corresponding event has set “Lamp On” and the De-

mIndicatorBehaviour of the Indicator referenced by DemProtectIndicatorRef is set to “DEM_INDICATOR_FAST_FLASH”. ⌋

7.8.4 FreezeFrame

[SWS_Dem_00899] ⌈ The function Dem_J1939DcmSetFreezeFrameFilter (refer to section 8.3.5.3.2) shall set the filter mask attributes to be used for the subsequent calls of Dem_J1939DcmGetNextFreezeFrame, and reset an internal counter to the first freeze frame. The filter mask attributes shall be used until the next call of Dem_J1939DcmSetFreezeFrameFilter or Dem initialization. The matching filter citreria are DEM_J1939DCM_FREEZEFRAME (filtering the data in DemJ1939FreezeFrameClassRef) or DEM_J1939DCM_EXPANDED_FREEZEFRAME (filtering the data in DemJ1939ExpandedFreezeFrameClassRef). ⌋*(SRS_Diag_04111)*

[SWS_Dem_00900] ⌈ Each function call of Dem_J1939DcmGetNextFreezeFrame shall step to the next (Expanded-)FreezeFrame for a specific NodeAddress defined by the filter criteria (FreezeFrameKind and NodeAddress) of Dem_J1939DcmSetFreezeFrameFilter. In case no more (Expanded-)FreezeFrames are matching the filter criteria, the function shall return DEM_FILTERED_NO_MATCHING_ELEMENT. The out parameters need not to be valid in this case. ⌋*(SRS_Diag_04111)*

[SWS_Dem_00901] ⌈ The function Dem_J1939DcmGetNextFreezeFrame shall trigger the DET error DEM_E_WRONG_CONDITION in case of not supported FreezeFrameKind. Valid values are DEM_J1939DCM_FREEZEFRAME and J1939DCM_EXPANDED_FREEZEFRAME. ⌋*(SRS_Diag_04111)*

[SWS_Dem_00902] ⌈ In case the search of [SWS_Dem_00901] needs to be interrupted due to internal implementations or limitations, the return value can be set to DEM_FILTERED_PENDING. The out parameter needs not to be valid in this case. The next call of Dem_J1939DcmGetNextFreezeFrame should continue after the interrupted element. ⌋*(SRS_Diag_04111)*

[SWS_Dem_00903] ⌈ In case the function Dem_J1939DcmGetNextFreezeFrame has found a FreezeFrame, the Dem shall:

- Check if the buffer in the BufSize parameter is big enough to hold the (Expanded-)FreezeFrame. If not, DEM_FILTERED_BUFFER_TOO_SMALL shall be returned without any further actions. The out parameters need not to be valid in this case.
- Copy the (Expanded-)FreezeFrame data into the buffer provided by the parameter DestBuffer (in case of Expanded FreezeFrames without any SPN informations). Unused bits shall be filled with “0”.

- Set the parameter J1939DTC to the corresponding J1939DTC value (refer DemDTCAttributes for details) and the parameter OccurrenceCounter to the corresponding occurrence counter value.
- Return with DEM_FILTERED_OK. In case the occurrence counter is above +126 (0x7F), the returned value shall be set to +126 (0x7F).

⌋(*SRS_Diag_04111*)

7.8.4.1 SPNs in ExpandedFreezeFrame

[SWS_Dem_00904] ⌈ Each call to Dem_J1939DcmGetNextSPNInFreezeFrame shall step to the next SPN of the ExpandedFreezeFrame definition for a specific NodeAddress. In case no more SPN of the ExpandedFreezeFrame definition is available, the function return value shall be DEM_FILTERED_NO_MATCHING_ELEMENT. The out parameter needs not to be valid in this case. ⌋(*SRS_Diag_04111*)

[SWS_Dem_00905] ⌈ In case the search of [SWS_Dem_00904] needs to be interrupted due to internal implementations or limitations, the return value can be set to DEM_FILTERED_PENDING. The out parameter needs not to be valid in this case. The next call of Dem_J1939DcmGetNextSPNInFreezeFrame should continue after the interrupted element. ⌋(*SRS_Diag_04111*)

[SWS_Dem_00906] ⌈ The function Dem_J1939DcmGetNextSPNInFreezeFrame shall trigger the DET error DEM_E_WRONG_CONDITION in case of not supported FreezeFrameKind. Valid value is Dem_SPNsInExpandedFreezeFrame. ⌋(*SRS_Diag_04057, SRS_Diag_04111*)

[SWS_Dem_00907] ⌈ In case Dem_J1939DcmGetNextSPNInFreezeFrame has found an SPN of the ExpandedFreezeFrame definition, the Dem shall:

- Check if the buffer in the BufSize parameter is big enough to hold the (Expanded-)FreezeFrame. If not, DEM_FILTERED_BUFFER_TOO_SMALL shall be returned without any further actions. The out parameters need not to be valid in this case.
- Copy the (Expanded-)FreezeFrame data into the buffer provided by the parameter DestBuffer (in case of Expanded FreezeFrames without any SPN informations). Unused bits shall be filled with “0”.
- Set the parameter J1939DTC to the corresponding J1939DTC value (refer DemDTCAttributes for details) and the parameter OccurrenceCounter to the corresponding occurrence counter value.
- Return with DEM_FILTERED_OK. In case the occurrence counter is above +126 (0x7F), the returned value shall be set to +126 (0x7F).

⌋(*SRS_Diag_04111*)

7.8.5 Diagnostic Readiness

The Dem module needs to calculate internally the (Non-)Continuously Monitored Systems support and status for “Diagnostic Readiness 1” (DM5), “Diagnostic Readiness 2” (DM21), and “Diagnostic Readiness 3” (DM26).

[SWS_Dem_00908] ⌈ The J1939 events shall use the configuration parameter DemEventOBDReadinessGroup (refer to DemObdDTC) to assign individual events to a Continuously or Non-Continuously Monitored System. ⌋

Dem_J1939DcmReadDiagnosticReadiness1 reports the diagnostics information that relates to diagnostic readiness according DM5.

[SWS_Dem_00909] ⌈ A call of Dem_J1939DcmReadDiagnosticReadiness1 shall report a response message based on the J193973 DM5 definition:

- “Active Trouble Codes” shall report the number of active DTCs identical to the number of filtered DTCs by Dem_J1939DcmSetDTCFilter(Dem_Active, DEM_DTC_KIND_ALL_DTCS).
- “Previously Active Diagnostic Trouble Codes” shall report the number of previously active DTCs identical to the number of filtered DTCs by Dem_J1939DcmSetDTCFilter(Dem_PreviouslyActive, DEM_DTC_KIND_ALL_DTCS).
- “OBD Compliance” shall be based on the configuration parameter DemOBDCompliancy (in DemGeneralOBD). For nonOBD ECUs the value five (5) shall be reported.
- The (Non-)Continuously Monitored Systems support and status shall be reported according to SAE J1939-73 chapter 5.7.5.4 to 5.7.5.6. The calculation shall be based on those events referencing the corresponding configuration parameter DemEventOBDReadinessGroup. In case there is no reference by any event parameter, the support bit shall be set to zero (0). The “monitoring status” bit shall be set to “test complete” (0) if all assigned event parameter have the TestNotCompletedSinceLastClear status bit set to zero (0), otherwise the “monitoring status” bit shall be set to “test not complete” (1).

⌋

Dem_J1939DcmReadDiagnosticReadiness2 reports the diagnostic information relevant to a second PGN conveying diagnostic readiness according to DM21 (see also DM5).

[SWS_Dem_00910] ⌈ A call of Dem_J1939DcmReadDiagnosticReadiness2 shall report a response message based on the J193973 DM21 definition:

- “Distance Traveled While MIL is Activated” shall be identical to PID 0x21, except the maximum value is 64255 (all values above 64255 shall be reported as 64255).

- “Distance Since Diagnostic Trouble Codes Cleared” shall be identical to PID 0x31, except the maximum value is 64255 (all values above 64255 shall be reported as 64255).
- “Minutes Run by Engine While MIL is Activated” shall be identical to PID 0x4D, except the maximum value is 64255 (all values above 64255 shall be reported as 64255).
- “Time Since Diagnostic Trouble Codes Cleared” shall be identical to PID 0x4E, except the maximum value is 64255 (all values above 64255 shall be reported as 64255).

⌋

Dem_J1939DcmReadDiagnosticReadiness3 reports the diagnostic information conveying the pending status of OBD system monitors for the current drive cycle according to DM26 (See also DM5 and DM21).

[SWS_Dem_00911] ⌈ A call of Dem_J1939DcmReadDiagnosticReadiness3 shall report a response message based on the J193973 DM26 definition:

- “Time Since Engine Start” shall be retrieved by the DemOBDTimeSinceEngineStart (in DemGeneralOBD)
- “Number of Warmups Since DTCs Cleared” shall be identical to PID 0x30, except the maximum value is 250 (all values above 250 shall be reported as 250).
- The (Non-)Continuously Monitored Systems Enable/Completed status shall be reported according to SAE J193973 chapter 5.7.26.3 to 5.7.26.5. The calculation shall be based on those events referencing the corresponding configuration parameter DemEventOBDReadinessGroup. In case there is no reference by any event parameter it shall shall indicate disabled and complete. Otherwise the disable state shall be based on the readiness group disabled status according to [SWS_Dem_00356], and the complete status shall be set to “monitor complete” (0) if all assigned event parameter have the TestNotCompletedSinceThisOperationCycle status bit set to zero (0); otherwise the status bit shall be set to “monitor not complete” (1).

⌋

7.8.6 Monitor Performance Ratio

For J1939 the In-Use-Monitor Performace Ratio (IUMPR) requires a reporting for each supported Applicable System Monitor only.

[SWS_Dem_00912] ⌈ The function Dem_J1939DcmSetRatioFilter (refer to section 8.3.5.4.1) shall reset an internal counter to the first valid SPN with DemRatioId defined to be used for the subsequent calls of Dem_J1939DcmGetNextFilteredRatio. Furthermore, it shall return the “Ignition Cycle Counter” according SAEJ193973 chapter

5.7.20.1, as well as “OBD Monitoring Conditions Encountered” according SAEJ193973 chapter 5.7.20.2 (CARB defines this as the general denominator). ⌋

[SWS_Dem_00913] ⌈ Each call of the function Dem_J1939DcmGetNextFilteredRatio shall skip to the next valid SPN for a specific NodeAddress with DemRatioId defined.

- The events referenced by this SPN via DemRatioId shall be used to calculate the return values SPN (for “SPN of Applicable System Monitor”), Numerator (for “Applicable System Monitor Numerator”), and Denominator (for “Applicable System Monitor Denominator”). The return value shall be DEM_FILTERED_OK.
- In case the calculation needs to be interrupted due to internal implementations or limitations, the return value can be set to DEM_FILTERED_PENDING. The out parameter needs not to be valid in this case. The next call of Dem_J1939DcmGetNextFilteredRatio should continue after the interrupted element.
- In case no more SPNs are available, the function return value shall be DEM_FILTERED_NO_MATCHING_ELEMENT. The out parameter needs not to be valid in this case.

⌋*(SRS_Diag_04113)*

7.9 Interaction with other Software Modules

7.9.1 Interaction with Software Components (SW-C)

In the AUTOSAR ECU architecture, the Diagnostic Event Manager implements an AUTOSAR Service. This means that the Dem module does not exclusively communicate with other BSW modules, but also with SWC via the RTE (refer to Figure 1).

From the viewpoint of the BSW “Cmodule” Dem, there are three kinds of dependencies between the Service and the AUTOSAR Software Components above the RTE:

- The application accesses the API (implemented as Cfunctions) of the Dem (by accessing the Dem Service Component).
- The application is optionally notified upon the outcome of requested asynchronous activity (via direct/indirect RTE API by the Dem).
- An initialization function of the SWC is invoked by the Dem.

These dependencies must be described in terms of the AUTOSAR metamodel, which will contribute to the SWC Description of the application component as well as to the Service Component Description of the Dem Service.

The service component description of the Dem Service will define the ports below the RTE. Each SWC, which uses the Dem Service, must contain respective ports in its own SWC description, which will be typed by the same (or compatible) interfaces and must

be connected to the ports of the Dem, so that the RTE can be generated. Ids used in these Dem-ports are abstracted with portdefined arguments.

[SWS_Dem_00512] ⌈ The name of the Dem Service Component shall be “Dem”. ⌋

The callbacks to SWCs do not use the mechanism of the portdefined arguments. Instead, the Dem configuration mechanism must ensure that the callback is delivered to the configured port and invokes the correct operation at this port using an RTE (direct or indirect) API call. For example, the EventId must not be passed as the first argument of the operation, because the monitors (and other SW-Cs) do not cope with EventIds explicitly.

In contrast to that, there are some special SWCs, that need to handle event status/data changes uniformly. These SWCs are notified by the Dem, which provides then also the EventId for the application, to be able, to request the respective data from the Dem. Therefore, general interfaces are provided (refer to chapters chapter 8.6.2.9, chapter 8.6.2.20, and chapter 8.6.2.3), which shall only be used in the way, that the EventIds (given by the Dem) shall not be interpreted by SWCs in any way, as well as the EventId values (used by the SW-Cs to request the event-specific data) have always to be provided by the Dem. Here only a handthrough mechanism is allowed to be used by SWCs for EventIds.

In the current Autosar release, the RTE analyzes the complete calltree of triggered runnables. Therefore, some limitations in the Dem API are introduced to avoid function calls from BSW modules, which rely on callbacks to SWCs via RTE.

7.9.2 Interaction with Diagnostic Communication Manager (Dcm)

The Dcm accesses the Dem module in order to delegate those diagnostic services, which are related to the event memory.

For detailed description of the interface usage between Dcm and Dem consult the Dcm SWS document [7, SWS Dcm]. There, especially the handling of freeze frame data is described.

[SWS_Dem_00171] ⌈ If the Dcm has requested an unavailable event memory / DTC origin, the Dem module functions with the respective return value shall return DEM_<...>_WRONG_DTCORIGIN. ⌋*(SRS_Diag_04058)*

[SWS_Dem_00172] ⌈ If the Dcm has requested a DTC that is not available at all or that is available but has a different event memory / DTC origin than the requested one, the Dem module functions with the respective return value shall return DEM_<...>_WRONG_DTC. ⌋

[SWS_Dem_00827] ⌈ The Dem shall provide the API Dem_DcmControlDTCStatusChangedNotification. ⌋*(SRS_Diag_04010)*

[SWS_Dem_00828] ⌈ If the API Dem_DcmControlDTCStatusChangedNotification with state value TRUE and the parameter DemTriggerDcmReports is en-

abled, the Dem starts to report changes of the DTC status to the Dcm via Dcm_DemTriggerOnDTCStatus. ⌋

[SWS_Dem_00829] ⌈ If the API Dem_DcmControlDTCStatusChangedNotification with state value FALSE is enabled, the Dem stops to report changes of the DTC status to the Dcm via Dcm_DemTriggerOnDTCStatus. ⌋*(SRS_Diag_04105)*

[SWS_Dem_00830] ⌈ If the DTC status changed by clearing the event memory Dcm_DemTriggerOnDTCStatus shall not be called. ⌋

Note: The configuration parameter DemTriggerDcmReports enables this function. The function Dcm_DemTriggerOnDTCStatus uses the same prototype as <Module>_DemTriggerOnDTCStatus (refer to chapter 8.4.3.1.4). This configuration parameter is for simplicity (so the generic parameter DemCallbackDTCStatusChangedFnc needs not to be used).

7.9.2.1 Access DTCs and Status Information

The following chapter defines the APIs, which shall be used to access the number of DTCs, and DTCs matching specific filter criteria and the associated status information.

[SWS_Dem_00231] ⌈ The function Dem_DcmGetTranslationType (refer to paragraph 8.3.4.1.1) shall provide the capability to get the configured translation format of the ECU (refer to DemTypeOfDTCSupported in DemGeneral). ⌋

[SWS_Dem_00060] ⌈ The function Dem_DcmGetDTCStatusAvailabilityMask shall provide the DTC status availability mask (refer to DemDtcStatusAvailabilityMask in DemGeneral), that means the DTC status information (according to [2, ISO 14229-1]) supported by the Dem module with respect to the tester client with UDS service 0x19. ⌋*(SRS_Diag_04067, SRS_Diag_04010)*

[SWS_Dem_00657] ⌈ The Dem module shall mask all DTC status bytes (API parameters DTCStatus) provided to the Dcm with the DTC status availability mask (by performing a bit-wise AND operation). ⌋*(SRS_Diag_04010)*

[SWS_Dem_00059] ⌈ The function Dem_DcmGetStatusOfDTC (refer to paragraph 8.3.4.1.3) shall copy the status of a DTC to the API parameter DTCStatus according to [2, ISO 14229-1]. ⌋*(SRS_Diag_04067)*

It is possible, that a DTC with different states exists depending on the location. If the secondary memory is used as a kind of protocol stack that gives information which services have been performed on the primary memory, different DTC states might appear (e.g. DTC has been deleted from the primary memory and is written to the secondary memory with its latest status).

[SWS_Dem_00057] ⌈ The function Dem_DcmSetDTCFilter (refer to chapter paragraph 8.3.4.1.6) shall set the filter criteria attributes to be used for the subsequent calls of Dem_DcmGetNumberOfFilteredDTC, Dem_DcmGetNextFilteredDTC, Dem_DcmGetNextFilteredDTCAndFDC, as well as

Dem_DcmGetNextFilteredDTCAndSeverity. The filter criteria attributes shall be used until the next call of Dem_DcmSetDTCFilter or Dem initialization. ⌋*(SRS_Diag_04057)*

[SWS_Dem_00649] ⌈ Each call of Dem_DcmSetDTCFilter shall lead to a reset of the sequence. ⌋*(SRS_Diag_04057)*

[SWS_Dem_01058] ⌈ The function Dem_DcmSetDTCFilter shall ignore unsupported bits (refer to configuration parameter DemDtcStatusAvailabilityMask) retrieved in DTCStatusMask. ⌋

[SWS_Dem_00061] ⌈ The function Dem_DcmGetNumberOfFilteredDTC (refer to paragraph 8.3.4.1.7) shall get the number of DTCs matching the filter criteria defined by the function call of Dem_DcmSetDTCFilter [SWS_Dem_00057]. ⌋*(SRS_Diag_04000)*

Dem_DcmSetDTCFilter has to be called prior to Dem_DcmGetNumberOfFilteredDTC.

[SWS_Dem_00216] ⌈ With each call to the function Dem_DcmGetNextFilteredDTC (refer to chapter 8.3.4.1.8) the Dem module shall return the next DTC and its associated status matching the filter criteria defined by the function call of Dem_DcmSetDTCFilter [SWS_Dem_00057]. ⌋*(SRS_Diag_04010)*

The Dcm calls the function Dem_DcmGetNextFilteredDTC continuously until the return value of the function is DEM_FILTERED_NO_MATCHING_ELEMENT, to receive all DTCs matching the filter criteria.

[SWS_Dem_00653] ⌈ The API Dem_DcmGetNextFilteredDTC shall not return the value DEM_FILTERED_PENDING when being used for emissionrelated services, indicated by any of these parameter combination:

- DTCKind=DEM_DTC_KIND_EMISSION_REL_DTCS and DTCStatusMask=0x08
- DTCKind=DEM_DTC_KIND_EMISSION_REL_DTCS and DTCStatusMask=0x04
- DTCKind=DEM_DTC_KIND_EMISSION_REL_DTCS and DTCOrigin=DEM_DTC_ORIGIN_PERMANENT

⌋

Rationale: The API Dem_DcmGetNextFilteredDTC is used for UDS service 0x19 as well as OBD services $03/$07/$0A (refer to SWS_Dem_00301). UDS service 0x19 is allowed to respond with NRC 0x78 (response pending) to the tester, while the OBD services are not.

[SWS_Dem_00228] ⌈ With each call to the function Dem_DcmGetNextFilteredDTCAndFDC (refer to chapter 8.3.4.1.9) the Dem module shall return the next DTC and its associated fault detection counter (FDC, refer to SWS_Dem_00264) matching the filter criteria defined by the function call Dem_DcmSetDTCFilter (refer to [SWS_Dem_00057]). ⌋*(SRS_Diag_04010)*

[SWS_Dem_00513] ⌈ If the callbackfunction GetFaultDetectionCounter (refer to chapter 8.3.3.19) returns other than E_OK for a DTC filtered by the function Dem_DcmGetNextFilteredDTCAndFDC, this DTC shall not match the filter criteria (i.e. assume FDC is 0). ⌋

The Dcm calls the function Dem_DcmGetNextFilteredDTCAndFDC continuously until the return value of the function is DEM_FILTERED_NO_MATCHING_ELEMENT, to receive all DTCs matching the filter criteria.

Note: Non-Dem-internal calculated fault detection counters are typically requested from SWCs through the RTE. To indicate an equivalent calltree for these runnables, a workaround is used: The Dcm main function specifies a trigger to the Dem interface GeneralEvtInfo (operation GetFaultDetectionCounter), which triggers the respective runnable (refer to RunnableEntity GetFaultDetectionCounter, chapter 8.3.3.19).

[SWS_Dem_00287] ⌈ With each call to the function Dem_DcmGetNextFilteredDTCAndSeverity (refer to paragraph 8.3.4.1.10) the Dem module shall return the next DTC and its associated fault severity matching the filter criteria defined by the function call Dem_DcmSetDTCFilter (refer to [SWS_Dem_00057]). ⌋

The Dcm calls the function Dem_DcmGetNextFilteredDTCAndSeverity continuously until the return value of the function is DEM_FILTERED_NO_MATCHING_ELEMENT, to receive all DTCs matching the filter criteria.

For information about the functions Dem_DcmGetSeverityOfDTC and Dem_DcmGetFunctionalUnitOfDTC, refer to chapter 7.2.

[SWS_Dem_00595] ⌈ The function Dem_DcmSetFreezeFrameRecordFilter (refer to chapter 8.3.4.1.11) shall set the static filter criteria attribute “all freeze frame records currently stored in the event memory” to be used for the subsequent calls of Dem_DcmGetNextFilteredRecord. The filter criteria attributes shall be used until the next call of Dem_DcmSetFreezeFrameRecordFilter or Dem initialization. ⌋

[SWS_Dem_00650] ⌈ Each call of Dem_DcmSetFreezeFrameRecordFilter shall lead to a reset of the sequence. ⌋*(SRS_Diag_04079)*

[SWS_Dem_00210] ⌈ The function Dem_DcmSetFreezeFrameRecordFilter shall retrieve the number of filtered freeze frame records. This filter always belongs to primary memory. ⌋*(SRS_Diag_04010)*

[SWS_Dem_00225] ⌈ With each call to the function Dem_DcmGetNextFilteredRecord (refer to chapter 8.3.4.1.12) the Dem module shall return the next DTC and its associated relative addressed freeze frame record numbers matching the filter criteria defined by the function call Dem_DcmSetFreezeFrameRecordFilter. ⌋*(SRS_Diag_04074)*

The Dcm calls the function Dem_DcmGetNextFilteredRecord continuously until the return value of the function is DEM_FILTERED_NO_MATCHING_ELEMENT, to receive all records matching the filter criteria.

[SWS_Dem_00219] ⌈ The function Dem_DcmGetDTCByOccurrenceTime (refer to chapter 8.3.4.1.13) shall provide the capability to get one DTC stored in the primary event memory according to the API parameter DTCRequest, which specifies the relevant occurrence time. ⌋*(SRS_Diag_04070)*

[SWS_Dem_00221] ⌈ The function Dem_DcmGetDTCByOccurrenceTime shall return DEM_OCCURR_NOT_AVAILABLE, if no DTC is matching the requested occurrence time (DTCRequest). ⌋*(SRS_Diag_04070)*

7.9.2.2 Access event related data

This section defines the APIs, to get access to the event related data (refer to chapter 7.3.7.1 and chapter 7.3.7.3) stored with the DTCs in the event memory of the Dem via diagnostics. Furthermore, access to OBDrelevant PIDs stored in a freeze frame is made available. Details concerning freeze frame handling can be found in ISO 14229-1 and ISO 15031-5.

[SWS_Dem_00686] ⌈ The mapping of the PID value to the corresponding DID value shall be done following ISO 142291 by preceding the value 0xF4 to the original PID value specified in ISO 150315. ⌋*(SRS_Diag_04000)*

Example for the mapping of a PID to a DID: PID $04 –> DID 0xF404

[SWS_Dem_00070] ⌈ The function Dem_DcmGetOBDFreezeFrameData (refer to chapter 8.3.4.4.11) shall return the DTC associated with the most important OBD freeze frame (which caused MIL on). ⌋*(SRS_Diag_04074)*

[SWS_Dem_00575] ⌈ The API Dem_DcmGetOBDFreezeFrameData shall provide the OBD equivalent freeze frame data record as specified in ISO 150315 (service $02 frame number 0x00), but each data preceded by the corresponding DID value instead of its PID value, with the format shown in Figure 50. ⌋*(SRS_Diag_04010, SRS_Diag_04082)*

Note: The API Dem_DcmGetOBDFreezeFrameData is only available, if the Dem module supports OBD (refer to DemOBDSupport).

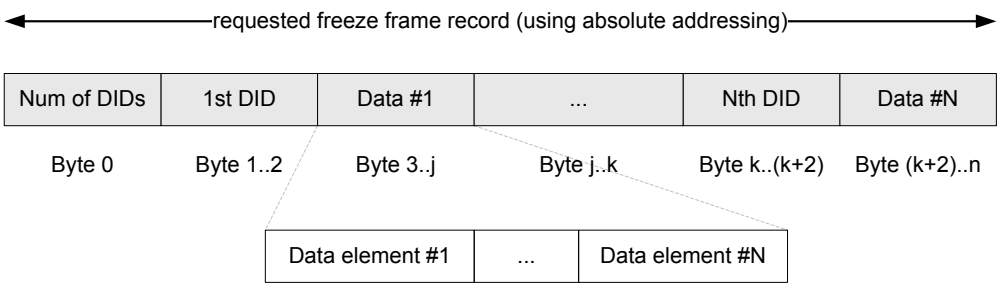

Figure 7.49: Buffer format used by Dem_DcmGetOBDFreezeFrameData

Note: The data returned include the DTCSnapshotRecordNumberOfIdentifiers (Num of DIDs) as defined by ISO 142291 for the response message to service 0x19 0x05.

[SWS_Dem_00071] ⌈ The function Dem_DcmGetFreezeFrameDataByDTC (refer to paragraph 8.3.4.2.3) shall copy the specific data identifiers (DIDs) of the requested freeze frame record to the destination buffer (DestBuffer). The function shall transmit these data as a complete record with the format shown in Figure 7.50. ⌋*(SRS_Diag_04074)*

[SWS_Dem_00576] ⌈ If the API parameter RecordNumber of Dem_DcmGetFreezeFrameDataByDTC is set to 0x00, the Dem module shall provide the event/DTCspecific OBD equivalent freeze frame record as specified in ISO 150315 (service $02 frame number 0x00), but each data preceded by the corresponding DID value instead of its PID value. ⌋*(SRS_Diag_04074)*

Note: With service 0x19 0x04 <DTC> 0x00 the OBD relevant freeze frame (if existing) of each single DTC can be retrieved whereas with service $19 $05 $00 only the most important OBD freeze frame can be retrieved (the one which will also be reported when subsequently using service $02 <PID> $00).

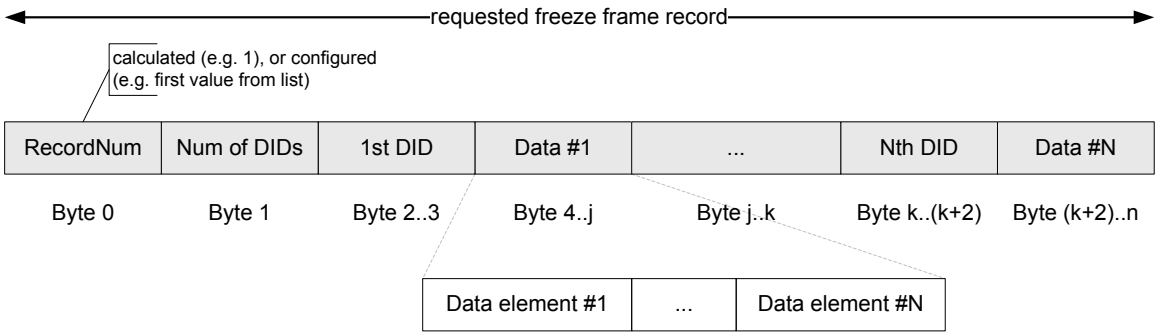

Figure 7.50: Buffer format used by Dem_DcmGetFreezeFrameDataByDTC

Note: The data returned include the DTCSnapshotRecordNumber (RecordNum) and DTCSnapshotRecordNumberOfIdentifiers (Num of DIDs) as defined by ISO 142291 for the response message to service 0x19 0x04. In contradiction to the Dcm PIDstructure handling, the DIDstructure is handled within the Dem module. Therefore, the Dem returns already full freeze frame records according to the response layout. This results in an optimized Dem/Dcm interface.

[SWS_Dem_00630] ⌈ If Dem_DcmGetFreezeFrameDataByDTC is called with a valid DTC and a valid freeze frame record number which is not stored, the Dem shall return DEM_GET_FFDATABYDTC_OK and BufSize 0 (empty buffer). ⌋*(SRS_Diag_04074)*

[SWS_Dem_00074] ⌈ The function Dem_DcmGetSizeOfFreezeFrameByDTC (refer to paragraph 8.3.4.2.4) shall return the size of the requested freeze frame record(s), which represents the number of user data bytes, including any freeze frame headerinformation (according to the format defined in [SWS_Dem_00071]). ⌋*(SRS_Diag_04074, SRS_Diag_04079)*

Note: If the record number value 0xFF is requested, the Dem considers the size of all stored freeze frame records according to [SWS_Dem_00074].

[SWS_Dem_00075] ⌈ The function Dem_DcmGetExtendedDataRecordByDTC (refer to chapter paragraph 8.3.4.2.5) shall copy the complete data of the requested extended data record to the destination buffer (DestBuffer). The function shall transmit these data as a complete record with the format shown in Figure 7.51. ⌋*(SRS_Diag_04074)*

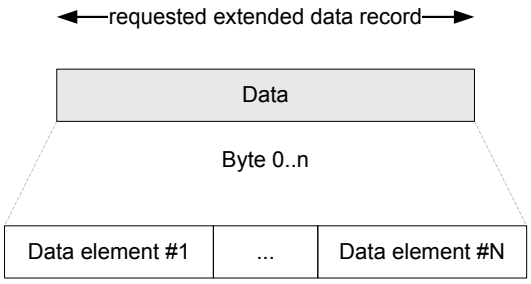

Figure 7.51: Buffer format used by Dem_DcmGetExtendedDataRecordByDTC

[SWS_Dem_00631] ⌈ If Dem_DcmGetExtendedDataRecordByDTC is called with a valid DTC and a valid extended data record number which is not stored, the Dem shall return DEM_RECORD_OK and BufSize 0 (empty buffer). ⌋*(SRS_Diag_04074)*

[SWS_Dem_00076] ⌈ The function Dem_DcmGetSizeOfExtendedDataRecordByDTC (refer to paragraph 8.3.4.2.6) shall return the size of the requested extended data record(s), which represents the number of user data bytes stored in the extended data record, including any extended data record headerinformation (i.e. the extended data record number, different to the format defined in [SWS_Dem_00075]). ⌋*(SRS_Diag_04074)*

Note: If the record number value 0xFE is requested, the Dem considers the size of all OBD stored extended data records in the range of 0x90 to 0xEF according to [SWS_Dem_00076].

Note: If the record number value 0xFF is requested, the Dem considers the size of all stored extended data records (in the range of 0x01 to 0xEF) according to [SWS_Dem_00076].

The Dcm uses the function Dem_DcmDisableDTCRecordUpdate, if the freeze frame or extended data of a specific DTC is about to be accessed by subsequent APIcalls. This is done to ensure, that the requested data of this DTC are not changed while the Dcm accesses those by several APIcalls.

[SWS_Dem_00270] ⌈ The function Dem_DcmDisableDTCRecordUpdate (refer to chapter paragraph 8.3.4.2.1) shall protect the event related data of the specified DTC within the specified origin from updating or deleting, to allow a consistent read for the following subsequent APIcalls:

- Dem_DcmGetSizeOfFreezeFrameByDTC and Dem_DcmGetFreezeFrameDataByDTC

- Dem_DcmGetSizeOfExtendedDataRecordByDTC and Dem_DcmGetExtendedDataRecordByDTC.

⌋

New and other events including their associated freeze frames and extended data records can still be added to and changed in the event memory as long as space is available.

Note: Event related data might still be updated in background (e.g. Deminternal data elements).

Note: The function Dem_DcmDisableDTCRecordUpdate does not affect the DTC status information update.

[SWS_Dem_00271] ⌈ The function Dem_DcmEnableDTCRecordUpdate (refer to paragraph 8.3.4.2.2) shall release the currently disabled DTC which has been protected by the function Dem_DcmDisableDTCRecordUpdate, so that the data can be updated again. ⌋*(SRS_Diag_04074)*

The function Dem_DcmEnableDTCRecordUpdate is the counterpart to the function Dem_DcmDisableDTCRecordUpdate.

The Dcm calls the function Dem_DcmEnableDTCRecordUpdate after the freeze frame and extended data of the specific DTC were protected by the function Dem_DcmDisableDTCRecordUpdate, after the access by subsequent APIcalls is finished.

[SWS_Dem_00648] ⌈ If development error detection is enabled and the function Dem_DcmDisableDTCRecordUpdate is called while another DTC is locked, the Dem module shall set the error code DEM_E_WRONG_CONDITION (refer also to [SWS_Dem_00370]). ⌋

7.9.2.3 Clear diagnostic information

For definition of locking the clearing process, refer to chapter 7.3.2.2.

[SWS_Dem_00009] ⌈ The Dem module shall provide the API Dem_DcmClearDTC (refer to chapter 8.3.4.3.1) to the Dcm for deleting single DTCs, DTC groups, and all DTCs from the event memory. This shall trigger also initializing of related SWCs / BSW modules according to [SWS_Dem_00003]. ⌋

The service ClearDiagnosticInformation (0x14) of ISO 142291 [2] defines and covers the required actions and the deletion of related memory areas like freeze frames. One DTC value is transmitted, which can represent either a single DTC, or a group of DTCs. The groups are defined in ISO 14229-1, Annex D.1.

Note: Dem_DcmClearDTC typically triggers further callbacks through the RTE. To indicate the respective calltree for these runnables, a workaround is used: The Dcm

triggers the DTC deletion using the Dem interface DcmIf (operation DcmClearDTC, refer to chapter chapter 8.6.2.14) instead of a direct C call.

[SWS_Dem_00077] ⌈ The function Dem_DcmClearDTC shall clear the status of all event(s) related to the specified DTC(s), as well as all associated event memory entries for these event(s). ⌋*(SRS_Diag_04065)*

Note: Exceptions may apply due to [SWS_Dem_00514].

Note: This ensures that Dem_DcmClearDTC can react to calls from different services in a safe way (e.g. preemptive protocol changes from ISO 14229 service 0x14 to ISO 15031 service $04).

[SWS_Dem_01042] ⌈ The Dem module shall release the locked clearing process as soon as the internal processing is finished and the Dem is able to process the next clear request. ⌋*(SRS_Diag_04130)*

Note: The caller might not receive a final E_OK from call of Dem_<...>ClearDTC API in case it was aborted by another client. In this case E_BUSY is finally returned and can be used as end criterion.

7.9.2.4 Control DTC setting

[SWS_Dem_00035] ⌈ The Dem module shall be capable of disabling (refer to Dem_DcmDisableDTCSetting) and enabling (refer to Dem_DcmEnableDTCSetting) the DTC setting of a specific DTC group. ⌋

Note: Dem_DcmEnableDTCSetting typically triggers further InitMonitorForEvent callbacks through the RTE. To indicate the respective calltree for these runnables, a workaround is used: The Dcm triggers the enabling of DTC setting using the Dem interface DcmIf (operation DcmEnableDTCSetting, refer to chapter 8.3.4.6) instead of a direct C call.

[SWS_Dem_00626] ⌈ When DTC setting is disabled, all status reports from SW-Cs (refer to Dem_SetEventStatus) and BSW modules (refer to Dem_ReportErrorStatus) for those events being assigned to this specific DTC group shall be ignored (no change of UDS DTC status byte) by the Dem. ⌋*(SRS_Diag_04115)*

Note: This is similar like the enable condition handling (refer to [SWS_Dem_00449]). It has no impact on Dem_ResetEventDebounceStatus, Dem_ResetEventStatus and Dem_<...>ClearDTC. In case of Dem-internal debouncing the related fault detection counter will be frozen or reset (refer to chapter Figure 7.28 and Figure 7.31).

[SWS_Dem_00079] ⌈ The function Dem_DcmDisableDTCSetting shall disable the storage of all events assigned to a specific DTC group in the event memory including the respective UDS DTC status byte updates. ⌋*(SRS_Diag_04010)*

The function Dem_DcmDisableDTCSetting is used in case of an induced failure situation in a system, e.g. during flashreprogramming of one ECU in a network. In that case

all the ECUs are commanded via diagnostic request (forwarded from Dcm by using Dem_DcmDisableDTCSetting / Dem_DcmEnableDTCSetting) to ignore DTC reports, as the flashed ECU is not participating in the normal communication anymore. Note: If one of the other networked ECUs needs one of the signals, which are now missing, for this case a failsafereaction of the ECU cannot be assigned to the UDS DTC status byte updates, as these are also suppressed during disabled DTC setting.

[SWS_Dem_00080] ⌈ The function Dem_DcmEnableDTCSetting shall enable the storage of all events assigned to specific a DTC group in the event memory including the respective UDS DTC status byte updates. ⌋*(SRS_Diag_04000, SRS_Diag_04115)*

Note : If a DTC belongs to multiple DTC groups [e.g. EMISSION and NETWORK], the DTC has to be consider when processing any of the group.

7.9.2.5 Asynchronous Dcm operations

Most of the APIs of the Dem/Dcm interface may depend on NVRAM data. Therefore, an asynchronous processing of these APIs can be realized by using the additional "PENDING" value of the respective API return types (refer to chapter 8.2.1.20).

7.9.3 Interaction with J1939 Diagnostic Manager (J1939)

[SWS_Dem_00971] ⌈ In case the TestFailed bit in the event status changes and a J1939 DTC number is assigned to that event parameter, the function J1939Dcm_DemTriggerOnDTCStatus shall be called. ⌋

7.9.4 Interaction with Function Inhibition Manager (FIM)

The purpose of the Function Inhibition Manager is to control (enable/disable) function entities within software components based on inhibit conditions such as detected errors. The Dem contribution to this functionality is to provide event status information to the FiM.

The FiM uses the information of dependencies provided by the SWCs.

[SWS_Dem_00029] ⌈ If DemTriggerFimReports (refer to DemGeneral) is enabled, the Dem module shall notify the FiM module (refer to [5]) on each event status change of all events (refer also to [SWS_Dem_00016]), by calling the function FiM_DemTriggerOnEventStatus. ⌋*(SRS_Diag_04031)*

Note: The function FiM_DemTriggerOnEventStatus is equal to <Module>_DemTriggerOnEventStatus (refer to chapter 8.4.3.2). This configuration parameter is for simplicity to avoid a reference of the function to each individual EventId.

The Dem module provides the function Dem_GetEventStatus for possible plausibility checks of the FiM, rebuilding, startup, etc. of inhibition relations by the FiM.

[SWS_Dem_00658] ⌈ If DemTriggerFimReports (refer to DemGeneral) is enabled, the Dem shall call FiM_DemInit (Dem implementation-specific) to re-initialize the FiM module in case the Dem detects a status change of a certain number of events, e.g. clearance of the Dem event memory (on request of ClearDiagnosticInformation). ⌋*(SRS_Diag_04031)*

7.9.5 Interaction with NVRAM Manager (NvM)

Typically, the Dem module uses nonvolatile memory blocks (configurable in size by the NVRAM Manager [8]) to achieve permanent storage of event status information, event related data and required internal states (e.g. retrieve status at startup). Each nonvolatile memory block used from the Dem, needs also to be configured (refer to DemNvRamBlockId). The number, the type, and the content of the used nonvolatile memory blocks are not prescribed. These shall be handled implementationspecific. The NvM usage can also be deactivated by configuration (refer to multiplicity of DemNvRamBlockId), so that the Dem will work based on RAM only.

[SWS_Dem_00339] ⌈ The Dem module shall verify the validity (which relates to block states), integrity (which relates to CRC results), and for general NvM-reading errors of its nonvolatile blocks (before using the respective data). ⌋*(SRS_Diag_04107)*

Usually this verification is done in the API Dem_Init (refer to chapter 8.3.2.2) by using NvM_GetErrorStatus for these blocks, which are read by the ECU State Manager (refer to API NvM_ReadAll).

Note: For the non-volatile data of the Dem module, it is recommended to configure a CRC in the NvM.

[SWS_Dem_00578] ⌈ If the NVM module was not able to read some nonvolatile data of the Dem module, the Dem module shall initialize all non-volatile data with their initial values. ⌋

Note: To avoid inconsistencies between readable blocks and erroneous blocks, all nonvolatile data are initialized. The initialization is done to allow the fault detection mechanism of the NvM module, to report the respective reading error(s) to the Dem module (refer to Dem_ReportErrorStatus). These errors denote the defective NVRAM.

[SWS_Dem_00340] ⌈ After the API Dem_Init has finished, the Dem shall be fully operational. ⌋*(SRS_BSW_00101)*

[SWS_Dem_00550] ⌈ The Dem module shall provide the configuration parameter DemImmediateNvStorage per DTC (refer to DemDTCAttributes) to trigger the storage of the respective event memory entry including its event related data in nonvolatile memory (refer to NvM_WriteBlock) immediately. ⌋*(SRS_BSW_00101)*

[SWS_Dem_00551] ⌈ If immediate nonvolatile storage is enabled for a specific DTC, the Dem module shall trigger the storage for new event memory entries and after every change of the event related data (event memory entry was updated). ⌋

Note: For event memory entries, which are stored immediately, it is necessary to ensure data consistency (e.g. with the event status byte) during Dem_Init.

Note: If the immediate nonvolatile storage is disabled, the event memory entry and its event related data are stored persistently during the shutdown phase (refer to SWS_Dem_00102, SWS_Dem_00341 and the note below).

[SWS_Dem_00552] ⌈ If immediate nonvolatile storage is enabled for a specific DTC, the Dem module shall not trigger further immediate write operations to NVRAM for this DTC, if its occurrence counter has reached the threshold defined by the configuration parameter DemImmediateNvStorageLimit (refer to DemGeneral). ⌋

Note: Write operations to NVRAM will perform in any case at ECU shutdown.

[SWS_Dem_00102] ⌈ The API Dem_Shutdown shall finalize all pending operations in the Dem module to prepare the internal states and data for transfer to the NVRAM. The event memory shall be locked and not modified until the API Dem_Init is recalled. ⌋*(SRS_BSW_00336)*

[SWS_Dem_00341] ⌈ For changed nonvolatile data, the Dem module shall trigger the storage to NVRAM before or during Dem_Shutdown. ⌋*(SRS_BSW_00336)*

Based on the Dem configuration and implementation, the copying process to NVRAM of those Demrelated NvMblocks to be stored after Dem_Shutdown, is performed by the API NvM_WriteAll called by the ECU State Manager.

If the ECU power supply is disconnected before the NvM has finished copying all data to NVRAM, these data will be incomplete/inconsistent or not stored. At next startup, the events of the last operating cycle could not be found anymore. Therefore, the NVRAM Manager configuration provides mechanisms for data consistency, like redundant data blocks.

[SWS_Dem_00164] ⌈ The Dem module shall use the APIs NvM_WriteBlock and NvM_ReadBlock of the NVRAM Manager, if there is the necessity to store and restore data between Dem_Init and Dem_Shutdown. ⌋*(SRS_Diag_04077)*

Note: The NvM module realizes a retry mechanism for block reading and writing. Therefore, the Dem module does not implement any retry mechanism for its nonvolatile blocks.

[SWS_Dem_00579] ⌈ If the NVM module was not able to write (some) nonvolatile data of the Dem module, the Dem module shall ignore the reported negative return values by the NvM. ⌋*(SRS_BSW_00171)*

Note: If writing of nonvolatile Dem data fails, the Dem module is not able to perform any adequate reaction.

7.9.6 Interaction with Development Error Tracer (Det)

The interaction with the Det is described in chapter 7.11.

7.9.7 Interaction with Diagnostic Log & Trace (Dlt)

[SWS_Dem_00517] ⌈ If DemTriggerDltReports (refer to DemGeneral is enabled, the Dem module shall notify the Dlt module (refer to [17]) on each event status change of all events (refer also to SWS_Dem_00016), by calling the function Dlt_DemTriggerOnEventStatus. ⌋*(SRS_Diag_04099)*

Note: The function Dlt_DemTriggerOnEventStatus is equal to <Module>_DemTriggerOnEventStatus (refer to chapter 8.4.3.1.3). This configuration parameter is for simplicity to avoid a reference of the function to each individual EventId.

After the Dlt module has been notified by Dlt_DemTriggerOnEventStatus, it will make use of the functions Dem_DltGetMostRecentFreezeFrameRecordData (refer to chapter 8.3.7.1) and Dem_DltGetAllExtendedDataRecords (refer to chapter 8.3.7.2) to collect the current event related data.

[SWS_Dem_00632] ⌈ If DemTriggerDltReports is enabled, the Dem module shall provide access on the most recent freeze frame record data (refer to chapter 8.3.7.1) to the Dlt module. ⌋*(SRS_Diag_04099)*

[SWS_Dem_00633] ⌈ The function Dem_DltGetMostRecentFreezeFrameRecordData shall report the data of the most recent freeze frame record of the requested diagnostic event. ⌋*(SRS_Diag_04099)*

[SWS_Dem_00992] ⌈ The format of the data in the destination buffer (DestBuffer) of the function Dem_DltGetMostRecentFreezeFrameRecordData is raw hexadecimal values and contains no headerinformation like RecordNumber or DataId. The size of the buffer equals to the configuration settings of all respective data elements. ⌋*(SRS_Diag_04099)*

[SWS_Dem_00634] ⌈ If DemTriggerDltReports is enabled, the Dem module shall provide access on all extended data records (refer to chapter 8.3.7.2) to the Dlt module. ⌋*(SRS_Diag_04099)*

[SWS_Dem_00635] ⌈ The function Dem_DltGetAllExtendedDataRecords shall report the data of all extended data records of the requested diagnostic event. ⌋*(SRS_Diag_04099)*

[SWS_Dem_00993] ⌈ The format of the data in the destination buffer (DestBuffer) of the function Dem_DltGetAllExtendedDataRecords is raw hexadecimal values and contains no headerinformation like RecordNumber. The size of the buffer equals to the configuration settings of all respective data elements. ⌋*(SRS_Diag_04099)*

7.9.8 Required data by the Dem module

The Dem module requires different information for internal computation/processing. If this information is provided by SWCs (as provide-ports), they are described in the respective ServiceNeeds (diagnostic capabilities).

One kind of information required by the Dem module are event related data (represented by freeze frames and extended data records, refer to chapter 7.3.7).

Note: For OBDrelevant ECUs, the Dem module could access (via the data element interface) on the following data values:

- engine temperature engine speed
- vehicle speed (refer to SWS_Dem_00346, PID $0D)
- distance information
- programming event
- ambient temperature
- ambient pressure
- accelerator pedal information

However, the list of data elements and their required size and resolution are implementationspecific and will be configured in OBD and handled during integration process. For example, the vehicle speed, accelerator pedal information, ambient temperature, etc. are necessary to evaluate the cycle conditions of the IUMPR General Denominator. Similar inputs are necessary for the PID computation (e.g. the engine temperature for the computation of the WarmUp cycle or the WarmUp cycle condition itself). These variables are typically accessed through the RTE.

7.10 Version check

For details refer to the chapter 5.1.8 “Version Check” in SWS_BSWGeneral.

7.11 Error classification

[SWS_Dem_00173] ⌈ The following errors shall be detectable by the Dem module depending on its configuration (development / production mode):

Type or error	*Relevance*	*Related error code*	*Value [hex]*
API function called with a parameter value, which is not allowed by active configuration	Development	DEM_E_PARAM_CONFIG	0x10

API function called with a NULL pointer	Development	DEM_E_PARAM_POINTER	0x11
API function called with wrong parameter value	Development	DEM_E_PARAM_DATA	0x12
API function called with wrong length parameter value	Development	DEM_E_PARAM_LENGTH	0x13
API function called before the Dem module has been full initialized (refer to [SWS_Dem_00124], [SWS_Dem_00364]) or after the Dem module has been shut down (refer to [SWS_Dem_00368])	Development	DEM_E_UNINIT	0x20
No valid data available by the SW-C	Development	DEM_E_NODATAAVAILABLE	0x30
Required conditions for the respective API call are not fulfilled (e.g. an invalid status change was initiated, or a filter was not set correctly, etc. - refer to [SWS_Dem_00518]).	Development	DEM_E_WRONG_CONDITION	0x40
If any Operation Cycle is running and is configured as volatile (configuration parameter “Dem-OperationCycleStatusStorage” is set to “FALSE”) when Dem_Shutdown is called (refer to [SWS_Dem_00988]).	Development	DEM_E_OPERATION_CYCLE_STARTED	0x41
The requested record number is not supported by the event.	Development	DEM_E_WRONG_RECORDNUMBER	0x31
The requested DID is not supported by the freeze frame	Development	DEM_E_WRONG_DIDNUMBER	0x32
API function called with a parameter which is not supported by Configuration.	Development	DEM_E_WRONG_CONFIGURATION	0x41

Table 7.5: Types of errors which can be detected by the Dem module

⌋

[SWS_Dem_00124] ⌈ If development error detection is enabled and any instance calls any Dem API, excluding Dem_ReportErrorStatus, Dem_ResetEventDebounceStatus and Dem_GetVersionInfo, before the Dem was fully initialized, the Dem module shall set the error code DEM_E_UNINIT. ⌋*(SRS_BSW_00406)*

Note: If development error detection is disabled and the Dem is not fully initialized, the behavior of the APIs is undefined.

[SWS_Dem_00364] ⌈ If development error detection is enabled and any instance calls Dem_ReportErrorStatus or Dem_ResetEventDebounceStatus before the Dem was preinitialized, the Dem module shall set the error code DEM_E_UNINIT. ⌋*(SRS_BSW_00406)*

Note: If development error detection is disabled and the Dem is not preinitialized, the behavior of these APIs is undefined.

[SWS_Dem_00368] ⌈ If development error detection is enabled and any instance calls any Dem API, excluding Dem_ReportErrorStatus, Dem_ResetEventDebounceStatus and Dem_GetVersionInfo, after Dem_Shutdown (refer to chapter 8.3.2.3) has been called, the Dem module shall set the error code DEM_E_UNINIT until Dem_Init is called again. ⌋

Note: If development error detection is disabled and the Dem is shut down, the behavior of these APIs is undefined.

[SWS_Dem_00518] ⌈ If development error detection is enabled and a Dem function is called with required preconditions not fulfilled, the Dem module shall set the error code DEM_E_WRONG_CONDITION. ⌋*(SRS_Diag_04057)*

Note: For example, Dem_DcmGetNextFilteredDTCAndFDC is called, after Dem_DcmSetDTCFilter with FilterForFaultDetectionCounter FALSE was called.

[SWS_Dem_00370] ⌈ If development error detection is enabled and a Dem function detects a development error, excluding DEM_E_NODATAAVAILABLE, then the Dem function shall return immediately with E_NOT_OK (in case of Std_ReturnType) or an appropriate negative return value, after the development error was reported. ⌋

7.12 Error detection

For details refer to the chapter 7.2.3.2 “Configuration of Development Errors” in SWS_BSWGeneral.

7.13 Error notification

For details refer to the chapter 7.2.3.3 “Reporting Development Errors” in SWS_BSWGeneral.

7.14 Support for Debugging

For details refer to the chapter 7.1.17 “Debugging support” in SWS_BSWGeneral.

8 API specification

The figures below show the interfaces between Dem and its surrounding SW-Cs and BSW modules. The description of the interface shall give a simple overview of these relations.

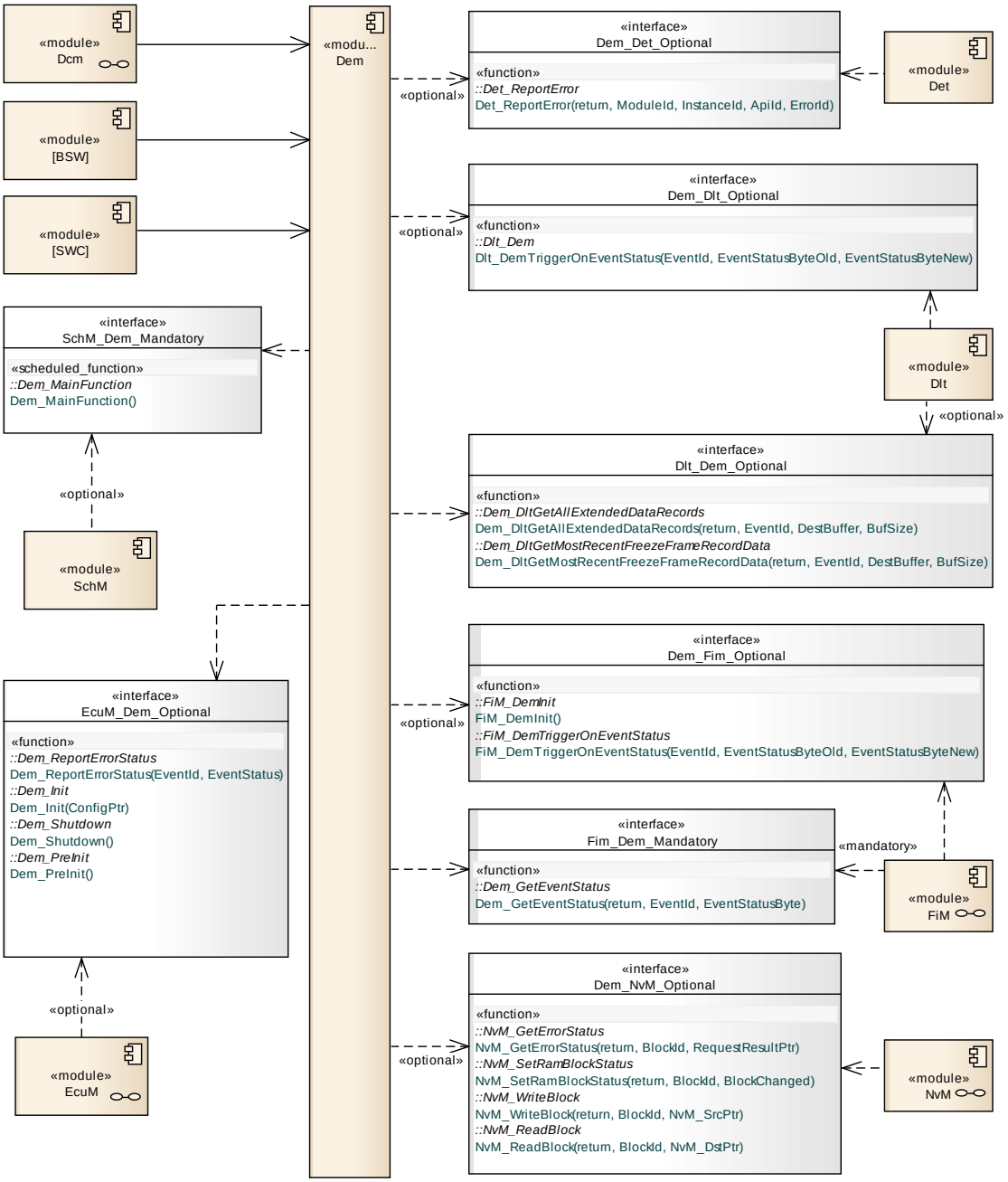

Figure 8.1: Overview of interfaces between the Dem and other BSW modules

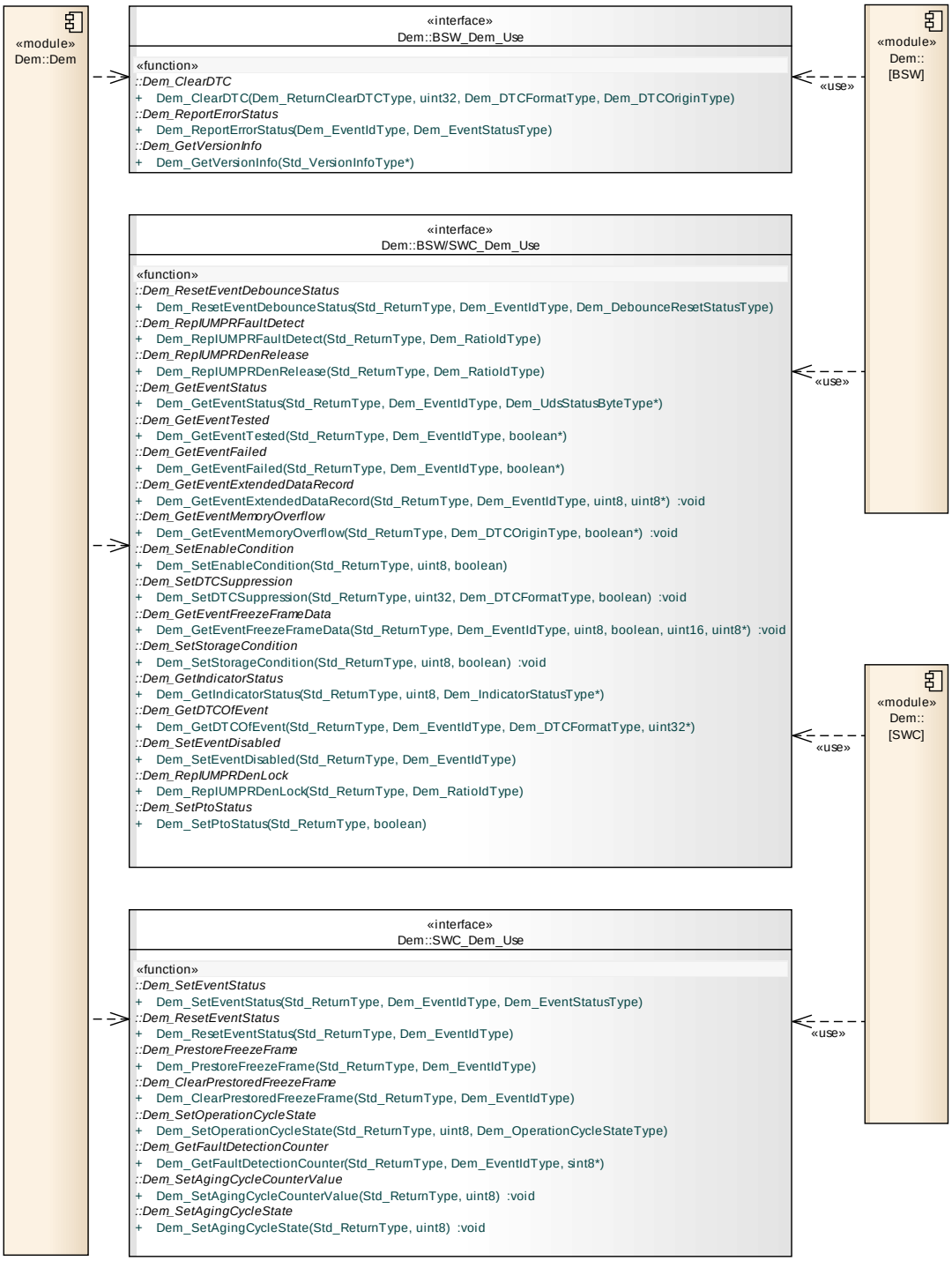

Figure 8.2: Overview of interfaces between the Dem and other BSW modules (in general)

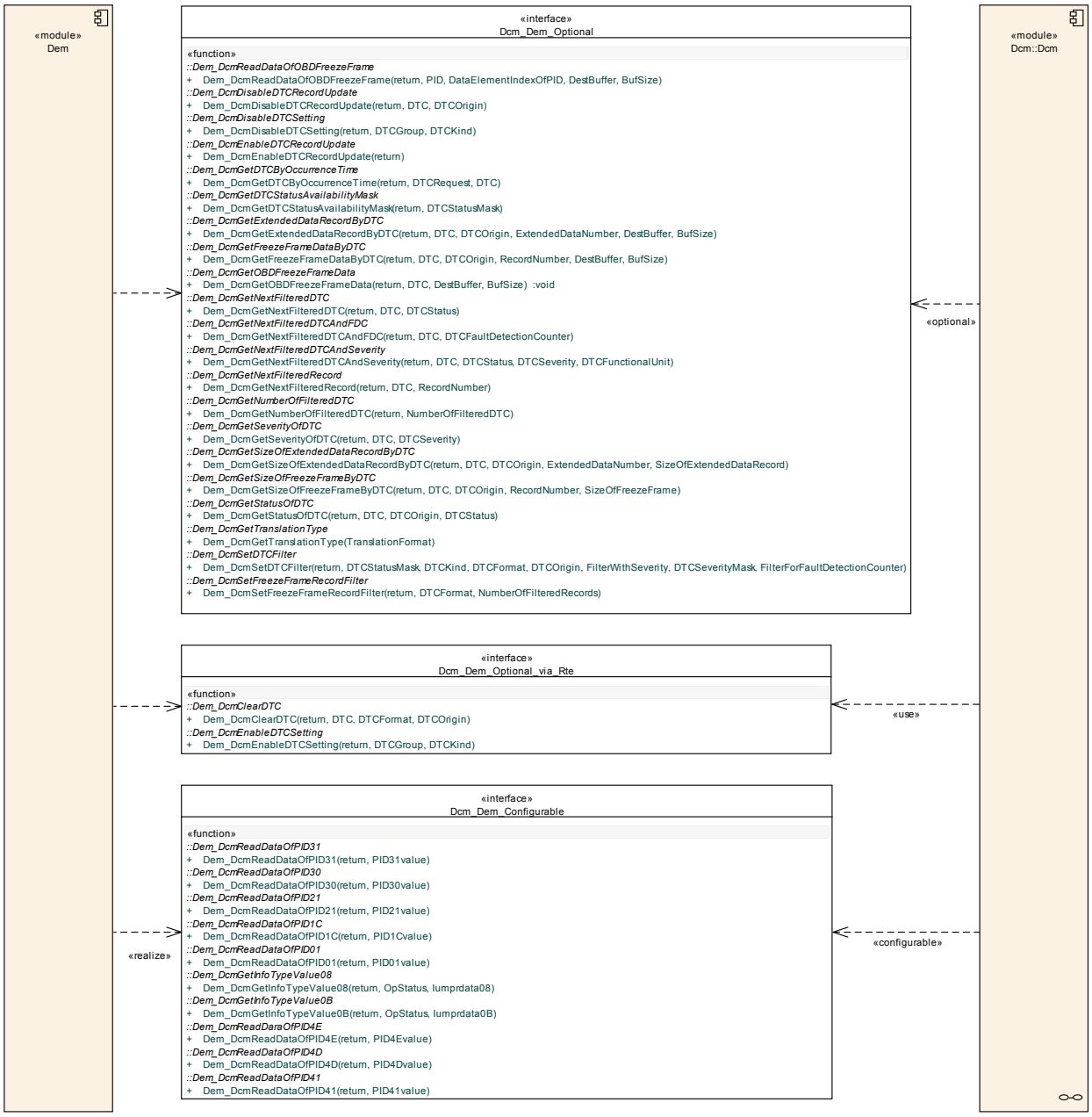

Figure 8.3: Overview of interfaces between the Dem and Dcm

8.1 Imported Types

In this chapter all types included from the following files are listed:

[SWS_Dem_00176] ⌈

Module	*Imported Type*
Dcm	Dcm_OpStatusType
NvM	NvM_BlockIdType NvM_RequestResultType
Std_Types	Std_ReturnType Std_VersionInfoType

Table 8.1: Dem_ImportedTypes

⌋

8.2 Type definitions

The following Data Types shall be used for the functions defined in this specification.

8.2.1 Dem data types

8.2.1.1 Dem_ConfigType

[SWS_Dem_00924] ⌈

Name	**Dem_ConfigType**		
Type	Structure		
Element:	void	implementation specific	–
Description	This type of the external data structure shall contain the post build initialization data for the Dem.		

Table 8.2: Dem_ConfigType

⌋

8.2.1.2 Dem_EventIdType

[SWS_Dem_00925] ⌈

Name:	**Dem_EventIdType**		
Type:	uint16		
Range:	1..65535	–	Internal identifier of a diagnostic event Remark: 0 is not a valid value
Description:	Identification of an event by assigned EventId. The EventId is assigned by the Dem. Example: 1 refers to monitor x, 2 refers to monitor y, etc.		

Table 8.3: Dem_EventIdType

⌋

8.2.1.3 Dem_EventStatusType

[SWS_Dem_00926] ⌈

Name:	**Dem_EventStatusType**		
Type:	uint8		
Range:	DEM_EVENT_STATUS_PASSED	0x00	Monitor reports qualified test result passed.
	DEM_EVENT_STATUS_FAILED	0x01	Monitor reports qualified test result failed.
	DEM_EVENT_STATUS_PREPASSED	0x02	Monitor reports non-qualified test result pre-passed (debounced
	DEM_EVENT_STATUS_PREFAILED	0x03	Monitor reports non-qualified test result pre-failed (debounced Dem-internally).
		0x04 - 0xFF	reserved
Description:	This type contains all monitor test result values, which can be reported via Dem_ReportErrorStatus() and Dem_SetEventStatus().		

Table 8.4: Dem_EventStatusType

⌋

8.2.1.4 Dem_DebouncingStateType

[SWS_Dem_01000] ⌈

Name:	**Dem_DebouncingStateType**		
Type:	uint8		
Range:	DEM_TEMPORARILY_DEFECTIVE	0x01	Bit 0: Temporarily Defective (corresponds to 0 < FDC < 127)
	DEM_FINALLY_DEFECTIVE	0x02	Bit 1: finally Defective (corresponds to FDC = 127)
	DEM_TEMPORARILY_HEALED	0x04	Bit 2: temporarily healed (corresponds to -128 < FDC < 0)
	DEM_TEST_COMPLETE	0x08	Bit 3: Test complete (corresponds to FDC = -128 or FDC = 127)
	DEM_DTR_UPDATE	0x10	Bit 4: DTR Update (= Test complete && Debouncing complete && enable conditions / storage conditions fulfilled)
Description:	–		

Table 8.5: Dem_DebouncingStateType

⌋

8.2.1.5 Dem_DebounceResetStatusType

[SWS_Dem_00927] ⌈

Name:	**Dem_DebounceResetStatusType**		
Type:	uint8		
Range:	DEM_DEBOUNCE_STATUS_FREEZE	0x00	Freeze the internal debounce counter/timer.
	DEM_DEBOUNCE_STATUS_RESET	0x01	Reset the internal debounce counter/timer.
		0x02 - 0xFF	reserved
Description:	This type contains all definitions to control an internal debounce counter/timer via the function Dem_ResetEventDebounceStatus().		

Table 8.6: Dem_DebounceResetStatusType

⌋

8.2.1.6 Dem_UdsStatusByteType

[SWS_Dem_00928] ⌈

Name:	**Dem_UdsStatusByteType**		
Type:	uint8		
Range:	range	0x00..0xFF	the possible range from 0x00 to 0xFF allows for all combinations of the individual status bits
	DEM_UDS_STATUS_TF	0x01	bit 0: TestFailed
	DEM_UDS_STATUS_TFTOC	0x02	bit 1: TestFailedThisOperationCycle
	DEM_UDS_STATUS_PDTC	0x04	bit 2: PendingDTC
	DEM_UDS_STATUS_CDTC	0x08	bit 3: ConfirmedDTC
	DEM_UDS_STATUS_TNCSLC	0x10	bit 4: TestNotCompletedSinceLastClear
	DEM_UDS_STATUS_TFSLC	0x20	bit 5: TestFailedSinceLastClear
	DEM_UDS_STATUS_TNCTOC	0x40	bit 6: TestNotCompletedThisOperationCycle
	DEM_UDS_STATUS_WIR	0x80	bit 7: WarningIndicatorRequested
Description:	In this data-type each bit has an individual meaning. The bit is set to 1 when the condition holds. For example, if the 2nd bit (0x02) is set to 1, this means that the test failed this operation cycle. If the bit is set to 0, it has not yet failed this cycle.		

Table 8.7: Dem_UdsStatusByteType

⌋

8.2.1.7 Dem_OperationCycleStateType

[SWS_Dem_00929] ⌈

Name:	**Dem_OperationCycleStateType**		
Type:	uint8		
Range:	DEM_CYCLE_STATE_START	0x00	Start/restart the operation cycle
	DEM_CYCLE_STATE_END	0x01	End the operation cycle
Description:	This type contains operation cycle state values, which can be reported via Dem_SetOperationCycleState()/ Dem_GetOperationCycleState().		

Table 8.8: Dem_OperationCycleStateType

⌋

8.2.1.8 Dem_IndicatorStatusType

[SWS_Dem_00930] ⌈

Name:	**Dem_IndicatorStatusType**		
Type:	uint8		
Range:	DEM_INDICATOR_OFF	0x00	Indicator off mode
	DEM_INDICATOR_CONTINUOUS	0x01	Indicator continuously on mode
	DEM_INDICATOR_BLINKING	0x02	Indicator blinking mode
	DEM_INDICATOR_BLINK_CONT	0x03	Indicator blinking or continuously on mode
	DEM_INDICATOR_SLOW_FLASH	0x04	Indicator slow flashing mode
	DEM_INDICATOR_FAST_FLASH	0x05	Indicator fast flashing mode
Description:	Indicator mode used by Dem_GetIndicatorStatus() and SetIndicatorStatus().		

Table 8.9: Dem_IndicatorStatusType

⌋

8.2.1.9 Dem_DTCKindType

[SWS_Dem_00932] ⌈

Name:	**Dem_DTCKindType**		
Type:	uint8		
Range:	DEM_DTC_KIND_ALL_DTCS	0x01	Select all DTCs
	DEM_DTC_KIND_EMISSION_REL_DTCS	0x02	Select OBD-relevant DTCs
Description:	This type is used to filter DTCs for their kind.		

Table 8.10: Dem_DTCKindType

⌋

8.2.1.10 Dem_DTCFormatType

[SWS_Dem_00933] ⌈

Name:	**Dem_DTCFormatType**		
Type:	uint8		
Range:	DEM_DTC_FORMAT_OBD	0	selects the 2-byte OBD DTC format (refer to configuration parameter DemObdDTC)
	DEM_DTC_FORMAT_UDS	1	selects the 3-byte UDS DTC format (refer to configuration parameter DemUdsDTC)
	DEM_DTC_FORMAT_J1939	2	selects the merged SPN + FMI to 3-byte J1939 DTC format (refer to DemJ1939DTC)
Description:	This type is used to select the format of the DTC value.		

Table 8.11: Dem_DTCFormatType

⌋

8.2.1.11 Dem_DTCOriginType

[SWS_Dem_00934] ⌈

Name:	**Dem_DTCOriginType**		
Type:	uint8		
Range:	DEM_DTC_ORIGIN_PRIMARY_MEMORY	0x01	Event information located in the primary memory
	DEM_DTC_ORIGIN_MIRROR_MEMORY	0x02	Event information located in the mirror memory
	DEM_DTC_ORIGIN_PERMANENT_MEMORY	0x03	The Event information is located in the permanent memory
	DEM_DTC_ORIGIN_SECONDARY_MEMORY	0x04	Event information located in the secondary memory
Description:	This enum is used to define the location of the events. The definition and use of the different memory types is OEM-specific.		

Table 8.12: Dem_DTCOriginType

⌋

8.2.1.12 Dem_DTCRequestType

[SWS_Dem_00935] ⌈

Name:	**Dem_DTCRequestType**		
Type:	uint8		
Range:	DEM_FIRST_FAILED_DTC	0x01	first failed DTC requested
	DEM_MOST_RECENT_FAILED_DTC	0x02	most recent failed DTC requested
	DEM_FIRST_DET_CONFIRMED_DTC	0x03	first detected confirmed DTC requested
	DEM_MOST_REC_DET_CONFIRMED_DTC	0x04	most recently detected confirmed DTC requested
Description:	This type is used to request a DTC with specific attributes.		

Table 8.13: Dem_DTCRequestType

⌋

8.2.1.13 Dem_DTCTranslationFormatType

[SWS_Dem_00936] ⌈

Name:	**Dem_DTCTranslationFormatType**		
Type:	uint8		
Range:	DEM_DTC_TRANSLATION_ISO15031_6	0x00	ISO15031-6 DTC format
	DEM_DTC_TRANSLATION_ISO14229_1	0x01	ISO14229-1 DTC format
	DEM_DTC_TRANSLATION_SAEJ1939_73	0x02	SAEJ1939-73 DTC format
	DEM_DTC_TRANSLATION_ISO11992_4	0x03	ISO11992-4 DTC format
Description:	DTC translation format as defined in ISO14229-1 Service 0x19 returned by Dem_DcmGetTranslationType().		

Table 8.14: Dem_DTCTranslationFormatType

⌋

8.2.1.14 Dem_DTCSeverityType

[SWS_Dem_00937] ⌈

Name:	**Dem_DTCSeverityType**		
Type:	uint8		
Range:	DEM_SEVERITY_NO_SEVERITY	0x00	No severity information available
	DEM_SEVERITY_MAINTENANCE_ONLY	0x20	maintenance required
	DEM_SEVERITY_CHECK_AT_NEXT_HALT	0x40	check at next halt
	DEM_SEVERITY_CHECK_IMMEDIATELY	0x80	Check immediately
Description:	Defines the type of a DTCSeverity according to ISO 14229-1 Annex D.3. If the type is used for a DTCSeverityMask, each bit (5-7) has an individual meaning. In this case it is possible to set multiple bits at once, to select multiple kinds of DTC severity simultaneously. For example, to select MAINTENANCE_ONLY and CHECK_IMMEDIATELY the value 0xA0 is used.		

Table 8.15: Dem_DTCSeverityType

⌋

8.2.1.15 Dem_RatioIdType

[SWS_Dem_00940] ⌈

Name:	**Dem_RatioIdType**		
Type:	uint8, uint16		
Range:	0..255, 0..65535	–	Configurable, size depends on system complexity (refer to range of configuration parameter DemRatioId)
Description:	OBD specific ratio Id (related to a specific event, a FID, and an IUMPR group). This type depends on the Dem configuration.		

Table 8.16: Dem_RatioIdType

⌋

8.2.1.16 Dem_DTRControlType

[SWS_Dem_00941] ⌈

Name:	**Dem_DTRControlType**		
Type:	uint8		
Range:	DEM_DTR_CTL_NORMAL	0x00	Values are reported and regarded as valid test result
	DEM_DTR_CTL_NO_MAX	0x01	Values are reported, but maximum limit is not available (not valid); upper limit value is ignored.
	DEM DTR_CTL_NO_MIN	0x02	Values are reported, but minimum limit is not available (not valid); lower limit value is ignored.

	DEM_DTR_CTL_RESET	0x03	Values are all ignored. External representation will be all zeros as initialized (e.g. after fault clear)
	DEM DTR_CTL_INVISIBLE	0x04	Values are all ignored. This DTR is treated for the external view (tester) as if not integrated.
Description:	Control parameter for the interpretation of the reported test results.		

Table 8.17: Dem_DTRControlType

⌋

8.2.1.17 Dem_InitMonitorReasonType

[SWS_Dem_00942] ⌈

Name:	**Dem_InitMonitorReasonType**		
Type:	uint8		
Range:	DEM_INIT_MONITOR_CLEAR	0x01	Event was cleared and all internal values and states are reset.
	DEM_INIT_MONITOR_RESTART	0x02	Operation cycle of the event was (re-)started.
	DEM_INIT_MONITOR_REENABLED	0x03	Enable conditions or DTC settings re-enabled.
Description:	(Re-)Initialization reason returned by the callback <Module>_DemInitMonitorFor<EventName>().		

Table 8.18: Dem_InitMonitorReasonType

⌋

8.2.1.18 Dem_IumprDenomCondIdType

[SWS_Dem_00943] ⌈

Name:	**Dem_IumprDenomCondIdType**
Type:	uint8

Range:	DEM_IUMPR_GENERAL_DENOMINATOR	0x01	IUMPR denominator condition "General Denominator"
	DEM_IUMPR_DEN_COND_COLDSTART	0x02	Additional IUMPR denominator condition "Cold Start"
	DEM_IUMPR_DEN_COND_EVAP	0x03	Additional IUMPR denominator condition "EVAP"
	DEM_IUMPR_DEN_COND_500MI	0x04	Additional IUMPR denominator condition "500 miles"
Description:	This type contains all possible additional IUMPR denominator conditions to be broadcasted among OBD-relevant ECUs.		

Table 8.19: Dem_IumprDenomCondIdType

⌋

8.2.1.19 Dem_IumprDenomCondStatusType

[SWS_Dem_00944] ⌈

Name:	**Dem_IumprDenomCondStatusType**		
Type:	uint8		
Range:	DEM_IUMPR_DEN_STATUS_NOT_REACHED	0x00	Condition of IUMPR-Denominator given by IUMPRDenCondId is not met (yet).
	DEM_IUMPR_DEN_STATUS_REACHED	0x01	Condition of IUMPR-Denominator given by IUMPRDenCondId is met
	DEM_IUMPR_DEN_STATUS_INHIBITED	0x02	Condition of IUMPR-Denominator given by IUMPRDenCondId is inhibited and cannot be reached.
		0x03 - 0xFF	reserved
Description:	This type contains all possible states of an additional IUMPR denominator condition to be broadcasted among OBD-relevant ECUs.		

Table 8.20: Dem_IumprDenomCondStatusType

⌋

8.2.1.20 Dem_J1939DcmDTCStatusFilterType

[SWS_Dem_00945] ⌈

Name:	**Dem_J1939DcmDTCStatusFilterType**		
Type:	uint8		
Range:	DEM_J1939DTC_ACTIVE	0	active DTCs
	DEM_J1939DTC_PREVIOUSLY_ACTIVE	1	previously active DTCs
	DEM_J1939DTC_PENDING	2	pending DTCs
	DEM_J1939DTC_PERMANENT	3	permanent DTCs
	DEM_J1939DTC_CURRENTLY_ACTIVE	4	currently active DTC
Description:	The type to distinguish which DTCs should be filtered.		

Table 8.21: Dem_J1939DcmDTCStatusFilterType

⌋

8.2.1.21 Dem_J1939DcmSetClearFilterType

[SWS_Dem_00946] ⌈

Name:	**Dem_J1939DcmSetClearFilterType**		
Type:	uint8		
Range:	DEM_J1939DTC_CLEAR_ALL	0	active DTCs
	DEM_J1939DTC_CLEAR_PREVIOUSLY_ACTIVE	1	previously active DTCs
Description:	The type to distinguish which DTCs gets cleared		

Table 8.22: Dem_J1939DcmSetClearFilterType

⌋

8.2.1.22 Dem_J1939DcmSetFreezeFrameFilterType

[SWS_Dem_00947] ⌈

Name:	**Dem_J1939DcmSetFreezeFrameFilterType**		
Type:	uint8		
Range:	DEM_J1939DCM_FREEZEFRAME	0	FreezeFrame (DM4)
	DEM_J1939DCM_EXPANDED_FREEZEFRAME	1	ExpandedFreezeFrame (DM25)
	DEM_J1939DCM_SPNS_IN_EXPANDED_FREEZEFRAME	2	SPNs in Expanded-FreezeFrame (DM24)
Description:	The type to distinguish which DTCs gets cleared		

Table 8.23: Dem_J1939DcmSetFreezeFrameFilterType

⌋

8.2.1.23 Dem_J1939DcmLampStatusType

[SWS_Dem_00948] ⌈

Name:	**Dem_J1939DcmLampStatusType**		
Type:	uint16		
Range:	bits 8-7	–	Malfunction Indicator Lamp Status
	bits 6-5	–	Red Stop Lamp Status
	bits 4-3	–	Amber Warning Lamp Status
	bits 2-1	–	Protect Lamp Status
	bits 8-7	–	Flash Malfunction Indicator Lamp
	bits 6-5	–	Flash Red Stop Lamp
	bits 4-3	–	Flash Amber Warning Lamp
	bits 2-1	–	Flash Protect Lamp
Description:	For details refer SAE J1939-73		

Table 8.24: Dem_J1939DcmLampStatusType

⌋

8.2.1.24 Dem_J1939DcmDiagnosticReadiness1Type

[SWS_Dem_00949] ⌈

Name	**Dem_J1939DcmDiagnosticReadiness1Type**		
Type	Structure		
Element:	uint8	ActiveTroubleCodes	Number of actice DTCs
	uint8	PreviouslyActive DiagnosticTrouble Codes	Number of previously actice DTCs
	uint8	OBDCompliance	OBD Compliance
	uint8	ContinuouslyMonitored SystemsSupport_ Status	Identifies the continuously monitored system support and status
	uint16	NonContinuously MonitoredSystems Support	Identifies the non-continuously monitored systems support
	uint16	NonContinuously MonitoredSystems Status	Identifies the non-continuously monitored systems status
Description	This structure represents all data elemets of the DM5 message. The encoding shall be done acording SAE J1939-73		

Table 8.25: Dem_J1939DcmDiagnosticReadiness1Type

⌋

8.2.1.25 Dem_J1939DcmDiagnosticReadiness2Type

[SWS_Dem_00950] ⌈

Name	**Dem_J1939DcmDiagnosticReadiness2Type**		
Type	Structure		
Element:	uint16	DistanceTraveledWhile MILisActivated	The kilometers accumulated while the MIL is activated
	uint16	DistanceSinceDTCs Cleared	Distance accumulated since emission related DTCs were cleared
	uint16	MinutesRunbyEngine WhileMILisActivated	Accumulated count (in minutes) while the MIL is activated (on)

	uint16	TimeSinceDiagnostic TroubleCodesCleared	Engine running time accumulated since emission related DTCs were cleared
Description	This structure represents all data elemets of the DM21 message. The encoding shall be done acording SAE J1939-73		

Table 8.26: Dem_J1939DcmDiagnosticReadiness2Type

⌋

8.2.1.26 Dem_J1939DcmDiagnosticReadiness3Type

[SWS_Dem_00951] ⌈

Name	**Dem_J1939DcmDiagnosticReadiness3Type**		
Type	Structure		
Element:	uint16	TimeSinceEngineStart	Time since key-on that the engine has been running.
	uint8	NumberofWarmups SinceDTCsCleared	Number of OBD warm-up cycles since all DTCs were cleared
	uint8	ContinuouslyMonitored SystemsEnable CompletedStatus	Identifies the continuously monitored system enable/completed support and status.
	uint16	NonContinuously MonitoredSystems EnableStatus	Enable status of non-continuous monitors this monitoring cycle
	uint16	NonContinuously MonitoredSystems	Completion status of non-continuous monitors this monitoring cycle
Description	This structure represents all data elemets of the DM26 message. The encoding shall be done acording SAE J1939-73		

Table 8.27: Dem_J1939DcmDiagnosticReadiness3Type

⌋

8.2.2 Dem return types

8.2.2.1 Dem_ReturnGetStatusOfDTCType

[SWS_Dem_00952] ⌈

Name:	**Dem_ReturnGetStatusOfDTCType**		
Type:	uint8		
Range:	DEM_STATUS_OK	0x00	Status of DTC is OK
	DEM_STATUS_WRONG_DTC	0x01	DTC value not existing (in this format)
	DEM_STATUS_WRONG_DTCORIGIN	0x02	Wrong DTC origin
	DEM_STATUS_FAILED	0x03	DTC failed
	DEM_STATUS_PENDING	0x04	The requested value is calculated asynchronously and currently not available. The caller can retry later.
Description:	Used to return the status of Dem_DcmGetStatusOfDTC.		

Table 8.28: Dem_ReturnGetStatusOfDTCType

⌋

8.2.2.2 Dem_ReturnGetSeverityOfDTCType

[SWS_Dem_00953] ⌈

Name:	**Dem_ReturnGetSeverityOfDTCType**		
Type:	uint8		
Range:	DEM_GET_SEVERITYOFDTC_OK	0x00	Severity successfully returned.
	DEM_GET_SEVERITYOFDTC_WRONG_DTC	0x01	DTC value not existing (in UDS format)
	DEM_GET_SEVERITYOFDTC_NOSEVERITY	0x02	Severity information is not available
	DEM_GET_SEVERITYOFDTC_PENDING	0x03	The requested value is calculated asynchronously and currently not available. The caller can retry later.
Description:	Used to return the status of Dem_DcmGetSeverityOfDTC.		

Table 8.29: Dem_ReturnGetSeverityOfDTCType

⌋

8.2.2.3 Dem_ReturnGetFunctionalUnitOfDTCType

[SWS_Dem_00954] ⌈

Name:	**Dem_ReturnGetFunctionalUnitOfDTCType**		
Type:	uint8		
Range:	DEM_GET_FUNCTIONALUNITOFDTC_OK	0x00	Functional unit successfully returned.
	DEM_GET_FUNCTIONALUNITOFDTC_WRONG_DTC	0x01	DTC value not existing (in UDS format)
Description:	Used to return the status of Dem_DcmGetFunctionalUnitOfDTC.		

Table 8.30: Dem_ReturnGetFunctionalUnitOfDTCType

⌋

8.2.2.4 Dem_ReturnSetFilterType

[SWS_Dem_00955] ⌈

Name:	**Dem_ReturnSetFilterType**		
Type:	uint8		
Range:	DEM_FILTER_ACCEPTED	0x00	Filter was accepted
	DEM_WRONG_FILTER	0x01	Wrong filter selected
Description:	Used to return the status of (re-)setting a specific filter.		

Table 8.31: Dem_ReturnSetFilterType

⌋

8.2.2.5 Dem_ReturnGetNumberOfFilteredDTCType

[SWS_Dem_00956] ⌈

Name:	**Dem_ReturnGetNumberOfFilteredDTCType**		
Type:	uint8		
Range:	DEM_NUMBER_OK	0x00	Getting number of filtered DTCs was successful.
	DEM_NUMBER_FAILED	0x01	Getting number of filtered DTCs failed.
	DEM_NUMBER_PENDING	0x02	The requested value is calculated asynchronously and currently not available. The caller can retry later.
Description:	Used to return the status of Dem_DcmGetNumberOfFilteredDTC.		

Table 8.32: Dem_ReturnGetNumberOfFilteredDTCType

⌋

8.2.2.6 Dem_ReturnGetNextFilteredElementType

[SWS_Dem_00957] ⌈

Name:	**Dem_ReturnGetNextFilteredElementType**		
Type:	uint8		
Range:	DEM_FILTERED_OK	0x00	Returned next filtered element
	DEM_FILTERED_NO_MATCHING_ELEMENT	0x01	No further element (matching the filter criteria) found
	DEM_FILTERED_PENDING	0x02	The requested value is calculated asynchronously and currently not available. The caller can retry later. Only used by asynchronous interfaces.
	DEM_FILTERED_BUFFER_TOO_SMALL	0x03	Buffer in the BufSize parameter is not huge enough
Description:	Used to return the status of Dem_DcmGetNextFilteredXxx and Dem_J1939DcmGetNextFilteredXxx.		

Table 8.33: Dem_ReturnGetNextFilteredElementType

⌋

8.2.2.7 Dem_ReturnGetDTCByOccurrenceTimeType

[SWS_Dem_00958] ⌈

Name:	**Dem_ReturnGetDTCByOccurrenceTimeType**		
Type:	uint8		
Range:	DEM_OCCURR_OK DEM_OCCURR_NOT_AVAILABLE	0x00 0x01	matching DTC available no DTC is matching the requested occurrence time
Description:	Used to return the status of Dem_DtcGetDTCByOccurrenceTime.		

Table 8.34: Dem_ReturnGetDTCByOccurrenceTimeType

⌋

8.2.2.8 Dem_ReturnDisableDTCRecordUpdateType

[SWS_Dem_00959] ⌈

Name:	**Dem_ReturnDisableDTCRecordUpdateType**		
Type:	uint8		
Range:	DEM_DISABLE_DTCRECUP_OK	0x00	Event memory update of DTC successfully disabled
	DEM_DISABLE_DTCRECUP_WRONG_DTC	0x01	DTC value not existing (in UDS format)
	DEM_DISABLE_DTCRECUP_WRONG_DTCORIGIN	0x02	Wrong DTC origin
	DEM_DISABLE_DTCRECUP_PENDING	0x03	Disabling is currently not possible. The caller can retry later.
Description:	Used to return the status of Dem_DcmDisableDTCRecordUpdate		

Table 8.35: Dem_ReturnDisableDTCRecordUpdateType

⌋

8.2.2.9 Dem_ReturnGetFreezeFrameDataByDTCType

[SWS_Dem_00960] ⌈

Name:	**Dem_ReturnGetFreezeFrameDataByDTCType**		
Type:	uint8		
Range:	DEM_GET_FFDATABYDTC_OK	0x00	Size successfully returned.
	DEM_GET_FFDATABYDTC_WRONG_DTC	0x01	DTC value not existing (in UDS format)
	DEM_GET_FFDATABYDTC_WRONG_DTCORIGIN	0x02	Wrong DTC origin
	DEM_GET_FFDATABYDTC_WRONG_RECORDNUMBER	0x03	Record number is not supported by configuration and therefore invalid
	DEM_GET_FFDATABYDTC_WRONG_BUFFERSIZE	0x04	provided buffer size to small
	DEM_GET_FFDATABYDTC_PENDING	0x05	The requested value is calculated asynchronously and currently not available. The caller can retry later.
Description:	Used to return the status of Dem_DcmGetFreezeFrameDataByDTC.		

Table 8.36: Dem_ReturnGetFreezeFrameDataByDTCType

⌋

8.2.2.10 Dem_ReturnGetExtendedDataRecordByDTCType

[SWS_Dem_00961] ⌈

Name:	**Dem_ReturnGetExtendedDataRecordByDTCType**		
Type:	uint8		
Range:	DEM_RECORD_OK	0x00	Extended data record successfully returned
	DEM_RECORD_WRONG_DTC	0x01	DTC value not existing (in UDS format)
	DEM_RECORD_WRONG_DTCORIGIN	0x02	Origin wrong

	DEM_RECORD_WRONG_NUMBER	0x03	Record number is not supported by configuration and therefore invalid
	DEM_RECORD_WRONG_BUFFERSIZE	0x04	Provided buffer to small
	DEM_RECORD_PENDING	0x05	The requested value is calculated asynchronously and currently not available. The caller can retry later.
Description:	Used to return the status of Dem_DcmGetExtendedDataRecordByDTC.		

Table 8.37: Dem_ReturnGetExtendedDataRecordByDTCType

⌋

8.2.2.11 Dem_ReturnGetSizeOfDataByDTCType

[SWS_Dem_00962] ⌈

Name:	**Dem_ReturnGetSizeOfDataByDTCType**		
Type:	uint8		
Range:	DEM_GETSIZEBYDTC_OK	0x00	Size successfully returned
	DEM_GETSIZEBYDTC_WRONG_DTC	0x01	DTC value not existing (in UDS format)
	DEM_GETSIZEBYDTC_WRONG_DTCORIGIN	0x02	Wrong DTC origin
	DEM_GETSIZEBYDTC_WRONG_RECNUM	0x03	Record number is not supported by configuration and therefore invalid
	DEM_GETSIZEBYDTC_PENDING	0x04	The requested value is calculated asynchronously and currently not available. The caller can retry later.
Description:	Used to return the status of Dem_DcmGetSizeOfFreezeFrameByDTC and Dem_DcmGetSizeOfExtendedDataRecordByDTC.		

Table 8.38: Dem_ReturnGetSizeOfDataByDTCType

⌋

8.2.2.12 Dem_ReturnClearDTCType

[SWS_Dem_00963] ⌈

Name:	**Dem_ReturnClearDTCType**		
Type:	uint8		
Range:	DEM_CLEAR_OK	0x00	DTC successfully cleared
	DEM_CLEAR_WRONG_DTC	0x01	DTC value not existing (in this format)
	DEM_CLEAR_WRONG_DTCORIGIN	0x02	Wrong DTC origin
	DEM_CLEAR_FAILED	0x03	DTC clearing failed
	DEM_CLEAR_PENDING	0x04	The DTC clearing is performed asynchronously and still pending. The caller can retry later.
	DEM_CLEAR_BUSY	0x05	DTC not cleared, as another clearing process is in progress. The caller can retry later.
	DEM_CLEAR_MEMORY_ERROR	0x06	An error occurred during erasing a memory location (e.g. if DemClearDTCBehavior in Dem is set to DEM_CLRRESP_NONVOLAT and erasing of non-volatile-block failed).
Description:	Used to return the status of Dem_<...>ClearDTC.		

Table 8.39: Dem_ReturnClearDTCType

⌋

8.2.2.13 Dem_ReturnControlDTCSettingType

[SWS_Dem_00964] ⌈

Name:	**Dem_ReturnControlDTCSettingType**

Type:	uint8		
Range:	DEM_CONTROL_DTC_SETTING_OK	0x00	DTC setting control successful
	DEM_CONTROL_DTC_SETTING_N_OK	0x01	DTC setting control not successful
	DEM_CONTROL_DTC_WRONG_DTCGROUP	0x02	DTC setting control not successful because group of DTC was wrong
Description:	Used to return the status of Dem_DcmDisableDTCSetting and Dem_DcmEnableDTCSetting.		

Table 8.40: Dem_ReturnControlDTCSettingType

⌋

8.3 Function definitions

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

8.3.1 Dem_GetVersionInfo

[SWS_Dem_00177] ⌈

Service name:	Dem_GetVersionInfo	
Syntax:	void Dem_GetVersionInfo(Std_VersionInfoType* versioninfo)	
Service ID[hex]:	0x00	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	None	
Parameters (in-out):	None	
Parameters (out):	versioninfo	Pointer to where to store the version information of this module.
Return value:	None	
Description:	Returns the version information of this module.	

Table 8.41: Dem_GetVersionInfo

⌋

8.3.2 Interface ECU State Manager <=> Dem

8.3.2.1 Dem_PreInit

[SWS_Dem_00179] ⌈

Service name:	Dem_PreInit
Syntax:	void Dem_PreInit(void)
Service ID[hex]:	0x01
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None
Parameters (in-out):	None
Parameters (out):	None
Return value:	None
Description:	Initializes the internal states necessary to process events reported by BSW-modules.

Table 8.42: Dem_PreInit

⌋

8.3.2.2 Dem_Init

[SWS_Dem_00181] ⌈

Service name:	Dem_Init	
Syntax:	void Dem_Init(const Dem_ConfigType* ConfigPtr)	
Service ID[hex]:	0x02	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	ConfigPtr	Pointer to the configuration set in VARIANT-POST-BUILD.
Parameters (in-out):	None	

Parameters (out):	None
Return value:	None
Description:	Initializes or reinitializes this module.

Table 8.43: Dem_Init

⌋(*SRS_BSW_00101*)

8.3.2.3 Dem_Shutdown

[SWS_Dem_00182] ⌈

Service name:	Dem_Shutdown
Syntax:	void Dem_Shutdown(void)
Service ID[hex]:	0x03
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None
Parameters (in-out):	None
Parameters (out):	None
Return value:	None
Description:	Shuts down this module.

Table 8.44: Dem_Shutdown

⌋(*SRS_BSW_00336*)

8.3.3 Interface BSW modules / SW-Components <=> Dem

8.3.3.1 Dem_ReportErrorStatus

[SWS_Dem_00206] ⌈

Service name:	Dem_ReportErrorStatus

Syntax:	void Dem_ReportErrorStatus(Dem_EventIdType EventId, Dem_EventStatusType EventStatus)	
Service ID[hex]:	0x0f	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant for different EventIds. Non reentrant for the same EventId.	
Parameters (in):	EventId	Identification of an event by assigned Event ID.
	EventStatus	Monitor test result
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	None	
Description:	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. OBD Events Suppression shall be ignored for this computation.	

Table 8.45: Dem_ReportErrorStatus

⌋

8.3.3.2 Dem_SetEventStatus

[SWS_Dem_00183] ⌈

Service name:	Dem_SetEventStatus	
Syntax:	Std_ReturnType Dem_SetEventStatus(Dem_EventIdType EventId, Dem_EventStatusType EventStatus)	
Service ID[hex]:	0x04	
Sync/Async:	Synchronous/Asynchronous	
Reentrancy:	Reentrant for different EventIds. Non reentrant for the same EventId.	
Parameters (in):	EventId	Identification of an event by assigned EventId.
	EventStatus	Monitor test result
Parameters (in-out):	None	

Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: set of event status was successful E_NOT_OK: set of event status failed or could not be accepted (e.g.: the operation cycle configured for this event has not been started, an according enable condition has been disabled)
Description:	Processes the events reported by SW-Cs via RTE. This API can only be used through the RTE, and therefore no declaration is exported via Dem.h. Some bits of the UDS DTC status byte changes synchronously or asynchronously (refer to Dem036 and Dem379). OBD Events Suppression shall be ignored for this computation.	

Table 8.46: Dem_SetEventStatus

⌋

8.3.3.3 Dem_ResetEventDebounceStatus

[SWS_Dem_00683] ⌈

Service name:	Dem_ResetEventDebounceStatus	
Syntax:	Std_ReturnType Dem_ResetEventDebounceStatus(Dem_EventIdType EventId, Dem_DebounceResetStatusType DebounceResetStatus)	
Service ID[hex]:	0x09	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different EventIds. Non reentrant for the same EventId.	
Parameters (in):	EventId	Identification of an event by assigned EventId.
	DebounceReset Status	Freeze or reset the internal debounce counter/timer of the specified event.
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Operation was successful E_NOT_OK: Only on development error

Description:	Control the internal debounce counter/timer by BSW modules and SW-Cs. The event qualification will not be affected by these debounce state changes. This API is available for BSW modules as soon as Dem_PreInit has been completed (refer to SWS_Dem_00438 and SWS_Dem_00167).

Table 8.47: Dem_ResetEventDebounceStatus

⌋

8.3.3.4 Dem_ResetEventStatus

[SWS_Dem_00185] ⌈

Service name:	Dem_ResetEventStatus	
Syntax:	Std_ReturnType Dem_ResetEventStatus(Dem_EventIdType EventId)	
Service ID[hex]:	0x05	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different EventIds. Non reentrant for the same EventId.	
Parameters (in):	EventId	Identification of an event by assigned EventId.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: reset of event status was successful E_NOT_OK: reset of event status failed or is not allowed, because the event is already tested in this operation cycle
Description:	Resets the event failed status. This API can only be used through the RTE and therefore no declaration is exported via Dem.h.	

Table 8.48: Dem_ResetEventStatus

⌋

8.3.3.5 Dem_PrestoreFreezeFrame

[SWS_Dem_00188] ⌈

Service name:	Dem_PrestoreFreezeFrame	
Syntax:	Std_ReturnType Dem_PrestoreFreezeFrame(Dem_EventIdType EventId)	
Service ID[hex]:	0x06	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different EventIds. Non reentrant for the same EventId.	
Parameters (in):	EventId	Identification of an event by assigned EventId.
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK Freeze frame prestorage was successful E_NOT_OK Freeze frame prestorage failed
Description:	Captures the freeze frame data for a specific event. This API can only be used through the RTE and therefore no declaration is exported via Dem.h.	

Table 8.49: Dem_PrestoreFreezeFrame

⌋

8.3.3.6 Dem_ClearPrestoredFreezeFrame

[SWS_Dem_00193] ⌈

Service name:	Dem_ClearPrestoredFreezeFrame	
Syntax:	Std_ReturnType Dem_ClearPrestoredFreezeFrame(Dem_EventIdType EventId)	
Service ID[hex]:	0x07	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different EventIds. Non reentrant for the same EventId.	
Parameters (in):	EventId	Identification of an event by assigned EventId.
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Clear prestored freeze frame was successful E_NOT_OK: Clear prestored freeze frame failed

Description:	Clears a prestored freeze frame of a specific event. This API can only be used through the RTE and therefore no declaration is exported via Dem.h.

Table 8.50: Dem_ClearPrestoredFreezeFrame

⌋

8.3.3.7 Dem_SetOperationCycleState

[SWS_Dem_00194] ⌈

Service name:	Dem_SetOperationCycleState	
Syntax:	Std_ReturnType Dem_SetOperationCycleState(uint8 OperationCycleId, Dem_OperationCycleStateType CycleState)	
Service ID[hex]:	0x08	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant	
Parameters (in):	OperationCycleId	Identification of operation cycle, like power cycle, driving cycle.
	CycleState	New operation cycle state: (re-)start or end
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: set of operation cycle was accepted and will be handled asynchronously E_NOT_OK: set of operation cycle was rejected
Description:	Sets an operation cycle state. This API can only be used through the RTE and therefore no declaration is exported via Dem.h. The interface has an asynchronous behavior to avoid exceeding of typical timing requirements on APIs if a large number of events has to be processed and during the re-initializations of the related monitors. The asynchronous acknowledgements are the related InitMonitorForEvent callbacks.	

Table 8.51: Dem_SetOperationCycleState

⌋

8.3.3.8 Dem_GetOperationCycleState

[SWS_Dem_00729] ⌈

Service name:	Dem_GetOperationCycleState	
Syntax:	Std_ReturnType Dem_GetOperationCycleState(uint8 OperationCycleId, Dem_OperationCycleStateType* CycleState)	
Service ID[hex]:	0x9e	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	OperationCycleId	Identification of operation cycle, like power cycle, driving cycle.
Parameters (in-out):	None	
Parameters (out):	CycleState	Cycle status information
Return value:	Std_ReturnType	E_OK: read out of operation cycle was successful E_NOT_OK: read out of operation cycle failed
Description:	Gets information about the status of a specific operation cycle. This API can only be used through the RTE and therefore no declaration is exported via Dem.h.	

Table 8.52: Dem_GetOperationCycleState

⌋

8.3.3.9 Dem_SetAgingCycleState

[SWS_Dem_00554] ⌈

Service name:	Dem_SetAgingCycleState	
Syntax:	Std_ReturnType Dem_SetAgingCycleState(uint8 AgingCycleId)	
Service ID[hex]:	0x11	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	AgingCycleId	Identification of aging cycle.
Parameters (in-out):	None	
Parameters (out):	None	

Return value:	Std_ReturnType	E_OK: set of aging cycle was successful E_NOT_OK: set of aging cycle failed
Description:	Triggers the next aging cycle state. This API can only be used through the RTE, and therefore no declaration is exported via Dem.h.	

Table 8.53: Dem_SetAgingCycleState

⌋

8.3.3.10 Dem_SetAgingCycleCounterValue

[SWS_Dem_00555] ⌈

Service name:	Dem_SetAgingCycleCounterValue	
Syntax:	Std_ReturnType Dem_SetAgingCycleCounterValue(uint8 CounterValue)	
Service ID[hex]:	0x12	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	CounterValue	Current external aging counter value.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: set of aging counter was successful E_NOT_OK: set of aging counter failed
Description:	Provides the value of the external aging counter. This API can only be used through the RTE, and therefore no declaration is exported via Dem.h.	

Table 8.54: Dem_SetAgingCycleCounterValue

⌋

8.3.3.11 Dem_SetWIRStatus

[SWS_Dem_00839] ⌈

Service name:	Dem_SetWIRStatus

Syntax:	Std_ReturnType Dem_SetWIRStatus(Dem_EventIdType EventId, boolean WIRStatus)	
Service ID[hex]:	0x7a	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different EventIds. Non reentrant for the same EventId.	
Parameters (in):	EventId	Identification of an event by assigned EventId. The Event Number is configured in the DEM. Min.: 1 (0: Indication of no Event or Failure) Max.:Result of configuration of Event Numbers in DEM (Max is either 255 or 65535)
	WIRStatus	Requested status of event related WIR-bit (regarding to the current status of function inhibition) WIRStatus = TRUE -> WIR-bit shall be set to "1" WIRStatus = FALSE -> WIR-bit shall be set to "0"
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: The request is accepted E_NOT_OK: not be accepted (e.g. disabled controlDTCSetting) and should be repeated.
Description:	Sets the WIR status bit via failsafe SW-Cs. This API can only be used through the RTE and therefore no declaration is exported via Dem.h.	

Table 8.55: Dem_SetWIRStatus

⌋

8.3.3.12 Dem_GetEventStatus

[SWS_Dem_00195] ⌈

Service name:	Dem_GetEventStatus
Syntax:	Std_ReturnType Dem_GetEventStatus(Dem_EventIdType EventId, Dem_UdsStatusByteType* EventStatusByte)
Service ID[hex]:	0x0a
Sync/Async:	Synchronous

Reentrancy:	Reentrant	
Parameters (in):	EventId	Identification of an event by assigned EventId.
Parameters (in-out):	None	
Parameters (out):	EventStatusByte	UDS DTC status byte of the requested event (refer to chapter "Status bit support"). If the return value of the function call is E_NOT_OK, this parameter does not contain valid data.
Return value:	Std_ReturnType	E_OK: get of event status was successful E_NOT_OK: get of event status failed
Description:	Gets the current extended event status of an event.	

Table 8.56: Dem_GetEventStatus

⌋

8.3.3.13 Dem_GetEventFailed

[SWS_Dem_00196] ⌈

Service name:	Dem_GetEventFailed	
Syntax:	Std_ReturnType Dem_GetEventFailed(Dem_EventIdType EventId, boolean* EventFailed)	
Service ID[hex]:	0x0b	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	EventId	Identification of an event by assigned EventId.
Parameters (in-out):	None	
Parameters (out):	EventFailed	TRUE - Last Failed FALSE - not Last Failed
Return value:	Std_ReturnType	E_OK: get of "EventFailed" was successful E_NOT_OK: get of "EventFailed" was not successful
Description:	Gets the event failed status of an event.	

Table 8.57: Dem_GetEventFailed

⌋

8.3.3.14 Dem_GetEventTested

[SWS_Dem_00197] ⌈

Service name:	Dem_GetEventTested	
Syntax:	Std_ReturnType Dem_GetEventTested(Dem_EventIdType EventId, boolean* EventTested)	
Service ID[hex]:	0x0c	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	EventId	Identification of an event by assigned EventId.
Parameters (in-out):	None	
Parameters (out):	EventTested	TRUE - event tested this cycle FALSE - event not tested this cycle
Return value:	Std_ReturnType	E_OK: get of event state "tested" successful E_NOT_OK: get of event state "tested" failed
Description:	Gets the event tested status of an event.	

Table 8.58: Dem_GetEventTested

⌋

8.3.3.15 Dem_GetDebouncingOfEvent

[SWS_Dem_00730] ⌈

Service name:	Dem_GetDebouncingOfEvent	
Syntax:	Std_ReturnType Dem_GetDebouncingOfEvent(Dem_EventIdType EventId, Dem_DebouncingStateType* DebouncingState)	
Service ID[hex]:	0x9f	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	EventId	Identification of an event by assigned EventId.
Parameters (in-out):	None	

Parameters (out):	Debouncing State	Bit 0 Temporarily Defective (corresponds to 0 < FDC < 127) Bit 1 finally Defective (corresponds to FDC = 127) Bit 2 temporarily healed (corresponds to -128 < FDC < 0) Bit 3 Test complete (corresponds to FDC = -128 or FDC = 127) Bit 4 DTR Update (= Test complete && Debouncing complete && enable conditions / storage conditions fulfilled)
Return value:	Std_ReturnType	E_OK: get of debouncing status per event state successful E_NOT_OK: get of debouncing per event state failed
Description:	Gets the debouncing status of an event. This function shall not be used for EventId with native debouncing within their functions. It is rather for EventIds using debouncing within the Dem.	

Table 8.59: Dem_GetDebouncingOfEvent

⌋

8.3.3.16 Dem_GetDTCOfEvent

[SWS_Dem_00198] ⌈

Service name:	Dem_GetDTCOfEvent	
Syntax:	Std_ReturnType Dem_GetDTCOfEvent(Dem_EventIdType EventId, Dem_DTCFormatType DTCFormat, uint32* DTCOfEvent)	
Service ID[hex]:	0x0d	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	EventId	Identification of an event by assigned EventId.
	DTCFormat	Defines the output-format of the requested DTC value.
Parameters (in-out):	None	
Parameters (out):	DTCOfEvent	Receives the DTC value in respective format returned by this function. If the return value of the function is other than E_OK this parameter does not contain valid data.

Return value:	Std_ReturnType	E_OK: get of DTC was successful E_NOT_OK: the call was not successful DEM_E_NO_DTC_AVAILABLE: there is no DTC configured in the requested format
Description:	Gets the DTC of an event.	

Table 8.60: Dem_GetDTCOfEvent

⌋

8.3.3.17 Dem_SetEnableCondition

[SWS_Dem_00201] ⌈

Service name:	Dem_SetEnableCondition	
Syntax:	Std_ReturnType Dem_SetEnableCondition(uint8 EnableConditionID, boolean ConditionFulfilled)	
Service ID[hex]:	0x39	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	EnableCondition ID	This parameter identifies the enable condition.
	ConditionFulfilled	This parameter specifies whether the enable condition assigned to the EnableConditionID is fulfilled (TRUE) or not fulfilled (FALSE).
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	In case the enable condition could be set successfully the API call returns E_OK. If the setting of the enable condition failed the return value of the function is E_NOT_OK.
Description:	Sets an enable condition.	

Table 8.61: Dem_SetEnableCondition

⌋

8.3.3.18 Dem_SetStorageCondition

[SWS_Dem_00556] ⌈

Service name:	Dem_SetStorageCondition	
Syntax:	Std_ReturnType Dem_SetStorageCondition(uint8 StorageConditionID, boolean ConditionFulfilled)	
Service ID[hex]:	0x38	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	StorageConditionID	This parameter identifies the storage condition.
	ConditionFulfilled	This parameter specifies whether the storage condition assigned to the StorageConditionID is fulfilled (TRUE) or not fulfilled (FALSE).
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	In case the storage condition could be set successfully the API call returns E_OK. If the setting of the storage condition failed the return value of the function is E_NOT_OK.
Description:	Sets a storage condition.	

Table 8.62: Dem_SetStorageCondition

⌋

8.3.3.19 Dem_GetFaultDetectionCounter

[SWS_Dem_00203] ⌈

Service name:	Dem_GetFaultDetectionCounter	
Syntax:	Std_ReturnType Dem_GetFaultDetectionCounter(Dem_EventIdType EventId, sint8* FaultDetectionCounter)	
Service ID[hex]:	0x3e	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	EventId	Identification of an event by assigned EventId.

Parameters (in-out):	None	
Parameters (out):	FaultDetection Counter	This parameter receives the Fault Detection Counter information of the requested EventId. If the return value of the function call is other than E_OK this parameter does not contain valid data. -128dec...127dec PASSED... FAILED according to ISO 14229-1
Return value:	Std_ReturnType	E_OK: request was successful E_NOT_OK: request failed DEM_E_NO_FDC_AVAILABLE: there is no fault detection counter available for the requested event
Description:	Gets the fault detection counter of an event. This API can only be used through the RTE, and therefore no declaration is exported via Dem.h.	

Table 8.63: Dem_GetFaultDetectionCounter

⌋

8.3.3.20 Dem_GetIndicatorStatus

[SWS_Dem_00205] ⌈

Service name:	Dem_GetIndicatorStatus	
Syntax:	Std_ReturnType Dem_GetIndicatorStatus(uint8 IndicatorId, Dem_IndicatorStatusType* IndicatorStatus)	
Service ID[hex]:	0x29	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	IndicatorId	Number of indicator
Parameters (in-out):	None	
Parameters (out):	IndicatorStatus	Status of the indicator, like off, on, or blinking.
Return value:	Std_ReturnType	E_OK: Operation was successful E_NOT_OK: Operation failed
Description:	Gets the indicator status derived from the event status.	

Table 8.64: Dem_GetIndicatorStatus

⌋

8.3.3.21 Dem_SetIndicatorStatus

[SWS_Dem_00732] ⌈

Service name:	Dem_SetIndicatorStatus	
Syntax:	Std_ReturnType Dem_SetIndicatorStatus(uint8 IndicatorId, Dem_IndicatorStatusType* IndicatorStatus)	
Service ID[hex]:	0xa1	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	IndicatorId	Number of indicator
	IndicatorStatus	Status of the indicator, like off, on, or blinking.
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Operation was successful E_NOT_OK: Operation failed or is not supported
Description:	Sets the indicator status included in the event status. API will only exist in a Master ECU a software component if the MIL Master functionality is solved within the Dem of the Master ECU.	

Table 8.65: Dem_SetIndicatorStatus

⌋

8.3.3.22 Dem_GetEventFreezeFrameData

[SWS_Dem_00558] ⌈

Service name:	Dem_GetEventFreezeFrameData
Syntax:	Std_ReturnType Dem_GetEventFreezeFrameData(Dem_EventIdType EventId, uint8 RecordNumber, boolean ReportTotalRecord, uint16 DataId, uint8* DestBuffer)

Service ID[hex]:	0x31	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	EventId	Identification of an event by assigned EventId.
	RecordNumber	This parameter is a unique identifier for a freeze frame record as defined in ISO15031-5 and ISO14229-1. **0xFF means most recent freeze frame record is returned.**
	ReportTotal Record	This parameter is obsolete and shall be set to FALSE. This function requests a single PID/DID.
	DataId	This parameter specifies the PID (ISO15031-5 mapped in UDS range 0xF400 - 0xF4FF) or DID (ISO14229-1) that shall be copied to the destination buffer.
Parameters (in-out):	None	
Parameters (out):	DestBuffer	This parameter contains a byte pointer that points to the buffer, to which the freeze frame data record shall be written to. The format is raw hexadecimal values and contains no header-information.
Return value:	Std_ReturnType	E_OK: Operation was successful DEM_E_NODATAAVAILABLE: The requested event data is not currently stored (but the request was valid) DEM_E_WRONG_RECORDNUMBER: The requested record number is not supported by the event DEM_E_WRONG_DIDNUMBER: The requested DID is not supported by the freeze frame
Description:	Gets the data of a freeze frame by event.	

Table 8.66: Dem_GetEventFreezeFrameData

⌋

8.3.3.23 Dem_GetEventExtendedDataRecord

[SWS_Dem_00557] ⌈

Service name:	Dem_GetEventExtendedDataRecord
Syntax:	Std_ReturnType Dem_GetEventExtendedDataRecord(Dem_EventIdType EventId, uint8 RecordNumber, uint8* DestBuffer)

Service ID[hex]:	0x30	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	EventId	Identification of an event by assigned EventId.
	RecordNumber	Identification of requested Extended data record. Valid values are between 0x01 and 0xEF as defined in ISO14229-1.
Parameters (in-out):	None	
Parameters (out):	DestBuffer	This parameter contains a byte pointer that points to the buffer, to which the extended data shall be written to. The format is raw hexadecimal values and contains no header-information.
Return value:	Std_ReturnType	E_OK: Operation was successful DEM_E_NODATAAVAILABLE: The requested event data is not currently stored (but the request was valid) DEM_E_WRONG_RECORDNUMBER: The requested record number is not supported by the event
Description:	Gets the data of an extended data record by event.	

Table 8.67: Dem_GetEventExtendedDataRecord

⌋

8.3.3.24 Dem_GetEventMemoryOverflow

[SWS_Dem_00559] ⌈

Service name:	Dem_GetEventMemoryOverflow	
Syntax:	Std_ReturnType Dem_GetEventMemoryOverflow(Dem_DTCOriginType DTCOrigin, boolean* OverflowIndication)	
Service ID[hex]:	0x32	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	DTCOrigin	If the Dem supports more than one event memory this parameter is used to select the source memory the overflow indication shall be read from.
Parameters (in-out):	None	

Parameters (out):	OverflowIndication	This parameter returns TRUE if the according event memory was overflowed, otherwise it returns FALSE.
Return value:	Std_ReturnType	E_OK: Operation was successful E_NOT_OK: Operation failed or is not supported
Description:	Gets the event memory overflow indication status.	

Table 8.68: Dem_GetEventMemoryOverflow

⌋(*SRS_Diag_04093*)

8.3.3.25 Dem_GetNumberOfEventMemoryEntries

[SWS_Dem_00652] ⌈

Service name:	Dem_GetNumberOfEventMemoryEntries	
Syntax:	Std_ReturnType Dem_GetNumberOfEventMemoryEntries(Dem_DTCOriginType DTCOrigin, uint8* NumberOfEventMemoryEntries)	
Service ID[hex]:	0x35	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	DTCOrigin	If the Dem supports more than one event memory this parameter is used to select the source memory the number of entries shall be read from.
Parameters (inout):	None	
Parameters (out):	NumberOfEventMemoryEntries	Number of entries currently stored in the requested event memory.
Return value:	Std_ReturnType	E_OK: Operation was successful E_NOT_OK: Operation failed
Description:	Returns the number of entries currently stored in the requested event memory.	

Table 8.69: Dem_GetNumberOfEventMemoryEntries

⌋(*SRS_Diag_04109*)

8.3.3.26 Dem_SetDTCSuppression

[SWS_Dem_01047] ⌈

Service name:	Dem_SetDTCSuppression	
Syntax:	Std_ReturnType Dem_SetDTCSuppression(uint32 DTC, Dem_DTCFormatType DTCFormat, boolean SuppressionStatus)	
Service ID[hex]:	0x33	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	DTC	Diagnostic Trouble Code
	DTCFormat	Defines the input-format of the provided DTC value.
	Suppression Status	This parameter specifies whether the respective DTC shall be disabled (TRUE) or enabled (FALSE).
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK (Operation was successful), E_NOT_OK (operation failed or event entry for this DTC still exists)
Description:	Set the suppression status of a specific DTC.	

Table 8.70: Dem_SetDTCSuppression

⌋

8.3.3.27 Dem_SetEventSuppression

[SWS_Dem_01048] ⌈

Service name:	Dem_SetEventSuppression	
Syntax:	Std_ReturnType Dem_SetEventSuppression(Dem_EventIdType EventId, boolean SuppressionStatus)	
Service ID[hex]:	0x36	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	EventId	Identification of an event by assigned EventId.
	Suppression Status	This parameter specifies whether the respective Event shall be disabled (TRUE) or enabled (FALSE).
Parameters (in-out):	None	

Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Operation was successful.
Description:	Set the suppression status of a specific Event.	

Table 8.71: Dem_SetEventSuppression

⌋

8.3.3.28 Dem_ClearDTC

[SWS_Dem_00665] ⌈

Service name:	Dem_ClearDTC	
Syntax:	Dem_ReturnClearDTCType Dem_ClearDTC(uint32 DTC, Dem_DTCFormatType DTCFormat, Dem_DTCOriginType DTCOrigin)	
Service ID[hex]:	0x23	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant	
Parameters (in):	DTC	Defines the DTC in respective format, that shall be cleared from the event memory. If the DTC fits to a DTC group number, all DTCs of the group shall be cleared.
	DTCFormat DTCOrigin	Defines the input-format of the provided DTC value. If the Dem supports more than one event memory this parameter is used to select the source memory the DTCs shall be read from.
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Dem_Return ClearDTCType	Status of the operation of type Dem_ReturnClearDTCType.
Description:	Clears single DTCs, as well as groups of DTCs. This API is intended for complex device driver. It can only be used through the RTE (due to work-around described below SWS_Dem_00659), and therefore no declaration is exported via Dem.h.	

Table 8.72: Dem_ClearDTC

⌋(*SRS_Diag_04122*)

8.3.4 Interface Dcm <=> Dem

8.3.4.1 Access DTCs and Status Information

8.3.4.1.1 Dem_DcmGetTranslationType

[SWS_Dem_00230] ⌈

Service name:	Dem_DcmGetTranslationType	
Syntax:	Dem_DTCTranslationFormatType Dem_DcmGetTranslationType(void)	
Service ID[hex]:	0x3c	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Dem_DTCTrans-lationFormat Type	Returns the configured DTC translation format. A combination of different DTC formats is not possible.
Description:	Gets the supported DTC formats of the ECU. The supported formats are configured via DemTypeOfDTCSupported.	

Table 8.73: Dem_DcmGetTranslationType

⌋

8.3.4.1.2 Dem_DcmGetDTCStatusAvailabilityMask

[SWS_Dem_00213] ⌈

Service name:	Dem_DcmGetDTCStatusAvailabilityMask
Syntax:	Std_ReturnType Dem_DcmGetDTCStatusAvailabilityMask(Dem_UdsStatusByteType* DTCStatusMask)
Service ID[hex]:	0x16
Sync/Async:	Synchronous

Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (in-out):	None	
Parameters (out):	DTCStatusMask	The value DTCStatusMask indicates the supported DTC status bits from the Dem. All supported information is indicated by setting the corresponding status bit to 1. See ISO14229-1.
Return value:	Std_ReturnType	E_OK: get of DTC status mask was successful E_NOT_OK: get of DTC status mask failed
Description:	Gets the DTC Status availability mask.	

Table 8.74: Dem_DcmGetDTCStatusAvailabilityMask

⌋

8.3.4.1.3 Dem_DcmGetStatusOfDTC

[SWS_Dem_00212] ⌈

Service name:	Dem_DcmGetStatusOfDTC	
Syntax:	Dem_ReturnGetStatusOfDTCType Dem_DcmGetStatusOfDTC(uint32 DTC, Dem_DTCOriginType DTCOrigin, uint8* DTCStatus)	
Service ID[hex]:	0x15	
Sync/Async:	Synchronous or Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	DTC	Diagnostic Trouble Code in UDS format.
	DTCOrigin	If the Dem supports more than one event memory this parameter is used to select the source memory the DTCs shall be read from.
Parameters (in-out):	None	
Parameters (out):	DTCStatus	This parameter receives the status information of the requested DTC. If the return value of the function call is other than DEM_STATUS_OK this parameter does not contain valid data. 0x00...0xFF match DTCStatusMask as defined in ISO14229-1

Return value:	Dem_ReturnGetStatusOfDTCType	Status of the operation of type Dem_ReturnGetStatusOfDTCType.
Description:	Gets the status of a DTC. For large configurations and DTC-calibration, the interface behavior can be asynchronous (splitting the DTC-search into segments). The DTCs of OBD Events Suppression shall be reported as DEM_STATUS_WRONG_DTC.	

Table 8.75: Dem_DcmGetStatusOfDTC

⌋(*SRS_Diag_04066*)

8.3.4.1.4 Dem_DcmGetSeverityOfDTC

[SWS_Dem_00232] ⌈

Service name:	Dem_DcmGetSeverityOfDTC	
Syntax:	Dem_ReturnGetSeverityOfDTCType Dem_DcmGetSeverityOfDTC(uint32 DTC, Dem_DTCSeverityType* DTCSeverity)	
Service ID[hex]:	0x0e	
Sync/Async:	Synchronous or Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	DTC	Diagnostic Trouble Code in UDS format.
Parameters (in-out):	None	
Parameters (out):	DTCSeverity	This parameter contains the DTCSeverity according to ISO 14229-1.
Return value:	Dem_ReturnGetSeverityOfDTCType	Status of the operation of type Dem_ReturnGetSeverityOfDTCType.
Description:	Gets the severity of the requested DTC. For large configurations and DTC-calibration, the interface behavior can be asynchronous (splitting the DTC-search into segments).	

Table 8.76: Dem_DcmGetSeverityOfDTC

⌋

8.3.4.1.5 Dem_DcmGetFunctionalUnitOfDTC

[SWS_Dem_00594] ⌈

Service name:	Dem_DcmGetFunctionalUnitOfDTC	
Syntax:	Dem_ReturnGetFunctionalUnitOfDTCType Dem_DcmGetFunctionalUnitOfDTC(uint32 DTC, uint8* DTCFunctionalUnit)	
Service ID[hex]:	0x34	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	DTC	Diagnostic Trouble Code in UDS format.
Parameters (in-out):	None	
Parameters (out):	DTCFunctional Unit	Functional unit value of this DTC
Return value:	Dem_ReturnGet FunctionalUnitOf DTCType	Status of the operation of type Dem_ReturnGetFunctionalUnitOfDTCType.
Description:	Gets the functional unit of the requested DTC.	

Table 8.77: Dem_DcmGetFunctionalUnitOfDTC

⌋

8.3.4.1.6 Dem_DcmSetDTCFilter

[SWS_Dem_00208] ⌈

Service name:	Dem_DcmSetDTCFilter
Syntax:	Dem_ReturnSetFilterType Dem_DcmSetDTCFilter(Dem_UdsStatusByteType DTCStatusMask, Dem_DTCKindType DTCKind, Dem_DTCFormatType DTCFormat, Dem_DTCOriginType DTCOrigin, boolean FilterWithSeverity, Dem_DTCSeverityType DTCSeverityMask, boolean FilterForFaultDetectionCounter)
Service ID[hex]:	0x13
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant

Parameters (in):	DTCStatusMask	Status-byte mask for DTC status-byte filtering Values: 0x00: Autosar-specific value to deactivate the status-byte filtering (different meaning than in ISO 14229-1) to report all supported DTCs (used for service 0x19 subfunctions 0x0A/0x15) 0x01..0xFF: Status-byte mask according to ISO 14229-1 DTCStatusMask (handed over by Dcm from service request directly) to filter for DTCs with at least one status bit set matching this status-byte mask
	DTCKind	Defines the functional group of DTCs to be reported (e.g. all DTC, OBD-relevant DTC). If passed value does not fit to Configuration, the DET error DEM_E_WRONG_CONFIGURATION shall be reported, e.g if DTCKind "DEM_DTC_KIND_EMISSION_REL_DTCS" is passed, but no emission related dtcs are configured.
	DTCFormat	Defines the output-format of the requested DTC values for the sub-sequent API calls. If passed value does not fit to Configuration, the DET error DEM_E_WRONG_CONFIGURATION shall be reported, e.g. if DTCFormat "DEM_DTC_FORMAT_OBD" is passed, but OBD is not supported per configuration.
	DTCOrigin	If the Dem supports more than one event memory this parameter is used to select the source memory the DTCs shall be read from. If passed value does not fit to Configuration, the DET error DEM_E_WRONG_CONFIGURATION shall be reported, e.g. if DTCOrigin "DEM_DTC_ORIGIN_SECONDARY_MEMORY" is passed, but no secondary memory is configured.
	FilterWithSeverity	This flag defines whether severity information (ref. to parameter below) shall be used for filtering. This is to allow for coexistence of DTCs with and without severity information.
	DTCSeverityMask	This parameter contains the DTCSeverityMask according to ISO14229-1 (see for example Service 0x19, subfunction 0x08). If the value is invalid the function returns DEM_WRONG_FILTER.

	FilterForFaultDetectionCounter	This flag defines whether the fault detection counter information shall be used for filtering. This is to allow for coexistence of DTCs with and without fault detection counter information. If fault detection counter information is filter criteria, only those DTCs with a fault detection counter value between 1 and 0x7E shall be reported. Remark: If the event does not use the debouncing inside Dem, then the Dem must request this information via GetFaultDetectionCounter.
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Dem_ReturnSetFilterType	Status of the operation to (re-)set a DTC filter.
Description:	Sets the DTC Filter. The server shall perform a bit-wise logical AND-ing operation between the parameter DTCStatusMask and the current DTC status in the server. In addition to the DTCStatusAvailabilityMask, the server shall return all DTCs for which the result of the AND-ing operation is non-zero [i.e. (statusOfDTC & DTCStatusMask) != 0]. The server shall process only the DTC Status bits that it is supporting. OBD Events Suppression shall be ignored for this computation. If no DTCs within the server match the masking criteria specified in the client's request, no DTC or status information shall be provided following the DTCStatusAvailabilityMask byte in the positive response message (((statusOfDTC & DTCStatusMask) != 0) && ((severity & DTCSeverityMask) != 0)) == TRUE	

Table 8.78: Dem_DcmSetDTCFilter

⌋

8.3.4.1.7 Dem_DcmGetNumberOfFilteredDTC

[SWS_Dem_00214] ⌈

Service name:	Dem_DcmGetNumberOfFilteredDTC

Syntax:	Dem_ReturnGetNumberOfFilteredDTCType Dem_DcmGetNumberOfFilteredDTC(uint16* NumberOfFilteredDTC)	
Service ID[hex]:	0x17	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (in-out):	None	
Parameters (out):	NumberOfFilteredDTC	The number of DTCs matching the defined status mask.
Return value:	Dem_ReturnGetNumberOfFilteredDTCType	Status of the operation to retrieve a number of DTC from the Dem
Description:	Gets the number of a filtered DTC.	

Table 8.79: Dem_DcmGetNumberOfFilteredDTC

⌋

8.3.4.1.8 Dem_DcmGetNextFilteredDTC

[SWS_Dem_00215] ⌈

Service name:	Dem_DcmGetNextFilteredDTC	
Syntax:	Dem_ReturnGetNextFilteredElementType Dem_DcmGetNextFilteredDTC(uint32* DTC, Dem_UdsStatusByteType* DTCStatus)	
Service ID[hex]:	0x18	
Sync/Async:	Synchronous or Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (in-out):	None	
Parameters (out):	DTC	Receives the DTC value in respective format of the filter returned by this function. If the return value of the function is other than DEM_FILTERED_OK this parameter does not contain valid data.

	DTCStatus	This parameter receives the status information of the requested DTC. It follows the format as defined in ISO14229-1 If the return value of the function call is other than DEM_FILTERED_OK this parameter does not contain valid data.
Return value:	Dem_Return GetNextFiltered ElementType	Status of the operation to retrieve a DTC from the Dem. The value DEM_FILTERED_PENDING is not always allowed (refer to SWS_DEM_00653).
Description:	Gets the next filtered DTC matching the filter criteria. For UDS services, the interface has an asynchronous behavior, because a large number of DTCs has to be processed.	

Table 8.80: Dem_DcmGetNextFilteredDTC

⌋

8.3.4.1.9 Dem_DcmGetNextFilteredDTCAndFDC

[SWS_Dem_00227] ⌈

Service name:	Dem_DcmGetNextFilteredDTCAndFDC	
Syntax:	Dem_ReturnGetNextFilteredElementType Dem_DcmGetNextFilteredDTCAndFDC(uint32* DTC, sint8* DTCFaultDetectionCounter)	
Service ID[hex]:	0x3b	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (in-out):	None	
Parameters (out):	DTC	Receives the DTC value in respective format of the filter returned by this function. If the return value of the function is other than DEM_FILTERED_OK this parameter does not contain valid data.
	DTCFaultDetec-tionCounter	This parameter receives the Fault Detection Counter information of the requested DTC. If the return value of the function call is other than DEM_FILTERED_OK this parameter does not contain valid data. -128dec...127dec PASSED...FAILED according to ISO 14229-1

Return value:	Dem_Return GetNextFiltered ElementType	Status of the operation to retrieve a DTC from the Dem.
Description:	Gets the next filtered DTC and its associated Fault Detection Counter (FDC) matching the filter criteria. The interface has an asynchronous behavior, because a large number of DTCs has to be processed and the FDC might be received asynchronously from a SW-C, too.	

Table 8.81: Dem_DcmGetNextFilteredDTCAndFDC

⌋

8.3.4.1.10 Dem_DcmGetNextFilteredDTCAndSeverity

[SWS_Dem_00281] ⌈

Service name:	Dem_DcmGetNextFilteredDTCAndSeverity	
Syntax:	Dem_ReturnGetNextFilteredElementType Dem_DcmGetNextFilteredDTCAndSeverity(uint32* DTC, Dem_UdsStatusByteType* DTCStatus, Dem_DTCSeverityType* DTCSeverity, uint8* DTCFunctionalUnit)	
Service ID[hex]:	0x3d	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	DTC	Receives the DTC value in respective format of the filter returned by this function. If the return value of the function is other than DEM_FILTERED_OK this parameter does not contain valid data.
	DTCStatus	This parameter receives the status information of the requested DTC. It follows the format as defined in ISO14229-1 If the return value of the function call is other than DEM_FILTERED_OK this parameter does not contain valid data.
	DTCSeverity	Receives the severity value returned by the function. If the return value of the function is other than DEM_FILTERED_OK this parameter does not contain valid data.

	DTCFunctional Unit	Receives the functional unit value returned by the function. If the return value of the function is other than DEM_FILTERED_OK this parameter does not contain valid data.
Return value:	Dem_Return GetNextFiltered ElementType	Status of the operation to retrieve a DTC from the Dem.
Description:	Gets the next filtered DTC and its associated Severity matching the filter criteria. The interface has an asynchronous behavior, because a large number of DTCs has to be processed.	

Table 8.82: Dem_DcmGetNextFilteredDTCAndSeverity

⌋

8.3.4.1.11 Dem_DcmSetFreezeFrameRecordFilter

[SWS_Dem_00209] ⌈

Service name:	Dem_DcmSetFreezeFrameRecordFilter	
Syntax:	Dem_ReturnSetFilterType Dem_DcmSetFreezeFrameRecordFilter(Dem_DTCFormatType DTCFormat, uint16* NumberOfFilteredRecords)	
Service ID[hex]:	0x3f	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	DTCFormat	Defines the output-format of the requested DTC values for the sub-sequent API calls.
Parameters (inout):	None	
Parameters (out):	NumberOfFilteredRecords	Number of freeze frame records currently stored in the event memory.
Return value:	Dem_ReturnSetFilterType	Status of the operation to (re-)set a freeze frame record filter.
Description:	Sets a freeze frame record filter.	

Table 8.83: Dem_DcmSetFreezeFrameRecordFilter

⌋

8.3.4.1.12 Dem_DcmGetNextFilteredRecord

[SWS_Dem_00224] ⌈

Service name:	Dem_DcmGetNextFilteredRecord	
Syntax:	Dem_ReturnGetNextFilteredElementType Dem_DcmGetNextFilteredRecord(uint32* DTC, uint8* RecordNumber)	
Service ID[hex]:	0x3a	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (in-out):	None	
Parameters (out):	DTC	Receives the DTC value in respective format of the filter returned by this function. If the return value of the function is other than DEM_FILTERED_OK this parameter does not contain valid data.
	RecordNumber	Freeze frame record number of the reported DTC (relative addressing). If the return value of the function is other than DEM_FILTERED_OK this parameter does not contain valid data.
Return value:	Dem_Return GetNextFiltered ElementType	Status of the operation to retrieve a DTC and its associated snapshot record number from the Dem.
Description:	Gets the next freeze frame record number and its associated DTC stored in the event memory. The interface has an asynchronous behavior, because NvRAM access might be required.	

Table 8.84: Dem_DcmGetNextFilteredRecord

⌋

8.3.4.1.13 Dem_DcmGetDTCByOccurrenceTime

[SWS_Dem_00218] ⌈

Service name:	Dem_DcmGetDTCByOccurrenceTime

Syntax:	Dem_ReturnGetDTCByOccurrenceTimeType Dem_DcmGetDTCByOccurrenceTime(Dem_DTCRequestType DTCRequest, uint32* DTC)	
Service ID[hex]:	0x19	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	DTCRequest	This parameter defines the request type of the DTC.
Parameters (in-out):	None	
Parameters (out):	DTC	Receives the DTC value in UDS format returned by the function. If the return value of the function is other than DEM_OCCURR_OK this parameter does not contain valid data.
Return value:	Dem_Return GetDTCByOc-currenceTime Type	Status of the operation of type Dem_ReturnGetDTCByOccurrenceTimeType.
Description:	Gets the DTC by occurrence time. There is no explicit parameter for the DTC-origin as the origin always is DEM_DTC_ORIGIN_PRIMARY_MEMORY.	

Table 8.85: Dem_DcmGetDTCByOccurrenceTime

⌋

8.3.4.1.14 Dem_DcmControlDTCStatusChangedNotification

[SWS_Dem_00846] ⌈

Service name:	Dem_DcmControlDTCStatusChangedNotification	
Syntax:	void Dem_DcmControlDTCStatusChangedNotification(boolean TriggerNotification)	
Service ID[hex]:	0xb0	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	TriggerNotifica-tion	This parameter specifies whether the triggering of the notification shall be enabled (TRUE) or disabled (FALSE).
Parameters (in-out):	None	

Parameters (out):	None
Return value:	None
Description:	Controls the triggering of Dcm_DemTriggerOnDTCStatus.

Table 8.86: Dem_DcmControlDTCStatusCN

⌋

8.3.4.2 Access extended data records and FreezeFrame data

8.3.4.2.1 Dem_DcmDisableDTCRecordUpdate

[SWS_Dem_00233] ⌈

Service name:	Dem_DcmDisableDTCRecordUpdate	
Syntax:	Dem_ReturnDisableDTCRecordUpdateType Dem_DcmDisableDTCRecordUpdate(uint32 DTC, Dem_DTCOriginType DTCOrigin)	
Service ID[hex]:	0x1a	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	DTC	Selects the DTC in UDS format, for which DTC record update shall be disabled.
	DTCOrigin	If the Dem supports more than one event memory, this parameter is used to select the source memory for which DTC record update shall be disabled.
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Dem_ReturnDisableDTCRecordUpdateType	Status of the operation to disable the event memory update of a specific DTC.
Description:	Disables the event memory update of a specific DTC (only one at one time).	

Table 8.87: Dem_DcmDisableDTCRecordUpdate

⌋

8.3.4.2.2 Dem_DcmEnableDTCRecordUpdate

[SWS_Dem_00234] ⌈

Service name:	Dem_DcmEnableDTCRecordUpdate	
Syntax:	Std_ReturnType Dem_DcmEnableDTCRecordUpdate(void)	
Service ID[hex]:	0x1b	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	Always E_OK is returned.
Description:	Enables the event memory update of the DTC disabled by Dem_DcmDisableDTCRecordUpdate() before.	

Table 8.88: Dem_DcmEnableDTCRecordUpdate

⌋

8.3.4.2.3 Dem_DcmGetFreezeFrameDataByDTC

[SWS_Dem_00236] ⌈

Service name:	Dem_DcmGetFreezeFrameDataByDTC	
Syntax:	Dem_ReturnGetFreezeFrameDataByDTCType Dem_DcmGetFreezeFrameDataByDTC(uint32 DTC, Dem_DTCOriginType DTCOrigin, uint8 RecordNumber, uint8* DestBuffer, uint16* BufSize)	
Service ID[hex]:	0x1d	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	DTC	Diagnostic Trouble Code in UDS format.
	DTCOrigin	If the Dem supports more than one event memory, this parameter is used to select the source memory the DTCs shall be read from.

	RecordNumber	This parameter is a unique identifier for a freeze frame record as defined in ISO 15031-5 and ISO 14229-1. This record number is unique per DTC (relative addressing). The value 0xFF is not allowed. The value 0x00 indicates the DTC-specific OBD freeze frame.
Parameters (in-out):	BufSize	When the function is called this parameter contains the maximum number of data bytes that can be written to the buffer. The function returns the actual number of written data bytes in this parameter.
Parameters (out):	DestBuffer	This parameter contains a byte pointer that points to the buffer, to which the freeze frame data record shall be written to. The format is: {RecordNumber, NumOfDIDs, DID[1], data[1], ..., DID[N], data[N]}
Return value:	Dem_ReturnGetFreezeFrameDataByDTCType	Status of the operation to retrieve freeze frame data by DTC.
Description:	Gets freeze frame data by DTC. The function stores the data in the provided DestBuffer.	

Table 8.89: Dem_DcmGetFreezeFrameDataByDTC

⌋(*SRS_Diag_04066*)

8.3.4.2.4 Dem_DcmGetSizeOfFreezeFrameByDTC

[SWS_Dem_00238] ⌈

Service name:	Dem_DcmGetSizeOfFreezeFrameByDTC	
Syntax:	Dem_ReturnGetSizeOfDataByDTCType Dem_DcmGetSizeOfFreezeFrameByDTC(uint32 DTC, Dem_DTCOriginType DTCOrigin, uint8 RecordNumber, uint16* SizeOfFreezeFrame)	
Service ID[hex]:	0x1f	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	DTC	Diagnostic Trouble Code in UDS format.
	DTCOrigin	If the Dem supports more than one event memory, this parameter is used to select the source memory the DTCs shall be read from.

	RecordNumber	This parameter is a unique identifier for a freeze frame record as defined in ISO 15031-5 and ISO 14229-1. This record number is unique per DTC (relative addressing). The value 0xFF is explicitly allowed to request the overall size.
Parameters (in-out):	None	
Parameters (out):	SizeOfFreeze Frame	Number of bytes in the requested freeze frame record.
Return value:	Dem_ReturnGet SizeOfDataByD TCType	Status of the operation to retrieve the size of freeze frame data.
Description:	Gets the size of freeze frame data by DTC.	

Table 8.90: Dem_DcmGetSizeOfFreezeFrameByDTC

⌋(*SRS_Diag_04066*)

8.3.4.2.5 Dem_DcmGetExtendedDataRecordByDTC

[SWS_Dem_00239] ⌈

Service name:	Dem_DcmGetExtendedDataRecordByDTC	
Syntax:	Dem_ReturnGetExtendedDataRecordByDTCType Dem_DcmGetExtendedDataRecordByDTC(uint32 DTC, Dem_DTCOriginType DTCOrigin, uint8 ExtendedDataNumber, uint8* DestBuffer, uint16* BufSize)	
Service ID[hex]:	0x20	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	DTC	Diagnostic Trouble Code in UDS format.
	DTCOrigin	If the Dem supports more than one event memory, this parameter is used to select the source memory the DTCs shall be read from.
	ExtendedData Number	Identification/Number of requested extended data record. The values 0xFE and 0xFF are not allowed.
Parameters (in-out):	BufSize	When the function is called this parameter contains the maximum number of data bytes that can be written to the buffer. The function returns the actual number of written data bytes in this parameter.

Parameters (out):	DestBuffer	This parameter contains a byte pointer that points to the buffer, to which the extended data record shall be written to. The format is raw hexadecimal values and contains no header-information.
Return value:	Dem_ReturnGetExtendedDataRecordByDTCType	Status of the operation to retrieve extended data by DTC.
Description:	Gets extended data by DTC. The function stores the data in the provided DestBuffer.	

Table 8.91: Dem_DcmGetExtendedDataRecordByDTC

⌋(*SRS_Diag_04066*)

8.3.4.2.6 Dem_DcmGetSizeOfExtendedDataRecordByDTC

[SWS_Dem_00240] ⌈

Service name:	Dem_DcmGetSizeOfExtendedDataRecordByDTC	
Syntax:	Dem_ReturnGetSizeOfDataByDTCType Dem_DcmGetSizeOfExtendedDataRecordByDTC(uint32 DTC, Dem_DTCOriginType DTCOrigin, uint8 ExtendedDataNumber, uint16* SizeOfExtendedDataRecord)	
Service ID[hex]:	0x21	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	DTC	Diagnostic Trouble Code in UDS format.
	DTCOrigin	If the Dem supports more than one event memory, this parameter is used to select the source memory the DTCs shall be read from.
	ExtendedDataNumber	Identification/Number of requested extended data record. Additionally the values 0xFE and 0xFF are explicitly allowed to request the overall size of all OBD records / all records.
Parameters (in-out):	None	
Parameters (out):	SizeOfExtendedDataRecord	Size of the requested extended data record(s) including record number size

Return value:	Dem_ReturnGetSizeOfDataByDTCType	Status of the operation to retrieve the size of extended data.
Description:	Gets the size of extended data by DTC.	

Table 8.92: Dem_DcmGetSizeOfExtendedDataRecordByDTC

⌋(*SRS_Diag_04066*)

8.3.4.3 DTC storage

8.3.4.3.1 Dem_DcmClearDTC

[SWS_Dem_00241] ⌈

Service name:	Dem_ClearDTC	
Syntax:	Dem_ReturnClearDTCType Dem_ClearDTC(uint32 DTC, Dem_DTCFormatType DTCFormat, Dem_DTCOriginType DTCOrigin)	
Service ID[hex]:	0x23	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant	
Parameters (in):	DTC	Defines the DTC in respective format, that shall be cleared from the event memory. If the DTC fits to a DTC group number, all DTCs of the group shall be cleared.
	DTCFormat DTCOrigin	Defines the input-format of the provided DTC value. If the Dem supports more than one event memory this parameter is used to select the source memory the DTCs shall be read from.
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Dem_ReturnClearDTCType	Status of the operation of type Dem_ReturnClearDTCType.
Description:	Clears single DTCs, as well as groups of DTCs. This API is intended for complex device driver. It can only be used through the RTE (due to work-around described below SWS_Dem_00659), and therefore no declaration is exported via Dem.h.	

Table 8.93: Dem_ClearDTC

⌋(*SRS_Diag_04066*)

8.3.4.3.2 Dem_DcmDisableDTCSetting

[SWS_Dem_00242] ⌈

Service name:	Dem_DcmDisableDTCSetting	
Syntax:	Dem_ReturnControlDTCSettingType Dem_DcmDisableDTCSetting(uint32 DTCGroup, Dem_DTCKindType DTCKind)	
Service ID[hex]:	0x24	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	DTCGroup	Defines the group of DTC that shall be disabled to store in event memory.
	DTCKind	This parameter defines the requested DTC kind, either only OBD-relevant DTCs or all DTCs
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Dem_ReturnControlDTCSettingType	Returns status of the operation
Description:	Disables the DTC setting for a DTC group.	

Table 8.94: Dem_DcmDisableDTCSetting

⌋

8.3.4.3.3 Dem_DcmEnableDTCSetting

[SWS_Dem_00243] ⌈

Service name:	Dem_DcmEnableDTCSetting

Syntax:	Dem_ReturnControlDTCSettingType Dem_DcmEnableDTCSetting(uint32 DTCGroup, Dem_DTCKindType DTCKind)	
Service ID[hex]:	0x25	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	DTCGroup	Defines the group of DTC that shall be enabled to store in event memory.
	DTCKind	This parameter defines the requested DTC kind, either only OBD-relevant DTCs or all DTCs
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Dem_Return ControlDTC SettingType	Returns the status of the operation
Description:	Enables the DTC setting for a DTC group. This API is intended for the Dcm. It can only be used through the RTE (due to work-around described below SWS_Dem_00035), and therefore no declaration is exported via Dem_Dcm.h.	

Table 8.95: Dem_DcmEnableDTCSetting

⌋

8.3.5 OBD-specific Dcm <=> Dem Interfaces

8.3.5.1 Dem_DcmGetInfoTypeValue08

[SWS_Dem_00316] ⌈

Service name:	Dem_DcmGetInfoTypeValue08
Syntax:	Std_ReturnType Dem_DcmGetInfoTypeValue08(Dcm_OpStatusType OpStatus, uint8* lumprdata08)
Service ID[hex]:	0x6b
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant

Parameters (in):	OpStatus	Only DCM_INITIAL will appear, because this API behaves synchronous.
Parameters (in-out):	None	
Parameters (out):	Iumprdata08	Buffer containing the contents of InfoType $08. The buffer is provided by the Dcm.
Return value:	Std_ReturnType	Always E_OK is returned.
Description:	Service is used for requesting IUMPR data according to InfoType $08. This interface is derived from the prototype <Module>_GetInfotypeValueData() defined by the Dcm. Therefore Dcm_OpStatusType and Std_ReturnType are contained. API is needed in OBD-relevant ECUs only	

Table 8.96: Dem_DcmGetInfoTypeValue08

⌋

8.3.5.2 Dem_DcmGetInfoTypeValue0B

[SWS_Dem_00317] ⌈

Service name:	Dem_DcmGetInfoTypeValue0B	
Syntax:	Std_ReturnType Dem_DcmGetInfoTypeValue0B(Dcm_OpStatusType OpStatus, uint8* Iumprdata0B)	
Service ID[hex]:	0x6c	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	OpStatus	Only DCM_INITIAL will appear, because this API behaves synchronous.
Parameters (in-out):	None	
Parameters (out):	Iumprdata0B	Buffer containing the contents of InfoType $0B. The buffer is provided by the Dcm.
Return value:	Std_ReturnType	Always E_OK is returned.
Description:	Service is used for requesting IUMPR data according to InfoType $0B. This interface is derived from the prototype <Module>_GetInfotypeValueData() defined by the Dcm. Therefore Dcm_OpStatusType and Std_ReturnType are contained. API is needed in OBD-relevant ECUs only	

Table 8.97: Dem_DcmGetInfoTypeValue0B

⌋

8.3.5.3 Dem_DcmReadDataOfPID01

[SWS_Dem_00318] ⌈

Service name:	Dem_DcmReadDataOfPID01	
Syntax:	Std_ReturnType Dem_DcmReadDataOfPID01(uint8* PID01value)	
Service ID[hex]:	0x61	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (in-out):	None	
Parameters (out):	PID01value	Buffer containing the contents of PID $01 computed by the Dem. The buffer is provided by the Dcm with the appropriate size, i.e. during configuration, the Dcm identifies the required size from the largest PID in order to configure a PIDBuffer.
Return value:	Std_ReturnType	Always E_OK is returned, as E_NOT_OK will never appear.
Description:	Service to report the value of PID $01 computed by the Dem. API is needed in OBD-relevant ECUs only	

Table 8.98: Dem_DcmReadDataOfPID01

⌋

8.3.5.4 Dem_DcmReadDataOfPID1C

[SWS_Dem_00325] ⌈

Service name:	Dem_DcmReadDataOfPID1C
Syntax:	Std_ReturnType Dem_DcmReadDataOfPID1C(uint8* PID1Cvalue)
Service ID[hex]:	0x63
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None

Parameters (in-out):	None	
Parameters (out):	PID1Cvalue	Buffer containing the contents of PID $1C computed by the Dem. The value of PID$1C is configuration within DemOBDCompliancy. The buffer is provided by the Dcm with the appropriate size, i.e. during configuration, the Dcm identifies the required size from the largest PID in order to configure a PID-Buffer.
Return value:	Std_ReturnType	Always E_OK is returned, as E_NOT_OK will never appear.
Description:	Service to report the value of PID $1C computed by the Dem. API is needed in OBD-relevant ECUs only	

Table 8.99: Dem_DcmReadDataOfPID1C

⌋

8.3.5.5 Dem_DcmReadDataOfPID21

[SWS_Dem_00319] ⌈

Service name:	Dem_DcmReadDataOfPID21	
Syntax:	Std_ReturnType Dem_DcmReadDataOfPID21(uint8* PID21value)	
Service ID[hex]:	0x64	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (in-out):	None	
Parameters (out):	PID21value	Buffer containing the contents of PID $21 computed by the Dem. The buffer is provided by the Dcm with the appropriate size, i.e. during configuration, the Dcm identifies the required size from the largest PID in order to configure a PIDBuffer.
Return value:	Std_ReturnType	Always E_OK is returned, as E_NOT_OK will never appear.
Description:	Service to report the value of PID $21 computed by the Dem. API is needed in OBD-relevant ECUs only	

Table 8.100: Dem_DcmReadDataOfPID21

⌋

8.3.5.6 Dem_DcmReadDataOfPID30

[SWS_Dem_00320] ⌈

Service name:	Dem_DcmReadDataOfPID30	
Syntax:	Std_ReturnType Dem_DcmReadDataOfPID30(uint8* PID30value)	
Service ID[hex]:	0x65	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (in-out):	None	
Parameters (out):	PID30value	Buffer containing the contents of PID $30 computed by the Dem. The buffer is provided by the Dcm with the appropriate size, i.e. during configuration, the Dcm identifies the required size from the largest PID in order to configure a PIDBuffer.
Return value:	Std_ReturnType	Always E_OK is returned, as E_NOT_OK will never appear.
Description:	Service to report the value of PID $30 computed by the Dem. API is needed in OBD-relevant ECUs only	

Table 8.101: Dem_DcmReadDataOfPID30

⌋

8.3.5.7 Dem_DcmReadDataOfPID31

[SWS_Dem_00321] ⌈

Service name:	Dem_DcmReadDataOfPID31
Syntax:	Std_ReturnType Dem_DcmReadDataOfPID31(uint8* PID31value)
Service ID[hex]:	0x66
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None

Parameters (in-out):	None	
Parameters (out):	PID31value	Buffer containing the contents of PID $31 computed by the Dem. The buffer is provided by the Dcm with the appropriate size, i.e. during configuration, the Dcm identifies the required size from the largest PID in order to configure a PIDBuffer.
Return value:	Std_ReturnType	Always E_OK is returned, as E_NOT_OK will never appear.
Description:	Service to report the value of PID $31 computed by the Dem. API is needed in OBD-relevant ECUs only	

Table 8.102: Dem_DcmReadDataOfPID31

⌋

8.3.5.8 Dem_DcmReadDataOfPID41

[SWS_Dem_00322] ⌈

Service name:	Dem_DcmReadDataOfPID41	
Syntax:	Std_ReturnType Dem_DcmReadDataOfPID41(uint8* PID41value)	
Service ID[hex]:	0x67	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (in-out):	None	
Parameters (out):	PID41value	Buffer containing the contents of PID $41 computed by the Dem. The buffer is provided by the Dcm with the appropriate size, i.e. during configuration, the Dcm identifies the required size from the largest PID in order to configure a PIDBuffer.
Return value:	Std_ReturnType	Always E_OK is returned, as E_NOT_OK will never appear.
Description:	Service to report the value of PID $41 computed by the Dem. API is needed in OBD-relevant ECUs only	

Table 8.103: Dem_DcmReadDataOfPID41

⌋

8.3.5.9 Dem_DcmReadDataOfPID4D

[SWS_Dem_00323] ⌈

Service name:	Dem_DcmReadDataOfPID4D	
Syntax:	Std_ReturnType Dem_DcmReadDataOfPID4D(uint8* PID4Dvalue)	
Service ID[hex]:	0x68	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (in-out):	None	
Parameters (out):	PID4Dvalue	Buffer containing the contents of PID $4D computed by the Dem. The buffer is provided by the Dcm with the appropriate size, i.e. during configuration, the Dcm identifies the required size from the largest PID in order to configure a PIDBuffer.
Return value:	Std_ReturnType	Always E_OK is returned, as E_NOT_OK will never appear.
Description:	Service to report the value of PID $4D computed by the Dem. API is needed in OBD-relevant ECUs only	

Table 8.104: Dem_DcmReadDataOfPID4D

⌋

8.3.5.10 Dem_DcmReadDataOfPID4E

[SWS_Dem_00324] ⌈

Service name:	Dem_DcmReadDataOfPID4E
Syntax:	Std_ReturnType Dem_DcmReadDataOfPID4E(uint8* PID4Evalue)
Service ID[hex]:	0x69
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None
Parameters (in-out):	None

Parameters (out):	PID4Evalue	Buffer containing the contents of PID $4E computed by the Dem. The buffer is provided by the Dcm with the appropriate size, i.e. during configuration, the Dcm identifies the required size from the largest PID in order to configure a PIDBuffer.
Return value:	Std_ReturnType	Always E_OK is returned, as E_NOT_OK will never appear.
Description:	Service to report the value of PID $4E computed by the Dem. API is needed in OBD-relevant ECUs only	

Table 8.105: Dem_DcmReadDataOfPID4E

⌋

8.3.5.11 Dem_DcmGetOBDFreezeFrameData

[SWS_Dem_00235] ⌈

Service name:	Dem_DcmGetOBDFreezeFrameData	
Syntax:	Std_ReturnType Dem_DcmGetOBDFreezeFrameData(uint32* DTC, uint8* DestBuffer, uint16* BufSize)	
Service ID[hex]:	0x1c	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (in-out):	BufSize	When the function is called this parameter contains the maximum number of data bytes that can be written to the buffer. The function returns the actual number of written data bytes in this parameter.
Parameters (out):	DTC	Receives the DTC value in UDS format returned by this function. If the return value of the function is other than E_OK this parameter does not contain valid data.
	DestBuffer	This parameter contains a byte pointer that points to the buffer, to which the freeze frame data record shall be written to. The format is: {NumOfDIDs, DID[1], data[1], ..., DID[N], data[N]}.
Return value:	Std_ReturnType	E_OK: DTC and OBD freeze frame data successfully reported E_NOT_OK: No DTC and OBD freeze frame data available

Description:	Gets the most important DTC and its OBD freeze frame (which caused MIL on) for the output on UDS (service 0x19). The function stores the freeze frame data in the provided DestBuffer. API is needed in OBD-relevant ECUs only

Table 8.106: Dem_DcmGetOBDFreezeFrameData

⌋

8.3.5.12 Dem_DcmReadDataOfOBDFreezeFrame

[SWS_Dem_00327] ⌈

Service name:	Dem_DcmReadDataOfOBDFreezeFrame	
Syntax:	Std_ReturnType Dem_DcmReadDataOfOBDFreezeFrame(uint8 PID, uint8 DataElementIndexOfPID, uint8* DestBuffer, uint16* BufSize)	
Service ID[hex]:	0x52	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	PID	This parameter is an identifier for a PID as defined in ISO15031-5.
	DataElement IndexOfPID	Data element index of this PID according to the Dcm configuration of service $02. It is zero-based and consecutive, and ordered by the data element positions (configured in Dcm, refer to SWS_Dem_00597).
Parameters (in-out):	DestBuffer	This parameter contains a byte pointer that points to the buffer, to which the data element of the PID shall be written to. The format is raw hexadecimal values and contains no header-information.
	BufSize	When the function is called this parameter contains the maximum number of data bytes that can be written to the buffer. The function returns the actual number of written data bytes in this parameter.
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK Freeze frame data was successfully reported E_NOT_OK Freeze frame data was not successfully reported

Description:	Gets data element per PID and index of the most important freeze frame being selected for the output of service $02. The function stores the data in the provided DestBuffer. API is needed in OBD-relevant ECUs only

Table 8.107: Dem_DcmReadDataOfOBDFreezeFrame

⌋

8.3.5.13 Dem_DcmGetDTCOfOBDFreezeFrame

[SWS_Dem_00624] ⌈

Service name:	Dem_DcmGetDTCOfOBDFreezeFrame	
Syntax:	Std_ReturnType Dem_DcmGetDTCOfOBDFreezeFrame(uint8 FrameNumber, uint32* DTC)	
Service ID[hex]:	0x53	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	FrameNumber	Unique identifier for a freeze frame record as defined in ISO 15031-5. The value 0x00 indicates the complete OBD freeze frame. Other values are reserved for future functionality.
Parameters (in-out):	None	
Parameters (out):	DTC	Diagnostic Trouble Code in ODB format. If the return value of the function is other than E_OK this parameter does not contain valid data.
Return value:	Std_ReturnType	E_OK: operation was successful E_NOT_OK: no DTC available
Description:	Gets DTC by freeze frame record number. API is needed in OBD-relevant ECUs only	

Table 8.108: Dem_DcmGetDTCOfOBDFreezeFrame

⌋

8.3.5.14 Dem_SetDTR

[SWS_Dem_00765] ⌈

Service name:	Dem_SetDTR	
Syntax:	Std_ReturnType Dem_SetDTR(uint16 DTRId, sint32 TestResult, sint32 LowerLimit, sint32 UpperLimit, Dem_DTRControlType CtrlVal)	
Service ID[hex]:	0xa2	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	DTRId	Identification of a DTR element by assigned DTRId.
	TestResult LowerLimit UpperLimit CtrlVal	Test result of DTR Lower limit of DTR Upper limit of DTR Control value of the DTR to support its interpretation Dem-internally.
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Report of DTR result successful E_NOT_OK: Report of DTR result failed
Description:	Reports a DTR result with lower and upper limit. The internal eventstatus serves as master whether the DTR values are forwarded or ignored, also taking the DTRUpdateKind into account. The EventId that is related to the DTR is assigned per configuration (and derived from ServiceNeeds). Processing takes enable/storage conditions into account. API is needed in OBD-relevant ECUs only	

Table 8.109: Dem_SetDTR

⌋

8.3.5.15 Dem_DcmGetAvailableOBDMIDs

[SWS_Dem_00766] ⌈

Service name:	Dem_DcmGetAvailableOBDMIDs

Syntax:	Std_ReturnType Dem_DcmGetAvailableOBDMIDs(uint8 Obdmid, uint32* Obdmidvalue)	
Service ID[hex]:	0xa3	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	Obdmid	Availablity OBDMID ($00,$20, $40. . .)
Parameters (in-out):	None	
Parameters (out):	Obdmidvalue	Bit coded information on the support of OBDMIDs.
Return value:	Std_ReturnType	E_OK: Report of DTR result successful E_NOT_OK: Report of DTR result failed
Description:	Reports the value of a requested "availability-OBDMID" to the DCM upon a Service $06 request. Derived from that the tester displays the supported tests a mechanic can select from. API is needed in OBD-relevant ECUs only	

Table 8.110: Dem_DcmGetAvailableOBDMIDs

⌋

8.3.5.16 Dem_DcmGetNumTIDsOfOBDMID

[SWS_Dem_00767] ⌈

Service name:	Dem_DcmGetNumTIDsOfOBDMID	
Syntax:	Std_ReturnType Dem_DcmGetNumTIDsOfOBDMID(uint8 Obdmid, uint8* numberOfTIDs)	
Service ID[hex]:	0xa4	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	Obdmid	OBDMID subject of the request to identify the number of assigned TIDs
Parameters (in-out):	None	
Parameters (out):	numberOfTIDs	Number of assigned TIDs for the requested OBDMID. Used as loop value for the DCM to retrieve all OBD/TID result data.

Return value:	Std_ReturnType	E_OK: get of debouncing status per event state successful E_NOT_OK: get of debouncing per event state failed
Description:	Gets the number of TIDs per (functional) OBDMID. This can be used by the DCM to iteratively request for OBD/TID result data within a loop from 0....numberOfTIDs-1 API is needed in OBD-relevant ECUs only	

Table 8.111: Dem_DcmGetNumTIDsOfOBDMID

⌋

8.3.5.17 Dem_DcmGetDTRData

[SWS_Dem_00768] ⌈

Service name:	Dem_DcmGetDTRData	
Syntax:	Std_ReturnType Dem_DcmGetDTRData(uint8 Obdmid, uint8 TIDindex, uint8* TIDvalue, uint8* UaSID, uint16* Testvalue, uint16* Lowlimvalue, uint16* Upplimvalue)	
Service ID[hex]:	0xa5	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	Obdmid	Identification of a DTR element by assigned DTRId.
	TIDindex	Index of the TID within the DEM. Runs from 0 to "numberOfTIDs" obtained in the call to Dem_DcmGetNumTIDsOfOBDMID()
Parameters (in-out):	None	
Parameters (out):	TIDvalue	TID to be put on the tester reponse
	UaSID Testvalue Lowlimvalue Upplimvalue	UaSID to be put on the tester reponse Latest test result Lower limit value associated to the latest test result Upper limit value associated to the latest test result
Return value:	Std_ReturnType	E_OK: Report of DTR result successful E_NOT_OK: Report of DTR result failed

Description:	Reports a DTR data along with TID-value, UaSID, test result with lower and upper limit. API is needed in OBD-relevant ECUs only

Table 8.112: Dem_DcmGetDTRData

⌋

8.3.6 Interface J1939Dcm <=> Dem

8.3.6.1 Access DTCs and Status Information

8.3.6.1.1 Dem_J1939DcmSetDTCFilter

[SWS_Dem_00970] ⌈

Service name:	Dem_J1939DcmSetDTCFilter	
Syntax:	Dem_ReturnSetFilterType Dem_J1939DcmSetDTCFilter(Dem_J1939DcmDTCStatusFilterType DTCStatusFilter, Dem_DTCKindType DTCKind, uint8 NodeAddress, Dem_J1939DcmLampStatusType* LampStatus)	
Service ID[hex]:	0x90	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	DTCStatusFilter	The following types are available: DEM_J1939DTC_ACTIVE DEM_J1939DTC_PREVIOUSLY_ACTIVE DEM_J1939DTC_PENDING DEM_J1939DTC_PERMANENT DEM_J1939DTC_CURRENTLY_ACTIVE
	DTCKind NodeAddress	Defines the functional group of DTCs to be reported (e.g. all DTC, OBD-relevant DTC) NodeAddress
Parameters (in-out):	None	
Parameters (out):	LampStatus	Receives the commulated lamp status.
Return value:	Dem_ReturnSet FilterType	Status of the operation to (re-)set a DTC filter.
Description:	The function set the DTC filter for a specifc node and returns the composit lamp status of the filtered DTCs.	

Table 8.113: Dem_J1939DcmSetDTCFilter

⌋

8.3.6.1.2 Dem_J1939DcmGetNumberOfFilteredDTC

[SWS_Dem_00972] ⌈

Service name:	Dem_J1939DcmGetNumberOfFilteredDTC	
Syntax:	Dem_ReturnGetNumberOfFilteredDTCType Dem_J1939DcmGetNumberOfFilteredDTC(uint16* NumberOfFilteredDTC)	
Service ID[hex]:	0x91	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (in-out):	None	
Parameters (out):	NumberOfFilteredDTC	The number of DTCs matching the defined status mask.
Return value:	Dem_ReturnGetNumberOfFilteredDTCType	Status of the operation to retrieve a number of DTC from the Dem
Description:	Gets the number of currently filtered DTCs set by the function Dem_J1939DcmSetDTCFilter.	

Table 8.114: Dem_J1939DcmGetNumberOfFilteredDTC

⌋

8.3.6.1.3 Dem_J1939DcmGetNextFilteredDTC

[SWS_Dem_00973] ⌈

Service name:	Dem_J1939DcmGetNextFilteredDTC
Syntax:	Dem_ReturnGetNextFilteredElementType Dem_J1939DcmGetNextFilteredDTC(uint32* J1939DTC, uint8* OccurenceCounter)

Service ID[hex]:	0x92	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (in-out):	None	
Parameters (out):	J1939DTC	Receives the J1939DTC value. If the return value of the function is other than DEM_FILTERED_OK this parameter does not contain valid data.
	Occurence Counter	This parameter receives the corresponding occurrence counter. If the return value of the function call is other than DEM_FILTERED_OK this parameter does not contain valid data.
Return value:	Dem_Return GetNextFiltered ElementType	Status of the operation to retrieve a DTC from the Dem.
Description:	Gets the next filtered J1939 DTC.	

Table 8.115: Dem_J1939DcmGetNextFilteredDTC

⌋

8.3.6.1.4 Dem_J1939DcmFirstDTCwithLampStatus

[SWS_Dem_00974] ⌈

Service name:	Dem_J1939DcmFirstDTCwithLampStatus	
Syntax:	void Dem_J1939DcmFirstDTCwithLampStatus(uint8 NodeAddress)	
Service ID[hex]:	0x93	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	NodeAddress	Node address of requesting client
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	None	
Description:	The function set the filter to the first applicable DTC for the DM31 response for a specifc node.	

Table 8.116: Dem_J1939DcmFirstDTCwithLampStatus

⌋

8.3.6.1.5 Dem_J1939DcmGetNextDTCwithLampStatus

[SWS_Dem_00975] ⌈

Service name:	Dem_J1939DcmGetNextDTCwithLampStatus	
Syntax:	Dem_ReturnGetNextFilteredElementType Dem_J1939DcmGetNextDTCwithLampStatus(Dem_J1939DcmLampStatusType* LampStatus, uint32* J1939DTC, uint8* OccurenceCounter)	
Service ID[hex]:	0x94	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (in-out):	None	
Parameters (out):	LampStatus	Receives the lamp status returned by this function. If the return value of the function is other than DEM_FILTERED_OK this parameter does not contain valid data.
	J1939DTC Occurence Counter	Receives the J1939DTC value. If the return value of the function is other than DEM_FILTERED_OK this parameter does not contain valid data. This parameter receives the corresponding occurrence counter. If the return value of the function call is other than DEM_FILTERED_OK this parameter does not contain valid data.
Return value:	Dem_Return GetNextFiltered ElementType	Status of the operation to retrieve a DTC from the Dem.
Description:	Gets the next filtered J1939 DTC for DM31 including current LampStatus.	

Table 8.117: Dem_J1939DcmGetNextDTCwithLampStatus

⌋

8.3.6.2 DTC storage

8.3.6.2.1 Dem_J1939DcmClearDTC

[SWS_Dem_00976] ⌈

Service name:	Dem_J1939DcmClearDTC	
Syntax:	Dem_ReturnClearDTCType Dem_J1939DcmClearDTC(Dem_J1939DcmSetClearFilterType DTCTypeFilter, uint8 NodeAddress)	
Service ID[hex]:	0x95	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	DTCTypeFilter	The following types are available: DEM_J1939DTC_CLEAR_ALL DEM_J1939DTC_CLEAR_PREVIOUSLY_ACTIVE
	NodeAddress	node address of requesting client
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Dem_ReturnClearDTCType	Status of the operation of type Dem_ReturnClearDTCType.
Description:	Clears active DTCs as well as previously avtive DTCs.	

Table 8.118: Dem_J1939DcmClearDTC

⌋

8.3.6.2.2 Dem_J1939DcmSetFreezeFrameFilters

[SWS_Dem_00977] ⌈

Service name:	Dem_J1939DcmSetFreezeFrameFilter
Syntax:	Dem_ReturnSetFilterType Dem_J1939DcmSetFreezeFrameFilter(Dem_J1939DcmSetFreezeFrameFilterType FreezeFrameKind, uint8 NodeAddress)
Service ID[hex]:	0x96
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant

Parameters (in):	FreezeFrame Kind	The following types are available: DEM_J1939DCM_FREEZEFRAME DEM_J1939DCM_EXPANDED_FREEZEFRAME DEM_J1939DCM_SPNS_IN_EXPANDED_FREEZEFRAME
	NodeAddress	Node address of requesting client
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Dem_ReturnSetFilterType	Status of the operation to (re-)set a FreezeFrame filter.
Description:	The function set the FreezeFrame filter for a specifc node.	

Table 8.119: Dem_J1939DcmSetFreezeFrameFilter

⌋

8.3.6.2.3 Dem_J1939DcmGetNextFreezeFrame

[SWS_Dem_00978] ⌈

Service name:	Dem_J1939DcmGetNextFreezeFrame	
Syntax:	Dem_ReturnGetNextFilteredElementType Dem_J1939DcmGetNextFreezeFrame(uint32* J1939DTC, uint8* OccurenceCounter, uint8* DestBuffer, uint16* BufSize)	
Service ID[hex]:	0x97	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (in-out):	DestBuffer	This parameter contains a byte pointer that points to the buffer, to which the freeze frame data record shall be written to.
	BufSize	When the function is called this parameter contains the maximum number of data bytes that can be written to the buffer. The function returns the actual number of written data bytes in DestBuffer
Parameters (out):	J1939DTC	Receives the J1939DTC value. If the return value of the function is other than DEM_FILTERED_OK this parameter does not contain valid data.

	Occurence Counter	This parameter receives the corresponding occurrence counter. If the return value of the function call is other than DEM_FILTERED_OK this parameter does not contain valid data.
Return value:	Dem_Return GetNextFiltered ElementType	Status of the operation to retrieve freeze frame data by DTC.
Description:	Gets next freeze frame data. The function stores the data in the provided DestBuffer.	

Table 8.120: Dem_J1939DcmGetNextFreezeFrame

⌋

8.3.6.2.4 Dem_J1939DcmGetNextSPNInFreezeFrame

[SWS_Dem_00979] ⌈

Service name:	Dem_J1939DcmGetNextSPNInFreezeFrame	
Syntax:	Dem_ReturnGetNextFilteredElementType Dem_J1939DcmGetNextSPNInFreezeFrame(uint32* SPNSupported, uint8* SPNDataLength)	
Service ID[hex]:	0x98	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	SPNSupported	This parameter contains the next SPN in the ExpandedFreezeFrame
	SPNDataLength	This parameter contains the corresponding dataLength of the SPN
Return value:	Dem_Return GetNextFiltered ElementType	Status of the operation to retrieve freeze frame data by DTC.
Description:	Gets next SPN.	

Table 8.121: Dem_J1939DcmGetNextSPNInFreezeFrame

⌋

8.3.6.3 Reporting

8.3.6.3.1 Dem_J1939DcmSetRatioFilter

[SWS_Dem_00980] ⌈

Service name:	Dem_J1939DcmSetRatioFilter	
Syntax:	Dem_ReturnSetFilterType Dem_J1939DcmSetRatioFilter(uint16* IgnitionCycleCounter, uint16* OBDMonitoringConditionsEncountered, uint8 NodeAddress)	
Service ID[hex]:	0x99	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	NodeAddress	Node address of requesting client
Parameters (in-out):	None	
Parameters (out):	IgnitionCycle Counter	Ignition Cycle Counter
	OBDMonitor-ingConditions Encountered	OBD Monitoring Conditions Encountered
Return value:	Dem_ReturnSet FilterType	Status of the operation to (re-)set a DTC filter.
Description:	The function set the Ratio filter for a specifc node and returns the corresponding Ignition Cycle Counter and General Denominator.	

Table 8.122: Dem_J1939DcmSetRatioFilter

⌋

8.3.6.3.2 Dem_J1939DcmGetNextFilteredRatio

[SWS_Dem_00981] ⌈

Service name:	Dem_J1939DcmGetNextFilteredRatio
Syntax:	Dem_ReturnGetNextFilteredElementType Dem_J1939DcmGetNextFilteredRatio(uint16* SPN, uint16* Numerator, uint16* Denominator)

Service ID[hex]:	0x9a	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (in-out):	None	
Parameters (out):	SPN	Receives the SPN of the applicaple system monitor. If the return value of the function is other than DEM_FILTERED_OK this parameter does not contain valid data.
	Numerator	Receives the Numerator of the applicable system monitor. If the return value of the function is other than DEM_FILTERED_OK this parameter does not contain valid data.
	Denominator	Receives the Denominator of the applicable system monitor. If the return value of the function is other than DEM_FILTERED_OK this parameter does not contain valid data.
Return value:	Dem_Return GetNextFiltered ElementType	0x00 DEM_FILTERED_OK Ratio available in out parameter 0x01 DEM_FILTERED_NO_FURTHER ELEMENT No further element available
Description:	Gets the next filtered Ratio.	

Table 8.123: Dem_J1939DcmGetNextFilteredRatio

⌋

8.3.6.3.3 Dem_J1939DcmReadDiagnosticReadiness1

[SWS_Dem_00982] ⌈

Service name:	Dem_J1939DcmReadDiagnosticReadiness1	
Syntax:	void Dem_J1939DcmReadDiagnosticReadiness1(Dem_J1939DcmDiagnosticReadiness1Type* DataValue, uint8 NodeAddress)	
Service ID[hex]:	0x9b	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	NodeAddress	Node address of requesting client
Parameters (in-out):	None	

Parameters (out):	DataValue	Buffer of 8 bytes containing the contents of Diagnostic Readiness 1 (DM5) computed by the Dem.
Return value:	None	
Description:	Service to report the value of Diagnostic Readiness 1 (DM5) computed by the Dem.	

Table 8.124: Dem_J1939DcmReadDiagnosticReadiness1

⌋

8.3.6.3.4 Dem_J1939DcmReadDiagnosticReadiness2

[SWS_Dem_00983] ⌈

Service name:	Dem_J1939DcmReadDiagnosticReadiness2	
Syntax:	void Dem_J1939DcmReadDiagnosticReadiness2(Dem_J1939DcmDiagnosticReadiness2Type* DataValue, uint8 NodeAddress)	
Service ID[hex]:	0x9c	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	NodeAddress	Node address of requesting client
Parameters (inout):	None	
Parameters (out):	DataValue	Buffer of 8 bytes containing the contents of Diagnostic Readiness 2 (DM21) computed by the Dem.
Return value:	None	
Description:	Service to report the value of Diagnostic Readiness 2 (DM21) computed by the Dem.	

Table 8.125: Dem_J1939DcmReadDiagnosticReadiness2

⌋

8.3.6.3.5 Dem_J1939DcmReadDiagnosticReadiness3

[SWS_Dem_00770] ⌈

Service name:	Dem_J1939DcmReadDiagnosticReadiness3

Syntax:	void Dem_J1939DcmReadDiagnosticReadiness3(Dem_J1939DcmDiagnosticReadiness3Type* DataValue, uint8 NodeAddress)	
Service ID[hex]:	0x9d	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	NodeAddress	Node address of requesting client
Parameters (in-out):	None	
Parameters (out):	DataValue	Buffer of 8 bytes containing the contents of Diagnostic Readiness 3 (DM26) computed by the Dem.
Return value:	None	
Description:	Service to report the value of Diagnostic Readiness 3 (DM26) computed by the Dem.	

Table 8.126: Dem_J1939DcmReadDiagnosticReadiness3

⌋

8.3.7 Interface Dlt <=> Dem

8.3.7.1 Dem_DltGetMostRecentFreezeFrameRecordData

[SWS_Dem_00636] ⌈

Service name:	Dem_DltGetMostRecentFreezeFrameRecordData	
Syntax:	Std_ReturnType Dem_DltGetMostRecentFreezeFrameRecordData(Dem_EventIdType EventId, uint8* DestBuffer, uint16* BufSize)	
Service ID[hex]:	0x41	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	EventId	Identification of an event by assigned EventId.
Parameters (in-out):	BufSize	When the function is called this parameter contains the maximum number of data bytes that can be written to the buffer. The function returns the actual number of written data bytes in this parameter.

Parameters (out):	DestBuffer	This parameter contains a byte pointer that points to the buffer, to which the freeze frame record shall be written to. The format is raw hexadecimal values and contains no header-information.
Return value:	Std_ReturnType	E_OK: Operation was successful. DEM_E_NODATAAVAILABLE: The requested event data is not currently stored (but the request was valid). DEM_E_PENDING: The requested data is currently transported from NvM and needs to be requested again.
Description:	Gets the data of an most recent freeze frame record by event.	

Table 8.127: Dem_DltGetMostRecentFreezeFrameRecord

⌋(*SRS_Diag_04099*)

8.3.7.2 Dem_DltGetAllExtendedDataRecords

[SWS_Dem_00637] ⌈

Service name:	Dem_DltGetAllExtendedDataRecords	
Syntax:	Std_ReturnType Dem_DltGetAllExtendedDataRecords(Dem_EventIdType EventId, uint8* DestBuffer, uint16* BufSize)	
Service ID[hex]:	0x40	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	EventId	Identification of an event by assigned EventId.
Parameters (in-out):	BufSize	When the function is called this parameter contains the maximum number of data bytes that can be written to the buffer. The function returns the actual number of written data bytes in this parameter.
Parameters (out):	DestBuffer	This parameter contains a byte pointer that points to the buffer, to which the extended data shall be written to. The format is raw hexadecimal values and contains no header-information.

Return value:	Std_ReturnType	E_OK: Operation was successful. DEM_E_NODATAAVAILABLE: The requested event data is not currently stored (but the request was valid). DEM_E_PENDING: The requested data is currently transported from NvM and needs to be requested again.
Description:	Gets the data of all extended data records of an event.	

Table 8.128: Dem_DltGetAllExtendedDataRecords

⌋(*SRS_Diag_04099*)

8.3.8 OBD-specific Interfaces

8.3.8.1 Dem_SetEventDisabled

[SWS_Dem_00312] ⌈

Service name:	Dem_SetEventDisabled	
Syntax:	Std_ReturnType Dem_SetEventDisabled(Dem_EventIdType EventId)	
Service ID[hex]:	0x51	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different EventIds. Non reentrant for the same EventId.	
Parameters (in):	EventId	Identification of an event by assigned EventId.
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK set of event to disabled was successfull. E_NOT_OK set of event disabled failed
Description:	Service for reporting the event as disabled to the Dem for the PID $41 computation. API is needed in OBD-relevant ECUs only	

Table 8.129: Dem_SetEventDisabled

⌋

8.3.8.2 Dem_ReplUMPRFaultDetect

[SWS_Dem_00313] ⌈

Service name:	Dem_ReplUMPRFaultDetect	
Syntax:	Std_ReturnType Dem_ReplUMPRFaultDetect(Dem_RatioIdType RatioID)	
Service ID[hex]:	0x73	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	RatioID	Ratio Identifier reporting that a respective monitor could have found a fault - only used when interface option "API" is selected
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK report of IUMPR result was successfully reported
Description:	Service for reporting that faults are possibly found because all conditions are fullfilled. API is needed in OBD-relevant ECUs only	

Table 8.130: Dem_ReplUMPRFaultDetect

⌋

8.3.8.3 Dem_SetIUMPRDenCondition

[SWS_Dem_00733] ⌈

Service name:	Dem_SetIUMPRDenCondition	
Syntax:	Std_ReturnType Dem_SetIUMPRDenCondition(Dem_IumprDenomCondIdType ConditionId, Dem_IumprDenomCondStatusType ConditionStatus)	
Service ID[hex]:	0xae	
Sync/Async:	Synchronous /Asynchronous	
Reentrancy:	Reentrant	
Parameters (in):	ConditionId	Identification of a IUMPR denominator condition ID (General Denominator, Cold start, EVAP, 500mi).
	ConditionStatus	Status of the IUMPR denominator condition (Not-reached, reached, not reachable / inhibited)

Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: set of IUMPR denominator condition was successful E_NOT_OK: set of IUMPR denominator condition failed or could not be accepted.
Description:	In order to communicate the status of the (additional) denominator conditions among the OBD relevant ECUs, the API is used to forward the condition status to a Dem of a particular ECU. API is needed in OBD-relevant ECUs only.	

Table 8.131: Dem_SetIUMPRDenCondition

⌋

8.3.8.4 Dem_GetIUMPRDenCondition

[SWS_Dem_00734] ⌈

Service name:	Dem_GetIUMPRDenCondition	
Syntax:	Std_ReturnType Dem_GetIUMPRDenCondition(Dem_IumprDenomCondIdType ConditionId, Dem_IumprDenomCondStatusType* ConditionStatus)	
Service ID[hex]:	0xaf	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	ConditionId	Identification of a IUMPR denominator condition ID (General Denominator, Cold start, EVAP, 500mi).
Parameters (in-out):	None	
Parameters (out):	ConditionStatus	Status of the IUMPR denominator condition (Not-reached, reached, not reachable / inhibited)
Return value:	Std_ReturnType	E_OK: get of IUMPR denominator condition status was successful E_NOT_OK: get of condition status failed
Description:	In order to communicate the status of the (additional) denominator conditions among the OBD relevant ECUs, the API is used to retrieve the condition status from the Dem of the ECU where the conditions are computed. API is needed in OBD-relevant ECUs only.	

Table 8.132: Dem_GetIUMPRDenCondition

⌋

8.3.8.5 Dem_ReplUMPRDenLock

[SWS_Dem_00314] ⌈

Service name:	Dem_ReplUMPRDenLock	
Syntax:	Std_ReturnType Dem_ReplUMPRDenLock(Dem_RatioIdType RatioID)	
Service ID[hex]:	0x71	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	RatioID	Ratio Identifier reporting that specific denominator is locked (for physical reasons - e.g. temperature conditions or minimum activity)
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK report of IUMPR denominator status was successfully reported E_NOK report of IUMPR denominator status was not successfully reported
Description:	Service is used to lock a denominator of a specific monitor. API is needed in OBD-relevant ECUs only.	

Table 8.133: Dem_ReplUMPRDenLock

⌋

8.3.8.6 Dem_ReplUMPRDenRelease

[SWS_Dem_00315] ⌈

Service name:	Dem_ReplUMPRDenRelease
Syntax:	Std_ReturnType Dem_ReplUMPRDenRelease(Dem_RatioIdType RatioID)
Service ID[hex]:	0x72
Sync/Async:	Synchronous
Reentrancy:	Reentrant

Parameters (in):	RatioID	Ratio Identifier reporting that specific denominator is released (for physical reasons - e.g. temperature conditions or minimum activity)
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK report of IUMPR denominator status was successfully reported E_NOK report of IUMPR denominator status was not successfully reported
Description:	Service is used to release a denominator of a specific monitor. API is needed in OBD-relevant ECUs only	

Table 8.134: Dem_RepIUMPRDenRelease

⌋

8.3.8.7 Dem_SetPtoStatus

[SWS_Dem_00627] ⌈

Service name:	Dem_SetPtoStatus	
Syntax:	Std_ReturnType Dem_SetPtoStatus(boolean PtoStatus)	
Service ID[hex]:	0x79	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	PtoStatus	sets the status of the PTO (TRUE==active; FALSE==inactive)
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	Returns E_OK when the new PTO-status has been adopted by the Dem; returns E_NOT_OK in all other cases.
Description:	API is needed in OBD-relevant ECUs only	

Table 8.135: Dem_SetPtoStatus

⌋

8.3.8.8 Dem_SetDataOfPID21

[SWS_Dem_00735] ⌈

Service name:	Dem_SetDataOfPID21	
Syntax:	Std_ReturnType Dem_SetDataOfPID21(uint8* PID21value)	
Service ID[hex]:	0xa6	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	PID21value	Buffer containing the contents of PID $21. The buffer is provided by the Dcm with the appropriate size, i.e. during configuration, the Dcm identifies the required size from the largest PID in order to configure a PIDBuffer.
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	Always E_OK is returned, as E_NOT_OK will never appear.
Description:	Service to set the value of PID $21 in the Dem by a software component. API is needed in OBD-relevant ECUs only.	

Table 8.136: Dem_SetDataOfPID21

⌋

8.3.8.9 Dem_SetDataOfPID31

[SWS_Dem_00736] ⌈

Service name:	Dem_SetDataOfPID31
Syntax:	Std_ReturnType Dem_SetDataOfPID31(uint8* PID31value)
Service ID[hex]:	0xa7
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant

Parameters (in):	PID31value	Buffer containing the contents of PID $31. The buffer is provided by the Dcm with the appropriate size, i.e. during configuration, the Dcm identifies the required size from the largest PID in order to configure a PIDBuffer.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	Always E_OK is returned, as E_NOT_OK will never appear.
Description:	Service to set the value of PID $31 in the Dem by a software component. API is needed in OBD-relevant ECUs only.	

Table 8.137: Dem_SetDataOfPID31

⌋

8.3.8.10 Dem_SetDataOfPID4D

[SWS_Dem_00737] ⌈

Service name:	Dem_SetDataOfPID4D	
Syntax:	Std_ReturnType Dem_SetDataOfPID4D(uint8* PID4Dvalue)	
Service ID[hex]:	0xa8	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	PID4Dvalue	Buffer containing the contents of PID $4D. The buffer is provided by the Dcm with the appropriate size, i.e. during configuration, the Dcm identifies the required size from the largest PID in order to configure a PIDBuffer.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	Always E_OK is returned, as E_NOT_OK will never appear.
Description:	Service to set the value of PID $4D in the Dem by a software component. API is needed in OBD-relevant ECUs only.	

Table 8.138: Dem_SetDataOfPID4D

⌋

8.3.8.11 Dem_SetDataOfPID4E

[SWS_Dem_00738] ⌈

Service name:	Dem_SetDataOfPID4E	
Syntax:	Std_ReturnType Dem_SetDataOfPID4E(uint8* PID4Evalue)	
Service ID[hex]:	0xa9	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	PID4Evalue	Buffer containing the contents of PID $4E. The buffer is provided by the Dcm with the appropriate size, i.e. during configuration, the Dcm identifies the required size from the largest PID in order to configure a PIDBuffer.
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	Always E_OK is returned, as E_NOT_OK will never appear.
Description:	Service to set the value of PID $4E in the Dem by a software component. API is needed in OBD-relevant ECUs only.	

Table 8.139: Dem_SetDataOfPID4E

⌋

8.3.8.12 Dem_SetPfcCycleQualified

[SWS_Dem_00739] ⌈

Service name:	Dem_SetPfcCycleQualified
Syntax:	Std_ReturnType Dem_SetPfcCycleQualified(void)
Service ID[hex]:	0xaa
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None

Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	Always E_OK is returned, as E_NOT_OK will never appear.
Description:	Marks the current OBD driving cycle as having met the criteria for the PFC cycle. API is needed in OBD-relevant ECUs only.	

Table 8.140: Dem_SetPfcCycleQualified

⌋

8.3.8.13 Dem_GetPfcCycleQualified

[SWS_Dem_00740] ⌈

Service name:	Dem_GetPfcCycleQualified	
Syntax:	Std_ReturnType Dem_GetPfcCycleQualified(boolean isqualified)	
Service ID[hex]:	0xab	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	isqualified	TRUE: During the current OBD driving cycle the criteria for the PFC cycle have been met. FALSE: During the current OBD driving cycle the criteria for the PFC cycle have not been met.
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	Always E_OK is returned, as E_NOT_OK will never appear.
Description:	Returns TRUE if the criteria for the PFC cycle have been met during the current OBD driving cycle. API is needed in OBD-relevant ECUs only.	

Table 8.141: Dem_GetPfcCycleQualified

⌋

8.3.8.14 Dem_SetClearDTC

[SWS_Dem_00741] ⌈

Service name:	Dem_SetClearDTC	
Syntax:	Std_ReturnType Dem_SetClearDTC(uint32 DTC, Dem_DTCFormatType DTCFormat, Dem_DTCOriginType DTCOrigin)	
Service ID[hex]:	0xac	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	DTC	Defines the DTC in respective format, that has been cleared from the event memory.
	DTCFormat DTCOrigin	Format of the provided DTC value. Event memory (e.g. MIRROR)
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	Return value unused - only for compatibility with according RTE operation.
Description:	API to inform the Dem (of a Dependend / Secondary ECU) about the reception of service $04 execution by a software component. API is needed in OBD Dependend / Secondary ECUs only.	

Table 8.142: Dem_SetClearDTC

⌋

8.4 Expected Interfaces

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

8.4.1 Mandatory Interfaces

This chapter defines all interfaces, which are required to fulfill the core functionality of the Dem module.

API function Description

API function	Description

8.4.2 Optional Interfaces

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

[SWS_Dem_00255] ⌈

API function	Description
Dcm_DemTriggerOnDTCStatus	Triggers on changes of the UDS DTC status byte. Allows to trigger on ROE Event for subservice OnDTCStatusChanged.
Det_ReportError	Service to report development errors.
Dlt_DemTriggerOnEventStatus	This service is provided by the Dem in order to call Dlt upon status changes.
FiM_DemInit	This service re-initializes the FIM.
FiM_DemTriggerOnEventStatus	This service is provided by the Dem in order to call FiM upon status changes.
J1939Dcm_DemTriggerOnDTCStatus	Trigger for DM1 message that a DTC status change has happened.
NvM_GetErrorStatus	Service to read the block dependent error/status information.
NvM_ReadBlock	Service to copy the data of the NV block to its corresponding RAM block.
NvM_SetRamBlockStatus	Service for setting the RAM block status of an NVRAM block.
NvM_WriteBlock	Service to copy the data of the RAM block to its corresponding NV block.

⌋(*SRS_BSW_00171*)

Note: Based on implementation strategy FiM_DemInit can be used (refer also to chapter chapter 7.9.3).

Note: Based on implementation strategy either NvM_[Read|Write]Block or NvM_SetRamBlockStatus can be omitted, or the NvM usage is deactivated by configuration completely (refer also to chapter chapter 7.9.5).

8.4.3 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 this kind of interfaces are not fixed because they are configurable.

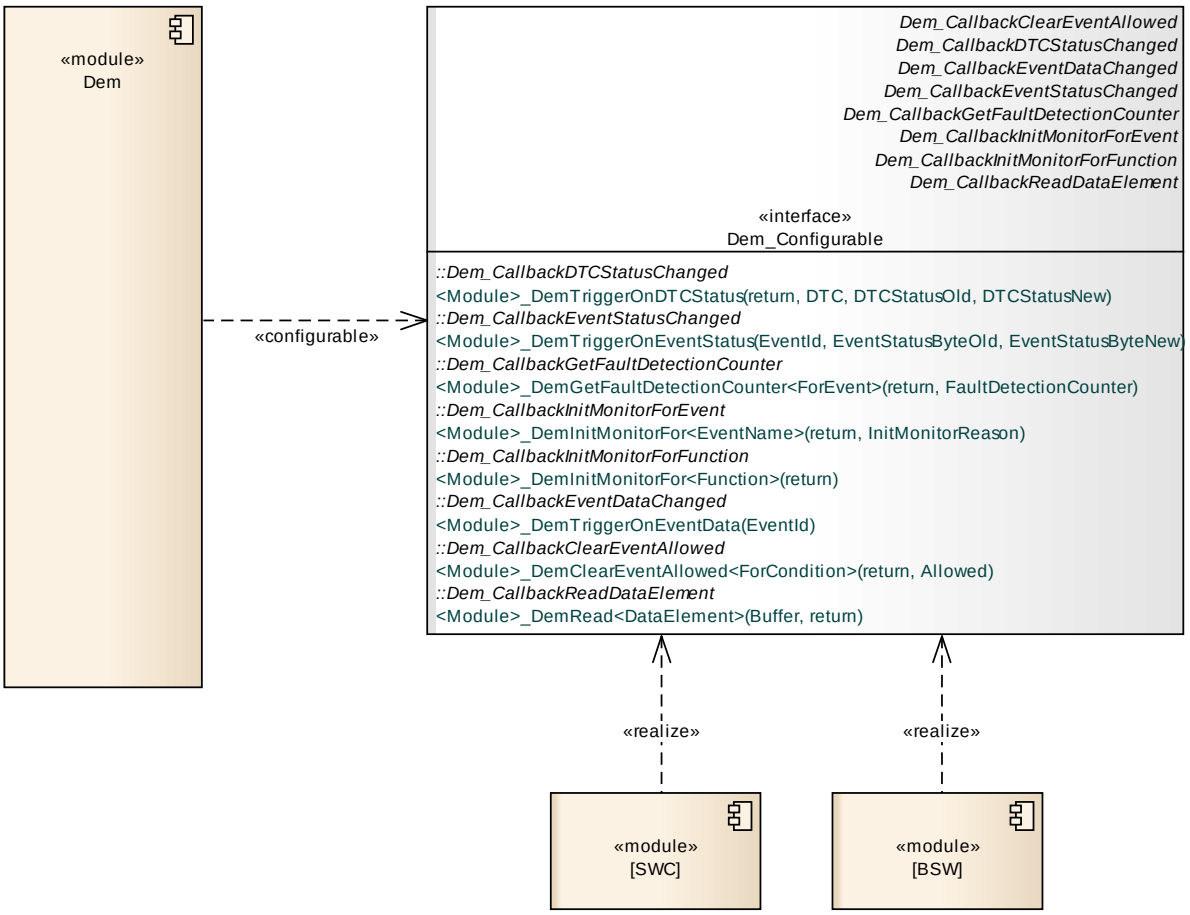

Figure 8.4: Configuration interfaces of the Dem module

8.4.3.1 Interface BSW modules / SW-Components <=> Dem

The callback interface from Dem to SW-Components is realized via RTE port interfaces. The following callback descriptions address the c-callbacks of other BSW modules.

8.4.3.1.1 InitMonitorForEvent

[SWS_Dem_00256] ⌈

Service name:	<Module>_DemInitMonitorFor<EventName>

Syntax:	Std_ReturnType <Module>_DemInitMonitorFor<EventName>(Dem_InitMonitorReasonType InitMonitorReason)	
Service ID[hex]:	–	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	InitMonitorReason	Specific (re-)initialization reason evaluated from the monitor to identify the initialization kind to be performed.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	Return value unused - only for compatibility with according RTE operation.
Description:	Inits the diagnostic monitor of a specific event. There is one separate callback per event (if configured), if no port interface is provided by the Dem.	

Table 8.145: InitMonitorForEvent

⌋

8.4.3.1.2 InitMonitorForFunction

[SWS_Dem_00258] ⌈

Service name:	<Module>_DemInitMonitorFor<Function>	
Syntax:	Std_ReturnType <Module>_DemInitMonitorFor<Function>(void)	
Service ID[hex]:	–	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	Return value unused - only for compatibility with according RTE operation.
Description:	Resets the <Function> of the according module.	

Table 8.146: InitMonitorForFunction

⌋

8.4.3.2 EventStatusChanged

[SWS_Dem_00259] ⌈

Service name:	<Module>_DemTriggerOnEventStatus	
Syntax:	void <Module>_DemTriggerOnEventStatus(Dem_EventIdType EventId, Dem_UdsStatusByteType EventStatusByteOld, Dem_UdsStatusByteType EventStatusByteNew)	
Service ID[hex]:	–	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	EventId	Identification of an event by assigned EventId.
	EventStatusByteOld EventStatusByteNew	UDS DTC status byte of event before change (refer to chapter "Status bit support"). UDS DTC status byte of event after change (refer to chapter "Status bit support").
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	None	
Description:	Triggers on changes of the UDS DTC status byte.	

Table 8.147: DemTriggerOnEventStatus

⌋

8.4.3.3 DTCStatusChanged

[SWS_Dem_00260] ⌈

Service name:	<Module>_DemTriggerOnDTCStatus

Syntax:	Std_ReturnType <Module>_DemTriggerOnDTCStatus(uint32 DTC, Dem_UdsStatusByteType DTCStatusOld, Dem_UdsStatusByteType DTCStatusNew)	
Service ID[hex]:	–	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	DTC	Diagnostic Trouble Code in UDS format.
	DTCStatusOld DTCStatusNew	DTC status before change DTC status after change
Parameters (in-out):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	Return value unused - only for compatibility with according RTE operation.
Description:	Triggers on changes of the UDS DTC status byte.	

Table 8.148: DemTriggerOnDTCStatus

⌋

8.4.3.3.1 SetClearDTC

[SWS_Dem_00747] ⌈

Service name:	<Module>_SetClearDTC	
Syntax:	Std_ReturnType <Module>_SetClearDTC(uint32 DTC, Dem_DTCFormatType DTCFormat, Dem_DTCOriginType DTCOrigin)	
Service ID[hex]:	0xad	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	DTC	Defines the DTC in respective format, that has been cleared from the event memory.
	DTCFormat DTCOrigin	Format of the provided DTC value. Event memory (e.g. MIRROR)
Parameters (in-out):	None	

Parameters (out):	None	
Return value:	Std_ReturnType	Return value unused - only for compatibility with according RTE operation.
Description:	API to inform a software component about the reception of service $04 execution by a Dem of a Primary or Master ECU API is needed in OBD-relevant ECUs only.	

Table 8.149: SetClearDTC

⌋

8.4.3.3.2 EventDataChanged

[SWS_Dem_00562] ⌈

Service name:	<Module>_DemTriggerOnEventData	
Syntax:	void <Module>_DemTriggerOnEventData(Dem_EventIdType EventId)	
Service ID[hex]:	–	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	EventId	Identification of an event by assigned EventId.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Triggers on changes of the event related data in the event memory.	

Table 8.150: DemTriggerOnEventData

⌋

8.4.3.3.3 ClearEventAllowed

[SWS_Dem_00563] ⌈

Service name:	<Module>_DemClearEventAllowed<ForCondition>

Syntax:	Std_ReturnType <Module>_DemClearEventAllowed<ForCondition>(boolean* Allowed)	
Service ID[hex]:	–	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (in-out):	None	
Parameters (out):	Allowed	True – clearance of event is allowed False – clearance of event is not allowed
Return value:	Std_ReturnType	E_OK: Operation was successful E_NOT_OK: Operation failed
Description:	Triggers on DTC-deletion, which is not allowed if the out-parameter returns False. There is one separate callback per condition, which can be assigned to one or several events, if no port interface is provided by the Dem.	

Table 8.151: DemClearEventAllowed

⌋

8.4.3.3.4 ReadDataElement

[SWS_Dem_00564] ⌈

Service name:	<Module>_DemRead<DataElement>	
Syntax:	Std_ReturnType <Module>_DemRead<DataElement>(uint8* Buffer)	
Service ID[hex]:	–	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (in-out):	None	
Parameters (out):	Buffer	Buffer containing the value of the data element
Return value:	Std_ReturnType	E_OK: Operation was successful E_NOT_OK: Operation failed
Description:	Requests the current value of the data element. There is one separate callback per data element, if no port interface is provided by the Dem.	

Table 8.152: DemRead

⌋

8.4.3.4 GetFaultDetectionCounter

[SWS_Dem_00263] ⌈

Service name:	<Module>_DemGetFaultDetectionCounter<ForEvent>	
Syntax:	Std_ReturnType <Module>_DemGetFaultDetectionCounter<ForEvent>(sint8* FaultDetectionCounter)	
Service ID[hex]:	–	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (in-out):	None	
Parameters (out):	FaultDetection Counter	This parameter receives the fault detection counter information of the requested EventId. If the return value of the function call is other than E_OK this parameter does not contain valid data. -128dec...127dec PASSED...FAILED according to ISO 14229-1
Return value:	Std_ReturnType	E_OK: request was successful E_NOT_OK: request failed
Description:	Gets the current fault detection counter value. There is one c-callback per event using monitor-internal debouncing, if no port interface is provided by the Dem.	

Table 8.153: GetFaultDetection

⌋

8.5 Scheduled functions

These functions are directly called by Basic Software Scheduler. The following functions shall have no return value and no parameter. All functions shall be non reentrant.

8.5.1 Dem_MainFunction

[SWS_Dem_00266] ⌈

Service name:	`Dem_MainFunction`
Syntax:	`void Dem_MainFunction( void )`
Service ID[hex]:	0x55
Description:	Processes all not event based Dem internal functions.

⌋

[SWS_Dem_00125] ⌈The function Dem_MainFunction shall process all not event based Dem module internal functions. ⌋

8.5.2 Runnable Entity MainFunction

```
RunnableEntity MainFunction
       symbol "Dem_MainFunction"
       canbeInvokedConcurrently = FALSE
       SSCP = port CBStatusEvt_*, EventStatusChanged
       SSCP = port GeneralCBStatusEvt, EventStatusChanged
       SSCP = port CBStatusDTC_*, DTCStatusChanged
       SSCP = port CBDataEvt_*, EventDataChanged
       SSCP = port GeneralCBDataEvt, EventDataChanged
       SSCP = port CBReadData_*, ReadData
```

8.6 Service Interfaces

8.6.1 Sender-Receiver-Interfaces

8.6.1.1 DataServices_Data

[SWS_Dem_00849] ⌈The Dem Service Component shall provide for each configured DemExternalSRDataElementClass a corresponding R-Port with SRDataServices_<SyncDataElement> with one data element having an ImplementationDataType of type DemDataElementDataType ⌋*(SRS_Diag_04030)*

[SWS_Dem_00850] ⌈Each R-Port of SRDataServices_<SyncDataElement> (refer to SWS_Dem_00847 shall derive the CompuMethod from its DemDiagnosisScaling container (if present) and add them to the DataType in the port interface.

Name	DataServices
Comment	–
IsService	false
Variation	({ecuc(Dem/DemGeneral/DemDataElementClass)} instanceof {ecuc(Dem/DemGeneral/DemDataElementClass/DemExternalSRDataElementClass)}) Data = {ecuc(Dem/DemGeneral/DemDataElementClass/SHORT-NAME)}

Data Elements	data	
	Type	boolean
	Variation	{ecuc(Dem/DemGeneral/DemDataElementClass/ DemExternalSRDataElementClass/ DemDataElementDataType)} == BOOLEAN
	data	
	Type	sint8
	Variation	{ecuc(Dem/DemGeneral/DemDataElementClass/ DemExternalSRDataElementClass/ DemDataElementDataType)} == SINT8
	data	
	Type	sint16
	Variation	{ecuc(Dem/DemGeneral/DemDataElementClass/ DemExternalSRDataElementClass/ DemDataElementDataType)} == SINT16
	data	
	Type	sint32
	Variation	{ecuc(Dem/DemGeneral/DemDataElementClass/ DemExternalSRDataElementClass/ DemDataElementDataType)} == SINT32
	data	
	Type	uint32
	Variation	{ecuc(Dem/DemGeneral/DemDataElementClass/ DemExternalSRDataElementClass/ DemDataElementDataType)} == UINT32
	data	
	Type	uint8
	Variation	{ecuc(Dem/DemGeneral/DemDataElementClass/ DemExternalSRDataElementClass/ DemDataElementDataType)} == UINT8
	data	
	Type	uint16
	Variation	{ecuc(Dem/DemGeneral/DemDataElementClass/ DemExternalSRDataElementClass/ DemDataElementDataType)} == UINT16

⌋(*SRS_Diag_04030*)

8.6.2 Client-Server-Interfaces

8.6.2.1 AgingCycle

[SWS_Dem_00602] ⌈The Dem Service Component shall provide the interface AgingCycle as defined below (to provide the capability to set an aging cycle state).

One port of this interface type is provided per aging cycle by the Dem Service Component. It has AgingCycleId as a port-defined argument.

Name	AgingCycle
Comment	–
IsService	true

Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

SetAgingCycleState		
Comment	–	
Variation	–	
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋(*SRS_Diag_04076*, *SRS_Diag_04061*)

8.6.2.2 CallbackClearEventAllowed

[SWS_Dem_00620] ⌈The Dem Service Component shall provide the interface CallbackClearEventAllowed as defined below (to get the permission before clearing a specific event from the SW-C), if configured.

For each event, there can be one port of this interface type.

Name	CallbackClearEventAllowed	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

ClearEventAllowed		
Comment	–	
Variation	–	
Parameters	Allowed	
	Comment	True - clearance of event is allowed False - clearance of event is not allowed
	Type	boolean
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋(*SRS_Diag_04117*)

8.6.2.3 CallbackEventStatusChange

[SWS_Dem_00615] ⌈The Dem Service Component shall provide the interface CallbackEventStatusChange as defined below (to trigger SW-Cs on event status byte changes), if configured.

For each event, there can be several ports of this interface type.

Name	CallbackEventStatusChange
Comment	–
IsService	true
Variation	–

Operations

EventStatusChanged		
Comment	–	
Variation	–	
Parameters	EventStatusByteOld	
	Comment	–
	Type	Dem_UdsStatusByteType
	Variation	–
	Direction	IN
	EventStatusByteNew	
	Comment	–
	Type	Dem_UdsStatusByteType
	Variation	–
	Direction	IN

⌋

8.6.2.4 CallbackEventDataChanged

[SWS_Dem_00618] ⌈The Dem Service Component shall provide the interface CallbackEventDataChanged as defined below (to trigger SW-Cs on event related data changes), if configured.

For each event, there can be one port of this interface type.

Name	CallbackEventDataChanged
Comment	–
IsService	true
Variation	–

Operations

EventDataChanged	
Comment	–

Variation	–

⌋

8.6.2.5 CallbackGetFaultDetectCounter

[SWS_Dem_00622] ⌈The Dem Service Component shall provide the interface CallbackGetFaultDetectionCounter as defined below (to get the monitor-internal fault detection counter value of a specific event from the SW-C), if configured.

Name	CallbackGetFaultDetectCounter	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

GetFaultDetectionCounter		
Comment	–	
Variation	–	
Parameters	FaultDetectionCounter	
	Comment	Value of FaultDetectionCounter
	Type	sint8
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋

8.6.2.6 CallbackInitMonitorForEvent

[SWS_Dem_00613] ⌈The Dem Service Component shall provide the interface CallbackInitMonitorForEvent as defined below (to trigger an event-specific initialization of the monitor part of the SW-C), if configured.

For each event, there can be one port of this interface type.

Name	CallbackInitMonitorForEvent	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

InitMonitorForEvent		
Comment	–	
Variation	–	
Parameters	InitMonitorReason	
	Comment	–
	Type	Dem_InitMonitorReasonType
	Variation	–
	Direction	IN
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋

8.6.2.7 CallbackInitMonitorForFunction

[SWS_Dem_00614] ⌈The Dem Service Component shall provide the interface CallbackInitMonitorForFunction as defined below, if configured.

Name	CallbackInitMonitorForFunction	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

InitMonitorForFunction		
Comment	–	
Variation	–	
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋

8.6.2.8 CallbackDTCStatusChange

[SWS_Dem_00617] ⌈The Dem Service Component shall provide the interface CallbackDTCStatusChange as defined below (to trigger SW-Cs on DTC status byte changes), if configured via DemCallbackDTCStatusChanged, DemCallbackOBDDTCStatusChanged, and / or DemCallbackJ1939DTCStatusChanged.

There can be several ports of this interface type, provided globally by the Dem Service Component.

Name	CallbackDTCStatusChange
Comment	–

IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

DTCStatusChanged		
Comment	–	
Variation	–	
Parameters	DTC	
	Comment	–
	Type	uint32
	Variation	–
	Direction	IN
	DTCStatusOld	
	Comment	–
	Type	Dem_UdsStatusByteType
	Variation	–
	Direction	IN
	DTCStatusNew	
	Comment	–
	Type	Dem_UdsStatusByteType
	Variation	–
	Direction	IN
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋

8.6.2.9 GeneralCallbackEventDataChanged

[SWS_Dem_00619] ⌈The Dem module shall provide the interface GeneralCallbackEventDataChanged as defined below (to also trigger SW-Cs on event related data changes, which are using this information in combination with the interface GeneralDiagnosticInfo), if at least on port CBDataEvt_<EventName> is configured.

One global port of this interface type is provided by the Dem Service Component.

Name	GeneralCallbackEventDataChanged
Comment	–
IsService	true
Variation	–

Operations

EventDataChanged	
Comment	–
Variation	–

Parameters	EventId	
	Comment	–
	Type	Dem_EventIdType
	Variation	–
	Direction	IN

⌋

8.6.2.10 GeneralCallbackEventStatusChange

[SWS_Dem_00616] ⌈The Dem module shall provide the interface GeneralCallbackEventStatusChange as defined below (to also trigger SW-Cs on event status byte changes, which are using this information in combination with the interface GeneralDiagnosticInfo), if at least on port CBStatusEvt_<EventName>_<SWC> is configured. One global port of this interface type is provided by the Dem Service Component.

Name	GeneralCallbackEventStatusChange
Comment	–
IsService	true
Variation	–

Operations

EventStatusChanged		
Comment	–	
Variation	–	
Parameters	EventId	
	Comment	–
	Type	Dem_EventIdType
	Variation	–
	Direction	IN
	EventStatusByteOld	
	Comment	–
	Type	Dem_UdsStatusByteType
	Variation	–
	Direction	IN
	EventStatusByteNew	
	Comment	–
	Type	Dem_UdsStatusByteType
	Variation	–
	Direction	IN

⌋(*SRS_Diag_04061*)

8.6.2.11 Service Interface CddIf

[SWS_Dem_00666] ⌈The Dem Service Component shall provide the interface CddIf as defined below (to provide the operations only related to complex device drivers). One port of this interface type is provided globally by the Dem Service Component.

Note: This port (Cdd) can only be connected to the respective port of the CDD module (refer to note below [SWS_Dem_00659]).

Name	CddIf	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	DEM_CLEAR_WRONG_DTC
	2	DEM_CLEAR_WRONG_DTCORIGIN
	3	DEM_CLEAR_FAILED
	4	DEM_CLEAR_PENDING
	5	DEM_CLEAR_BUSY

Operations

ClearDTC		
Comment	–	
Variation	–	
Parameters	DTC	
	Comment	–
	Type	uint32
	Variation	–
	Direction	IN
	DTCFormat	
	Comment	–
	Type	Dem_DTCFormatType
	Variation	–
	Direction	IN
	DTCOrigin	
	Comment	–
	Type	Dem_DTCOriginType
	Variation	–
	Direction	IN
Possible Errors	E_OK	Operation successful
	DEM_CLEAR_WRONG_DTC	
	DEM_CLEAR_WRONG_DTCORIGIN	
	DEM_CLEAR_FAILED	
	DEM_CLEAR_PENDING	
	DEM_CLEAR_BUSY	

⌋

8.6.2.12 Service Interface DTCSuppression

[SWS_Dem_00608] ⌈The Dem Service Component shall provide the interface DTC-Suppression as defined below (to provide the capability to control the suppression of DTCs).

One port of this interface type is provided globally by the Dem Service Component.

Name	DTCSuppression	
Comment	–	
IsService	true	
Variation	(({ecuc(Dem/DemGeneral/DemSuppressionSupport)} == DEM_DTC_SUPPRESSION) ||({ecuc(Dem/DemGeneral/ DemSuppressionSupport)} == DEM_EVENT_AND_DTC_SUPPRESSION))	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

SetDTCSuppression		
Comment	–	
Variation	–	
Parameters	DTC	
	Comment	–
	Type	uint32
	Variation	–
	Direction	IN
	DTCFormat	
	Comment	–
	Type	Dem_DTCFormatType
	Variation	–
	Direction	IN
	SuppressionStatus	
	Comment	–
	Type	boolean
	Variation	–
	Direction	IN
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋(*SRS_Diag_04030*)

8.6.2.13 DataServices_<SyncDataElement>

[SWS_Dem_00621] ⌈The Dem Service Component shall provide the interface DataServices_<SyncDataElement> as defined below (to get the data element value contained in a DID, a PID, or an extended data record from the respective SW-C via client/server or sender/receiver communication, refer to Figure 7.45), if configured.

For each data element, one port of this interface type is provided by the SW-Cs.

Name	DataServices	
Comment	–	
IsService	true	
Variation	(({ecuc(Dem/DemGeneral/DemDataElementClass)} instanceof {ecuc(Dem/DemGeneral/DemDataElementClass/DemExternalCSDataElementClass)})&& ({ecuc(Dem/DemGeneral/DemDataElementClass/DemExternalCSDataElementClass/DemDataElementUsePort)} == true)) Data = {ecuc(Dem/DemGeneral/DemDataElementClass/SHORT-NAME)}	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

ReadData		
Comment	The server is not allowed to return E_NOT_OK, but shall always provide a valid data value (e.g. a default/replacement value in an error-case) to Dcm/Dem nevertheless the signature of the operation includes E_NOT_OK to ensure compatibility between server runnable and RTE Call API, since the RTE may return negative Std_Return values in certain cases (e.g. partition of server stopped)	
Variation	–	
Parameters	Data	
	Comment	–
	Type	DataArrayType
	Variation	Data = {ecuc(Dem/DemGeneral/DemDataElementClass/SHORT-NAME)}
	Direction	OUT
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋(*SRS_Diag_04061*)

Note: The Dem- and Dcm-interfaces of this type are compatible, to be connected to the same provide-port of the application.

Note: The Dem Service Component supports synchronous client/server interfaces only (due to the used mechanisms for the event memory) and is compatible with the Dcm interface DataServices_<Data> with the setting for USE_DATA_SYNCH_CLIENT_SERVER. All further operations contained in the Dcm interface CSDataServices_<Data> like WriteData, ReadDataLength (relates to data elements with a variable length), ConditionCheckRead, etc. are provided by the SW-C and used by the Dcm, but are not required/considered by the Dem module.

Note: The Dem Service Component is compatible with the Dcm interface DataServices_<Data> with the setting for USE_DATA_SENDER_RECEIVER.

8.6.2.14 DcmIf

[SWS_Dem_00609] ⌈The Dem Service Component shall provide the interface DcmIf as defined below (to provide the operations only related to the Dcm).

One port of this interface type is provided globally by the Dem Service Component.

Name	DcmIf	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	DEM_CONTROL_DTC_SETTING_N_OK
	1	DEM_CLEAR_WRONG_DTC
	2	DEM_CONTROL_DTC_WRONG_DTCGROUP
	2	DEM_CLEAR_WRONG_DTCORIGIN
	3	DEM_CLEAR_FAILED
	4	DEM_CLEAR_PENDING
	5	DEM_CLEAR_BUSY

Operations

DcmClearDTC		
Comment	–	
Variation	–	
Parameters	DTC	
	Comment	–
	Type	uint32
	Variation	–
	Direction	IN
	DTCFormat	
	Comment	–
	Type	Dem_DTCFormatType
	Variation	–
	Direction	IN
	DTCOrigin	
	Comment	–
	Type	Dem_DTCOriginType
	Variation	–
	Direction	IN
Possible Errors	E_OK	Operation successful
	DEM_CLEAR_WRONG_DTC	
	DEM_CLEAR_WRONG_DTCORIGIN	
	DEM_CLEAR_FAILED	
	DEM_CLEAR_PENDING	
	DEM_CLEAR_BUSY	
DcmEnableDTCSetting		
Comment	–	
Variation	–	
Parameters	DTCKind	
	Comment	–

	Type	Dem_DTCKindType
	Variation	–
	Direction	IN
	DTCGroup	
	Comment	–
	Type	uint32
	Variation	–
	Direction	IN
Possible Errors	E_OK	Operation successful
	DEM_CONTROL_DTC_SETTING_N_OK	
	DEM_CONTROL_DTC_WRONG_DTCGROUP	

⌋

Note: This port (Dcm) can only be connected to the respective port of the Dcm module (refer to note below [SWS_Dem_00009] and [SWS_Dem_00035]).

8.6.2.15 DTRCentralReport

[SWS_Dem_00769] ⌈The Dem Service Component shall provide the interface DTRCentralReport as defined), if OBD support is configured.

Name	DTRCentralReport	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

SetDTR		
Comment	–	
Variation	–	
Parameters	TestResult	
	Comment	–
	Type	sint32
	Variation	–
	Direction	IN
	LowerLimit	
	Comment	–
	Type	sint32
	Variation	–
	Direction	IN
	UpperLimit	
	Comment	–
	Type	sint32
	Variation	–
	Direction	IN
	CtrlVal	
	Comment	–

	Type	Dem_DTRControlType
	Variation	–
	Direction	IN
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋

8.6.2.16 EnableCondition

[SWS_Dem_00604] ⌈The Dem Service Component shall provide the interface EnableCondition as defined below (to provide the capability to set an enable condition), if at least one enable condition is configured.

One port of this interface type is provided per enable condition by the Dem Service Component. It has EnableConditionId as a port-defined argument.

Name	EnableCondition	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

SetEnableCondition		
Comment	–	
Variation	–	
Parameters	ConditionFulfilled	
	Comment	–
	Type	boolean
	Variation	–
	Direction	IN
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋

8.6.2.17 DiagnosticMonitor

[SWS_Dem_00598] ⌈The Dem Service Component shall provide the interface DiagnosticMonitor as defined below (to provide the capability to modify the event information). One port of this interface type is provided per application-related diagnostic event by the Dem Service Component. It has EventId as a port-defined argument.

Name	DiagnosticMonitor
Comment	–

IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

ClearPrestoredFreezeFrame		
Comment	–	
Variation	{ecuc(Dem/DemConfigSet/DemEventParameter/DemEventClass/ DemFFPrestorageSupported)} == true	
Possible Errors	E_OK	Operation successful
	E_NOT_OK	
PrestoreFreezeFrame		
Comment	–	
Variation	{ecuc(Dem/DemConfigSet/DemEventParameter/DemEventClass/ DemFFPrestorageSupported)} == true	
Possible Errors	E_OK	Operation successful
	E_NOT_OK	
ResetEventDebounceStatus		
Comment	–	
Variation	(({ecuc(Dem/DemConfigSet/DemEventParameter/DemEventClass/ DemDebounceAlgorithmClass)} instanceof {ecuc(Dem/ DemConfigSet/DemEventParameter/DemEventClass/ DemDebounceAlgorithmClass/DemDebounceCounterBased)}) ||({ecuc(Dem/DemConfigSet/DemEventParameter/DemEventClass/ DemDebounceAlgorithmClass)} instanceof {ecuc(Dem/ DemConfigSet/DemEventParameter/DemEventClass/ DemDebounceAlgorithmClass/DemDebounceTimeBase)}))	
Parameters	DebounceResetStatus	
	Comment	–
	Type	Dem_DebounceResetStatusType
	Variation	–
	Direction	IN
Possible Errors	E_OK	Operation successful
	E_NOT_OK	
ResetEventStatus		
Comment	–	
Variation	–	
Possible Errors	E_OK	Operation successful
	E_NOT_OK	
SetEventDisabled		
Comment	–	
Variation	{ecuc(Dem/DemGeneral/DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT	
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

SetEventStatus		
Comment	–	
Variation	–	
Parameters	EventStatus	
	Comment	–
	Type	Dem_EventStatusType
	Variation	–
	Direction	IN
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋

Note: Each port of the DiagnosticMonitor interface is only connected to one monitor port.

8.6.2.18 EventStatus

[SWS_Dem_00838] ⌈The Dem Service Component shall provide the interface EventStatus as defined below (to provide the capability modify the event status).

One port of this interface type is provided per application-related diagnostic event by the Dem Service Component. It has EventId as a port-defined argument.

Name	EventStatus	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

SetWIRStatus		
Comment	–	
Variation	–	
Parameters	WIRStatus	
	Comment	–
	Type	boolean
	Variation	–
	Direction	IN
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋

Note: Each port of the EventStatus interface is only connected to one port of the failsafe SW-C.

8.6.2.19 DiagnosticInfo

[SWS_Dem_00599] ⌈The Dem Service Component shall provide the interface DiagnosticInfo as defined below (to provide the capability to obtain the event information).

One port of this interface type is provided per diagnostic event by the Dem Service Component. It has EventId as a port-defined argument.

Name	DiagnosticInfo	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK
	2	DEM_E_NO_DTC_AVAILABLE
	3	DEM_E_NO_FDC_AVAILABLE
	48	DEM_E_NODATAAVAILABLE
	49	DEM_E_WRONG_RECORDNUMBER
	50	DEM_E_WRONG_DIDNUMBER

Operations

GetDTCOfEvent		
Comment	–	
Variation	–	
Parameters	DTCFormat	
	Comment	–
	Type	Dem_DTCFormatType
	Variation	–
	Direction	IN
	DTCOfEvent	
	Comment	–
	Type	uint32
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	E_NOT_OK	
	DEM_E_NO_DTC_AVAILABLE	there is no DTC configured in the requested format
GetDebouncingOfEvent		
Comment	–	
Variation	(({ecuc(Dem/DemConfigSet/DemEventParameter/DemEventClass/DemDebounceAlgorithmClass)} instanceof {ecuc(Dem/DemConfigSet/DemEventParameter/DemEventClass/DemDebounceAlgorithmClass/DemDebounceCounterBased)}) ||({ecuc(Dem/DemConfigSet/DemEventParameter/DemEventClass/DemDebounceAlgorithmClass)} instanceof {ecuc(Dem/DemConfigSet/DemEventParameter/DemEventClass/DemDebounceAlgorithmClass/DemDebounceTimeBase)}))	
Parameters	DebouncingState	

	Comment	Bit 0 Temporarily Defective (corresponds to 0 < FDC < 127) Bit 1 finally Defective (corresponds to FDC = 127) Bit 2 temporarily healed (corresponds to -128 < FDC < 0) Bit 3 Test complete (corresponds to FDC = -128 or FDC = 127) Bit 4 DTR Update (= Test complete && Debouncing complete && enable conditions / storage conditions fulfilled)
	Type	Dem_DebouncingStateType
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

GetEventExtendedDataRecord		
Comment	–	
Variation	–	
Parameters	RecordNumber	
	Comment	–
	Type	uint8
	Variation	–
	Direction	IN
	DestBuffer	
	Comment	–
	Type	Dem_MaxDataValueType
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	DEM_E_NODATAAVAILABLE	The requested event data is not currently stored (but the request was valid)
	DEM_E_WRONG_RECORDNUMBER	The requested record number is not supported by the event

GetEventFailed		
Comment	–	
Variation	–	
Parameters	EventFailed	
	Comment	TRUE - Last Failed FALSE - not Last Failed
	Type	boolean
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

GetEventFreezeFrameData		
Comment	–	
Variation	–	
Parameters	RecordNumber	
	Comment	–
	Type	uint8

	Variation	–
	Direction	IN
	ReportTotalRecord	
	Comment	–
	Type	boolean
	Variation	–
	Direction	IN
	DataId	
	Comment	–
	Type	uint16
	Variation	–
	Direction	IN
	DestBuffer	
	Comment	–
	Type	Dem_MaxDataValueType
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	DEM_E_NODATAAVAILABLE	The requested event data is not currently stored (but the request was valid)
	DEM_E_WRONG_RECORDNUMBER	The requested record number is not supported by the event
	DEM_E_WRONG_DIDNUMBER	The requested DID is not supported by the freeze frame
GetEventStatus		
Comment	–	
Variation	–	
Parameters	EventStatusByte	
	Comment	–
	Type	Dem_UdsStatusByteType
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	E_NOT_OK	
GetEventTested		
Comment	–	
Variation	–	
Parameters	EventTested	
	Comment	TRUE - event tested this cycle FALSE - event not tested this cycle
	Type	boolean
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	E_NOT_OK	
GetFaultDetectionCounter		
Comment	–	

Variation	–	
Parameters	FaultDetectionCounter	
	Comment	–
	Type	sint8
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	E_NOT_OK	
	DEM_E_NO_FDC_AVAILABLE	there is no fault detection counter available for the requested event

⌋

Note: These DiagnosticInfo ports are also available for each BSW event (so that also SW-Cs can access on).

8.6.2.20 GeneralDiagnosticInfo

[SWS_Dem_00600] ⌈The Dem Service Component shall provide the interface GeneralDiagnosticInfo as defined below (to also provide the capability to obtain event information, but in comparision with the DiagnosticInfo interface, it is provided for those SW-Cs, which must access on these data uniformly, refer to chapter autoref-sub:InteractionWithSoftwareComponents).

One global port of this interface type is provided by the Dem Service Component.

Name	GeneralDiagnosticInfo	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK
	2	DEM_E_NO_DTC_AVAILABLE
	3	DEM_E_NO_FDC_AVAILABLE
	48	DEM_E_NODATAAVAILABLE
	49	DEM_E_WRONG_RECORDNUMBER
	50	DEM_E_WRONG_DIDNUMBER

Operations

GetDTCOfEvent		
Comment	–	
Variation	–	
Parameters	EventId	
	Comment	–
	Type	Dem_EventIdType
	Variation	–
	Direction	IN
	DTCFormat	
	Comment	–

	Type	Dem_DTCFormatType
	Variation	–
	Direction	IN
	DTCOfEvent	
	Comment	–
	Type	uint32
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	E_NOT_OK	
	DEM_E_NO_DTC_AVAILABLE	there is no DTC configured in the requested format
GetDebouncingOfEvent		
Comment	–	
Variation	(({ecuc(Dem/DemConfigSet/DemEventParameter/DemEventClass/DemDebounceAlgorithmClass)} instanceof {ecuc(Dem/DemConfigSet/DemEventParameter/DemEventClass/DemDebounceAlgorithmClass/DemDebounceCounterBased)}) ||({ecuc(Dem/DemConfigSet/DemEventParameter/DemEventClass/DemDebounceAlgorithmClass)} instanceof {ecuc(Dem/DemConfigSet/DemEventParameter/DemEventClass/DemDebounceAlgorithmClass/DemDebounceTimeBase)}))	
Parameters	EventId	
	Comment	–
	Type	Dem_EventIdType
	Variation	–
	Direction	IN
	DebouncingState	
	Comment	Bit 0 Temporarily Defective (corresponds to 0 < FDC < 127) Bit 1 finally Defective (corresponds to FDC = 127) Bit 2 temporarily healed (corresponds to -128 < FDC < 0) Bit 3 Test complete (corresponds to FDC = -128 or FDC = 127) Bit 4 DTR Update (= Test complete && Debouncing complete && enable conditions / storage conditions fulfilled)
	Type	Dem_DebouncingStateType
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	E_NOT_OK	
GetEventExtendedDataRecord		
Comment	–	
Variation	–	
Parameters	EventId	
	Comment	–
	Type	Dem_EventIdType
	Variation	–
	Direction	IN

	RecordNumber	
	Comment	–
	Type	uint8
	Variation	–
	Direction	IN
	DestBuffer	
	Comment	–
	Type	Dem_MaxDataValueType
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	DEM_E_NODATAAVAILABLE	The requested event data is not currently stored (but the request was valid)
	DEM_E_WRONG_RECORDNUMBER	The requested record number is not supported by the event

GetEventFailed		
Comment	–	
Variation	–	
Parameters	EventId	
	Comment	–
	Type	Dem_EventIdType
	Variation	–
	Direction	IN
	EventFailed	
	Comment	–
	Type	boolean
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

GetEventFreezeFrameData		
Comment	–	
Variation	–	
Parameters	EventId	
	Comment	–
	Type	Dem_EventIdType
	Variation	–
	Direction	IN
	RecordNumber	
	Comment	–
	Type	uint8
	Variation	–
	Direction	IN
	ReportTotalRecord	
	Comment	–
	Type	boolean
	Variation	–
	Direction	IN
	DataId	

	Comment	–
	Type	uint16
	Variation	–
	Direction	IN
	DestBuffer	
	Comment	–
	Type	Dem_MaxDataValueType
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	DEM_E_NODATAAVAILABLE	The requested event data is not currently stored (but the request was valid)
	DEM_E_WRONG_RECORDNUMBER	The requested record number is not supported by the event
	DEM_E_WRONG_DIDNUMBER	The requested DID is not supported by the freeze frame
GetEventStatus		
Comment	–	
Variation	–	
Parameters	EventId	
	Comment	–
	Type	Dem_EventIdType
	Variation	–
	Direction	IN
	EventStatusByte	
	Comment	–
	Type	Dem_UdsStatusByteType
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	E_NOT_OK	
GetEventTested		
Comment	–	
Variation	–	
Parameters	EventId	
	Comment	–
	Type	Dem_EventIdType
	Variation	–
	Direction	IN
	EventTested	
	Comment	–
	Type	boolean
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	E_NOT_OK	
GetFaultDetectionCounter		

Comment	–	
Variation	–	
Parameters	EventId	
	Comment	–
	Type	Dem_EventIdType
	Variation	–
	Direction	IN
	FaultDetectionCounter	
	Comment	–
	Type	sint8
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	E_NOT_OK	
	DEM_E_NO_FDC_AVAILABLE	there is no fault detection counter available for the requested event

⌋

8.6.2.21 ExternalAgingCycle

[SWS_Dem_00603] ⌈The Dem Service Component shall provide the interface ExternalAgingCycle as defined below (to provide the capability to set the current value of the aging counter Dem-externally).

One global port of this interface type is provided by the Dem Service Component.

Name	ExternalAgingCycle	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

SetAgingCycleCounterValue		
Comment	–	
Variation	–	
Parameters	CounterValue	
	Comment	–
	Type	uint8
	Variation	–
	Direction	IN
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋*(SRS_Diag_04030)*

8.6.2.22 IndicatorStatus

[SWS_Dem_00606] ⌈The Dem Service Component shall provide the interface IndicatorStatus as defined below (to provide the capability to set the status of an indicator), if at least one indicator is configured.

One port of this interface type is provided per indicator by the Dem Service Component. It has IndicatorId as a port-defined argument.

Name	IndicatorStatus	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

GetIndicatorStatus		
Comment	–	
Variation	–	
Parameters	IndicatorStatus	
	Comment	–
	Type	Dem_IndicatorStatusType
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	E_NOT_OK	
SetIndicatorStatus		
Comment	–	
Variation	{ecuc(Dem/DemGeneral/DemOBDSupport)} == DEM_OBD_MASTER_ECU	
Parameters	IndicatorStatus	
	Comment	–
	Type	Dem_IndicatorStatusType
	Variation	–
	Direction	IN
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋(*SRS_Diag_04128*)

8.6.2.23 IUMPRDenominator

[SWS_Dem_00611] ⌈The Dem Service Component shall provide the interface IUMPRDenominator as defined below (to provide the capability to define the number of times the vehicle operation has been fulfilled), if OBD support is configured.

One port of this interface type is provided per ratio Id by the Dem Service Component. It has RatioID as a port-defined argument.

Name	IUMPRDenominator	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

ReplUMPRDenLock		
Comment	–	
Variation	–	
Possible Errors	E_OK	Operation successful
	E_NOT_OK	
ReplUMPRDenRelease		
Comment	–	
Variation	–	
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋

8.6.2.24 IUMPRDenominatorCondition

[SWS_Dem_00742] ⌈The Dem Service Component shall provide the interface IUMPRDenominatorCondition as defined below (to broadcast the status information of the General Denominator and additional denominator conditions among all OBD relevant ECUs), if OBD support is configured.

One port of this interface type is provided per denominator condition Id by the Dem Service Component. It has Dem_IumprDenomCondId as a port-defined argument.

Name	IUMPRDenominatorCondition	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

GetIUMPRDenCondition		
Comment	–	
Variation	–	
Parameters	ConditionStatus	
	Comment	–
	Type	Dem_IumprDenomCondStatusType
	Variation	–
	Direction	OUT

Possible Errors	E_OK	Operation successful
	E_NOT_OK	
SetIUMPRDenCondition		
Comment	–	
Variation	–	
Parameters	ConditionStatus	
	Comment	–
	Type	Dem_IumprDenomCondStatusType
	Variation	–
	Direction	IN
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋

8.6.2.25 IUMPRNumerator

[SWS_Dem_00610] ⌈The Dem Service Component shall provide the interface IUMPR-Numerator as defined below (to provide the capability to define the number of times a fault could have been found), if OBD support is configured.

One port of this interface type is provided per ratio Id by the Dem Service Component. It has RatioID as a port-defined argument.

Name	IUMPRNumerator	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

ReplUMPRFaultDetect		
Comment	–	
Variation	–	
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋

8.6.2.26 OperationCycle

[SWS_Dem_00601] ⌈The Dem Service Component shall provide the interface Operation-Cycle as defined below (to provide the capability to set the state of an operation cycle).

One port of this interface type is provided per operation cycle by the Dem Service Component. It has OperationCycleId as a port-defined argument.

Name	OperationCycle	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

GetOperationCycleState		
Comment	–	
Variation	–	
Parameters	CycleState	
	Comment	–
	Type	Dem_OperationCycleStateType
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	E_NOT_OK	
SetOperationCycleState		
Comment	–	
Variation	–	
Parameters	CycleState	
	Comment	–
	Type	Dem_OperationCycleStateType
	Variation	–
	Direction	IN
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋(*SRS_Diag_04076*)

8.6.2.27 EvMemOverflowIndication

[SWS_Dem_00607] ⌈The Dem Service Component shall provide the interface EvMemOverflowIndication as defined below (to provide the status of the event memory), if the respective event memory is configured.

One port of this interface type is provided per supported event memory by the Dem Service Component. It has DTCOrigin as a port-defined argument.

Name	EvMemOverflowIndication	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK

	1	E_NOT_OK

Operations

GetEventMemoryOverflow		
Comment	–	
Variation	–	
Parameters	OverflowIndication	
	Comment	–
	Type	boolean
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	E_NOT_OK	
GetNumberOfEventMemoryEntries		
Comment	–	
Variation	–	
Parameters	GetNumberOfEventMemoryEntries	
	Comment	–
	Type	uint8
	Variation	–
	Direction	OUT
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋(*SRS_Diag_04105*)

8.6.2.28 PfcCyclePfcCycleQualified

[SWS_Dem_00743] ⌈The Dem Service Component shall provide the interface PfcCyclePfcCycleQualified as defined below, if the interface is used by a SW-C.

Name	PfcCyclePfcCycleQualified	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

GetPfcCycleQualified		
Comment	–	
Variation	–	
Parameters	pfcCycleisqualified	
	Comment	–
	Type	boolean
	Variation	–
	Direction	IN

Possible Errors	E_OK	Operation successful
	E_NOT_OK	
SetPfcCycleSetPfcCycleQualified		
Comment	–	
Variation	–	
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋

8.6.2.29 PowerTakeOff

[SWS_Dem_00612] ⌈The Dem Service Component shall provide the interface PowerTakeOff as defined below (to provide the capability to set the PTO status), if OBD support is configured.

One port of this interface type is provided by the Dem Service Component.

Name	PowerTakeOff	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

SetPtoStatus		
Comment	–	
Variation	–	
Parameters	PtoStatus	
	Comment	–
	Type	boolean
	Variation	–
	Direction	IN
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋

8.6.2.30 SetClearDTC

[SWS_Dem_00744] ⌈The Dem Service Component shall provide the interface SetClearDTC as defined below, if the interface is used by a SW-C.

Name	SetClearDTC
Comment	–

IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

SetClearDTC		
Comment	–	
Variation	–	
Parameters	DTC	
	Comment	–
	Type	uint32
	Variation	–
	Direction	IN
	DTCFormat	
	Comment	–
	Type	Dem_DTCFormatType
	Variation	–
	Direction	IN
	DTCOrigin	
	Comment	–
	Type	Dem_DTCOriginType
	Variation	–
	Direction	IN
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋(*SRS_Diag_04030*)

8.6.2.31 SetDataOfPID21

[SWS_Dem_00745] ⌈The Dem Service Component shall provide the interface SetDataOfPID21 as defined below, if the interface is used by a SW-C.

Name	SetDataOfPID21	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

SetDataOfPID21		
Comment	–	
Variation	–	
Parameters	PID21value	
	Comment	–
	Type	Dem_PID21valueType

	Variation	–
	Direction	IN
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋(*SRS_Diag_04061*, *SRS_Diag_04001*)

8.6.2.32 SetDataOfPID31

[SWS_Dem_00746] ⌈The Dem Service Component shall provide the interface SetDataOfPID31 as defined below, if the interface is used by a SW-C.

Name	SetDataOfPID31	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK
	1	E_NOT_OK

Operations

SetDataOfPID31		
Comment	–	
Variation	–	
Parameters	PID31value	
	Comment	–
	Type	Dem_PID31valueType
	Variation	–
	Direction	IN
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋(*SRS_Diag_04082*)

8.6.2.33 StorageCondition

[SWS_Dem_00605] ⌈The Dem Service Component shall provide the interface StorageCondition as defined below (to provide the capability to set an enable condition) if at least one storage condition is configured.

One port of this interface type is provided per storage condition by the Dem Service Component. It has StorageConditionId as a port-defined argument.

Name	StorageCondition	
Comment	–	
IsService	true	
Variation	–	
Possible Errors	0	E_OK

	1	E_NOT_OK

Operations

SetStorageCondition		
Comment	–	
Variation	–	
Parameters	ConditionFulfilled	
	Comment	This parameter specifies whether the enable condition assigned to the EnableConditionID is fulfilled (TRUE) or not fulfilled (FALSE).
	Type	boolean
	Variation	–
	Direction	IN
Possible Errors	E_OK	Operation successful
	E_NOT_OK	

⌋(*SRS_Diag_04095*)

8.6.3 Implementation Data Types

8.6.4 Ports

8.6.4.1 Dem_AgingCycle

[SWS_Dem_01001] ⌈

Name	AgingCycle_{Name}		
Kind	ProvidedPort	***Interface***	AgingCycle
Description	–		
Variation	(({ecuc(Dem/DemGeneral/DemAgingCycleCounterProcessing)} == DEM_PROCESS_AGINGCTR_INTERN) && ({ecuc(Dem/DemConfigSet/DemEventParameter/DemEventClass/DemAgingCycleRef)} instanceof {ecuc(Dem/DemGeneral/DemAgingCycle)})) Name = {ecuc(Dem/DemGeneral/DemAgingCycle/SHORT-NAME)}		

⌋

8.6.4.2 Dem_CBDataEvt

[SWS_Dem_01003] ⌈

Name	CBDataEvt_{Name}		
Kind	RequiredPort	***Interface***	CallbackEventDataChanged
Description	–		

Variation	(({ecuc(Dem/DemConfigSet/DemEventParameter/ DemCallbackEventDataChanged)} != NULL) &&({ecuc(Dem/ DemConfigSet/DemEventParameter/ DemCallbackEventDataChanged/ DemCallbackEventDataChangedFnc)} == NULL)) Name = {ecuc(Dem/DemConfigSet/DemEventParameter/ SHORT-NAME)}

⌋

8.6.4.3 Dem_CBFaultDetectCtr

[SWS_Dem_01004] ⌈

Name	CBFaultDetectCtr_{Name}		
Kind	RequiredPort	***Interface***	CallbackGetFaultDetectCounter
Description	–		
Variation	Name = {ecuc(Dem/DemConfigSet/DemEventParameter/ DemEventClass/DemDebounceAlgorithmClass/ DemDebounceMonitorInternal/DemCallbackGetFDC/ SHORT-NAME)}		

⌋

8.6.4.4 Dem_CBInitEvt

[SWS_Dem_01005] ⌈

Name	CBInitEvt_{Name}		
Kind	RequiredPort	***Interface***	CallbackInitMonitorForEvent
Description	–		
Variation	(({ecuc(Dem/DemConfigSet/DemEventParameter/ DemCallbackInitMForE)} != NULL) &&({ecuc(Dem/DemConfigSet/ DemEventParameter/DemCallbackInitMForE/ DemCallbackInitMForEFnc)} == NULL)) Name = {ecuc(Dem/DemConfigSet/DemEventParameter/ SHORT-NAME)}		

⌋

8.6.4.5 Dem_CBInitFct

[SWS_Dem_01006] ⌈

Name	CBInitFct_{Name}		
Kind	RequiredPort	***Interface***	CallbackInitMonitorForFunction
Description	–		

Variation	(({ecuc(Dem/DemConfigSet/DemDTCAttributes/ DemCallbackInitMForF)} != NULL) &&({ecuc(Dem/DemConfigSet/ DemDTCAttributes/DemCallbackInitMForF/ DemCallbackInitMForFFnc)} == NULL)) Name = {ecuc(Dem/DemConfigSet/DemDTCAttributes/ DemCallbackInitMForF/SHORT-NAME)}

⌋

8.6.4.6 Dem_CBStatusDTC

[SWS_Dem_01007] ⌈

Name	CBStatusDTC_{Name}		
Kind	RequiredPort	***Interface***	CallbackDTCStatusChange
Description	–		
Variation	Name = {ecuc(Dem/DemGeneral/DemCallbackDTCStatusChanged/ SHORT-NAME)}		

⌋

8.6.4.7 Dem_CBStatusEvt

[SWS_Dem_01008] ⌈

Name	CBStatusEvt_{EventName}_{CallbackName}		
Kind	RequiredPort	***Interface***	CallbackEventStatusChange
Description	–		
Variation	EventName = {ecuc(Dem/DemConfigSet/DemEventParameter/ SHORT-NAME)} CallbackName = {ecuc(Dem/DemConfigSet/DemEventParameter/ DemCallbackEventStatusChanged/SHORT-NAME)}		

⌋

8.6.4.8 Dem_Cdd

[SWS_Dem_01009] ⌈

Name	Cdd		
Kind	ProvidedPort	***Interface***	CddIf
Description	–		
Variation	–		

⌋

8.6.4.9 Dem_ControlDTCSuppression

[SWS_Dem_01010] ⌈

Name	ControlDTCSuppression		
Kind	ProvidedPort	***Interface***	DTCSuppression
Description	–		
Variation	(({ecuc(Dem/DemGeneral/DemSuppressionSupport)} == DEM_DTC_SUPPRESSION) ||({ecuc(Dem/DemGeneral/ DemSuppressionSupport)} == DEM_EVENT_AND_DTC_SUPPRESSION))		

⌋

8.6.4.10 Dem_ControlEventSuppression

[SWS_Dem_01011] ⌈

Name	ControlEventSuppression		
Kind	ProvidedPort	***Interface***	EventSuppression
Description	–		
Variation	(({ecuc(Dem/DemGeneral/DemSuppressionSupport)} == DEM_EVENT_SUPPRESSION) ||({ecuc(Dem/DemGeneral/ DemSuppressionSupport)} == DEM_EVENT_AND_DTC_SUPPRESSION))		

⌋

8.6.4.11 DataServices

[SWS_Dem_01012] ⌈

Name	DataServices_{Data}		
Kind	RequiredPort	***Interface***	DataServices
Description	–		
Variation	Data = {ecuc(Dem/DemGeneral/DemDataElementClass/ SHORT-NAME)} (({ecuc(Dem/DemGeneral/DemDataElementClass)} instanceof {ecuc(Dem/DemGeneral/DemDataElementClass/ DemExternalCSDataElementClass)})&& ({ecuc(Dem/DemGeneral/ DemDataElementClass/DemExternalCSDataElementClass/ DemDataElementUsePort)} == TRUE))||({ecuc(Dem/DemGeneral/ DemDataElementClass)} instanceof {ecuc(Dem/DemGeneral/ DemDataElementClass/DemExternalSRDataElementClass)}) ({ecuc(Dem/DemGeneral/DemDataElementClass)} instanceof {ecuc(Dem/DemGeneral/DemDataElementClass/ DemExternalSRDataElementClass)})		

⌋

8.6.4.12 Dem_Dcm

[SWS_Dem_01040] ⌈

Name	Dcm		
Kind	ProvidedPort	***Interface***	DcmIf
Description	–		
Variation	–		

⌋

8.6.4.13 Dem_DTR

[SWS_Dem_01039] ⌈

Name	DTR_{Name}		
Kind	ProvidedPort	***Interface***	DTRCentralReport
Description	–		
Variation	Name = {ecuc(Dem/DemConfigSet/DemDtr/SHORT-NAME)}		

⌋

8.6.4.14 Dem_EnableCond

[SWS_Dem_01038] ⌈

Name	EnableCond_{Name}		
Kind	ProvidedPort	***Interface***	EnableCondition
Description	–		
Variation	Name = {ecuc(Dem/DemGeneral/DemEnableCondition/ SHORT-NAME)}		

⌋

8.6.4.15 Dem_Event

[SWS_Dem_01037] ⌈

Name	Event_{Name}		
Kind	ProvidedPort	***Interface***	DiagnosticMonitor
Description	–		
Variation	Name = {ecuc(Dem/DemConfigSet/DemEventParameter/ SHORT-NAME)}		

⌋

8.6.4.16 Dem_EventStatus

[SWS_Dem_01036] ⌈

Name	EventStatus_{Name}		
Kind	ProvidedPort	***Interface***	EventStatus
Description	–		
Variation	Name = {ecuc(Dem/DemConfigSet/DemEventParameter/ SHORT-NAME)}		

⌋

8.6.4.17 Dem_EvtInfo

[SWS_Dem_01034] ⌈

Name	EventInfo_{Name}		
Kind	ProvidedPort	***Interface***	DiagnosticInfo
Description	–		
Variation	Name = {ecuc(Dem/DemConfigSet/DemEventParameter/ DemEventId/SHORT-NAME)}		

⌋

8.6.4.18 Dem_ExtAgingCycle

[SWS_Dem_01033] ⌈

Name	ExtAgingCycle		
Kind	ProvidedPort	***Interface***	ExternalAgingCycle
Description	–		
Variation	({ecuc(Dem/DemGeneral/DemAgingCycleCounterProcessing)} == DEM_PROCESS_AGINGCTR_EXTERN)		

⌋

8.6.4.19 Dem_GeneralCBDataEvt

[SWS_Dem_01041] ⌈

Name	GeneralCBDataEvt		
Kind	RequiredPort	***Interface***	GeneralCallbackEventDataChanged
Description	–		
Variation	({ecuc(Dem/DemGeneral/DemGeneralInterfaceSupport)} == True)		

⌋

8.6.4.20 Dem_GeneralCBStatusEvt

[SWS_Dem_01032] ⌈

Name	GeneralCBStatusEvt		
Kind	RequiredPort	***Interface***	GeneralCallbackEventStatusChange
Description	–		
Variation	({ecuc(Dem/DemGeneral/DemGeneralInterfaceSupport)} == True)		

⌋

8.6.4.21 Dem_GeneralEvtInfo

[SWS_Dem_01031] ⌈

Name	GeneralEvtInfo		
Kind	ProvidedPort	***Interface***	GeneralDiagnosticInfo
Description	–		
Variation	({ecuc(Dem/DemGeneral/DemGeneralInterfaceSupport)} == True)		

⌋

8.6.4.22 Dem_IndStatus

[SWS_Dem_01030] ⌈

Name	IndStatus_{Name}		
Kind	ProvidedPort	***Interface***	IndicatorStatus
Description	–		
Variation	Name = {ecuc(Dem/DemGeneral/DemIndicator/SHORT-NAME)}		

⌋

8.6.4.23 Dem_IUMPRDenominator

[SWS_Dem_01029] ⌈

Name	IUMPRDenominator_{Name}		
Kind	ProvidedPort	***Interface***	IUMPRDenominator
Description	–		
Variation	Name = {ecuc(Dem/DemGeneral/DemRatioId/SHORT-NAME)}		

⌋

8.6.4.24 Dem_IUMPRDenominatorCondition

[SWS_Dem_01028] ⌈

Name	IUMPRDenominatorCondition_{Name}		
Kind	ProvidedPort	***Interface***	IUMPRDenominatorCondition
Description	–		
Variation	Name = {ecuc(Dem/DemGeneral/DemRatioId/SHORT-NAME)}		

⌋

8.6.4.25 Dem_IUMPRNumerator

[SWS_Dem_01027] ⌈

Name	IUMPRNumerator_{Name}		
Kind	ProvidedPort	***Interface***	IUMPRNumerator
Description	–		
Variation	Name = {ecuc(Dem/DemGeneral/DemRatioId/SHORT-NAME)}		

⌋

8.6.4.26 Dem_OpCycle

[SWS_Dem_01026] ⌈

Name	OpCycle_{Name}		
Kind	ProvidedPort	***Interface***	OperationCycle
Description	–		
Variation	Name = {ecuc(Dem/DemGeneral/DemOperationCycle/ SHORT-NAME)}		

⌋

8.6.4.27 Dem_OverflowIndMirrorMemory

[SWS_Dem_01025] ⌈

Name	OverflowIndMirrorMemory		
Kind	ProvidedPort	***Interface***	EvMemOverflowIndication
Description	–		
Variation	({ecuc(Dem/DemGeneral/DemMaxNumberEventEntryMirror)} > 0)		

⌋

8.6.4.28 Dem_OverflowIndPermanentMemory

[SWS_Dem_01024] ⌈

Name	OverflowIndPermanentMemory		
Kind	ProvidedPort	***Interface***	EvMemOverflowIndication
Description	–		
Variation	({ecuc(Dem/DemGeneral/DemMaxNumberEventEntryPermanent)} > 0)		

⌋

8.6.4.29 Dem_OverflowIndPrimaryMemory

[SWS_Dem_01023] ⌈

Name	OverflowIndPrimaryMemory		
Kind	ProvidedPort	***Interface***	EvMemOverflowIndication
Description	–		
Variation	–		

⌋

8.6.4.30 Dem_OverflowIndSecondaryMemory

[SWS_Dem_01022] ⌈

Name	OverflowIndSecondaryMemory		
Kind	ProvidedPort	***Interface***	EvMemOverflowIndication
Description	–		
Variation	({ecuc(Dem/DemGeneral/DemMaxNumberEventEntrySecondary)} > 0)		

⌋

8.6.4.31 Dem_PfcCycleQualified

[SWS_Dem_01021] ⌈

Name	PfcCycleQualified		
Kind	ProvidedPort	***Interface***	PfcCyclePfcCycleQualified
Description	–		
Variation	({ecuc(Dem/DemGeneral/DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT) &&({ecuc(Dem/DemGeneral/ DemMaxNumberEventEntryPermanent)} > 0)		

⌋

8.6.4.32 Dem_PowerTakeOffStatus

[SWS_Dem_01020] ⌈

Name	PowerTakeOffStatus		
Kind	ProvidedPort	***Interface***	PowerTakeOff
Description	–		
Variation	{ecuc(Dem/DemGeneral/DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT		

⌋

8.6.4.33 Dem_SetClearDTC_dependend

[SWS_Dem_01019] ⌈

Name	SetClearDTC_dependend		
Kind	ProvidedPort	***Interface***	SetClearDTC
Description	Port is used in dependend / secondary ECUs		
Variation	{ecuc(Dem/DemGeneral/DemOBDSupport)} == DEM_OBD_DEP_SEC_ECU		

⌋

8.6.4.34 Dem_SetClearDTC_master

[SWS_Dem_01018] ⌈

Name	SetClearDTC_master		
Kind	RequiredPort	***Interface***	SetClearDTC
Description	port is used in Master ECU		
Variation	{ecuc(Dem/DemGeneral/DemOBDSupport)} == DEM_OBD_MASTER_ECU		

⌋

8.6.4.35 Dem_SetDataOfPID21

[SWS_Dem_01017] ⌈

Name	SetDataOfPID21		
Kind	ProvidedPort	***Interface***	SetDataOfPID21
Description	–		
Variation	{ecuc(Dem/DemGeneralOBD/DemOBDCentralizedPID21Handling)} == true		

⌋

8.6.4.36 Dem_SetDataOfPID31

[SWS_Dem_01016] ⌈

Name	SetDataOfPID31		
Kind	ProvidedPort	***Interface***	SetDataOfPID31
Description	–		
Variation	{ecuc(Dem/DemGeneralOBD/DemOBDCentralizedPID31Handling)} == true		

⌋

8.6.4.37 Dem_StorageCond

[SWS_Dem_01015] ⌈

Name	StorageCond_{Name}		
Kind	ProvidedPort	***Interface***	StorageCondition
Description	–		
Variation	Name = {ecuc(Dem/DemGeneral/DemStorageCondition/ SHORT-NAME)}		

⌋

9 Sequence Diagrams

9.1 ControlDTCSetting

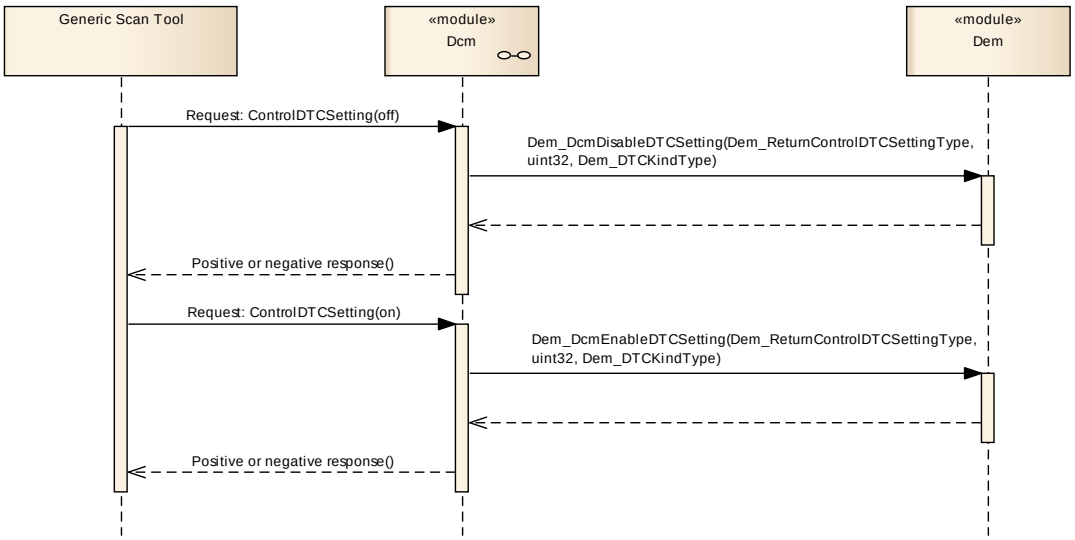

Figure 9.1: Sequence diagram of ControlDTCSetting

9.2 Dem_DcmClearDTC

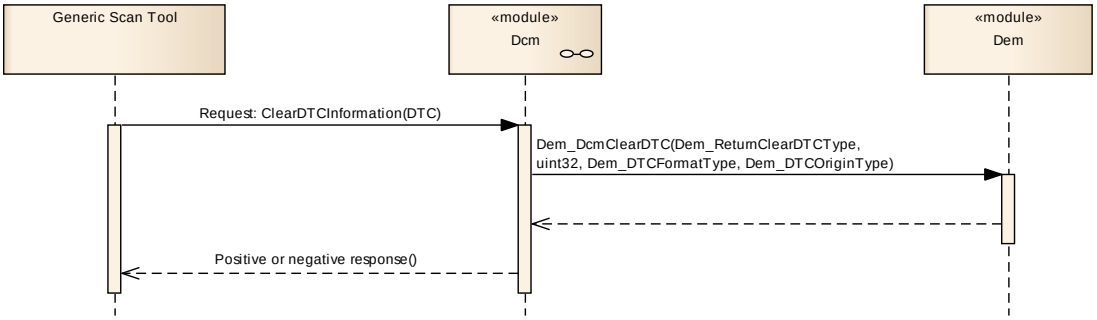

Figure 9.2: Sequence diagram of Dem_DcmClearDTC

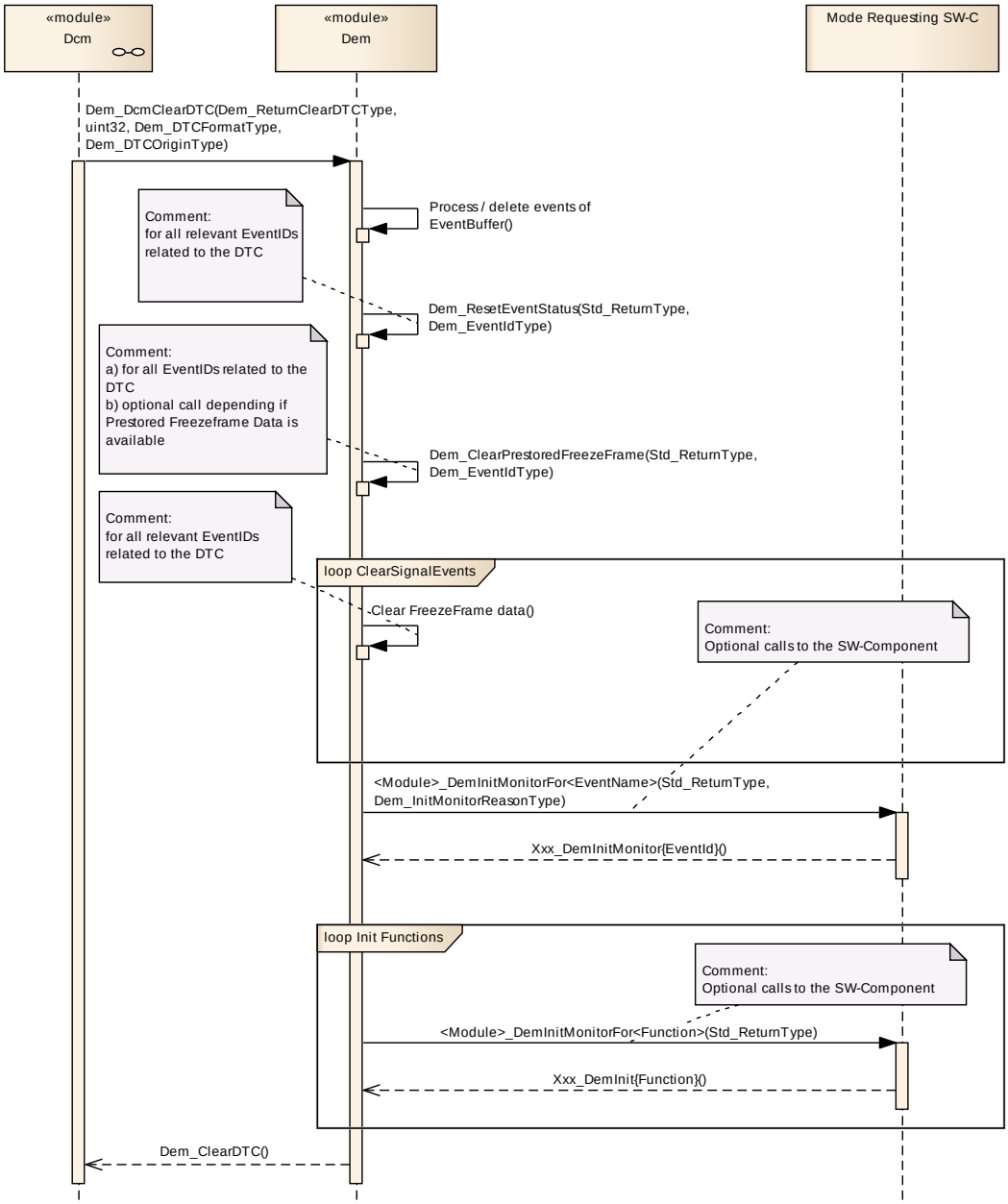

Figure 9.3: Sequence diagram of Dem_DcmClearDTC clearing a single DTC Dem-internally

9.3 Dem_DcmGetDTCByOccurrenceTime

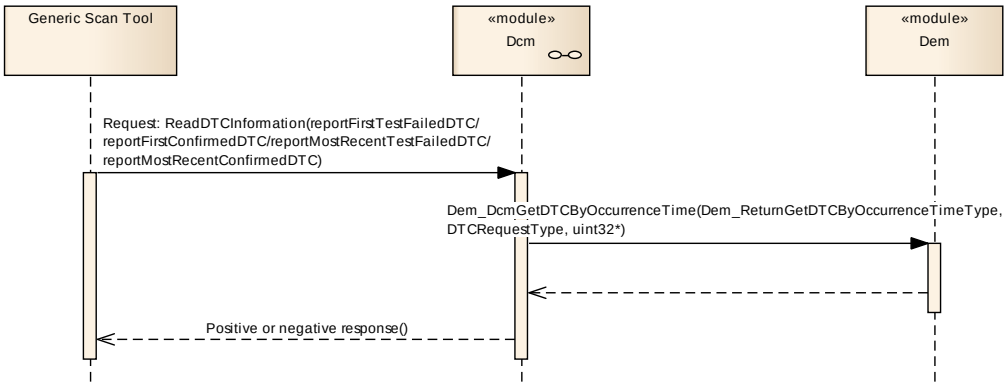

Figure 9.4: Sequence diagram of Dem_DcmGetDTCByOccurenceTime

9.4 Dem_DcmGetExtendedDataRecordByDTC

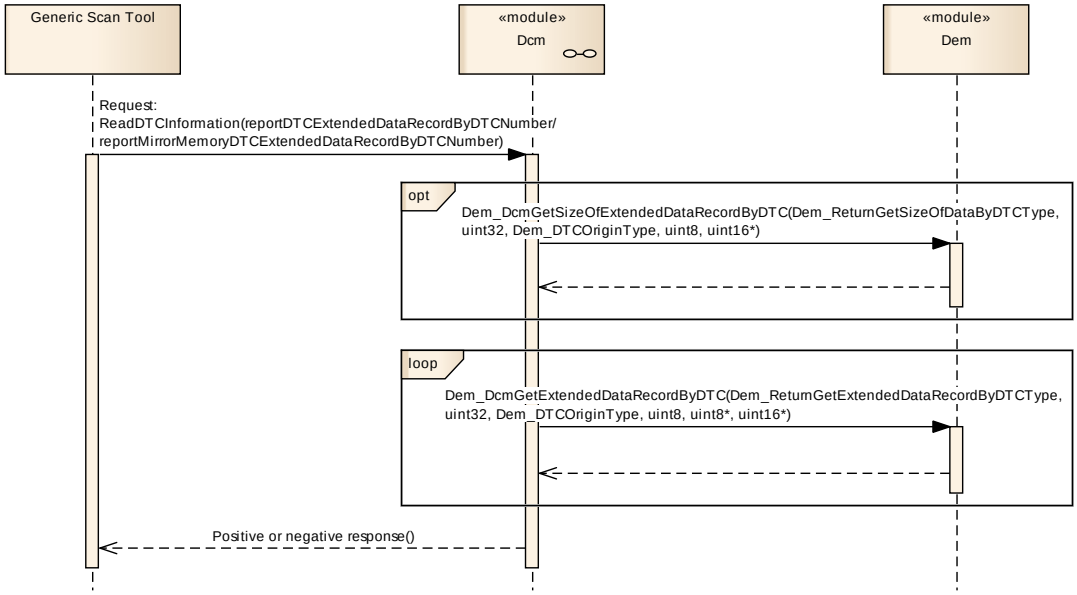

Figure 9.5: Sequence diagram of Dem_DcmGetExtendedDataRecordByDTC

9.5 Dem_DcmGetStatusOfDTC

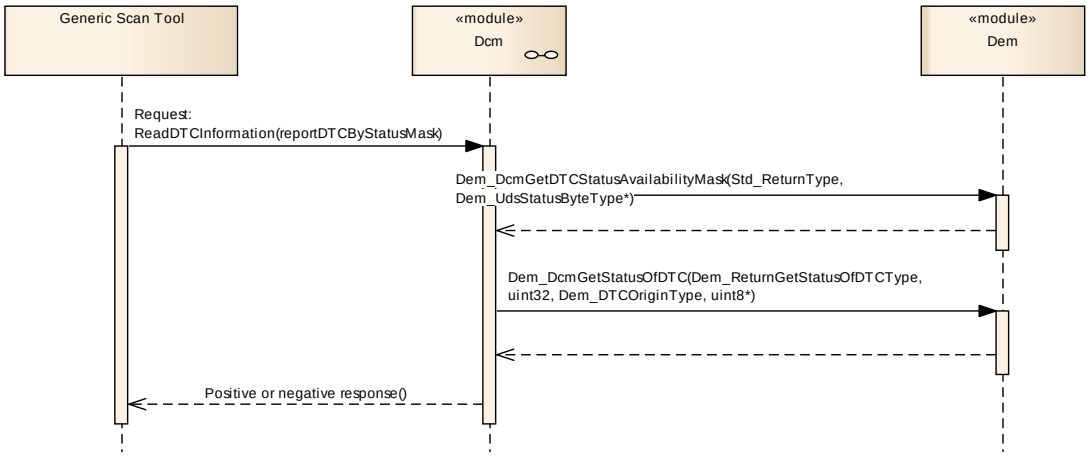

Figure 9.6: Sequence diagram of Dem_DcmGetStatusOfDTC

9.6 Dem_DcmGetFreezeFrameDataByDTC

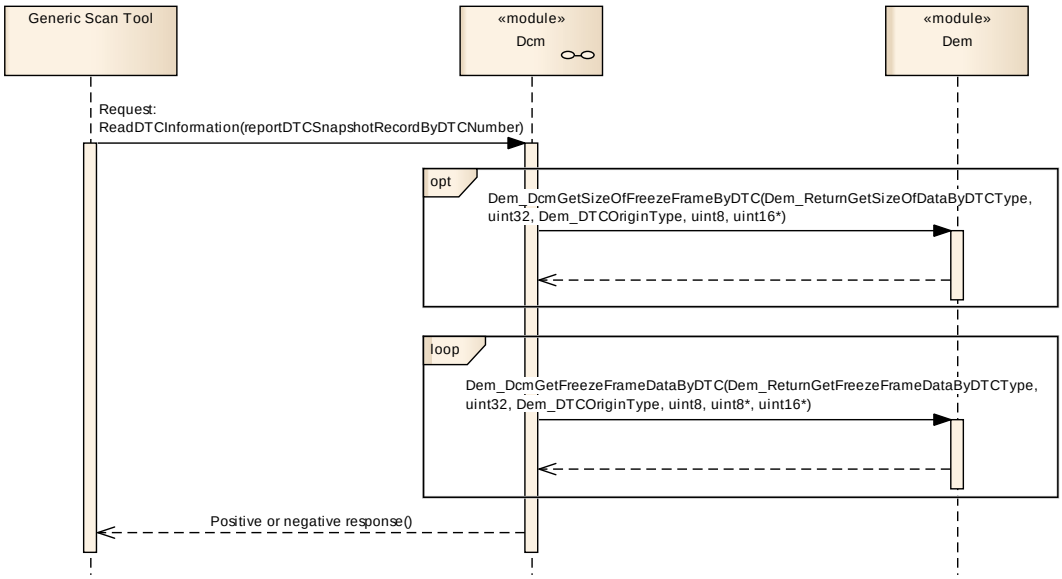

Figure 9.7: Sequence diagram of Dem_DcmGetFreezeFrameDataByDTC

9.7 GetOBDFaultInformation

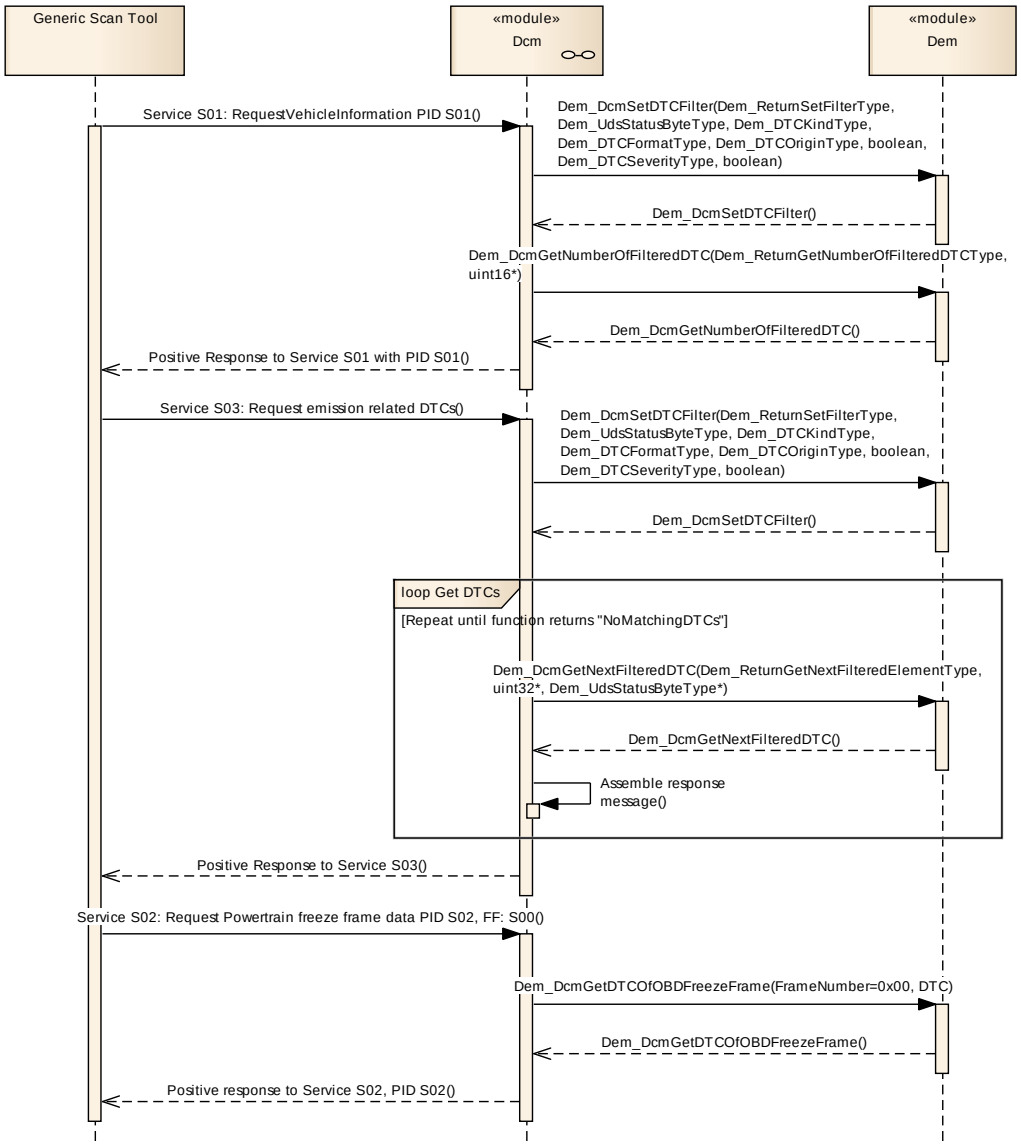

Figure 9.8: Sequence diagram of GetOBDFaultInformation

9.8 ReportDTCByStatusMask

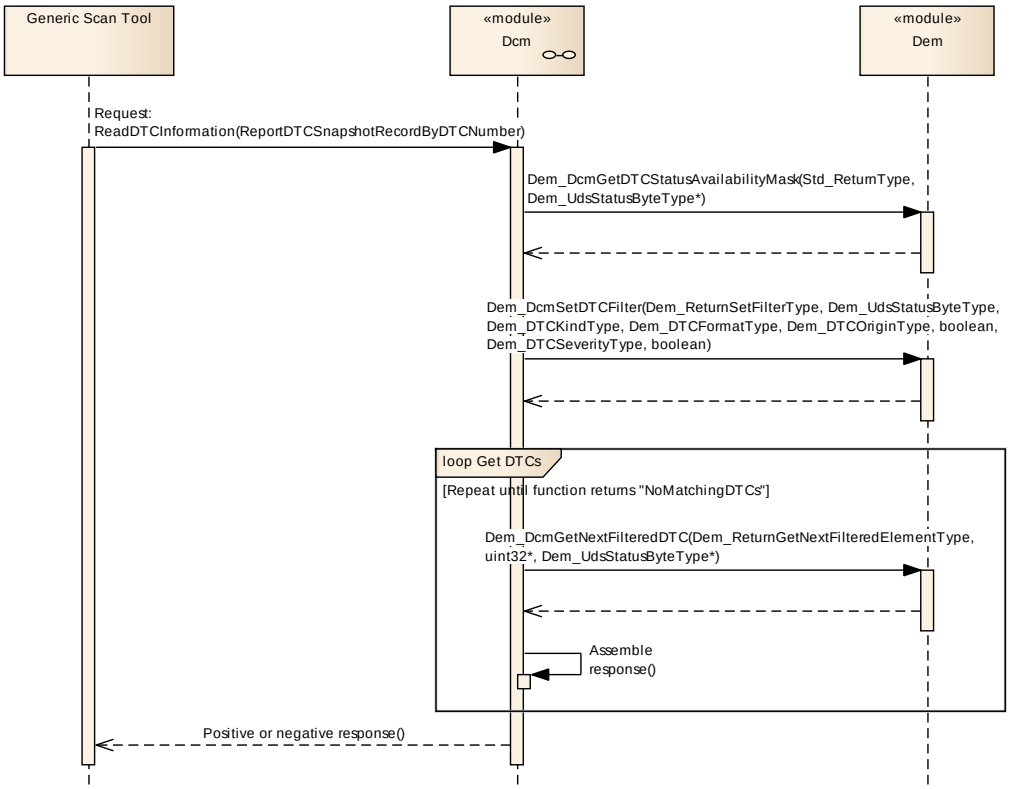

Figure 9.9: Sequence diagram of ReportDTCStatusMask

9.9 FiM_DemTriggerOnEventStatus

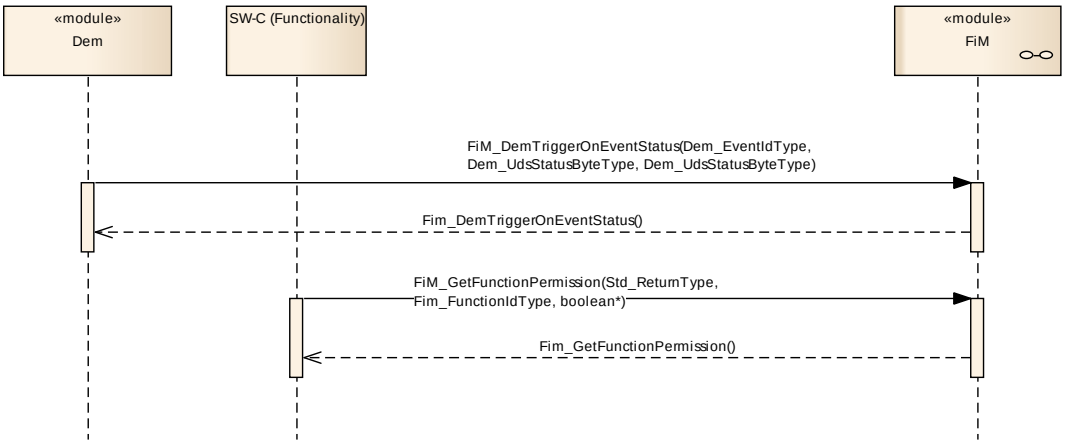

Figure 9.10: Sequence diagram of FiM_DemTriggerOnEventStatus

9.10 ProcessEvent (Example)

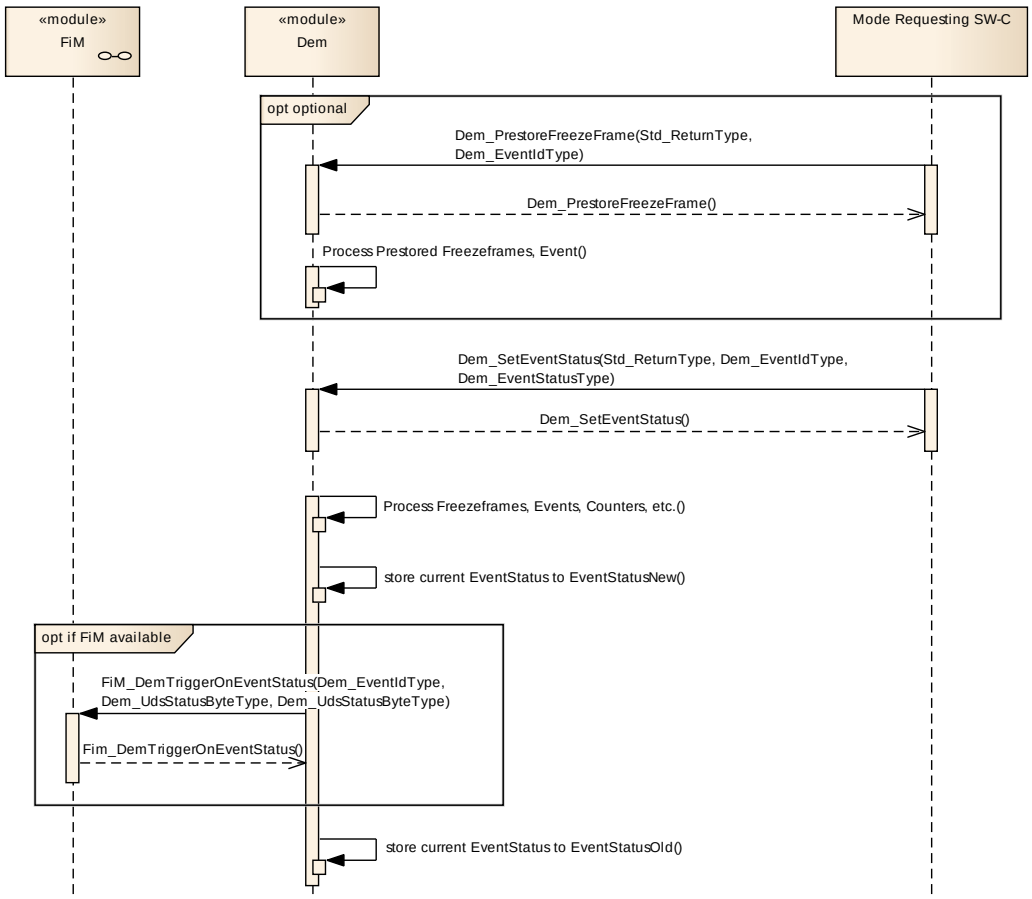

Figure 9.11: Sequence diagram for an example of ProcessEvent Dem-internally

10 Configuration specification

In general, this chapter defines configuration parameters and their clustering into containers. In order to support the specification Chapter chapter 10.1 describes fundamentals. It also specifies a template (table) you shall use for the parameter specification. We intend to leave chapter 10.1 in the specification to guarantee comprehension.

- chapter 10.2 specifies the structure (containers) and the parameters of the module Dem
- chapter 10.3 specifies published information of the module Dem

10.1 How to read this chapter

For details refer to the chapter 10.1 “Introduction to configuration specification” in SWS_BSWGeneral

10.2 Containers and configuration parameters

The following chapters summarize all configuration parameters. The detailed meaning of the parameters are described in chapter 7 and chapter 8.

10.2.1 Variants

The following use-cases and concepts for the Dem are the base for the different Dem configuration variants:

1. Simple ECU configuration, with optimized ROM usage: results in a pre-compile time variant
2. Multiple ECU configuration support: results in a post-built time selectable variant (in combination with a superset method). For this use-case, some DTC values have different values in the different configuration sets. Additionally it is possible, to omit events/DTCs (for the outside world) completely in some of the configuration sets.
3. Support of different PID values for Europe/USA results in a post-built time variant (refer to DemPidIdentifier and DemPidDataElementClassRef)
4. Support of different DTC kind for Europe/USA results in a post-built time variant

The parameter DemEventId and the whole debouncing-configuration (including vendor-specific debounce extensions) are always pre-compile time variant, independent of configuration concepts above, according to the pre-initialization handling (refer to chapter 7.6).

The following configuration parameters shall be available:

- **[SWS_Dem_00267]** ⌈VARIANT-PRE-COMPILE: only pre-compile time configuration parameters. ⌋*(SRS_BSW_00334, SRS_BSW_00345, SRS_BSW_00396, SRS_BSW_00397, SRS_BSW_00398, SRS_BSW_00399, SRS_BSW_00400, SRS_BSW_00401, SRS_BSW_00404, SRS_BSW_00405)*
- **[SWS_Dem_00268]** ⌈VARIANT-POST-BUILD: mix of pre-compile- and post build time-configuration parameters (using post build selectable, refer to chapter 4.1)⌋*(SRS_BSW_00334, SRS_BSW_00345, SRS_BSW_00396, SRS_BSW_00397, SRS_BSW_00398, SRS_BSW_00399, SRS_BSW_00400, SRS_BSW_00401, SRS_BSW_00404, SRS_BSW_00405)*

Link time configurable parameters (VARIANT-LINK-TIME) are not used in this specification.

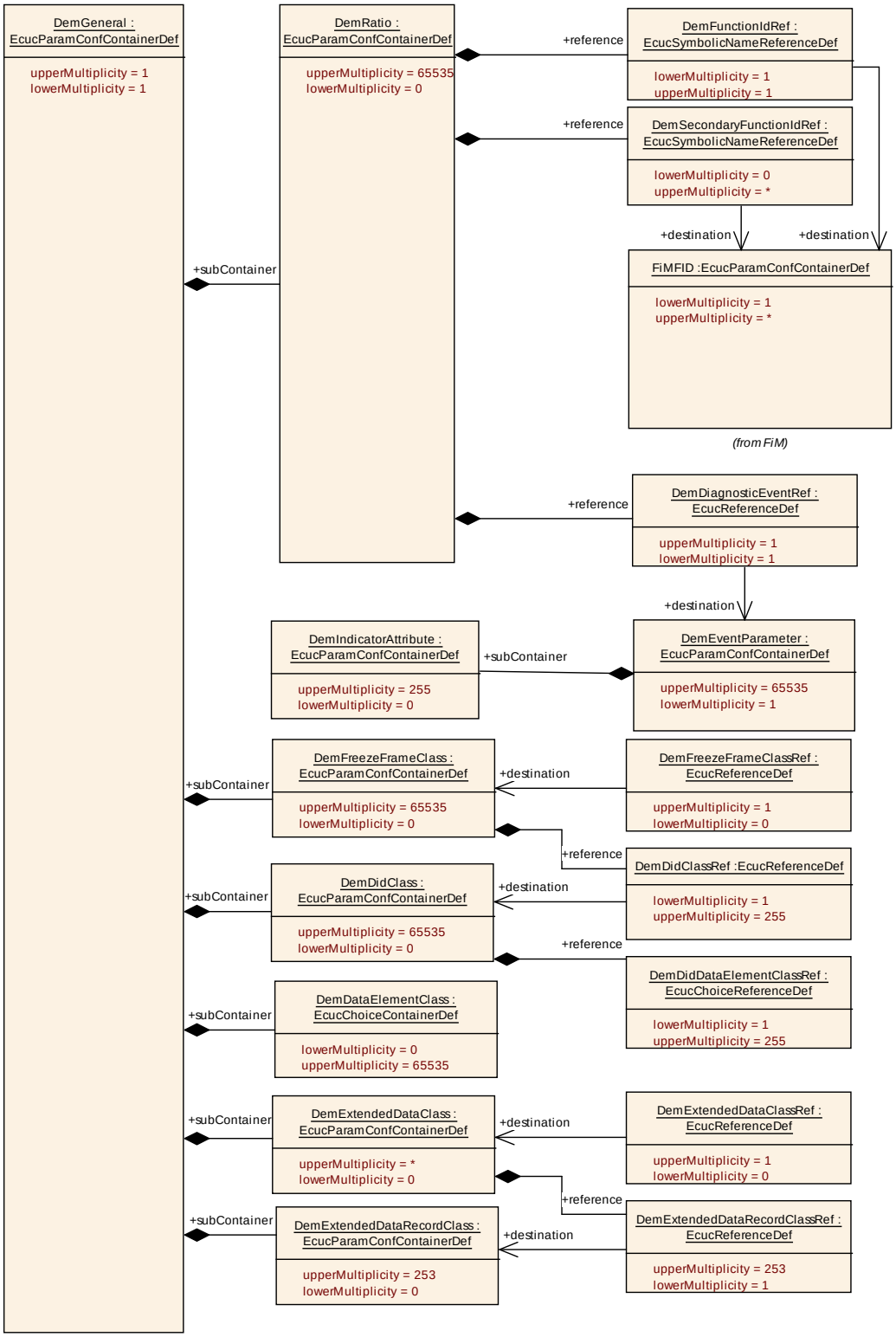

Figure 10.1: Configuration overview for DemGeneral (part I)

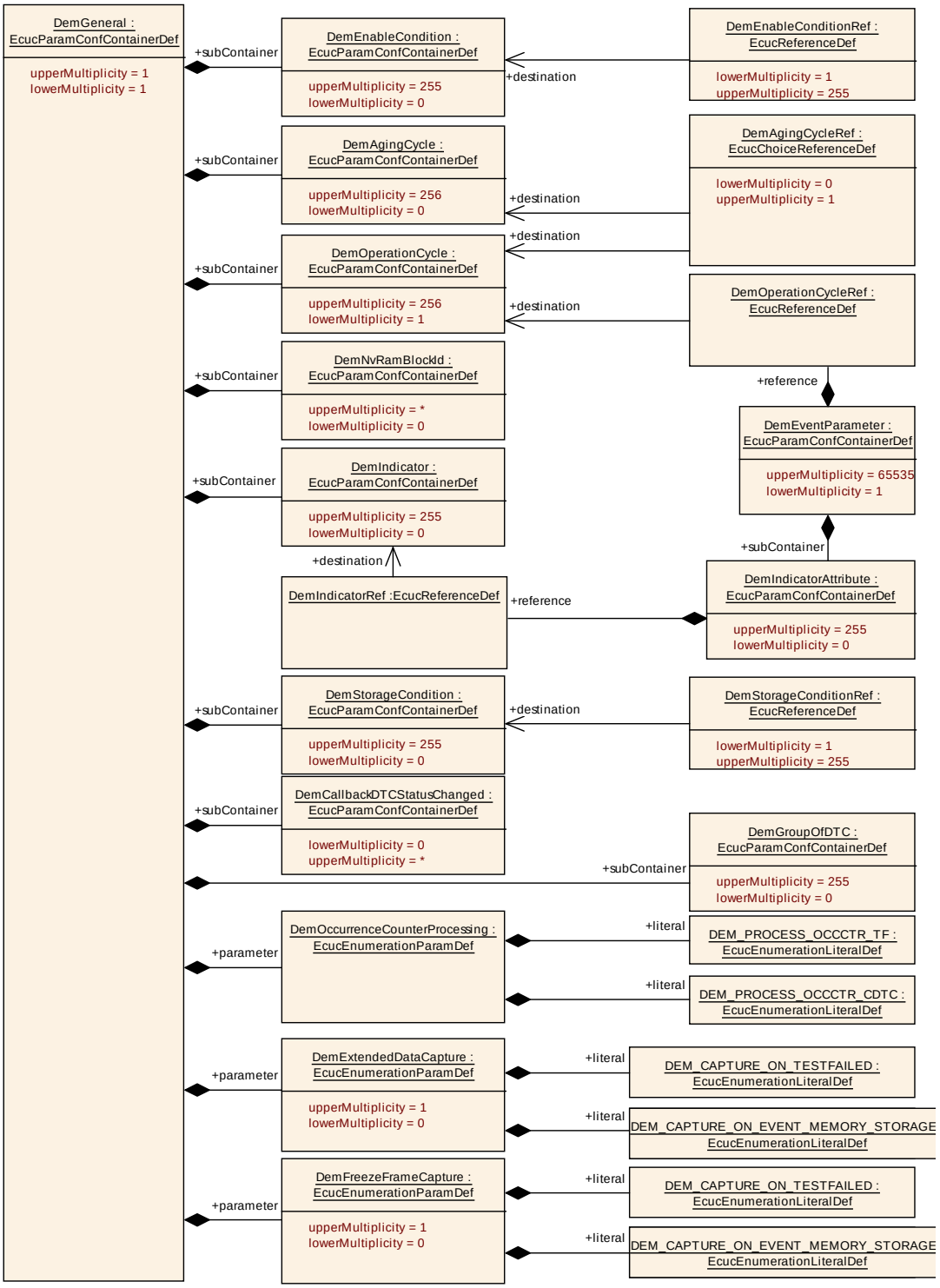

Figure 10.2: Configuration overview for DemGeneral (part II)

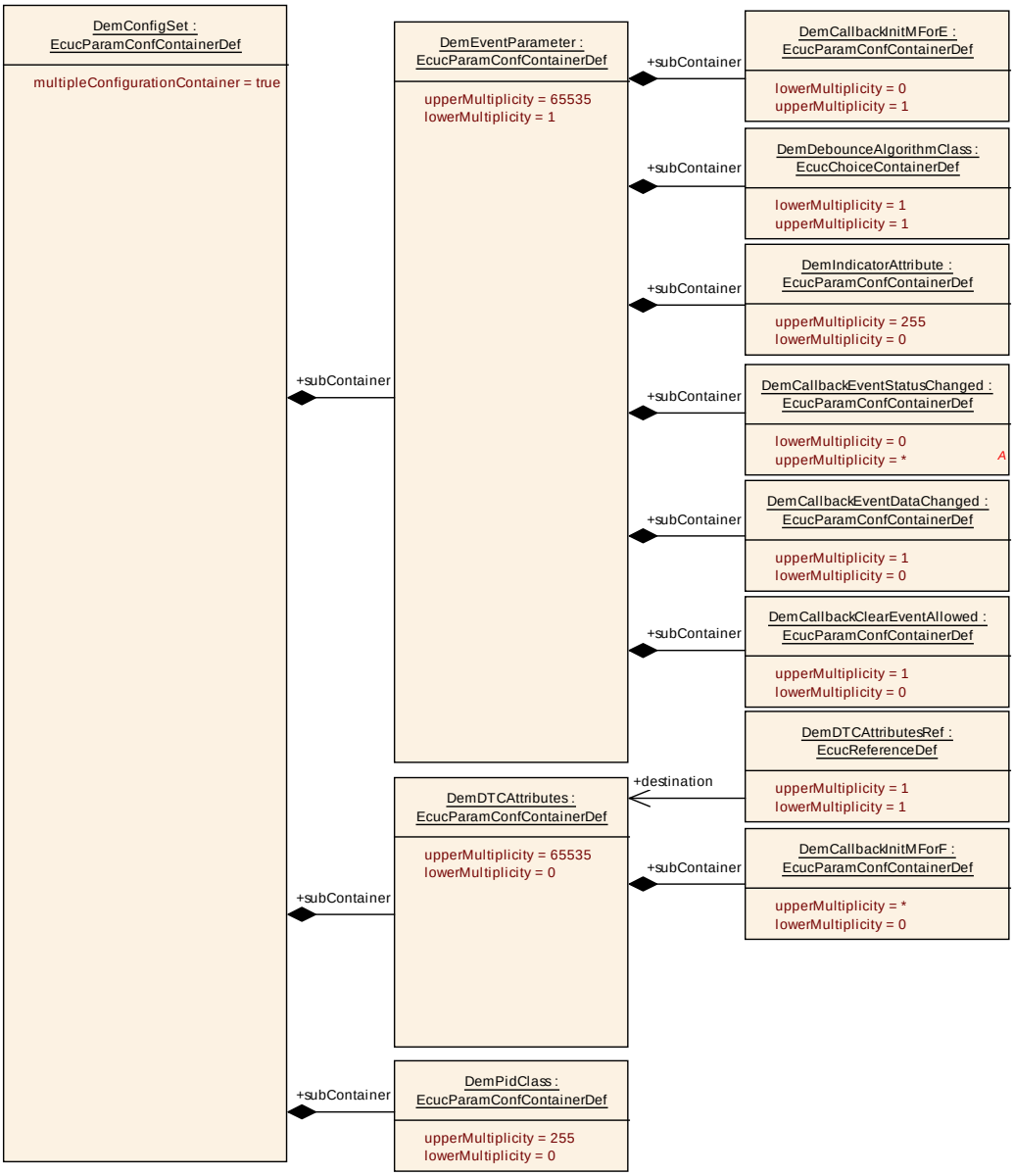

Figure 10.3: Configuration overview for DemConfigSet

10.2.2 Dem

Module Name	Dem	
Module Description	Configuration of the Dem (Diagnostic Event Manager) module.	
Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
DemConfigSet	1	This container contains the configuration parameters and sub containers of the Dem module supporting multiple configuration sets. This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set.

Container Name	Multiplicity	Scope / Dependency
DemGeneral	1	This container contains the configuration (parameters) of the BSW Dem

10.2.3 DemGeneral

DemGeneral

SWS Item	[ECUC_Dem_00677]
Container Name	DemGeneral
Description	This container contains the configuration (parameters) of the BSW Dem
Configuration Parameters	

Name	DemAgingCycleCounterProcessing [ECUC_Dem_00603]		
Description	This configuration switch defines, whether the aging counter is calculated Dem-internally or provided via Dem_SetAgingCycleCounterValue.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_PROCESS_AGING CTR_EXTERN	based on API Dem_SetAgingCycleCounterValue	
	DEM_PROCESS_AGING CTR_INTERN	based on reported cycle states	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemAgingRequieresTestedCycle [ECUC_Dem_00877]		
Description	Defines if the aging cycle counter is processed every aging cycles or if only tested aging cycle are considered. true: only tested aging cycle are considered for aging cycle counter false: aging cycle counter is processed every aging cycle		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemAvailabilitySupport [ECUC_Dem_00878]		
Description	This configuration switch defines, whether support for availability is enabled or not.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_EVENT_AVAILABILITY	Support availability by Event	
	DEM_NO_AVAILABILITY	Availability is not supported	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemBswErrorBufferSize [ECUC_Dem_00625]		
Description	Maximum number of elements in buffer for handling of BSW errors (ref. to Dem107).		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemClearDTCBehavior [ECUC_Dem_00766]		
Description	Defines the clearing process of diagnostic information for volatile and non-volatile memory and the positive response handling for the Dcm module.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_CLRRESP_NONVOLATILE_FINISH	Return DEM_CLEAR_OK after volatile and non-volatile event memory data cleared.	
	DEM_CLRRESP_NONVOLATILE_TRIGGER	Return DEM_CLEAR_OK after volatile event memory data cleared and non-volatile event memory clearing is triggered	
	DEM_CLRRESP_VOLATILE	Return DEM_CLEAR_OK after volatile event memory data cleared	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemClearDTCLimitation [ECUC_Dem_00790]		
Description	Defines the supported Dem_<...>ClearDTC API scope.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_ALL_SUPPORTED_DTCS	Dem_<...>ClearDTC accepts all supported DTC values, as well as all DTC values which are configured in DemGroupDTCs and DEM_DTC_GROUP_ALL_DTCS. (default)	
	DEM_ONLY_CLEAR_ALL_DTCS	Dem_<...>ClearDTC accepts ClearAllDTCs only.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDataElementDefaultEndianness [ECUC_Dem_00858]		
Description	Defines the default endianness of the data belonging to a data element which is applicable if the DemExternalSRDataElementClass does not define a endianness.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	BIG_ENDIAN	Most significant byte shall come at the lowest address.	
	LITTLE_ENDIAN	Most significant byte shall come highest address	
	OPAQUE	opaque data endianness	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemDebounceCounterBasedSupport [ECUC_Dem_00724]		
Description	This configuration switch defines, whether support for counter based debouncing is enabled or not. true: counter based debouncing support is enabled false: counter based debouncing support is disabled		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDebounceTimeBasedSupport [ECUC_Dem_00725]		
Description	This configuration switch defines, whether support for time based debouncing is enabled or not. true: time based debouncing support is enabled false: time based debouncing support is disabled		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDevErrorDetect [ECUC_Dem_00648]		
Description	Activate/Deactivate the Development Error Detection and Notification. true: Development Error Detection and Notification activated false: Development Error Detection and Notification deactivated		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemDtcStatusAvailabilityMask [ECUC_Dem_00652]		
Description	Mask for the supported DTC status bits by the Dem. This mask is used by UDS service 0x19.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemEventCombinationSupport [ECUC_Dem_00740]	
Description	This parameter defines the type of event combination supported by the Dem.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	DEM_EVCOMB_DISABLED	No event combination supported
	DEM_EVCOMB_ONRETRIEVAL	Event combination on retrieval

	DEM_EVCOMB_ONSTORAGE	Event combination on storage	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemEventDisplacementStrategy [ECUC_Dem_00742]		
Description	This configuration switch defines, whether support for event displacement is enabled or not, and which displacement strategy is followed.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_DISPLACEMENT_FULL	Event memory entry displacement is enabled, by consideration of priority active/passive status, and occurrence.	
	DEM_DISPLACEMENT_NONE	Event memory entry displacement is disabled.	
	DEM_DISPLACEMENT_PRIO_OCC	Event memory entry displacement is enabled, by consideration of priority and occurrence (but without active/passive status).	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemEventMemoryEntryStorageTrigger [ECUC_Dem_00797]		
Description	Configures the primary trigger to allocate an event memory entry.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_TRIGGER_ON_CONFIRMED		
	DEM_TRIGGER_ON_FDC_THRESHOLD		
	DEM_TRIGGER_ON_PENDING		
	DEM_TRIGGER_ON_TEST_FAILED		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemExtendedDataCapture [ECUC_Dem_00663]		
Description	This parameter defines the point in time, when the extended data collection is done for the initial event memory entry.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	DEM_CAPTURE_ON_EVENT_MEMORY_STORAGE	Triggers the collection of extended data if the event is stored in fault memory.	
	DEM_CAPTURE_ON_TESTFAILED	Triggers the collection of extended data if the UDS DTC status bit 0 (TestedFailed) changes from 0 to 1.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemFreezeFrameCapture [ECUC_Dem_00672]		
Description	This parameter defines the point in time, when the freeze frame data collection is done for the initial event memory entry.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	DEM_CAPTURE_ON_EVENT_MEMORY_STORAGE	Triggers the collection of freeze frame data if the event is stored in fault memory.	
	DEM_CAPTURE_ON_TESTFAILED	Triggers the collection of freeze frame data if the UDS DTC status bit 0 (TestedFailed) changes from 0 to 1.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemTypeOfFreezeFrameRecordNumeration == DEM_FF_RECNUM_CALCULATED		

Name	DemGeneralInterfaceSupport [ECUC_Dem_00880]		
Description	The interfaces GeneralEvtInfo, GeneralCallbackEventDataChanged and GeneralCallbackEventStatusChange are provided if DemGeneralInterfaceSupport is equal to true.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemHeaderFileInclusion [ECUC_Dem_00722]		
Description	Name of the header file(s) to be included by the Dem module containing the used C-callback declarations.		
Multiplicity	0..*		
Type	EcucStringParamDef		
Default Value			
Regular Expression	[a-zA-Z0-9_]([a-zA-Z0-9\._])*		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemImmediateNvStorageLimit [ECUC_Dem_00738]		
Description	This parameter defines the maximum number of occurrences, a specific event memory entry is allowed, to be stored in NVRAM immediately (refer to DemImmediateNvStorage).		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemMILIndicatorRef [ECUC_Dem_00723]		
Description	This parameter defines the indicator representing the MIL. This parameter is mandatory for ECUs supporting OBD (refer to DemOBDSupport).		
Multiplicity	0..1		
Type	Reference to DemIndicator		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemMaxNumberEventEntryMirror [ECUC_Dem_00688]		
Description	Maximum number of events which can be stored in the mirror memory		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemMaxNumberEventEntryPermanent [ECUC_Dem_00689]		
Description	Maximum number of events which can be stored in the permanent memory. The assignment of an event to this memory type is dynamic and used for emission-related events only.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemMaxNumberEventEntryPrimary [ECUC_Dem_00690]		
Description	Maximum number of events which can be stored in the primary memory		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemMaxNumberEventEntrySecondary [ECUC_Dem_00691]		
Description	Maximum number of events which can be stored in the secondary memory		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemMaxNumberPrestoredFF [ECUC_Dem_00692]		
Description	Defines the maximum number for prestored freeze frames. If set to 0, then freeze frame prestorage is not supported by the ECU.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default Value			

Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOBDSupport [ECUC_Dem_00698]		
Description	This configuration switch defines OBD support and kind of OBD ECU.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_OBD_DEP_SEC_ECU	Kind of OBD ECU: OBD Dependend / Secondary ECU	
	DEM_OBD_MASTER_ECU	Kind of OBD ECU: Master ECU	
	DEM_OBD_NO_OBD_SUPPORT	OBD is not supported within this ECU	
	DEM_OBD_PRIMARY_ECU	Kind of OBD ECU: Pimary ECU	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOccurrenceCounterProcessing [ECUC_Dem_00767]		
Description	This configuration switch defines the consideration of the fault confirmation process for the occurrence counter. For OBD and mixed systems (OBD/non OBD, refer to DemOBDSupport) configuration switch shall always set to DEM_PROCESS_OCCCTR_TF.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_PROCESS_OCCCTR_CDTC	the occurrence counter is triggered by the TestFailed bit if the fault confirmation was successful (ConfirmedDTC bit is set)	
	DEM_PROCESS_OCCCTR_TF	the occurrence counter is only triggered by the TestFailed bit (and the fault confirmation is not considered) This parameter is mandatory in case of J1939.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport, DemGeneralJ1939		

Name	DemOperationCycleProcessing [ECUC_Dem_00783] (Obsolete)		
Description	This configuration switch defines, whether the operation cycles are triggered by DEM_CYCLE_STATE_START or collecting an external counter value, which results in respective state changes. This parameter is obsolete since the option DEM_PROCESS_OPCYC_COUNTER is not longer supported by AUTOSAR. **Tags:** atp.Status=obsolete atp.StatusComment=This parameter is obsolete since the option DEM_PROCESS_OPCYC_COUNTER is not longer supported by AUTOSAR. atp.StatusRevisionBegin=4.1.1		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	DEM_PROCESS_OPCYC_COUNTER	operation cycle processing is based on API Dem_SetOperationCycleCntValue	
	DEM_PROCESS_OPCYC_STATE	operation cycle processing is based on API Dem_SetOperationCycleState (default)	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOperationCycleStatusStorage [ECUC_Dem_00764]		
Description	Defines if the operation cycle state is available over the power cycle (stored non-volatile) or not. true: the operation cycle state is stored non-volatile false: the operation cycle state is only stored volatile		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemPTOSupport [ECUC_Dem_00704]
Description	This configuration switch defines, whether PTO support (and therefore PID $1E support) is enabled or not.
Multiplicity	1
Type	EcucBooleanParamDef
Default Value	

Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemResetConfirmedBitOnOverflow [ECUC_Dem_00799]		
Description	This configuration switch defines, whether the confirmed bit is reset or not while an event memory entry will be displaced.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value	true		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemStatusBitHandlingTestFailedSinceLastClear [ECUC_Dem_00784]		
Description	This configuration switch defines, whether the aging and displacement mechanism shall be applied to the "TestFailedSinceLastClear" status bits.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_STATUS_BIT_AGING_AND_DISPLACEMENT	the "TestFailedSinceLastClear" status bits are reset to 0, if aging or displacement applies (like done for the "ConfirmedDTC" status bits)	
	DEM_STATUS_BIT_NORMAL	aging and displacement has no impact on the "TestFailedSinceLastClear" status bits (default)	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemStatusBitStorageTestFailed [ECUC_Dem_00714]		
Description	Activate/Deactivate the permanent storage of the "TestFailed" status bits. true: storage activated false: storage deactivated		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemSuppressionSupport [ECUC_Dem_00793]		
Description	This configuration switch defines, whether support for suppression is enabled or not.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_DTC_SUPPRESSION	Support suppression by DTC	
	DEM_EVENT_AND_DTC_SUPPRESSION	Support suppression by Event and DTC	
	DEM_EVENT_SUPPRESSION	Support suppression by Event	
	DEM_NO_SUPPRESSION	Suppression is not supported	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemTaskTime [ECUC_Dem_00715]		
Description	Allow to configure the time for the periodic cyclic task. Please note: This configuration value shall be equal to the value in the Basic Software Scheduler configuration of the RTE module. The AUTOSAR configuration standard is to use SI units, so this parameter is defined as float value in seconds. Dem configuration tools must convert this float value to the appropriate value format for the use in the software implementation of Dem. min: A negative value is not allowed. max: After event status was reported, processing shall be completed within 100ms in order to have the fault entry status information updated as soon as possible (e.g. for PID $01). upperMultiplicity: Exactly one TaskTime must be specified per configuration. lowerMultiplicity: Exactly one TaskTime must be specified per configuration.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	0.001 .. 0.1		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemTriggerDcmReports [ECUC_Dem_00754]		
Description	Activate/Deactivate the notification to the Diagnostic Communication Manager for ROE processing. true: Dcm ROE notification activated false: Dcm ROE notification deactivated		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemTriggerDltReports [ECUC_Dem_00718]		
Description	Activate/Deactivate the notification to the Diagnostic Log and Trace. true: Dlt notification activated false: Dlt notification deactivated		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemTriggerFiMReports [ECUC_Dem_00719]		
Description	Activate/Deactivate the notification to the Function Inhibition Manager. true: FiM notification activated false: FiM notification deactivated		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemTriggerMonitorInitBeforeClearOk [ECUC_Dem_00765]		
Description	Defines if the monitor re-initialization has to be triggered before or after the Dem module returns DEM_CLEAR_OK. true: trigger re-initialization before DEM_CLEAR_OK false: trigger re-initialization after DEM_CLEAR_OK		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			

Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemClearDTCBehavior		

Name	DemTypeOfDTCSupported [ECUC_Dem_00720]		
Description	This parameter defines the format returned by Dem_DcmGetTranslationType and does not relate to/influence the supported Dem functionality.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_DTC_TRANSLATION_ISO11992_4	ISO11992-4 DTC format	
	DEM_DTC_TRANSLATION_ISO14229_1	ISO14229-1 DTC format (3 byte format)	
	DEM_DTC_TRANSLATION_ISO15031_6	ISO15031-6 DTC format (2 byte format)	
	DEM_DTC_TRANSLATION_SAEJ1939_73	SAEJ1939-73 DTC format	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemTypeOfFreezeFrameRecordNumeration [ECUC_Dem_00778]		
Description	This parameter defines the type of assigning freeze frame record numbers for event-specific freeze frame records.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_FF_RECNUM_CALCULATED	freeze frame records will be numbered consecutive starting by 1 in their chronological order	
	DEM_FF_RECNUM_CONFIGURED	freeze frame records will be numbered based on the given configuration in their chronological order	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemVersionInfoApi [ECUC_Dem_00721]
Description	Activate/Deactivate the version information API. true: version information activated false: version information deactivated
Multiplicity	1
Type	EcucBooleanParamDef
Default Value	

Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
DemAgingCycle	0..256	Note that this container definition does not explicitly define a symbolic name parameter. Instead, the short name of the container will be used in the Ecu Configuration Description to specify the symbolic name of the aging cycle name. These aging cycles are reported via API Dem_SetAgingCycleState only.
DemCallbackDTCStatus Changed	0..*	The presence of this container indicates, that the Dem has access to a "DTCStatusChanged" callback, which the Dem will call to notify other components about the change in the status of a DTC. In case there is a DemCallbackDTCStatusChangedFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackDTCStatusChangedFnc, the Dem will have an R-Port requiring the interface CallbackDTCStatusChanged whose name is generated by using the unique callback-prefix followed by the event name. Status change notifications are supported for DTCs in primary memory only.
DemDataElementClass [DemExternalCSData ElementClass, Dem ExternalSRDataElement Class, DemInternalData ElementClass]	0..65535	This container contains the configuration (parameters) for an internal/external data element class.
DemDidClass	0..65535	This container contains the configuration (parameters) for a data Id class. It is assembled out of one or several data elements.
DemEnableCondition	0..255	This container contains the configuration (parameters) for enable conditions.
DemEnableCondition Group	0..255	This container contains the configuration (parameters) for enable condition groups.
DemExtendedDataClass	0..*	This class contains the combinations of extended data records for an extended data class.
DemExtendedData RecordClass	0..253	This container contains the configuration (parameters) for an extended data record class. It is assembled out of one or several data elements.
DemFreezeFrameClass	0..65535	This container contains the combinations of DIDs for a non OBD relevant freeze frame class.

DemFreezeFrameRec NumClass	0..255	This container contains a list of dedicated, different freeze frame record numbers assigned to an event. The order of record numbers in this list is assigned to the chronological order of the according freeze frame records. dependency: DemTypeOfFreezeFrameRecordNumeration = DEM_FF_RECNUM_CONFIGURED
DemFreezeFrameRecord Class	0..255	This container contains a list of dedicated, different freeze frame record numbers.
DemGeneralJ1939	0..1	This container contains the general J1939-specific configuration (parameters) of the Dem module. If the container exists the J1939 support is enabled.
DemGeneralOBD	0..1	This container contains the general OBD-specific configuration (parameters) of the Dem module.
DemGroupOfDTC	0..255	This container contains the configuration (parameters) for DTC groups.
DemIndicator	0..255	This container contains the configuration (parameters) for Indicators. Note that this container definition does not explicitly define a symbolic name parameter. Instead, the short name of the container will be used in the Ecu Configuration Description to specify the symbolic name of the INDICATOR_NAME.
DemNvRamBlockId	0..*	This container contains the configuration (parameters) for a non-volatile memory block, which is used from the Dem. If no permanent storage of event memory entries is required, no block needs to be configured. The number of blocks which are necessary depends on the implementation and configuration (e.g. number of used event memories) of the Dem module.
DemOperationCycle	1..256	Note that this container definition does not explicitly define a symbolic name parameter. Instead, the short name of the container will be used in the Ecu Configuration Description to specify the symbolic name of the operation cycle name.
DemRatio	0..65535	This container contains the OBD-specific in-use-monitor performance ratio configuration. It is related to a specific event, a FID, and an IUMPR group.
DemStorageCondition	0..255	This container contains the configuration (parameters) for storage conditions.
DemStorageCondition Group	0..255	This container contains the configuration (parameters) for storage condition groups.

10.2.4 DemGeneralOBD

DemGeneralOBD

SWS Item	[ECUC_Dem_00756]
Container Name	DemGeneralOBD
Description	This container contains the general OBD-specific configuration (parameters) of the Dem module.
Configuration Parameters	

Name	DemOBDCentralizedPID21Handling [ECUC_Dem_00794]		
Description	Switch to enable the centralized handling of PID $21. true: centralized handling of PID $21 enabled false: centralized handling of PID $21 disabled		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemOBDCentralizedPID31Handling [ECUC_Dem_00879]		
Description	Switch to enable the centralized handling of PID $31. true: centralized handling of PID $31 enabled false: centralized handling of PID $31 disabled		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemOBDCompliancy [ECUC_Dem_00795]	
Description	Configuration value to define the appropriate value to PID$1C "OBD requirements to which vehicle or engine is certified." according to the respective standards, e.g. OBD, OBDII, JOBD etc. Notice as well J1979 or the "DiagnosticReadiness 1" DM5 message of J1939-73	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 255	

Default Value			
Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU dependency: (DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU) or (DemJ1939Readiness1Support == true)		

Name	DemOBDDestinationOfEvents [ECUC_Dem_00836]		
Description	The destination of events assigns where the OBD events shall be stored.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_DTC_ORIGIN_PRIMARY_MEMORY	OBD Events are located in the primary memory.	
	DEM_DTC_ORIGIN_SECONDARY_MEMORY	OBD Events are located in the secondary memory.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemOBDEngineType [ECUC_Dem_00900]		
Description	Switch to provide either Gasoline or Diesel parameters.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	DEM_IGNITION_COMPRESSION	Diesel engine type	
	DEM_IGNITION_SPARK	Gasoline engine type	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemOBDEventDisplacement [ECUC_Dem_00796]
Description	Activate/Deactivate a different displacement behavior for OBD events. OBD events with special Conditions (e.g. Pending, MIL_On...) shall not be displaced.
Multiplicity	1
Type	EcucBooleanParamDef
Default Value	

Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemOBDInputAcceleratorPedalInformation [ECUC_Dem_00763]		
Description	Input variable for the accelerator padal information, which is assigned to a specific data element used as interface for the Dem-internal PID calculations.		
Multiplicity	0..1		
Type	Choice Reference to [DemExternalCSDataElement-Class,DemExternalSRDataElementClass]		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemOBDInputAmbientPressure [ECUC_Dem_00762]		
Description	Input variable for the ambient pressure, which is assigned to a specific data element used as interface for the Dem-internal PID calculations.		
Multiplicity	0..1		
Type	Choice Reference to [DemExternalCSDataElement-Class,DemExternalSRDataElementClass]		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemOBDInputAmbientTemperature [ECUC_Dem_00761]		
Description	Input variable for the ambient temperature, which is assigned to a specific data element used as interface for the Dem-internal PID calculations.		
Multiplicity	0..1		
Type	Choice Reference to [DemExternalCSDataElement-Class,DemExternalSRDataElementClass]		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemOBDInputDistanceInformation [ECUC_Dem_00759]		
Description	Input variable for the distance information, which is assigned to a specific data element used as interface for the Dem-internal PID calculations.		
Multiplicity	0..1		
Type	Choice Reference to [DemExternalCSDataElement-Class,DemExternalSRDataElementClass]		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemOBDInputEngineSpeed [ECUC_Dem_00757]		
Description	Input variable for the engine speed, which is assigned to a specific data element used as interface for the Dem-internal PID calculations.		
Multiplicity	0..1		
Type	Choice Reference to [DemExternalCSDataElement-Class,DemExternalSRDataElementClass]		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemOBDInputEngineTemperature [ECUC_Dem_00772]		
Description	Input variable for the engine temperature, which is assigned to a specific data element used as interface for the Dem-internal PID calculations.		
Multiplicity	0..1		
Type	Choice Reference to [DemExternalCSDataElement-Class,DemExternalSRDataElementClass]		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemOBDInputProgrammingEvent [ECUC_Dem_00760]		
Description	Input variable for the programming event, which is assigned to a specific data element used as interface for the Dem-internal PID calculations.		
Multiplicity	0..1		
Type	Choice Reference to [DemExternalCSDataElement-Class,DemExternalSRDataElementClass]		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemOBDInputVehicleSpeed [ECUC_Dem_00758]		
Description	Input variable for the vehicle speed, which is assigned to a specific data element used as interface for the Dem-internal PID calculations.		
Multiplicity	0..1		
Type	Choice Reference to [DemExternalCSDataElement-Class,DemExternalSRDataElementClass]		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemOBDTimeSinceEngineStart [ECUC_Dem_00827]		
Description	Input variable for the Time Since Engine Start information, which is assigned to a specific data element.		
Multiplicity	0..1		
Type	Choice Reference to [DemExternalCSDataElement-Class,DemExternalSRDataElementClass,DemInternalDataElementClass]		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**

DemCallbackOBDDTC StatusChanged	0..*	The presence of this container indicates, that the Dem has access to a "DTCStatusChanged" callback, which the Dem will call to notify other components about the change in the status of a DTC. In case there is a DemCallbackDTCStatusChangedFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackDTCStatusChangedFnc, the Dem will have an R-Port requiring the interface CallbackDTCStatusChanged whose name is generated by using the unique callback-prefix followed by the event name. Status change notifications are supported for DTCs in primary memory only.

10.2.5 DemGeneralJ19139

DemGeneralJ1939

SWS Item	[ECUC_Dem_00864]
Container Name	DemGeneralJ1939
Description	This container contains the general J1939-specific configuration (parameters) of the Dem module. If the container exists the J1939 support is enabled.
Configuration Parameters	

Name	DemAmberWarningLampIndicatorRef [ECUC_Dem_00821]		
Description	This parameter defines the indicator representing the AmberWarningLamp . This parameter may be used for ECUs supporting J1939.		
Multiplicity	0..1		
Type	Reference to DemIndicator		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemJ1939ClearDtcSupport [ECUC_Dem_00872]
Description	This configuration switch defines whether clearing J1939 DTCs (DM3 und DM11) is supported or not. This switches on and off the API Dem_J1939DcmClearDTC.
Multiplicity	1
Type	EcucBooleanParamDef
Default Value	

Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemJ1939Dm31Support [ECUC_Dem_00868]		
Description	This configuration switch defines whether J1939 DM31 is supported or not. This switches on and off the APIs Dem_J1939DcmFirstDTCwithLampStatus and Dem_J1939DcmGetNextDTCwithLampStatus. .		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemJ1939ExpandedFreezeFrameSupport [ECUC_Dem_00873]		
Description	This configuration switch defines whether J1939 expanded freeze frames are supported or not. This switches on and off the APIs Dem_J1939DcmSetFreezeFrameFilter, Dem_J1939DcmGetNextFreezeFrame and Dem_J1939DcmGetNextSPNInFreezeFrame.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemJ1939FreezeFrameSupport [ECUC_Dem_00866]		
Description	This configuration switch defines whether J1939 freeze frames are supported or not. This switches on and off the APIs Dem_J1939DcmSetFreezeFrameFilter and Dem_J1939DcmGetNextFreezeFrame.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			

Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemJ1939RatioSupport [ECUC_Dem_00867]		
Description	This configuration switch defines whether J1939 performance ratios are supported or not. This switches on and off the APIs Dem_J1939DcmSetRatioFilter and Dem_J1939DcmGetNextFilteredRatio.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemJ1939Readiness1Support [ECUC_Dem_00869]		
Description	This configuration switch defines whether J1939 diagnostic readiness 1 is supported or not. This switches on and off the API Dem_J1939DcmReadDiagnosticReadiness1.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemJ1939Readiness2Support [ECUC_Dem_00870]		
Description	This configuration switch defines whether J1939 diagnostic readiness 2 is supported or not. This switches on and off the API Dem_J1939DcmReadDiagnosticReadiness2.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemJ1939Readiness3Support [ECUC_Dem_00871]		
Description	This configuration switch defines whether J1939 diagnostic readiness 3 is supported or not. This switches on and off the API Dem_J1939DcmReadDiagnosticReadiness3.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemJ1939ReadingDtcSupport [ECUC_Dem_00865]		
Description	This configuration switch defines whether J1939 DTC readout is supported or not. This switches on and off the APIs Dem_J1939DcmSetDTCFilter, Dem_J1939DcmGetNumberOfFilteredDTC and Dem_J1939DcmGetNextFilteredDTC.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemProtectLampIndicatorRef [ECUC_Dem_00822]		
Description	This parameter defines the indicator representing the ProtectLamp. This parameter may be used for ECUs supporting J1939.		
Multiplicity	0..1		
Type	Reference to DemIndicator		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemRedStopLampIndicatorRef [ECUC_Dem_00820]		
Description	This parameter defines the indicator representing the RedStopLamp. This parameter may be used for ECUs supporting J1939.		
Multiplicity	0..1		
Type	Reference to DemIndicator		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
DemCallbackJ1939DTC StatusChanged	0..*	The presence of this container indicates, that the Dem has access to a "DTCStatusChanged" callback, which the Dem will call to notify other components about the change in the status of a DTC. In case there is a DemCallbackDTCStatusChangedFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackDTCStatusChangedFnc, the Dem will have an R-Port requiring the interface CallbackDTCStatusChanged whose name is generated by using the unique callback-prefix followed by the event name. Status change notifications are supported for DTCs in primary memory only.
DemJ1939FreezeFrame Class	0..255	This container contains the combinations of SPNs s for a J1939 relevant freeze frame.
DemSPNClass	0..*	This container contains the configuration (parameters) for a SPN.

10.2.6 DemOperationCycle

DemOperationCycle

SWS Item	[ECUC_Dem_00701]
Container Name	DemOperationCycle
Description	Note that this container definition does not explicitly define a symbolic name parameter. Instead, the short name of the container will be used in the Ecu Configuration Description to specify the symbolic name of the operation cycle name.
Configuration Parameters	

Name	DemOperationCycleAutomaticEnd [ECUC_Dem_00837]		
Description	If DemOperationCycleAutomaticEnd is configured to TRUE, Dem shall automatically end the driving cycle at either Dem_Shutdown() or Dem_Init().		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value	false		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECUECU || DEM_OBD_PRIMARY_ECU		

Name	DemOperationCycleAutostart [ECUC_Dem_00805]		
Description	The autostart property defines if the operation cycles is automatically (re-)started during Dem_PreInit.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value	false		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOperationCycleType [ECUC_Dem_00703]		
Description	Operation cycles types for the Dem to be supported by cycle-state APIs. Further cycle types can be specified as part of the Dem delivery.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_OPCYC_IGNITION	Ignition ON / OFF cycle	
	DEM_OPCYC_OBD_DCY	OBD Driving cycle	
	DEM_OPCYC_OTHER	further operation cycle	
	DEM_OPCYC_POWER	Power ON / OFF cycle	
	DEM_OPCYC_TIME	Time based operation cycle	
	DEM_OPCYC_WARMUP	OBD OBD Warm up cycle	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.7 DemAgingCycle

DemAgingCycle

SWS Item	[ECUC_Dem_00785]
Container Name	DemAgingCycle
Description	Note that this container definition does not explicitly define a symbolic name parameter. Instead, the short name of the container will be used in the Ecu Configuration Description to specify the symbolic name of the aging cycle name. These aging cycles are reported via API Dem_SetAgingCycleState only.
Configuration Parameters	
No Included Containers	

10.2.8 DemEnableCondition

DemEnableCondition

SWS Item	[ECUC_Dem_00653]
Container Name	DemEnableCondition
Description	This container contains the configuration (parameters) for enable conditions.
Configuration Parameters	

Name	DemEnableConditionId [ECUC_Dem_00654]		
Description	Defines a unique enable condition Id. This parameter should not be changeable by user, because the Id should be generated by Dem itself to prevent gaps and multiple use of an Id. The enable conditions should be sequentially ordered beginning with 0 and no gaps in between.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemEnableConditionStatus [ECUC_Dem_00656]		
Description	Defines the initial status for enable or disable of acceptance of event reports of a diagnostic event. The value is the initialization after power up (before this condition is reported the first time). true: acceptance of a diagnostic event enabled false: acceptance of a diagnostic event disabled		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.9 DemEnableConditionGroup

DemEnableConditionGroup

SWS Item	[ECUC_Dem_00745]
Container Name	DemEnableConditionGroup
Description	This container contains the configuration (parameters) for enable condition groups.
Configuration Parameters	

Name	DemEnableConditionRef [ECUC_Dem_00655]		
Description	References an enable condition.		
Multiplicity	1..255		
Type	Reference to DemEnableCondition		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.10 DemStorageCondition

DemStorageCondition

SWS Item	[ECUC_Dem_00728]
Container Name	DemStorageCondition
Description	This container contains the configuration (parameters) for storage conditions.
Configuration Parameters	

Name	DemStorageConditionId [ECUC_Dem_00730]		
Description	Defines a unique storage condition Id. This parameter should not be changeable by user, because the Id should be generated by Dem itself to prevent gaps and multiple use of an Id. The storage conditions should be sequentially ordered beginning with 0 and no gaps in between.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemStorageConditionStatus [ECUC_Dem_00731]		
Description	Defines the initial status for enable or disable of storage of a diagnostic event. The value is the initialization after power up (before this condition is reported the first time). true: storage of a diagnostic event enabled false: storage of a diagnostic event disabled		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

10.2.11 DemStorageConditionGroup

DemStorageConditionGroup

SWS Item	[ECUC_Dem_00773]
Container Name	DemStorageConditionGroup
Description	This container contains the configuration (parameters) for storage condition groups.
Configuration Parameters	

Name	DemStorageConditionRef [ECUC_Dem_00768]		
Description	References an enable condition.		
Multiplicity	1..255		
Type	Reference to DemStorageCondition		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.12 DemIndicator

DemIndicator

SWS Item	[ECUC_Dem_00680]
Container Name	DemIndicator
Description	This container contains the configuration (parameters) for Indicators. Note that this container definition does not explicitly define a symbolic name parameter. Instead, the short name of the container will be used in the Ecu Configuration Description to specify the symbolic name of the INDICATOR_NAME.
Configuration Parameters	

Name	DemIndicatorID [ECUC_Dem_00683]		
Description	Unique identifier of an indicator.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.13 DemNvRamBlockId

DemNvRamBlockId

SWS Item	[ECUC_Dem_00696]
Container Name	DemNvRamBlockId
Description	This container contains the configuration (parameters) for a non-volatile memory block, which is used from the Dem. If no permanent storage of event memory entries is required, no block needs to be configured. The number of blocks which are necessary depends on the implementation and configuration (e.g. number of used event memories) of the Dem module.
Configuration Parameters	

Name	DemNvRamBlockIdRef [ECUC_Dem_00697]		
Description	This reference contains the link to a non-volatile memory block. For post build time configurations worst case scenario shall be used.		
Multiplicity	1		
Type	Symbolic name reference to NvMBlockDescriptor		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.14 DemGroupOfDTC

DemGroupOfDTC

SWS Item	[ECUC_Dem_00679]
Container Name	DemGroupOfDTC
Description	This container contains the configuration (parameters) for DTC groups.
Configuration Parameters	

Name	DemGroupDTCs [ECUC_Dem_00678]		
Description	DTC values of the selected group of DTC (Range: 3 byte, 0xFFFFFF is reserved for 'all DTCs', according to ISO14229-1 Annex D.1) The DTC group 'all DTCs' is always available and will not be configured. The following ranges are reserved by ISO 14229-1 : 0x00000000 to 0x000000ff and 0x00ffff00 to 0x00ffffff.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	256 .. 16776959		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.15 DemRatio

DemRatio

SWS Item	[ECUC_Dem_00734]
Container Name	DemRatio
Description	This container contains the OBD-specific in-use-monitor performance ratio configuration. It is related to a specific event, a FID, and an IUMPR group.
Configuration Parameters	

Name	DemDiagnosticEventRef [ECUC_Dem_00735]		
Description	This reference contains the link to a diagnostic event.		
Multiplicity	1		
Type	Reference to DemEventParameter		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU		

Name	DemFunctionIdRef [ECUC_Dem_00736]		
Description	This reference contains the link to a function identifier within the FiM which is used as a primary FID.		
Multiplicity	1		
Type	Symbolic name reference to FiMFID		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU		

Name	DemIUMPRDenGroup [ECUC_Dem_00838]	
Description	This parameter specifies the assigned denominator type which is applied in addition to the General Denominator conditions.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	DEM_IUMPR_DEN_500MILL	Additional condition based on definition of 500miles conditions as defined for OBD2.
	DEM_IUMPR_DEN_COLDSTART	Additional condition based on definition of "cold start" as defined for EU5+
	DEM_IUMPR_DEN_EVAP	Additional condition based on definition of "EVAP" conditions as defined for OBD2.
	DEM_IUMPR_DEN_NONE	No further condition. Denominator increments based on General Denominator only.

	DEM_IUMPR_DEN_PHYS_API	Additional physical condition (component activity) computed wihtin the SW-C and reported via Dem_RepIUMPRDenLock / Dem_RepIUMPRDenRelease.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemIUMPRGroup [ECUC_Dem_00737]		
Description	This parameter specifies the assigned IUMPR group of the ratio Id.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_IUMPR_BOOSTPRS		
	DEM_IUMPR_CAT1		
	DEM_IUMPR_CAT2		
	DEM_IUMPR_EGR		
	DEM_IUMPR_EGSENSOR		
	DEM_IUMPR_EVAP		
	DEM_IUMPR_FLSYS		
	DEM_IUMPR_NMHCCAT		
	DEM_IUMPR_NOXADSORB		
	DEM_IUMPR_NOXCAT		
	DEM_IUMPR_OXS1		
	DEM_IUMPR_OXS2		
	DEM_IUMPR_PMFILTER		
	DEM_IUMPR_PRIVATE		
	DEM_IUMPR_SAIR		
	DEM_IUMPR_SECOXS1		
	DEM_IUMPR_SECOXS2		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU		

Name	DemRatioId [ECUC_Dem_00787]		
Description	Defines a unique ratio Id. This parameter should not be changeable by user, because the Id should be generated by Dem itself to prevent gaps and multiple use of an Id. The ratio Ids should be sequentially ordered beginning with 0 and no gaps in between.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 65535		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU		

Name	DemRatioKind [ECUC_Dem_00741]		
Description	This parameter defines whether the ratio will be calculated API or observer based.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_RATIO_API	API based ratio Id	
	DEM_RATIO_OBSERVER	Observer based ratio Id	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemSecondaryFunctionIdRef [ECUC_Dem_00782]		
Description	This reference contains the link to a function identifier within the FiM which is used as a secondary FID. The "primary" and all "secondary" FID inhibitions are combined by "OR".		
Multiplicity	0..*		
Type	Symbolic name reference to FiMFID		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU		

No Included Containers

10.2.16 DemCallbackDTCStatusChanged

DemCallbackDTCStatusChanged

SWS Item	[ECUC_Dem_00626]
Container Name	DemCallbackDTCStatusChanged
Description	The presence of this container indicates, that the Dem has access to a "DTCStatusChanged" callback, which the Dem will call to notify other components about the change in the status of a DTC. In case there is a DemCallbackDTCStatusChangedFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackDTCStatusChangedFnc, the Dem will have an R-Port requiring the interface CallbackDTCStatusChanged whose name is generated by using the unique callback-prefix followed by the event name. Status change notifications are supported for DTCs in primary memory only.
Configuration Parameters	

Name	DemCallbackDTCStatusChangedFnc [ECUC_Dem_00627]		
Description	Function name of prototype "DTCStatusChanged". Note: If the parameter DemTriggerDcmReports is enabled, this parameter shall not be "Dcm_DemTriggerOnDTCStatus".		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default Value			
Regular Expression			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemTriggerDcmReport		

No Included Containers

10.2.17 DemCallbackOBDDTCStatusChanged

DemCallbackOBDDTCStatusChanged

SWS Item	[ECUC_Dem_00825]
Container Name	DemCallbackOBDDTCStatusChanged
Description	The presence of this container indicates, that the Dem has access to a "DTCStatusChanged" callback, which the Dem will call to notify other components about the change in the status of a DTC. In case there is a DemCallbackDTCStatusChangedFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackDTCStatusChangedFnc, the Dem will have an R-Port requiring the interface CallbackDTCStatusChanged whose name is generated by using the unique callback-prefix followed by the event name. Status change notifications are supported for DTCs in primary memory only.
Configuration Parameters	

Name	DemCallbackDTCStatusChangedFnc [ECUC_Dem_00826]		
Description	Function name of prototype "DTCStatusChanged". Note: If the parameter DemTriggerDcmReports is enabled, this parameter shall not be "Dcm_DemTriggerOnDTCStatus".		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default Value			
Regular Expression			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemTriggerDcmReport		

No Included Containers

10.2.18 DemCallbackJ1939DTCStatusChanged

DemCallbackJ1939DTCStatusChanged

SWS Item	[ECUC_Dem_00823]
Container Name	DemCallbackJ1939DTCStatusChanged
Description	The presence of this container indicates, that the Dem has access to a "DTCStatusChanged" callback, which the Dem will call to notify other components about the change in the status of a DTC. In case there is a DemCallbackDTCStatusChangedFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackDTCStatusChangedFnc, the Dem will have an R-Port requiring the interface CallbackDTCStatusChanged whose name is generated by using the unique callback-prefix followed by the event name. Status change notifications are supported for DTCs in primary memory only.
Configuration Parameters	

Name	DemCallbackDTCStatusChangedFnc [ECUC_Dem_00824]		
Description	Function name of prototype "DTCStatusChanged". Note: If the parameter DemTriggerDcmReports is enabled, this parameter shall not be "Dcm_DemTriggerOnDTCStatus".		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default Value			
Regular Expression			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemTriggerDcmReport		

No Included Containers

10.2.19 DemConfigSet

DemConfigSet

SWS Item	[ECUC_Dem_00634]
Container Name	DemConfigSet[Multi Config Container]
Description	This container contains the configuration parameters and sub containers of the Dem module supporting multiple configuration sets. This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set.
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
DemDTC	0..65535	This container contains the configuration (parameters) for DemUdsDTC.
DemDTCAttributes	0..65535	This container contains the configuration (parameters) for DemDTCAttributes.
DemDebounceCounterBasedClass	0..65535	This container contains the configuration of Debounce Counter Based Class
DemDebounceTimeBaseClass	0..65535	This container contains the configuration of Debounce Counter Based Class
DemDtr	0..1	This container holds the configuration of one individual DTR.
DemEventParameter	1..65535	This container contains the configuration (parameters) for events. Note that this container definition does not explicitly define a symbolic name parameter. Instead, the short name of the container will be used in the Ecu Configuration Description to specify the symbolic name of the diagnostic event.
DemJ1939NodeAddress	0..255	Contains the parameters for the support of a logical J1939 node (identified by an ECU address).
DemObdDTC	0..65535	This container contains the configuration (parameters) for DemObdDTC.
DemPidClass	0..255	This container contains the different PIDs for the single global OBD relevant freeze frame class. It is assembled out of one or several data elements.

10.2.20 DemObdDTC

DemObdDTC

SWS Item	[ECUC_Dem_00884]
Container Name	DemObdDTC
Description	This container contains the configuration (parameters) for DemObdDTC.
Configuration Parameters	

Name	DemConsiderPtoStatus [ECUC_Dem_00602]		
Description	This parameter is TRUE, when the event is affected by the Dem PTO handling.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemDtcValue [ECUC_Dem_00885]		
Description	Unique Diagnostic Trouble Code value for OBD		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 65535		
Default Value			
Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemEventOBDReadinessGroup [ECUC_Dem_00755]	
Description	This parameter specifies the Event OBD Readiness group for PID $01 and PID $41 computation. This parameter is only applicable for emission-related ECUs.	
Multiplicity	0..1	
Type	EcucEnumerationParamDef	
Range	DEM_OBD_RDY_AC	A/C system component - spark
	DEM_OBD_RDY_BOOSTPR	Boost Pressure System - compr.
	DEM_OBD_RDY_CAT	Catalyst - spark
	DEM_OBD_RDY_CMPRCMPT	Comprehensive component - spark, compr.
	DEM_OBD_RDY_EGSENS	Exhaust Gas Sensor - compr.
	DEM_OBD_RDY_ERG	EGR system - spark, compr.
	DEM_OBD_RDY_EVAP	Evaporative system - spark
	DEM_OBD_RDY_FLSYS	Fuel system - spark, compr.
	DEM_OBD_RDY_FLSYS_NONCONT	Non Contious Fuel system - spark, compr
	DEM_OBD_RDY_HCCAT	Non-Methan HC Catalyst - compr.
	DEM_OBD_RDY_HTCAT	Heated catalyst - spark
	DEM_OBD_RDY_MISF	Misfire - spark, compr.
	DEM_OBD_RDY_NONE	None - spark, compr.

	DEM_OBD_RDY_NOXCAT	NOx Catalyst - compr.	
	DEM_OBD_RDY_O2SENS	Oxygen sensor - spark	
	DEM_OBD_RDY_O2SENSHT	Oxygen sensor heater - spark	
	DEM_OBD_RDY_PMFLT	Particle Matters Filter - compr.	
	DEM_OBD_RDY_SECAIR	Secondary air system - spark	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU		

Name	DemJ1939DTCValue [ECUC_Dem_00892]		
Description	Unique Diagnostic Trouble Code value for J1939 (consiting of SPN and FMI)		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 16777214		
Default Value			
Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

No Included Containers

10.2.21 DemDTC

DemDTC

SWS Item	[ECUC_Dem_00886]
Container Name	DemDTC
Description	This container contains the configuration (parameters) for DemUdsDTC.
Configuration Parameters	

Name	DemDTCAttributesRef [ECUC_Dem_00642]		
Description	This parameter defines the DTC Attributes associated with the DemDTC.		
Multiplicity	1		
Type	Reference to DemDTCAttributes		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name	DemDTCFunctionalUnit [ECUC_Dem_00643]		
Description	DTCFuncitonalUnit is a 1-byte value which identifies the corresponding basic vehicle / system function which reports the DTC. This parameter is necessary for the report of severity information. If this parameter is configured for no DTC, the Dem provides no DTC functional unit information.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDTCSeverity [ECUC_Dem_00645]		
Description	DTC severity according to ISO 14229-1. This parameter depends on the automotive manufacturer. If it is not configured, the value is counted as 'no severity'. If this parameter is configured for no DTC, the Dem provides no DTC severity information.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	DEM_SEVERITY_CHECK_AT_NEXT_HALT	Check at next halt	
	DEM_SEVERITY_CHECK_IMMEDIATELY	Check immediately	
	DEM_SEVERITY_MAINTENANCE_ONLY	Maintenance required	
	DEM_SEVERITY_NO_SEVERITY	No severity information available	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDtcValue [ECUC_Dem_00887]		
Description	Unique Diagnostic Trouble Code value for UDS (Range: 0x000000 and 0xFFFFFF are reserved for DTC groups by ISO 14229-1)		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 16777214		
Default Value			
Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

Name	DemObdDTCRef [ECUC_Dem_00889]		
Description	This parameter defines the OBD DTC configuration associated with the DemDTC. It is allowed to have events without a OBD DTC.		
Multiplicity	0..1		
Type	Reference to DemObdDTC		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.22 DemJ1939NodeAddress

DemJ1939NodeAddress

SWS Item	[ECUC_Dem_00817]
Container Name	DemJ1939NodeAddress
Description	Contains the parameters for the support of a logical J1939 node (identified by an ECU address). **Attributes:** postBuildChangeable=true
Configuration Parameters	

Name	DemJ1939NmNodeRef [ECUC_Dem_00818]		
Description	Reference to the corresponding J1939Nm node.		
Multiplicity	1..255		
Type	Symbolic name reference to J1939NmNode		
Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

No Included Containers

10.2.23 DemPidClass

DemPidClass

SWS Item	[ECUC_Dem_00729]
Container Name	DemPidClass
Description	This container contains the different PIDs for the single global OBD relevant freeze frame class. It is assembled out of one or several data elements.
Configuration Parameters	

Name	DemPidIdentifier [ECUC_Dem_00705]		
Description	identifier of the PID		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
DemPidDataElement	1..255	This container contains the different data elements contained in the specific PID.

10.2.24 DemPidDataElement

DemPidDataElement

SWS Item	[ECUC_Dem_00896]
Container Name	DemPidDataElement
Description	This container contains the different data elements contained in the specific PID.
Configuration Parameters	

Name	DemPidDataElementClassRef [ECUC_Dem_00733]		
Description	This reference contains the link to a data element class.		
Multiplicity	1		
Type	Choice Reference to [DemExternalCSDataElement-Class,DemExternalSRDataElementClass,DemInternalDataElementClass]		
Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

No Included Containers

10.2.25 DemDTCAttributes

DemDTCAttributes

SWS Item	[ECUC_Dem_00641]
Container Name	DemDTCAttributes
Description	This container contains the configuration (parameters) for DemDTCAttributes.
Configuration Parameters	

Name	DemAgingAllowed [ECUC_Dem_00622]		
Description	Switch to allow aging/unlearning of the event or not. true: aging allowed false: aging not allowed		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemAgingCycleCounterThreshold [ECUC_Dem_00623]		
Description	Number of aging cycles needed to unlearn/delete the event.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 256		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemAgingAllowed		

Name	DemAgingCycleRef [ECUC_Dem_00624]		
Description	Reference to the cycle which is triggering the aging of the event. This can either be the same as the operation cycle of the event, or a separate aging cycle reported via API Dem_SetAgingCycleState. If external aging is configured (refer to DemAgingCycleCounterProcessing), this parameter is not used.		
Multiplicity	0..1		
Type	Choice Reference to [DemAgingCycle,DemOperationCycle]		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemAgingAllowed, DemOperationCycleRef, DemAgingCycleCounterProcessing		

Name	DemEventMemoryEntryFdcThresholdStorageValue [ECUC_Dem_00798]		
Description	Threshold to allocate an event memory entry and to capture the Freeze Frame.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: DemFreezeFrameRecordTrigger, DemExtendedDataRecordTrigger, DemEventMemoryEntryStorageTrigger = DEM_STORAGE_ON_FDC_THRESHOLD		

Name	DemEventPriority [ECUC_Dem_00662]	
Description	Priority of the event, in view of full event buffer.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	1 .. 256	

Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemEventSignificance [ECUC_Dem_00779]		
Description	Significance of the event, which indicates additional information concerning fault classification and resolution. It can be mapped as Dem-internal data element. It shall be configured, if it is a part of event related data.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	DEM_EVENT_SIGNIFICANCE_FAULT	failure, which affects the component/ECU itself	
	DEM_EVENT_SIGNIFICANCE_OCCURRENCE	issue, which indicates additional information concerning insufficient system behavior	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemExtendedDataClassRef [ECUC_Dem_00667]		
Description	This reference defines the link to an extended data class sampler.		
Multiplicity	0..1		
Type	Reference to DemExtendedDataClass		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name	DemFreezeFrameClassRef [ECUC_Dem_00674]		
Description	These references define the links to a freeze frame class sampler.		
Multiplicity	0..1		
Type	Reference to DemFreezeFrameClass		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name	DemFreezeFrameRecNumClassRef [ECUC_Dem_00776]		
Description	This parameter defines the list of dedicated freeze frame record numbers associated with the diagnostic event. These record numbers are assigned to the freeze frame records (instead of calculated record numbers). This parameter is only required for configured record numeration (refer to DemTypeOfFreezeFrameRecordNumeration).		
Multiplicity	0..1		
Type	Reference to DemFreezeFrameRecNumClass		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemTypeOfFreezeFrameRecordNumeration = DEM_FF_RECNUM_CONFIGURED		

Name	DemImmediateNvStorage [ECUC_Dem_00739]		
Description	Switch to enable immediate storage triggering of an according event memory entriy persistently to NVRAM. true: immediate non-volatile storage triggering enabled false: immediate non-volatile storage triggering disabled		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemJ1939DTC_NodeAddressRef [ECUC_Dem_00819]		
Description	Reference to a J1939 NodeAddress		
Multiplicity	0..1		
Type	Reference to DemJ1939NodeAddress		
Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency			

Name	DemJ1939ExpandedFreezeFrameClassRef [ECUC_Dem_00834]		
Description	These references define the links to a J1939 freeze frame class sampler.		
Multiplicity	0..1		
Type	Reference to DemJ1939FreezeFrameClass		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name	DemJ1939FreezeFrameClassRef [ECUC_Dem_00835]		
Description	These references define the links to a J1939 freeze frame class sampler.		
Multiplicity	0..1		
Type	Reference to DemJ1939FreezeFrameClass		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name	DemMaxNumberFreezeFrameRecords [ECUC_Dem_00605]		
Description	This parameter defines the number of according freeze frame records, which can maximal be stored for this event. Therefore all these freeze frame records have the same freeze frame class. This parameter is only required for calculated record numeration (refer to DemTypeOfFreezeFrameRecordNumeration).		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemTypeOfFreezeFrameRecordNumeration = DEM_FF_RECNUM_CALCULATED		

Name	DemMemoryDestination [ECUC_Dem_00890]		
Description	The event destination assigns events to none, one or multiple origins. If no event destination is assigned to a specific event, the event is handled internally and is not visible externally to the Dcm. If more than one event destination is assigned to a specific event, the event can be present in the corresponding origins.		
Multiplicity	0..2		
Type	EcucEnumerationParamDef		
Range	DEM_DTC_ORIGIN_MIRROR_MEMORY	Event information located in the mirror memory.	
	DEM_DTC_ORIGIN_PRIMARY_MEMORY	Event information located in the primary memory.	
	DEM_DTC_ORIGIN_SECONDARY_MEMORY	Event information located in the secondary memory.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**

DemCallbackInitMForF	0..*	The presence of this container indicates, that the Dem has access to an "InitMonitorForFunction" callback, which the Dem will call to initialize a monitor. In case the container has a DemCallbackInitMForFFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackInitMForFFnc, the Dem will have an R-Port requiring the interface CallbackInitMonitorForFunction, whose name is generated by using the unique callback-prefix followed by the event name.

10.2.26 DemCallbackInitMForF

DemCallbackInitMForF

SWS Item	[ECUC_Dem_00600]
Container Name	DemCallbackInitMForF
Description	The presence of this container indicates, that the Dem has access to an "InitMonitorForFunction" callback, which the Dem will call to initialize a monitor. In case the container has a DemCallbackInitMForFFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackInitMForFFnc, the Dem will have an R-Port requiring the interface CallbackInitMonitorForFunction, whose name is generated by using the unique callback-prefix followed by the event name.
Configuration Parameters	

Name	DemCallbackInitMForFFnc [ECUC_Dem_00633]		
Description	Function name of prototype "InitMonitorForFunction".		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default Value			
Regular Expression			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.27 DemEventParameter

DemEventParameter

SWS Item	[ECUC_Dem_00661]
Container Name	DemEventParameter
Description	This container contains the configuration (parameters) for events. Note that this container definition does not explicitly define a symbolic name parameter. Instead, the short name of the container will be used in the Ecu Configuration Description to specify the symbolic name of the diagnostic event.
Configuration Parameters	

Name	DemDTCRef [ECUC_Dem_00888]		
Description	This parameter defines the DTC configuration (typically Uds) associated with the diagnostic event. It is allowed to have events without a DTC (e.g. for ECU-internal events triggering safety reactions without being reported via diagnostic communication). The same DemDTCAttributes can be used from several events, to combine these (refer to chapter "Combination of diagnostic event").		
Multiplicity	0..1		
Type	Reference to DemDTC		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name	DemEnableConditionGroupRef [ECUC_Dem_00746]		
Description	References an enable condition group.		
Multiplicity	0..1		
Type	Reference to DemEnableConditionGroup		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemEventAvailable [ECUC_Dcm_00792]		
Description	This parameter configures an Event as unavailable. It is treated by Dem as if it does not exist. true = Event is available false = Event is not available		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

Name	DemEventFailureCycleCounterThreshold [ECUC_Dem_00753]		
Description	Defines the number of failure cycles for the event based fault confirmation. If this parameter is enabled, fault confirmation of the event is enabled accordingly.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemEventFailureCycleRef [ECUC_Dem_00752]		
Description	Kind of failure cycle for the event based fault confirmation. If fault confirmation of an event is enabled, but this parameter is disabled, the operation cycle (refer to DemOperationCycleRef) is used for fault confirmation.		
Multiplicity	0..1		
Type	Reference to DemOperationCycle		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOperationCycleRef		

Name	DemEventId [ECUC_Dem_00659]		
Description	Unique identifier of a diagnostic event. This parameter should not be changeable by user, because the Id should be generated by Dem itself to prevent gaps and multiple use of an Id. The events should be sequentially ordered beginning with 1 and no gaps in between.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	1 .. 65535		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemEventKind [ECUC_Dem_00660]		
Description	This parameter is used to distinguish between SW-C and BSW events. SW-C events are reported by Dem_SetEventStatus API and BSW events are reported by Dem_ReportErrorStatus API.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_EVENT_KIND_BSW	The event is a assigned to a BSW module	
	DEM_EVENT_KIND_SWC	The event is a assigned to a SW-C	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name	DemFFPrestorageSupported [ECUC_Dem_00671]		
Description	If this parameter is set to true, then the Prestorage of FreezeFrames is supported by the assigned event. This parameter is useful to calculate the buffer size.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOBDGroupingAssociativeEventsRef [ECUC_Dem_00839]		
Description	This parameter defines a reference which points to a representative event of one group of associate events. The "reverence event" must refer to it self. Note: One event is only allowed to be reverenced to only one group of associate events.		
Multiplicity	0..1		
Type	Reference to DemEventParameter		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemOperationCycleRef [ECUC_Dem_00702]		
Description	Kind of operation cycle for the event (e.g. power cycle, driving cycle, ...)		
Multiplicity	1		
Type	Reference to DemOperationCycle		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemStorageConditionGroupRef [ECUC_Dem_00769]		
Description	References a storage condition group.		
Multiplicity	0..1		
Type	Reference to DemStorageConditionGroup		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
DemCallbackClearEvent Allowed	0..1	The presence of this container indicates that the Dem has access to a "ClearEventAllowed" callback. In case there is a DemCallbackClearEventAllowedFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackClearEventAllowedFnc, the Dem will have an R-Port requiring the interface CallbackClearEventAllowed whose name is generated by using the unique callback-prefix followed by the event name.
DemCallbackEventData Changed	0..1	The presence of this container indicates that the Dem has access to an "EventDataChanged" callback. In case there is a DemCallbackEventDataChangedFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackEventDataChangedFnc, the Dem will have an R-Port requiring the interface CallbackEventDataChanged whose name is generated by using the unique callback-prefix followed by the event name.
DemCallbackEventStatus Changed	0..*	The presence of this container indicates, that the Dem has access to an "EventStatusChanged" callback, which the Dem will call to notify other components about the change in the status of an event. In case there is a DemCallbackEvenStatusChangedFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackEvenStatusChangedFnc, the Dem will have an R-Port requiring the interface CallbackEventStatusChanged, whose name is generated by using the unique callback-prefix followed by the event name.

DemCallbackInitMForE	0..1	The presence of this container indicates, that the Dem has access to an "InitMonitorForEvent" callback, which the Dem will call to initialize a monitor. In case the container has a DemCallbackInitMForEFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackInitMForEFnc, the Dem will have an R-Port requiring the interface CallbackInitMonitorForEvent, whose name is generated by using the unique callback-prefix followed by the event name.
DemDebounceAlgorithm Class [DemDebounceCounter Based, DemDebounce MonitorInternal, Dem DebounceTimeBase]	1	Debounce algorithm class: counter based, time based, or monitor internal.
DemIndicatorAttribute	0..255	This container contains the event specific configuration of Indicators.

10.2.28 DemIndicatorAttribute

DemIndicatorAttribute

SWS Item	[ECUC_Dem_00681]
Container Name	DemIndicatorAttribute
Description	This container contains the event specific configuration of Indicators.
Configuration Parameters	

Name	DemIndicatorBehaviour [ECUC_Dem_00682]	
Description	Behaviour of the linked indicator	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	DEM_INDICATOR_BLINKING	The indicator blinks when the event has status FAILED Not relevant with J1939.
	DEM_INDICATOR_BLINK_CONT	The indicator is active and blinks when the event has status FAILED Not relevant with J1939.
	DEM_INDICATOR_CONTINUOUS	The indicator is active when the even has status FAILED
	DEM_INDICATOR_FAST_FLASH	Flash Indicator Lamp should be set to 'Fast Flash'

	DEM_INDICATOR_SLOW_FLASH	Flash Indicator Lamp should be set to 'Slow Flash'	
Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

Name	DemIndicatorFailureCycleCounterThreshold [ECUC_Dem_00750]		
Description	Defines the number of failure cycles for the WarningIndicatorOnCriteria.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemIndicatorFailureCycleRef [ECUC_Dem_00751] (Obsolete)		
Description	Kind of failure cycle for the indicator controlled by the according event used for the WarningIndicatorOnCriteria. **Tags:** atp.Status=obsolete atp.StatusRevisionBegin=4.1.3		
Multiplicity	0..1		
Type	Reference to DemOperationCycle		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemIndicatorFailureCycleSource [ECUC_Dem_00747]	
Description	This parameter defines, which failure cycle is used for the WarningIndicatorOnCriteria handling.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	DEM_FAILURE_CYCLE_EVENT	The event based failure cycle configured in DemEventParameter is used. Therefore, the parameters DemIndicatorFailureCycleRef and DemIndicatorFailureCycleCounterThreshold are not used for this indicator attribute of the event.

	DEM_FAILURE_CYCLE_INDICATOR	An indicator based failure cycle is used, defined by DemIndicatorFailureCycleRef and DemIndicatorFailureCycleCounterThreshold.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemIndicatorHealingCycleCounterThreshold [ECUC_Dem_00748]		
Description	Defines the number of healing cycles for the WarningIndicatorOffCriteria.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemIndicatorHealingCycleRef [ECUC_Dem_00749] (Obsolete)		
Description	Kind of healing cycle for the indicator controlled by the according event used for the WarningIndicatorOffCriteria. **Tags:** atp.Status=obsolete atp.StatusRevisionBegin=4.1.3		
Multiplicity	1		
Type	Reference to DemOperationCycle		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemIndicatorRef [ECUC_Dem_00687]		
Description	Reference to the used indicator.		
Multiplicity	1		
Type	Reference to DemIndicator		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.29 DemDebounceAlgorithmClass

DemDebounceAlgorithmClass

SWS Item	[ECUC_Dem_00604]
Container Name	DemDebounceAlgorithmClass
Description	Debounce algorithm class: counter based, time based, or monitor internal.
Configuration Parameters	
No Included Containers	

10.2.30 DemDebounceCounterBased

DemDebounceCounterBased

SWS Item	[ECUC_Dem_00711]
Container Name	DemDebounceCounterBased
Description	This container contains the configuration (parameters) for counter based debouncing.
Configuration Parameters	

Name	DemDebounceCounterBasedClassRef [ECUC_Dem_00883]		
Description	This reference selects the DemDebounceCounterBasedClass applied for the debouncing of the DemEventParameter.		
Multiplicity	1		
Type	Reference to DemDebounceCounterBasedClass		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.31 DemDebounceCounterBasedClass

DemDebounceCounterBasedClass

SWS Item	[ECUC_Dem_00881]
Container Name	DemDebounceCounterBasedClass
Description	This container contains the configuration of Debounce Counter Based Class
Configuration Parameters	

Name	DemDebounceBehavior [ECUC_Dem_00786]		
Description	This parameter defines how the event debounce algorithm will behave, if a related enable condition is not fulfilled or ControlDTCSetting of the related event is disabled.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_DEBOUNCE_FREEZE	The event debounce counter will be frozen with the current value and will not change while a related enable condition is not fulfilled or ControlDTCSetting of the related event is disabled. After all related enable conditions are fulfilled and ControlDTCSetting of the related event is enabled again, the event qualification will continue with the next report of the event (i.e. SetEventStatus).	
	DEM_DEBOUNCE_RESET	The event debounce counter will be reset to initial value if a related enable condition is not fulfilled or ControlDTCSetting of the related event is disabled. The qualification of the event will be restarted with the next valid event report.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemDebounceCounterDecrementStepSize [ECUC_Dem_00635]		
Description	Defines the step size for decrementation of the internal debounce counter (PREPASSED).		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 32768		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDebounceCounterFailedThreshold [ECUC_Dem_00618]	
Description	Defines the value of the internal debounce counter, which indicates the failed status.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	1 .. 32767	
Default Value		

Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDebounceCounterIncrementStepSize [ECUC_Dem_00637]		
Description	Defines the step size for incrementation of the internal debounce counter (PREFAILED).		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 32767		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDebounceCounterJumpDown [ECUC_Dem_00685]		
Description	Switch for the activation of Jump-Down. true: Jump-Down activated false: Jump-Down deactivated		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDebounceCounterJumpDownValue [ECUC_Dem_00638]		
Description	Jump-Down value of the internal debounce counter which is taken as initialization value for the counter when the respective step-down occurs.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	-32768 .. 32767		
Default Value	0		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemDebounceCounterJumpDown		

Name	DemDebounceCounterJumpUp [ECUC_Dem_00686]		
Description	Switch for the activation of Jump-Up. true: Jump-Up activated false: Jump-Up deactivated		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDebounceCounterJumpUpValue [ECUC_Dem_00639]		
Description	Jump-Up value of the internal debounce counter which is taken as initialization value for the counter when the respective step-up occurs.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	-32768 .. 32767		
Default Value	0		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemDebounceCounterJumpUp		

Name	DemDebounceCounterPassedThreshold [ECUC_Dem_00636]		
Description	Defines the value of the internal debounce counter, which indicates the passed status.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	-32768 .. -1		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDebounceCounterStorage [ECUC_Dem_00791]
Description	Switch to store the debounce counter value non-volatile or not. true: debounce counter value shall be stored non-volatile false: debounce counter value is volatile
Multiplicity	1
Type	EcucBooleanParamDef
Default Value	false

Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: DemOperationCycleStatusStorage = True		

No Included Containers

10.2.32 DemDebounceTimeBase

DemDebounceTimeBase

SWS Item	[ECUC_Dem_00713]
Container Name	DemDebounceTimeBase
Description	This container contains the configuration (parameters) for time based debouncing.
Configuration Parameters	

Name	DemDebounceTimeBaseRef [ECUC_Dem_00891]		
Description	This reference selects the DemDebounceTimeBaseClass applied for the debouncing of the DemEventParameter.		
Multiplicity	1		
Type	Reference to DemDebounceTimeBaseClass		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.33 DemDebounceTimeBaseClass

DemDebounceTimeBaseClass

SWS Item	[ECUC_Dem_00882]
Container Name	DemDebounceTimeBaseClass
Description	This container contains the configuration of Debounce Counter Based Class
Configuration Parameters	

Name	DemDebounceBehavior [ECUC_Dem_00789]		
Description	This parameter defines how the event debounce algorithm will behave, if a related enable condition is not fulfilled or ControlDTCSetting of the related event is disabled.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_DEBOUNCE_FREEZE	The event debounce timer will be frozen with the current value and will not change while a related enable condition is not fulfilled or ControlDTCSetting of the related event is disabled. After all related enable conditions are fulfilled and ControlDTCSetting of the related event is enabled again, the event qualification will continue with the next report of the event (i.e. SetEventStatus).	
	DEM_DEBOUNCE_RESET	The event debounce timer will be reset to initial value if a related enable condition is not fulfilled or ControlDTCSetting of the related event is disabled. The qualification of the event will be restarted with the next valid event report.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemDebounceTimeFailedThreshold [ECUC_Dem_00716]		
Description	Defines the time out duration for "Event Failed" qualification. The AUTOSAR configuration standard is to use SI units, so this parameter is defined as float value in seconds. Dem configuration tools must convert this float value to the appropriate value format for the use in the software implementation of Dem.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	0.001 .. 3600		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDebounceTimePassedThreshold [ECUC_Dem_00717]		
Description	Defines the time out duration for "Event Passed" qualification. The AUTOSAR configuration standard is to use SI units, so this parameter is defined as float value in seconds. Dem configuration tools must convert this float value to the appropriate value format for the use in the software implementation of Dem.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	0.001 .. 3600		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.34 DemDebounceMonitorInternal

DemDebounceMonitorInternal

SWS Item	[ECUC_Dem_00712]
Container Name	DemDebounceMonitorInternal
Description	This container contains the configuration (parameters) for monitor internal debouncing.
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
DemCallbackGetFDC	0..1	The presence of this container indicates, that the Dem has access to a "GetFaultDetectionCounter" callback, which the Dem will call to obtain the value of the fault detection counter. In case the container has a DemCallbackGetFDCFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackGetFDCFnc, the Dem will have a R-Port requiring the interface CallbackGetFaultDetectionCounter, whose name is generated by using the unique callback-prefix followed by the event name.

10.2.35 DemCallbackGetFDC

DemCallbackGetFDC

SWS Item	[ECUC_Dem_00630]
Container Name	DemCallbackGetFDC
Description	The presence of this container indicates, that the Dem has access to a "GetFaultDetectionCounter" callback, which the Dem will call to obtain the value of the fault detection counter. In case the container has a DemCallbackGetFDCFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackGetFDCFnc, the Dem will have a R-Port requiring the interface CallbackGetFaultDetectionCounter, whose name is generated by using the unique callback-prefix followed by the event name.
Configuration Parameters	

Name	DemCallbackGetFDCFnc [ECUC_Dem_00631]		
Description	Function name of prototype "GetFaultDetectionCounter".		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default Value			
Regular Expression			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.36 DemCallbackClearEventAllowed

DemCallbackClearEventAllowed

SWS Item	[ECUC_Dem_00607]
Container Name	DemCallbackClearEventAllowed
Description	The presence of this container indicates that the Dem has access to a "ClearEventAllowed" callback. In case there is a DemCallbackClearEventAllowedFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackClearEventAllowedFnc, the Dem will have an R-Port requiring the interface CallbackClearEventAllowed whose name is generated by using the unique callback-prefix followed by the event name.
Configuration Parameters	

Name	DemCallbackClearEventAllowedFnc [ECUC_Dem_00609]		
Description	Function name of prototype "ClearEventAllowed".		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default Value			
Regular Expression			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemClearEventAllowedBehavior [ECUC_Dem_00788]		
Description	Defines the resulting UDS status byte for the related event, which must not be cleared according to the ClearEventAllowed callback.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_NO_STATUS_BYTE_CHANGE	The event status byte keeps unchanged. (default)	
	DEM_ONLY_THIS_CYCLE_AND_READINESS	The <...>ThisOperationCycle and readiness bits of the event status byte are reset.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

10.2.37 DemCallbackEventDataChanged

DemCallbackEventDataChanged

SWS Item	[ECUC_Dem_00606]
Container Name	DemCallbackEventDataChanged
Description	The presence of this container indicates that the Dem has access to an "EventDataChanged" callback. In case there is a DemCallbackEventDataChangedFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackEventDataChangedFnc, the Dem will have an R-Port requiring the interface CallbackEventDataChanged whose name is generated by using the unique callback-prefix followed by the event name.
Configuration Parameters	

Name	DemCallbackEventDataChangedFnc [ECUC_Dem_00608]		
Description	Function name of prototype "EventDataChanged"		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default Value			
Regular Expression			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.38 DemCallbackEventStatusChanged

DemCallbackEventStatusChanged

SWS Item	[ECUC_Dem_00628]
Container Name	DemCallbackEventStatusChanged
Description	The presence of this container indicates, that the Dem has access to an "EventStatusChanged" callback, which the Dem will call to notify other components about the change in the status of an event. In case there is a DemCallbackEvenStatusChangedFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackEvenStatusChangedFnc, the Dem will have an R-Port requiring the interface CallbackEventStatusChanged, whose name is generated by using the unique callback-prefix followed by the event name.
Configuration Parameters	

Name	DemCallbackEventStatusChangedFnc [ECUC_Dem_00629]		
Description	Function name of prototype "EventStatusChanged"		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default Value			
Regular Expression			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.39 DemCallbackInitMForE

DemCallbackInitMForE

SWS Item	[ECUC_Dem_00632]
Container Name	DemCallbackInitMForE
Description	The presence of this container indicates, that the Dem has access to an "InitMonitorForEvent" callback, which the Dem will call to initialize a monitor. In case the container has a DemCallbackInitMForEFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackInitMForEFnc, the Dem will have an R-Port requiring the interface CallbackInitMonitorForEvent, whose name is generated by using the unique callback-prefix followed by the event name.
Configuration Parameters	

Name	DemCallbackInitMForEFnc [ECUC_Dem_00601]		
Description	Function name of prototype "InitMonitorForEvent".		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default Value			
Regular Expression			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.40 DemDtr

DemDtr

SWS Item	[ECUC_Dem_00806]
Container Name	DemDtr
Description	This container holds the configuration of one individual DTR.
Configuration Parameters	

Name	DemDtrCompuDenominator0 [ECUC_Dem_00815]		
Description	Part of the conversion between the binary representation and the physical meaning analogous to the SW-C Template conversion CompuRationalCoeffs with 2 numerator coefficients and 1 denominator coefficient in the direction compuInternalToPhys. This is the only one supported denominator value, a constant divisor. The value 0 is not allowed.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	-INF .. INF		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemDtrCompuNumerator0 [ECUC_Dem_00813]		
Description	Part of the conversion between the binary representation and the physical meaning analogous to the SW-C Template conversion CompuRationalCoeffs with 2 numerator coefficients and 1 denominator coefficient in the direction compuInternalToPhys. This is the first numerator value, which is multiplied with x^ 0, i.e., the offset.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	-INF .. INF		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDtrCompuNumerator1 [ECUC_Dem_00814]		
Description	Part of the conversion between the binary representation and the physical meaning analogous to the SW-C Template conversion CompuRationalCoeffs with 2 numerator coefficients and 1 denominator coefficient in the direction compuInternalToPhys. This is the second numerator value, which is multiplied with x^ 1, i.e., the factor.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	-INF .. INF		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemDtrEventRef [ECUC_Dem_00808]		
Description	Reference to the DemEventParameter this DTR is related to. If the related event is not configured, the Dem cannot ensure consistency between the DTR and the event.		
Multiplicity	0..1		
Type	Reference to DemEventParameter		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemDtrId [ECUC_Dem_00807]		
Description	The index identifier value assigned to this DTR. The value is generated during the Dem configuration process.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 65535		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemDtrMid [ECUC_Dem_00809]		
Description	The OBDMID of the DTR. The values 0x00, 0x20, 0x40, 0x60, 0x80, 0xA0, 0xC0, 0xE0 are reserved.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemDtrTid [ECUC_Dem_00810]		
Description	The OBDTID of the DTR.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default Value			

Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemDtrUasid [ECUC_Dem_00811]		
Description	The UaSId the DTR data shall be scaled to, and reported together with the rescaled DTR data.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU		

Name	DemDtrUpdateKind [ECUC_Dem_00812]		
Description	Update conditions applied by the Dem to reports of DTR values. Only supported if a related Event is configured. If no related Event is configured, the Dem behaves as if DemDtrUpdateKind is configured to "DEM_DTR_UPDATE_ALWAYS".		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	DEM_DTR_UPDATE_ALWAYS	Any DTR result reported by the monitor is used by the Dem.	
	DEM_DTR_UPDATE_STEADY	The Dem accepts reported DTRs only when the configured debouncing mechanism is stable at the FAIL or PASS limit.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: DemOBDSupport == DEM_OBD_MASTER_ECU || DEM_OBD_PRIMARY_ECU, DemDtrEventRef		

No Included Containers

10.2.41 DemFreezeFrameClass

DemFreezeFrameClass

SWS Item	[ECUC_Dem_00673]
Container Name	DemFreezeFrameClass
Description	This container contains the combinations of DIDs for a non OBD relevant freeze frame class.
Configuration Parameters	

Name	DemDidClassRef [ECUC_Dem_00707]		
Description	For OBD relevant data Multiple PIDs can be relevant per freeze frame.		
Multiplicity	1..255		
Type	Reference to DemDidClass		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.42 DemDidClass

DemDidClass

SWS Item	[ECUC_Dem_00706]
Container Name	DemDidClass
Description	This container contains the configuration (parameters) for a data Id class. It is assembled out of one or several data elements.
Configuration Parameters	

Name	DemDidDataElementClassRef [ECUC_Dem_00617]		
Description	This reference contains the link to a data element class.		
Multiplicity	1..255		
Type	Choice Reference to [DemExternalCSDataElement-Class,DemExternalSRDataElementClass,DemInternalDataElementClass]		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDidIdentifier [ECUC_Dem_00650]	
Description	Identifier of the Data ID.	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 65535	

Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.43 DemJ1939FreezeFrameClass

DemJ1939FreezeFrameClass

SWS Item	[ECUC_Dem_00828]
Container Name	DemJ1939FreezeFrameClass
Description	This container contains the combinations of SPNs s for a J1939 relevant freeze frame.
Configuration Parameters	

Name	DemSPNClassRef [ECUC_Dem_00829]		
Description	Reference to an SPN. This reference defines requiresIndex = true since it represents a ordered list of references where the order describes the order of single SPNs in the J1939 Freeze Frame. **Attributes:** requiresIndex=true		
Multiplicity	1..255		
Type	Reference to DemSPNClass		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.44 DemSPNClass

DemSPNClass

SWS Item	[ECUC_Dem_00830]
Container Name	DemSPNClass
Description	This container contains the configuration (parameters) for a SPN.
Configuration Parameters	

Name	DemSPNDataElementClassRef [ECUC_Dem_00832]		
Description	This reference contains the link to a data element class.		
Multiplicity	1		
Type	Choice Reference to [DemExternalCSDataElement-Class,DemExternalSRDataElementClass,DemInternalDataElementClass]		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemSPNId [ECUC_Dem_00831]		
Description	Suspect parameter number		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 524287		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

10.2.45 DemFreezeFrameRecordClass

DemFreezeFrameRecordClass

SWS Item	[ECUC_Dem_00801]
Container Name	DemFreezeFrameRecordClass
Description	This container contains a list of dedicated, different freeze frame record numbers.
Configuration Parameters	

Name	DemFreezeFrameRecordNumber [ECUC_Dem_00777]		
Description	This parameter defines a record number for a freeze frame record. This record number is unique per freeze frame record number class. The range of this value is defined by ISO 14229-1 (0x01 .. 0xFE).		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 254		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemFreezeFrameRecordTrigger [ECUC_Dem_00803]		
Description	Configures the primary trigger to allocate an event memory entry.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_TRIGGER_ON_CONFIRMED		
	DEM_TRIGGER_ON_FDC_THRESHOLD		
	DEM_TRIGGER_ON_PENDING		
	DEM_TRIGGER_ON_TEST_FAILED		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: DemTypeOfFreezeFrameRecordNumeration == DEM_FF_RECNUM_CONFIGURED		

Name	DemFreezeFrameRecordUpdate [ECUC_Dem_00802] (Obsolete)		
Description	This parameter defines the case, when the freeze frame record is stored/updated. **Tags:** atp.Status=obsolete atp.StatusRevisionBegin=4.1.1		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_UPDATE_RECORD_NO	This record is only captured for new event memory entries.	
	DEM_UPDATE_RECORD_YES	This record is captured every time.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.46 DemFreezeFrameRecNumClass

DemFreezeFrameRecNumClass

SWS Item	[ECUC_Dem_00775]
Container Name	DemFreezeFrameRecNumClass
Description	This container contains a list of dedicated, different freeze frame record numbers assigned to an event. The order of record numbers in this list is assigned to the chronological order of the according freeze frame records. dependency: DemTypeOfFreezeFrameRecordNumeration = DEM_FF_RECNUM_CONFIGURED
Configuration Parameters	

Name	DemFreezeFrameRecordClassRef [ECUC_Dem_00800]		
Description	This parameter references record number(s) for a freeze frame record. **Attributes:** requiresIndex=true		
Multiplicity	1..254		
Type	Reference to DemFreezeFrameRecordClass		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemTypeOfFreezeFrameRecordNumeration = DEM_FF_RECNUM_CONFIGURED		

No Included Containers

10.2.47 DemExtendedDataClass

DemExtendedDataClass

SWS Item	[ECUC_Dem_00664]
Container Name	DemExtendedDataClass
Description	This class contains the combinations of extended data records for an extended data class.
Configuration Parameters	

Name	DemExtendedDataRecordClassRef [ECUC_Dem_00774]		
Description	This reference contains the link to an extended data class record.		
Multiplicity	1..253		
Type	Reference to DemExtendedDataRecordClass		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.48 DemExtendedDataRecordClass

DemExtendedDataRecordClass

SWS Item	[ECUC_Dem_00665]
Container Name	DemExtendedDataRecordClass
Description	This container contains the configuration (parameters) for an extended data record class. It is assembled out of one or several data elements.
Configuration Parameters	

Name	DemDataElementClassRef [ECUC_Dem_00771]		
Description	This reference contains the link to a data element class.		
Multiplicity	1..255		
Type	Choice Reference to [DemExternalCSDataElement-Class,DemExternalSRDataElementClass,DemInternalDataElementClass]		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name	DemExtendedDataRecordNumber [ECUC_Dem_00666]	
Description	This configuration parameter specifies an unique identifier for an extended data record. One or more extended data records can be assigned to one diagnostic event/DTC. 0x00 is reserved by ISO (therefore the minimal value equals 1) 0xF0 to 0xFF are reserved by ISO (therefore the maximal value equals 239)	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	1 .. 239	
Default Value		

Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemExtendedDataRecordTrigger [ECUC_Dem_00804]		
Description	Configures the primary trigger to allocate an event memory entry.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	DEM_TRIGGER_ON_CONFIRMED		
	DEM_TRIGGER_ON_FDC_THRESHOLD		
	DEM_TRIGGER_ON_PENDING		
	DEM_TRIGGER_ON_TEST_FAILED		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemExtendedDataRecordUpdate [ECUC_Dem_00621]		
Description	This extended data record is captured if the configured trigger condition in "DemExtendedDataRecordTrigger" is fulfilled.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	DEM_UPDATE_RECORD_NO	This extended data record is only captured for new event memory entries.	
	DEM_UPDATE_RECORD_YES	This extended data record is captured every time.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.49 DemDataElementClass

DemDataElementClass

SWS Item	[ECUC_Dem_00610]
Container Name	DemDataElementClass
Description	This container contains the configuration (parameters) for an internal/external data element class.
Configuration Parameters	
No Included Containers	

10.2.50 DemDataElementInstance

DemDataElementInstance

SWS Item	[ECUC_Dem_00875]
Container Name	DemDataElementInstance
Description	Instance Reference to the primitive data in a port where the data element is typed with an ApplicationPrimitveDataType or an ImplementationDataType.
Configuration Parameters	

Name	DemDataElementInstanceRef [ECUC_Dem_00861]		
Description	Instance Reference to the primitive data which shall be read or written. Supported are VariableDataPrototypes in SenderReceiverInterfaces and NvDataInterfaces and ParameterDataPrototypes in ParameterInterfaces (read only). This reference is applicable if the AutosarDataPrototype is typed with a ApplicationPrimitiveDataType of category VALUE or BOOLEAN or if the AutosarDataPrototype is typed with a ImplementationDataType of category VALUE or TYPE_REFERENCE that in turn boils down to VALUE		
Multiplicity	1		
Type	Instance reference to AUTOSAR-DATA-PROTOTYPE context: ROOT-SW-COMPOSITION-PROTOTYPE SW-COMPONENT-PROTOTYPE PORT-PROTOTYPE		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.51 DemInternalDataElementClass

DemInternalDataElementClass

SWS Item	[ECUC_Dem_00684]
Container Name	DemInternalDataElementClass
Description	This container contains the configuration (parameters) for an internal data element class.
Configuration Parameters	

Name	DemDataElementDataSize [ECUC_Dem_00614]		
Description	Defines the size of the data element in bytes.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemInternalDataElement		

Name	DemInternalDataElement [ECUC_Dem_00616]	
Description	This parameter defines the Dem-internal data value, which is mapped to the data element.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	DEM_AGINGCTR_DOWNCNT	map down-counting Dem-internal aging counter, max. range: 1 byte
	DEM_AGINGCTR_UPCNT	map up-counting Dem-internal aging counter, max. range: 1 byte
	DEM_CURRENT_FDC	map Dem-internal fault detection counter, max. range: 1 byte
	DEM_CYCLES_SINCE_FIRST_FAILED	map Dem-internal Operation Cycle Counter - Cycles since first failed, max. range: 1 byte
	DEM_CYCLES_SINCE_LAST_FAILED	map Dem-internal Operation Cycle Counter - Cycles since last failed, max. range: 1 byte
	DEM_FAILED_CYCLES	map Dem-internal Operation Cycle Counter - Failed cycles, max. range: 1 byte
	DEM_MAX_FDC_DURING_CURRENT_CYCLE	map Dem-internal DTC Fault Detection Counter maximum value during current operation cycle, max. range: 1 byte
	DEM_MAX_FDC_SINCE_LAST_CLEAR	map Dem-internal DTC Fault Detection Counter maximum value since last clear, max. range: 1 byte
	DEM_OCCCTR	map Dem-internal occurrence counter, max. range: 1 byte

	DEM_OVFLIND	map Dem-internal overflow indication, max. range: 1 byte (0 = False, 1 = True)	
	DEM_SIGNIFICANCE	map (static) Dem-internal event significance (refer to DemEventSignificance), max. range: 1 byte (0 = OCCURRENCE, 1 = FAULT)	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.52 DemExternalCSDataElementClass

DemExternalCSDataElementClass

SWS Item	[ECUC_Dem_00668]
Container Name	DemExternalCSDataElementClass
Description	This container contains the configuration (parameters) for an external client/server based data element class. It defines, how the Dem can obtain the value of the data element from either a SW-C or another BSW module. Whether a client/server port or a C function-call is used, is defined by DemDataElementUsePort.
Configuration Parameters	

Name	DemDataElementDataSize [ECUC_Dem_00646]		
Description	Defines the size of the data element in bytes.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDataElementReadFnc [ECUC_Dem_00619]
Description	In case of DemDataElementUsePort is false, this parameter defines the prototype of the C function "ReadDataElement" used to get the according value.
Multiplicity	0..1
Type	EcucFunctionNameDef
Default Value	
Regular Expression	

Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDataElementUsePort [ECUC_Dem_00647]		
Description	If the parameter is set to True, a R-Port is generated, to obtain the data element (interface DataServices_<SyncDataElement>). If the parameter is set to False, the information is obtained by C function-call on another BSW module specified by the parameter DemDataElementReadFnc.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.53 DemExternalSRDataElementClass

DemExternalSRDataElementClass

SWS Item	[ECUC_Dem_00669]
Container Name	DemExternalSRDataElementClass
Description	This container contains the configuration (parameters) for an external sender/receiver based data element class. It defines, how the Dem can obtain the value of the data element from a SW-C, by using a sender/receiver port.
Configuration Parameters	

Name	DemDataElementDataSize [ECUC_Dem_00615]		
Description	Defines the size of the data element in bits.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDataElementDataType [ECUC_Dem_00840]		
Description	Provide the implementation data type of data belonging to a external data.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	BOOLEAN	type of the data is boolean.	
	SINT16	type of the data is sint16.	
	SINT32	type of the data is sint32.	
	SINT8	type of the data is sint8.	
	UINT16	type of the data is uint16.	
	UINT32	type of the data is uint32.	
	UINT8	type of the data is uint8.	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU dependency: DemDataSize		

Name	DemDataElementEndianness [ECUC_Dem_00841]		
Description	Defines the endianness of the data belonging to an external data. If no DemDataElementEndianness is defined the value of DemDataElementDefaultEndianness is applicable.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	BIG_ENDIAN	Most significant byte shall come at the lowest address.	
	LITTLE_ENDIAN	Most significant byte shall come highest address	
	OPAQUE	opaque data endianness	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local dependency: DemDataElementDefaultEndianness		

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
DemDiagnosisScaling [DemAlternativeData Interface, DemAlternative DataProps, Dem AlternativeDataType]	0..1	This container contains the configuration (parameters) of an alternative Diagnosis Representation. Out if this the scaling between Diagnosis and ECU internal representation and vice versa can be calculated.
DemSRDataElement Class [DemDataElement Instance, DemSub ElementInDataElement Instance, DemSub ElementInImplData ElementInstance]	1	This container defines the source of data in a provided port which shall be read for a external data element This container shall contain either one DemSubElementInDataElementInstance OR DemDataElementInstance OR DemSubElementInImplDataElementInstance reference.

10.2.54 DemSRDataElementClass

DemSRDataElementClass

SWS Item	[ECUC_Dem_00842]
Container Name	DemSRDataElementClass
Description	This container defines the source of data in a provided port which shall be read for a external data element This container shall contain either one DemSubElementInDataElementInstance OR DemDataElementInstance OR DemSubElementInImplDataElementInstance reference.
Configuration Parameters	
No Included Containers	

10.2.55 DemSubElementInDataElementInstance

DemSubElementInDataElementInstance

SWS Item	[ECUC_Dem_00874]
Container Name	DemSubElementInDataElementInstance
Description	Instance Reference to the primitve sub-element (at any level) of composite data in a port where the data element is typed with an ApplicationCompositeDataType.
Configuration Parameters	

Name	DemSubElementInDataElementInstanceRef [ECUC_Dem_00860]		
Description	Instance Reference to the primitve sub-element (at any level) of composite data in a port which shall be read or written. Supported are VariableDataPrototypes in SenderReceiverInterfaces and NvDataInterfaces and ParameterDataPrototypes in ParameterInterfaces (read only). This reference is applicable if the AutosarDataPrototype is typed with a ApplicationCompositeDataType.		
Multiplicity	1		
Type	Instance reference to AUTOSAR-DATA-PROTOTYPE context: ROOT-SW-COMPOSITION-PROTOTYPE SW-COMPONENT-PROTOTYPE PORT-PROTOTYPE AUTOSAR-DATA-PROTOTYPE APPLICATION-COMPOSITE-ELEMENT-DATA-PROTOTYPE*		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.56 DemSubElementInImplDataElementInstance

DemSubElementInImplDataElementInstance

SWS Item	[ECUC_Dem_00876]
Container Name	DemSubElementInImplDataElementInstance
Description	Instance Reference to the primitve sub-element (at any level) of composite data in a port where the data element is typed with an ImplementationDataType of category STRUCTURE or ARRAY.
Configuration Parameters	

Name	DemSubElementInImplDataElementInstanceRef [ECUC_Dem_00862]		
Description	Instance Reference to the primitve sub-element (at any level) of composite data in a port which shall be read or written. Supported are VariableDataPrototypes in SenderReceiverInterfaces and NvDataInterfaces and ParameterDataPrototypes in ParameterInterfaces (read only). This reference is applicable if the AutosarDataPrototype is typed with a ImplementationDataType of category STRUCTURE or ARRAY. Please note that in case of ARRAY the index attribute in the target reference has to be set to select a single array element.		
Multiplicity	1		
Type	Instance reference to IMPLEMENTATION-DATA-TYPE-ELEMENT context: ROOT-SW-COMPOSITION-PROTOTYPE SW-COMPONENT-PROTOTYPE PORT-PROTOTYPE AUTOSAR-DATA-PROTOTYPE I MPLEMENTATION-DATA-TYPE-ELEMENT*		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.57 DemDiagnosisScaling

DemDiagnosisScaling

SWS Item	[ECUC_Dem_00843]
Container Name	DemDiagnosisScaling
Description	This container contains the configuration (parameters) of an alternative Diagnosis Representation. Out if this the scaling between Diagnosis and ECU internal representation and vice versa can be calculated.
Configuration Parameters	
No Included Containers	

10.2.58 DemAlternativeDataInterface

DemAlternativeDataInterface

SWS Item	[ECUC_Dem_00844]
Container Name	DemAlternativeDataInterface
Description	This container contains the configuration (parameters) of an alternative Diagnosis Representation by the means of a VariableDataPrototype in a DataInterface. Additionally a reference to PortInterfaceMapping can be defined which provide already the mapping rules between the VariableDataPrototype in a DataInterface used by the software component (DemSRDataElementClass) and the intended Diagnosis Representation defined by DemExternalSRDataElementClass.
Configuration Parameters	

Name	DemDataElement [ECUC_Dem_00845]		
Description	Alternative Diagnosis Representation for the data defined by the means of a VariableDataPrototype in a DataInterface.		
Multiplicity	1		
Type	Foreign reference to VARIABLE-DATA-PROTOTYPE		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name	DemPortInterfaceMapping [ECUC_Dem_00846]		
Description	Optional reference to PortInterfaceMapping which defines the mapping rules.		
Multiplicity	0..1		
Type	Foreign reference to PORT-INTERFACE-MAPPING		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.59 DemAlternativeDataType

DemAlternativeDataType

SWS Item	[ECUC_Dem_00847]
Container Name	DemAlternativeDataType
Description	This container contains the configuration (parameters) of an alternative Diagnosis Representation by the means of an ApplicationDataType. Additionally the definition of a text table mapping can be a defined for ApplicationDataTypes of category TEXTTABLE and SCALE_LINEAR_AND_TEXTTABLE.
Configuration Parameters	

Name	DemApplicationDataType [ECUC_Dem_00848]		
Description	Alternative Diagnosis Representation for the data defined by the means of a ApplicationPrimitiveDataType of category VALUE or BOOLEAN.		
Multiplicity	1		
Type	Foreign reference to APPLICATION-PRIMITIVE-DATA-TYPE		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Included Containers		
Container Name	**Multiplicity**	**Scope / Depedency**
DemTextTableMapping	0..*	This container contains the configuration (parameters) of the mapping a DataPrototype typed by AutosarDataType that refer to a CompuMethods of category TEXTTABLE or SCALE_LINEAR_AND_TEXTTABLE. Each DemTextTableMapping defines a value pair which is used to map the ECU internal value (DemInternalDataValue) to the vale used in the diagnosis representation (DemDiagnosisRepresentationDataValue) and vice versa. The set of all DemTextTableMapping defines the whole mapping of an data.

10.2.60 DemAlternativeDataProps

DemAlternativeDataProps

SWS Item	[ECUC_Dem_00852]
Container Name	DemAlternativeDataProps
Description	This container contains the configuration (parameters) of an alternative Diagnosis Representation by the means of ECU configuration parameters. The physical unit of the alternative data representation is defined by the DataPrototype referenced by DemSRDataElementClass. Additionally the definition of a text table mapping can be a defined for DemDataTypeCategory TEXTTABLE and SCALE_LINEAR_AND_TEXTTABLE
Configuration Parameters	

Name	DemDataTypeCategory [ECUC_Dem_00857]		
Description	Data category of the alternative Diagnosis Representation.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	SCALE_LINEAR_AND_TEXTTABLE	category is SCALE_LINEAR_AND_TEXTTABLE	
	TEXTTABLE	category is TEXTTABLE	
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Included Containers

Container Name	Multiplicity	Scope / Depedency
DemLinearScale	0..1	This container contains the configuration (parameters) of an linear scale of the alternative Diagnosis Representation.
DemTextTableMapping	0..*	This container contains the configuration (parameters) of the mapping a DataPrototype typed by AutosarDataType that refer to a CompuMethods of category TEXTTABLE or SCALE_LINEAR_AND_TEXTTABLE. Each DemTextTableMapping defines a value pair which is used to map the ECU internal value (DemInternalDataValue) to the vale used in the diagnosis representation (DemDiagnosisRepresentationDataValue) and vice versa. The set of all DemTextTableMapping defines the whole mapping of an data.

10.2.61 DemLinearScale

DemLinearScale

SWS Item	[ECUC_Dem_00859]
Container Name	DemLinearScale
Description	This container contains the configuration (parameters) of an linear scale of the alternative Diagnosis Representation.
Configuration Parameters	

Name	DemDiagnosisRepresentationDataLowerRange [ECUC_Dem_00856]		
Description	Lower Range for this scale of the data in the alternative Diagnosis Representation.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	-INF .. INF		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDiagnosisRepresentationDataOffset [ECUC_Dem_00854]		
Description	Data offset of the alternative Diagnosis Representation for this scale.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	0 .. INF		
Default Value	0		
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDiagnosisRepresentationDataResolution [ECUC_Dem_00853]		
Description	Data resolution of the alternative Diagnosis Representation for this scale.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	0 .. INF		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDiagnosisRepresentationDataUpperRange [ECUC_Dem_00855]		
Description	Upper Range for this scale of the data in the alternative Diagnosis Representation.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	-INF .. INF		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.62 DemTextTableMapping

DemTextTableMapping

SWS Item	[ECUC_Dem_00849]
Container Name	DemTextTableMapping
Description	This container contains the configuration (parameters) of the mapping a DataPrototype typed by AutosarDataType that refer to a CompuMethods of category TEXTTABLE or SCALE_LINEAR_AND_TEXTTABLE. Each DemTextTableMapping defines a value pair which is used to map the ECU internal value (DemInternalDataValue) to the vale used in the diagnosis representation (DemDiagnosisRepresentationDataValue) and vice versa. The set of all DemTextTableMapping defines the whole mapping of an data.
Configuration Parameters	

Name	DemDiagnosisRepresentationDataValue [ECUC_Dem_00851]		
Description	The data value in the diagnosis representation.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 18446744073709551615		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemInternalDataValue [ECUC_Dem_00850]		
Description	The ECU internal data value.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 18446744073709551615		
Default Value			
Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.3 Published information

For details refer to the chapter 10.3 “Published Information” in SWS_BSWGeneral.