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

Document Status	Final
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	4.3.1

Document Change History			
Date	**Release**	**Changed by**	**Description**
2017-12-08	4.3.1	AUTOSAR Release Management	• Update and optimized interaction between Dcm and Dem • Made many functions asynchronous • Shifted constraint handling to explicit requirements instead of informative text in ECUC tables • minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation
2016-11-30	4.3.0	AUTOSAR Release Management	• Removal of context workarounds and reworked callback behaviour. Introduced monitor status and updated point in time of callback processing. • Introduced client concept for multiple access to the Dem. Optimized APIs for better runtime performance and aligned return values to allow the Dcm mapping to return values according to ISO 14229-1 [1]. • Supporting event memories for multiple diagnostic servers • Clarified thresholds and operation cycle handling • minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation

2015-07-31	4.2.2	AUTOSAR Release Management	• New APIs Dem_GetEventFreezeFrameDataEx and Dem_GetEventExtendedDataRecordEx with buffersize as parameter and corrected return value defintions. • Providing OBD FreezFrame for UDS service 0x19 0x05 • ISO 14229-1:2013[2] NRC handling for service 0x14 • Refined service interfaces for DataElements • minor corrections / clarifications / editorial changes; For details please refer to the ChangeDocumentation
2014-10-31	4.2.1	AUTOSAR Release Management	• Support of ISO 27145 (WWH-OBD / Euro VI)[3] • Update to support ISO 14229-1:2013[1] • Introduction of event dependencies • Refined DTC/Event suppression
2014-03-31	4.1.3	AUTOSAR Release Management	• Further clarification of event combination • Clarification of DTC groups • Editorial changes
2013-10-31	4.1.2	AUTOSAR Release Management	• Added API table for service interfaces • Clarification of event combination • Editorial changes • Removed chapter(s) on change documentation

2013-03-15	4.1.1	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
2011-12-22	4.0.3	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
2009-12-18	4.0.1	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

2010-02-02	3.1.4	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
2008-08-13	3.1.1	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
2007-12-21	3.0.1	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
2007-01-24	2.1.115	AUTOSAR Administration	• “Advice for users” revised • “Revision Information” added

2006-11-28	2.1	AUTOSAR Administration	• Complete rework and extension of debouncing part • Dem_ClearGroupOfDTC and Dem_ClearSingleDTC replaced by Dem_SingleDTC • Dem_GetNextFilteredDTCAndFDC, Dem_SetDTCFilterForRecords, Dem_GetSizeOfFreezeFrame, Dem_SetValueByOemId, Dem_SetEnableCondition, Xxx_DemGetFaultDetectionCounter 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
2006-05-16	2.0	AUTOSAR Administration	• Layout Adaptations

Disclaimer

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 Dem module that are not included in the [4, AUTOSAR glossary].

2.1 Acronyms

Acronym	Description
Activation Mode 1	Absence of malfunction - The MIL shall blink for one flash.
Activation Mode 2	"on-demand-MI" - The MIL shall show blink for two flashes if the OBD system would command an on-demand-MI according to the discriminatory display strategy.
Activation Mode 3	"short-MI" - The MIL shall blink for three flashes if the OBD system would command a short-MI according to the discriminatory display strategy.
Activation Mode 4	"continuous-MI" - The MIL shall remain continuously ON ("continuous-MI") if the OBD system would command a continuous-MI according to the discriminatory display strategy.
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.
Class B1 counter	Number of engine hours during which a Class B1 malfunction has been Confirmed and TestFailed.
Combined DTC	Normal DTC, but referenced by multiple events reported by several monitors (e.g. ECU Defect, consisting of different HW defects).
Continuous-MI counter	Hours run by the engine while a continuous MI is commanded.
Cumulative Continuous-MI counter	Number of engine hours during which MI has been continuously commanded to be on during its lifetime.
Debounce counter	Internal counter for counter-based debouncing algorithm(s).
DemComponent / Monitored Component	A monitored component is a part of the system, which is checked for proper operation by one or several monitorings. (see chapter 7.5)
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.
Dependent / Secondary ECUs	Dependent / Secondary (or dep. / sec.) ECUs are always related to a Master or a Primary ECU.
Directed acyclic graph	Dependency graph without circular dependencies.
Displacement	Replacing the the most insignificant event memory entry by a more significant event memory entry which needs to be stored.

Acronym	Description
DTC group	Uniquely identifies an set of DTCs. A DTC group is mapped into the range of valid DTCs. By providing a group of DTCs it is expressed that a certain operation is requested on all DTCs of that group. The DTC group definition is provided by ISO 14229-1[2] and OEM/supplier specific.
DtcGroupAllDtcs	Grouping of all configured DTCS (representation is 0xFFFFFF).
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 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 status bit 3 (ConfirmedDTC) is set.
Event memory	An event memory (e.g. Primary memory) consists of several event memory entries.
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 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.
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 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 status byte	Status byte as defined in ISO 14229-1 [1], based on event level.
Extended data record	An extended data record is a record to store specific information assigned to a fault.
Failure counter	The Failure counter represents the Trip Counter according to ISO14229-1 [2]. The Trip Counter counts the number of operation cycles (driving cycles) where a malfunction occurred. If the counter reaches the threshold (e.g., 2 driving cycles) the confirmed bit changes from 0 to 1.
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 ISO-14229-1[2].
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

Acronym	Description
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.
NvM is marked for NvM_WriteAll	The Dem has called NvM_SetRamBlockStatus() to set the according NvM Block to be written by NvM_WriteAll()
Operating cycle	An ‘operation cycle’ is the base of the event qualifying and also Dem scheduling (e.g. ignition key off-on cycles, driving cycles, etc.)
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 ECUprimary ECUs (several per vehicle) • Dependent / Secondary ECUs (several per vehicle)
P-Code	Power train code
PFC cycle	Permanent fault code - driving cycle (OBD Term)
Positive Callback from NvM	The Dem module shall use the APIs NvM_WriteBlock and NvM_GetErrorStatus of the NVRAMManager [5], if there is the necessity to store data between Dem_Init and Dem_Shutdown. Furthermore the API NvM_GetErrorStatus shall wait for positive response if writing of block completed successfully.
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
RBM cycle	OBD Term: General Nominator / Rate-based monitoring - driving cycle (OBD Term)
Readiness	The readiness refers to the tested bits TestNotCompletedSinceLastClear (bit 4) and TestNotCompleteThisOperationCycle (bit 6) of the UDS status byte.
Triggered to NvM	The Dem module shall use the API NvM_WriteBlock of the NVRAMManager [5], if there is the necessity to trigger the storage of data between Dem_Init and Dem_Shutdown. Furthermore the Dem module shall wait for positive response of NvM_WriteBlock if request has been accepted.
UDS status bit 0	testFailed bit of the UDS status byte. Indicates the result of the most recently performed test.
UDS status bit 1	testFailedThisOperationCycle bit of the UDS status byte. Indicates whether or not a diagnostic test has reported a testFailed result at any time during the current operation cycle.

Acronym	Description
UDS status bit 2	pendingDTC bit of the UDS status byte. Indicates whether or not a diagnostic test has reported a testFailed result at any time during the current or last completed operation cycle.
UDS status bit 3	confirmedDTC bit of the UDS status byte. Indicates whether a malfunction was detected enough times to warrant that the DTC is desired to be stored in long-term memory.
UDS status bit 4	testNotCompletedSinceLastClear bit of the UDS status byte. Indicates whether a DTC test has ever run and completed since the last time a call was made to ClearDiagnosticInformation.
UDS status bit 5	testFailedSinceLastClear bit of the UDS status byte. Indicates whether a DTC test has completed with a failed result since the last time a call was made to ClearDiagnosticInformation.
UDS status bit 6	testNotCompletedThisOperationCycle bit of the UDS status byte. Indicates whether a DTC test has ever run and completed during the current operation cycle.
UDS status bit 7	warningIndicatorRequested bit of the UDS status byte. Report the status of any warning indicators associated with a particular DTC.
UDS status byte	Status byte as defined in ISO 14229-1 [1], based on DTC level.

2.2 Abbreviations

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	Default Error Tracer
DID	Data Identifier
Dlt	Diagnostic Log and Trace
DTC	Diagnostic Trouble Code
DTR	Diagnostic Test Result
DYC	OBD Term: Driving Cycle (OBD Term)
ECU	Electronic Control Unit
EcuM	Electronic Control Unit Manager
FDC	Fault Detection Counter
Fim	Function Inhibition Manager
FMI	Failure Mode Indicator (SAE J1939)
FTB	Failure Type Byte
HW	Hardware
ID	Identification/Identifier
ISO	International Standardization Organization
IUMPR	In Use Monitoring Performance Ratio (OBD Term)
J1939Dcm	SAEJ1939 Diagnostic Communication Manager
MIL	Malfunction Indicator Light (SAE J1979) or Lamp (SAE J1939)
NVRAM	Non volatile RAM

Abbreviation	Description
OBD	On-Board-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
VOBD	Vehicle On-Board-Diagnostic
WUC	OBD Term: Warm up cycle (OBD Term)
WIR	Warning Indicator Request
WWH-OBD	World Wide Harmonized On-Board-Diagnostic

3 Related documentation

3.1 Input documents & related standards and norms

Bibliography

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

[2] Unified diagnostic services (UDS) – Part 1: Specification and requirements (Release 2013-03)
http://www.iso.org

[3] Road vehicles – Implementation of World-Wide Harmonized On-Board Diagnostics (WWH-OBD) communication requirements – Part 1: General information and use case definition
http://www.iso.org

[4] Glossary
AUTOSAR_TR_Glossary

[5] Specification of NVRAM Manager
AUTOSAR_SWS_NVRAMManager

[6] General Specification of Basic Software Modules
AUTOSAR_SWS_BSWGeneral

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

[8] Requirements on Function Inhibition Manager
AUTOSAR_SRS_FunctionInhibitionManager

[9] Specification of Diagnostic Log and Trace
AUTOSAR_SWS_DiagnosticLogAndTrace

[10] Specification of Diagnostic Communication Manager
AUTOSAR_SWS_DiagnosticCommunicationManager

[11] Requirements on Diagnostic
AUTOSAR_SRS_Diagnostic

[12] General Requirements on Basic Software Modules
AUTOSAR_SRS_BSWGeneral

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

[14] SAE J2012-DA Digital

[15] SAE J1939-73 Application Layer – Diagnostics

[16] Road vehicles – Interchange of digital information on electrical connections between towing and towed vehicles – Part 4: Diagnostic communication
http://www.iso.org

[17] ISO 17356-3: Road vehicles – Open interface for embedded automotive applications – Part 3: OSEK/VDX Operating System (OS)

[18] SAE J1979

[19] SAE J1979-DA Digital Annex of E/E Diagnostic Test Modes

[20] Title 13, California Code Regulations, Section 1971.1, On-Board Diagnostic System Requirements for 2013 and Subsequent Model-Year Heavy-Duty Engines (HD OBD)

[21] Title 13, California Code Regulations, Section 1968.2, Malfunction and Diagnostic 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

[22] Road vehicles – Implementation of World-Wide Harmonized On-Board Diagnostics (WWH-OBD) communication requirements – Part 3: Common message dictionary
http://www.iso.org

[23] Software Component Template
AUTOSAR_TPS_SoftwareComponentTemplate

[24] Diagnostic Extract Template
AUTOSAR_TPS_DiagnosticExtractTemplate

3.2 Related specification

AUTOSAR provides a General Specification on Basic Software modules [6, 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.

Callbacks for Event and DTCStatusChange are not deterministic in rare cases, wherefore it shall not be used for safety-relevant use-cases. Consider the general statement in ISO-14229-1[2] that it is not recommended to link the DTC status with failsafe strategies.

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 used defined memory), but the specific event memory processing is not defined in detail.

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 subsubsection 7.11.2.4).

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 [7, SWS J1939 Dcm]). The SPN Conversion Method is limited to Version 4 (CM = 0).

AUTOSAR only supports user defined memories with IDs from 0x10 to 0xFF as proposed to ISO 14229:1[2] by AUTOSAR.

Indicators are DemEventMemorySet local only. The Dem does not support configurations where indicators are used on more than one DemEventMemorySet.

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:

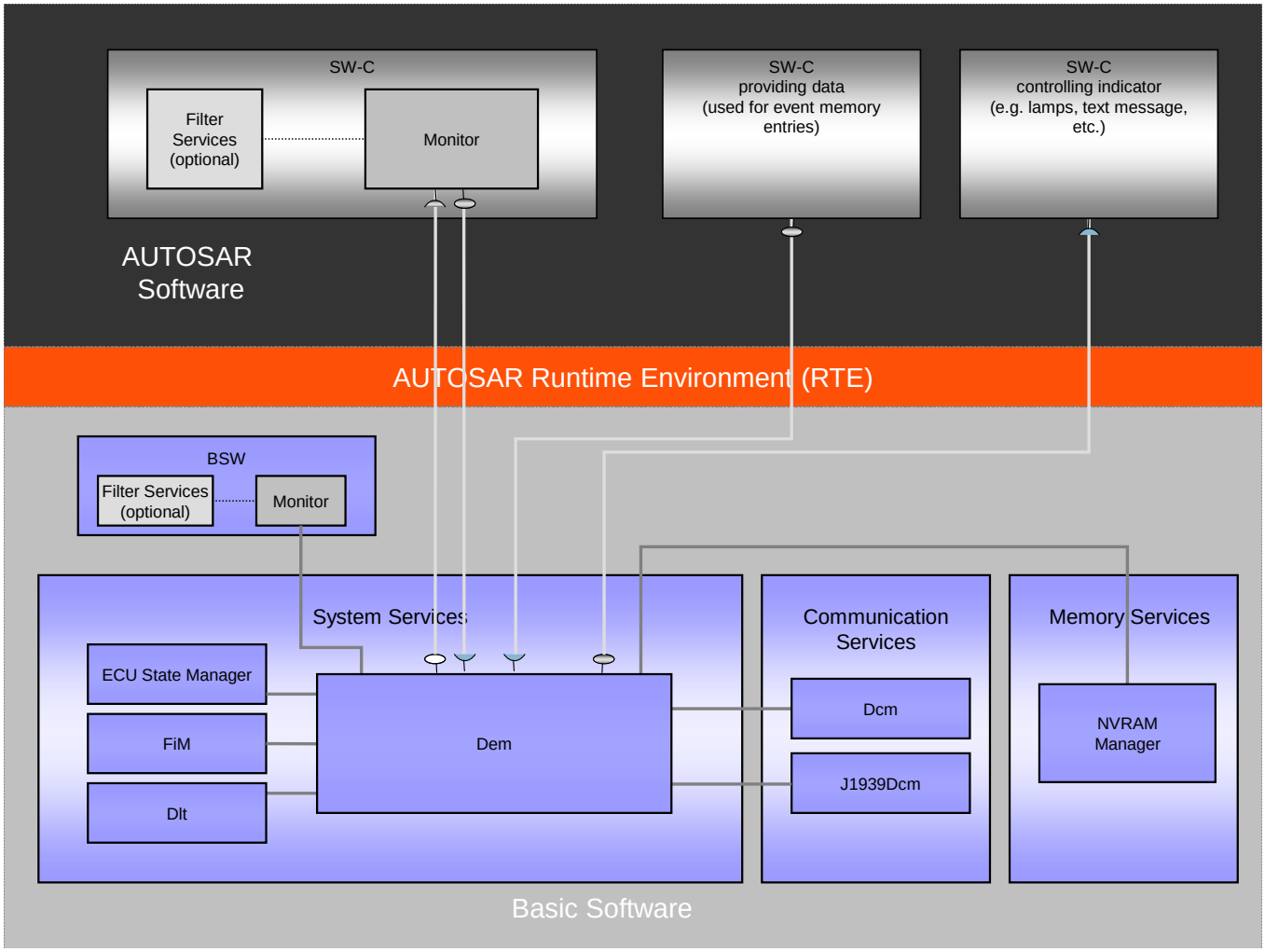


Figure 5.1: Dependencies of the Dem to other software modules

- The **Function Inhibition Manager (FiM)** (refer to [8, 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 monitor status in order to stop or release function entities according to assigned dependencies.
- The **Diagnostic Log and Trace (Dlt)** (refer to [9, 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 UDS 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 [10, 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 [7, 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 monitor status and UDS 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 [5]) 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 UDS 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 [6] (sub chapters “Link time configuration source” and “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.

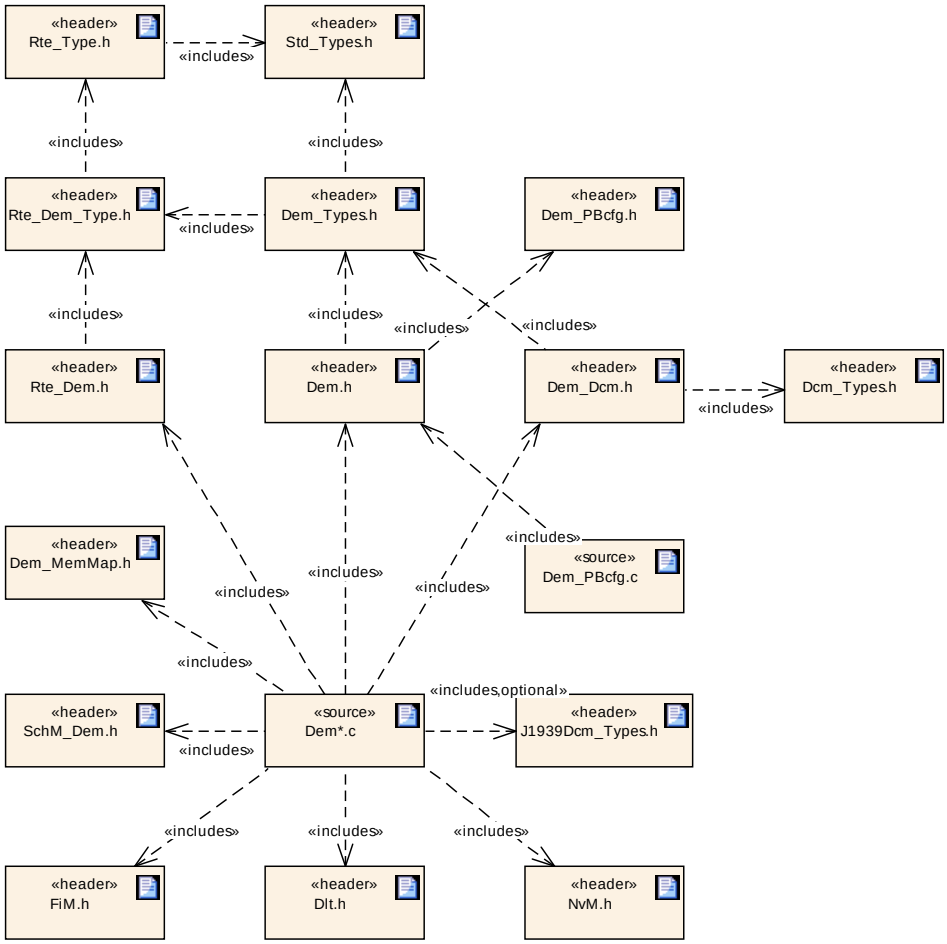


Figure 5.2: Header file structure

5.2 Integration Hints

Due to cross-module communication it is recommend to map the Dem and J1939Dcm to the same Task to have for the RTE the correct call-context configured. There might be other BSW vendor-specific solutions possible allowing a more flexible mapping (e.g. by describing this cross-module communication in the BSW internal-behavior or by a similar mechanism with the Dcm <-> Dem interaction).

6 Requirements traceability

The following table references the requirements specified in [11] as well as [12] and links to the fulfillment of these.

Requirement	Description	Satisfied by
[SRS_BSW_00005]	Modules of the μC Abstraction Layer (MCAL) may not have hard coded horizontal interfaces	[SWS_Dem_00999]
[SRS_BSW_00101]	The Basic Software Module shall be able to initialize variables and hardware in a separate initialization function	[SWS_Dem_00180] [SWS_Dem_00181] [SWS_Dem_00256] [SWS_Dem_00340]
[SRS_BSW_00161]	The AUTOSAR Basic Software shall provide a microcontroller abstraction layer which provides a standardized interface to higher software layers	[SWS_Dem_00999]
[SRS_BSW_00162]	The AUTOSAR Basic Software shall provide a hardware abstraction layer	[SWS_Dem_00999]
[SRS_BSW_00164]	The Implementation of interrupt service routines shall be done by the Operating System, complex drivers or modules	[SWS_Dem_00999]
[SRS_BSW_00168]	SW components shall be tested by a function defined in a common API in the Basis-SW	[SWS_Dem_00999]
[SRS_BSW_00170]	The AUTOSAR SW Components shall provide information about their dependency from faults, signal qualities, driver demands	[SWS_Dem_00999]
[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_00300]	All AUTOSAR Basic Software Modules shall be identified by an unambiguous name	[SWS_Dem_00512]
[SRS_BSW_00301]	All AUTOSAR Basic Software Modules shall only import the necessary information	[SWS_Dem_00176]
[SRS_BSW_00310]	API naming convention	[SWS_Dem_00256]

Requirement	Description	Satisfied by
[SRS_BSW_00327]	Error values naming convention	[SWS_Dem_00999]
[SRS_BSW_00331]	All Basic Software Modules shall strictly separate error and status information	[SWS_Dem_00999]
[SRS_BSW_00336]	Basic SW module shall be able to shutdown	[SWS_Dem_00102] [SWS_Dem_00182] [SWS_Dem_00341]
[SRS_BSW_00337]	Classification of development errors	[SWS_Dem_00368]
[SRS_BSW_00339]	Reporting of production relevant error status	[SWS_Dem_00167] [SWS_Dem_00207] [SWS_Dem_00851] [SWS_Dem_01079] [SWS_Dem_01212] [SWS_Dem_01289]
[SRS_BSW_00341]	Module documentation shall contains all needed informations	[SWS_Dem_00999]
[SRS_BSW_00347]	A Naming seperation of different instances of BSW drivers shall be in place	[SWS_Dem_00999]
[SRS_BSW_00348]	All AUTOSAR standard types and constants shall be placed and organized in a standard type header file	[SWS_Dem_00999]
[SRS_BSW_00350]	All AUTOSAR Basic Software Modules shall allow the enabling/disabling of detection and reporting of development errors.	[SWS_Dem_00999]
[SRS_BSW_00353]	All integer type definitions of target and compiler specific scope shall be placed and organized in a single type header	[SWS_Dem_00999]
[SRS_BSW_00357]	For success/failure of an API call a standard return type shall be defined	[SWS_Dem_00999]
[SRS_BSW_00359]	All AUTOSAR Basic Software Modules callback functions shall avoid return types other than void if possible	[SWS_Dem_00999]
[SRS_BSW_00360]	AUTOSAR Basic Software Modules callback functions are allowed to have parameters	[SWS_Dem_00999]
[SRS_BSW_00361]	All mappings of not standardized keywords of compiler specific scope shall be placed and organized in a compiler specific type and keyword header	[SWS_Dem_00999]

Requirement	Description	Satisfied by
[SRS_BSW_00369]	All AUTOSAR Basic Software Modules shall not return specific development error codes via the API	[SWS_Dem_01255] [SWS_Dem_01261] [SWS_Dem_01270] [SWS_Dem_01271] [SWS_Dem_01274] [SWS_Dem_01299] [SWS_Dem_01300]
[SRS_BSW_00373]	The main processing function of each AUTOSAR Basic Software Module shall be named according the defined convention	[SWS_Dem_00125]
[SRS_BSW_00374]	All Basic Software Modules shall provide a readable module vendor identification	[SWS_Dem_00999]
[SRS_BSW_00375]	Basic Software Modules shall report wake-up reasons	[SWS_Dem_00999]
[SRS_BSW_00379]	All software modules shall provide a module identifier in the header file and in the module XML description file.	[SWS_Dem_00999]
[SRS_BSW_00387]	No description	[SWS_Dem_01046]
[SRS_BSW_00402]	Each module shall provide version information	[SWS_Dem_00177]
[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_00407]	Each BSW module shall provide a function to read out the version information of a dedicated module implementation	[SWS_Dem_00177]
[SRS_BSW_00433]	Main processing functions are only allowed to be called from task bodies provided by the BSW Scheduler	[SWS_Dem_00999]
[SRS_BSW_00447]	Standardizing Include file structure of BSW Modules Implementing Autosar Service	[SWS_Dem_00151]
[SRS_BSW_00457]	- Callback functions of Application software components shall be invoked by the Basis SW	[SWS_Dem_00003] [SWS_Dem_00284] [SWS_Dem_00613] [SWS_Dem_00986] [SWS_Dem_00987] [SWS_Dem_01005]
[SRS_Com_02041]	The AUTOSAR module shall handle complex data types as a consistent set of data	[SWS_Dem_00918] [SWS_Dem_00919] [SWS_Dem_00920]
[SRS_Diag_04000]	No description	[SWS_Dem_00933]

Requirement	Description	Satisfied by
[SRS_Diag_04001]	No description	[SWS_Dem_00299] [SWS_Dem_00359] [SWS_Dem_00645] [SWS_Dem_00716] [SWS_Dem_00745] [SWS_Dem_00751] [SWS_Dem_00753] [SWS_Dem_00761] [SWS_Dem_00763] [SWS_Dem_00965] [SWS_Dem_01076] [SWS_Dem_01077] [SWS_Dem_01092] [SWS_Dem_01220] [SWS_Dem_01248]
[SRS_Diag_04002]	The Diagnostic event (fault) management shall be established as Basic SW Module	[SWS_Dem_00126]
[SRS_Diag_04005]	Manage Security Access level handling	[SWS_Dem_00999]
[SRS_Diag_04006]	Manage session handling	[SWS_Dem_00999]
[SRS_Diag_04007]	Provide a diagnostic service handling for the applications involved in diagnostic functionality	[SWS_Dem_00999]
[SRS_Diag_04010]	No description	[SWS_Dem_00060] [SWS_Dem_00079] [SWS_Dem_00172] [SWS_Dem_00204] [SWS_Dem_00210] [SWS_Dem_00213] [SWS_Dem_00216] [SWS_Dem_00228] [SWS_Dem_00230] [SWS_Dem_00264] [SWS_Dem_00277] [SWS_Dem_00281] [SWS_Dem_00287] [SWS_Dem_00415] [SWS_Dem_00417] [SWS_Dem_00439] [SWS_Dem_00513] [SWS_Dem_00584] [SWS_Dem_00599] [SWS_Dem_00624] [SWS_Dem_00645] [SWS_Dem_00657] [SWS_Dem_00671] [SWS_Dem_00828] [SWS_Dem_00947] [SWS_Dem_01253] [SWS_Dem_01254] [SWS_Dem_01256] [SWS_Dem_01257] [SWS_Dem_01258] [SWS_Dem_01269] [SWS_Dem_01273] [SWS_Dem_01290] [SWS_Dem_01296] [SWS_Dem_01304] [SWS_Dem_01305] [SWS_Dem_01306]
[SRS_Diag_04015]	Timing handling according to ISO15765-3	[SWS_Dem_00999]
[SRS_Diag_04016]	Support ”Busy handling” by sending a negative response 0x78	[SWS_Dem_00999]
[SRS_Diag_04019]	Confirm transmitting if complete to continue processing	[SWS_Dem_00999]
[SRS_Diag_04020]	Suppress responses to diagnostic tool requests	[SWS_Dem_00999]
[SRS_Diag_04021]	Handling of different diagnostic sessions in parallel	[SWS_Dem_00999]

Requirement	Description	Satisfied by
[SRS_Diag_04024]	Access and handle specific data elements and data element groups if requested by an external scan tool	[SWS_Dem_00479] [SWS_Dem_01194]
[SRS_Diag_04025]	No description	[SWS_Dem_00622]
[SRS_Diag_04031]	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_01189]
[SRS_Diag_04032]	Different diagnostic addresses shall be supported by multiple (physical) channels	[SWS_Dem_00999]
[SRS_Diag_04033]	Support the upload/download services for reading/writing data in an ECU in an extended and manufacturer specific diagnostic session	[SWS_Dem_00999]
[SRS_Diag_04057]	Classification of events for series production, OBD and expert usage	[SWS_Dem_00057] [SWS_Dem_00370] [SWS_Dem_00518] [SWS_Dem_00649] [SWS_Dem_00906] [SWS_Dem_01066]
[SRS_Diag_04058]	Ability to access different event memories	[SWS_Dem_00171]
[SRS_Diag_04059]	Configuration of timing parameters	[SWS_Dem_00999]
[SRS_Diag_04061]	No description	[SWS_Dem_00153] [SWS_Dem_00598] [SWS_Dem_00616] [SWS_Dem_00617] [SWS_Dem_00621] [SWS_Dem_00745] [SWS_Dem_01092]
[SRS_Diag_04063]	Process a dedicated event identifier for each monitoring path to support an autonomous handling of different events/faults	[SWS_Dem_00153] [SWS_Dem_00154] [SWS_Dem_00942] [SWS_Dem_01034] [SWS_Dem_01037]
[SRS_Diag_04064]	Buffers of scalable sizes for the storage of the events, status information and environmental data	[SWS_Dem_00999]
[SRS_Diag_04065]	No description	[SWS_Dem_00570] [SWS_Dem_00571] [SWS_Dem_00572] [SWS_Dem_00573] [SWS_Dem_00661] [SWS_Dem_00664] [SWS_Dem_01057] [SWS_Dem_01059] [SWS_Dem_01061] [SWS_Dem_01075] [SWS_Dem_01185] [SWS_Dem_01198]

Requirement	Description	Satisfied by
[SRS_Diag_04066]	No description	[SWS_Dem_00010] [SWS_Dem_00162] [SWS_Dem_00212] [SWS_Dem_00236] [SWS_Dem_00238] [SWS_Dem_00239] [SWS_Dem_00240] [SWS_Dem_00548] [SWS_Dem_00648] [SWS_Dem_00934] [SWS_Dem_01199] [SWS_Dem_01207]
[SRS_Diag_04067]	Provide the diagnostic status information according to ISO 14229-1	[SWS_Dem_00006] [SWS_Dem_00011] [SWS_Dem_00016] [SWS_Dem_00051] [SWS_Dem_00059] [SWS_Dem_00060] [SWS_Dem_00187] [SWS_Dem_00212] [SWS_Dem_00213] [SWS_Dem_00330] [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_00409] [SWS_Dem_00421] [SWS_Dem_00431] [SWS_Dem_00523] [SWS_Dem_00524] [SWS_Dem_00525] [SWS_Dem_00529] [SWS_Dem_00539] [SWS_Dem_00566] [SWS_Dem_00638] [SWS_Dem_00823] [SWS_Dem_00936] [SWS_Dem_01186] [SWS_Dem_01276] [SWS_Dem_01277] [SWS_Dem_01278] [SWS_Dem_01280] [SWS_Dem_01281] [SWS_Dem_01282] [SWS_Dem_01283] [SWS_Dem_01284] [SWS_Dem_01285] [SWS_Dem_01286] [SWS_Dem_01287] [SWS_Dem_91008]
[SRS_Diag_04068]	Event specific debounce algorithms	[SWS_Dem_00019] [SWS_Dem_00204] [SWS_Dem_00264] [SWS_Dem_00343] [SWS_Dem_00344] [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_00431] [SWS_Dem_00432] [SWS_Dem_00433] [SWS_Dem_00434] [SWS_Dem_00435] [SWS_Dem_00437] [SWS_Dem_00438] [SWS_Dem_00439] [SWS_Dem_00526] [SWS_Dem_00527] [SWS_Dem_00643] [SWS_Dem_00684] [SWS_Dem_00685] [SWS_Dem_00730] [SWS_Dem_00772] [SWS_Dem_00774] [SWS_Dem_00778] [SWS_Dem_00779] [SWS_Dem_00818] [SWS_Dem_00844] [SWS_Dem_00985] [SWS_Dem_01213] [SWS_Dem_01279]

Requirement	Description	Satisfied by
[SRS_Diag_04069]	No description	[SWS_Dem_00046] [SWS_Dem_00501] [SWS_Dem_00503] [SWS_Dem_00535] [SWS_Dem_00567] [SWS_Dem_00701] [SWS_Dem_00865] [SWS_Dem_00886] [SWS_Dem_00896] [SWS_Dem_00967] [SWS_Dem_01233] [SWS_Dem_01303]
[SRS_Diag_04071]	Process events according to their defined importance like priority and/or severity	[SWS_Dem_00232] [SWS_Dem_00382] [SWS_Dem_00383] [SWS_Dem_00692] [SWS_Dem_01291] [SWS_Dem_01292] [SWS_Dem_01293] [SWS_Dem_01294]
[SRS_Diag_04072]	No description	[SWS_Dem_00218]
[SRS_Diag_04073]	No description	[SWS_Dem_00024] [SWS_Dem_00163] [SWS_Dem_00440] [SWS_Dem_00441] [SWS_Dem_00442] [SWS_Dem_00536] [SWS_Dem_00539] [SWS_Dem_00540] [SWS_Dem_00541] [SWS_Dem_00542] [SWS_Dem_00672] [SWS_Dem_01050] [SWS_Dem_01051] [SWS_Dem_01052] [SWS_Dem_01053] [SWS_Dem_01295]
[SRS_Diag_04074]	No description	[SWS_Dem_00039] [SWS_Dem_00040] [SWS_Dem_00050] [SWS_Dem_00071] [SWS_Dem_00075] [SWS_Dem_00076] [SWS_Dem_00188] [SWS_Dem_00189] [SWS_Dem_00191] [SWS_Dem_00193] [SWS_Dem_00225] [SWS_Dem_00234] [SWS_Dem_00271] [SWS_Dem_00282] [SWS_Dem_00464] [SWS_Dem_00465] [SWS_Dem_00564] [SWS_Dem_00576] [SWS_Dem_00582] [SWS_Dem_00585] [SWS_Dem_00595] [SWS_Dem_00630] [SWS_Dem_00631] [SWS_Dem_00775] [SWS_Dem_00796] [SWS_Dem_00807] [SWS_Dem_00918] [SWS_Dem_00919] [SWS_Dem_00920] [SWS_Dem_00969] [SWS_Dem_00989] [SWS_Dem_00995] [SWS_Dem_00996] [SWS_Dem_00997] [SWS_Dem_01062] [SWS_Dem_01268] [SWS_Dem_01272]
[SRS_Diag_04075]	No description	[SWS_Dem_00198]
[SRS_Diag_04076]	No description	[SWS_Dem_00194] [SWS_Dem_00601] [SWS_Dem_00929] [SWS_Dem_01021] [SWS_Dem_01026]
[SRS_Diag_04077]	Uses standard mechanisms provided by persistency modules	[SWS_Dem_00164] [SWS_Dem_00329] [SWS_Dem_00551] [SWS_Dem_01237] [SWS_Dem_01238]

Requirement	Description	Satisfied by
[SRS_Diag_04082]	No description	[SWS_Dem_00315] [SWS_Dem_00321] [SWS_Dem_00323] [SWS_Dem_00325] [SWS_Dem_00610] [SWS_Dem_00612] [SWS_Dem_00627] [SWS_Dem_00735] [SWS_Dem_00738] [SWS_Dem_00742] [SWS_Dem_00743] [SWS_Dem_00746] [SWS_Dem_00766] [SWS_Dem_00769] [SWS_Dem_00933] [SWS_Dem_01073] [SWS_Dem_01187]
[SRS_Diag_04085]	The Default Error Tracer shall provide an interface to receive error reports	[SWS_Dem_00463]
[SRS_Diag_04086]	Report errors shall contain a dedicated set of information	[SWS_Dem_00999]
[SRS_Diag_04087]	The Default Error Tracer shall provide a development error report reception service	[SWS_Dem_00999]
[SRS_Diag_04089]	The DET module shall support fan-out of received error reports	[SWS_Dem_00999]
[SRS_Diag_04090]	A configurable list of error report receivers shall be provided	[SWS_Dem_00999]
[SRS_Diag_04091]	Notification about valid freeze frame data to applications	[SWS_Dem_00999]
[SRS_Diag_04093]	Memory overflow indication	[SWS_Dem_00397] [SWS_Dem_00398] [SWS_Dem_00399] [SWS_Dem_00559] [SWS_Dem_01023]
[SRS_Diag_04095]	No description	[SWS_Dem_00201] [SWS_Dem_00202] [SWS_Dem_00233] [SWS_Dem_00243] [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] [SWS_Dem_00733] [SWS_Dem_01113] [SWS_Dem_01210]
[SRS_Diag_04097]	Decentralized and modular diagnostic configuration in applications	[SWS_Dem_00999]
[SRS_Diag_04098]	Interact with standard bootloader	[SWS_Dem_00999]
[SRS_Diag_04099]	No description	[SWS_Dem_00636] [SWS_Dem_00637]
[SRS_Diag_04100]	Interface for logging and tracing	[SWS_Dem_00999]
[SRS_Diag_04101]	The DET module shall forward its trace events to the DLT	[SWS_Dem_00999]
[SRS_Diag_04102]	No description	[SWS_Dem_00935]

Requirement	Description	Satisfied by
[SRS_Diag_04105]	Event memory management	[SWS_Dem_00580] [SWS_Dem_00607] [SWS_Dem_00683] [SWS_Dem_00780] [SWS_Dem_00781] [SWS_Dem_00783] [SWS_Dem_00784] [SWS_Dem_00785] [SWS_Dem_00786] [SWS_Dem_00922] [SWS_Dem_00923] [SWS_Dem_01000]
[SRS_Diag_04107]	Provide defensive behavior	[SWS_Dem_00339]
[SRS_Diag_04109]	Provide an interface to retrieve the number of event memory entries	[SWS_Dem_00651] [SWS_Dem_00652]
[SRS_Diag_04110]	SAE J1939 lamp status	[SWS_Dem_00546] [SWS_Dem_00858] [SWS_Dem_00859] [SWS_Dem_00860] [SWS_Dem_00861] [SWS_Dem_00862] [SWS_Dem_00863] [SWS_Dem_00866] [SWS_Dem_00867] [SWS_Dem_00868] [SWS_Dem_00869] [SWS_Dem_00870] [SWS_Dem_00872] [SWS_Dem_00873] [SWS_Dem_00883] [SWS_Dem_00884] [SWS_Dem_00885] [SWS_Dem_00887] [SWS_Dem_00888] [SWS_Dem_00889] [SWS_Dem_00891] [SWS_Dem_00892] [SWS_Dem_00893] [SWS_Dem_00894] [SWS_Dem_00895] [SWS_Dem_00897] [SWS_Dem_00898] [SWS_Dem_00948] [SWS_Dem_00974]
[SRS_Diag_04111]	SAE J1939 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 J1939	[SWS_Dem_00645] [SWS_Dem_00845] [SWS_Dem_00855] [SWS_Dem_00856] [SWS_Dem_00864] [SWS_Dem_00874] [SWS_Dem_00875] [SWS_Dem_00910] [SWS_Dem_00911] [SWS_Dem_00912] [SWS_Dem_00933] [SWS_Dem_00945] [SWS_Dem_00946] [SWS_Dem_00970] [SWS_Dem_00971] [SWS_Dem_00976] [SWS_Dem_00977] [SWS_Dem_00978] [SWS_Dem_00979] [SWS_Dem_00981]
[SRS_Diag_04113]	Support a set of SAE J1939 DM-messages	[SWS_Dem_00770] [SWS_Dem_00880] [SWS_Dem_00881] [SWS_Dem_00882] [SWS_Dem_00908] [SWS_Dem_00909] [SWS_Dem_00913] [SWS_Dem_00921] [SWS_Dem_00949] [SWS_Dem_00950] [SWS_Dem_00951] [SWS_Dem_00982] [SWS_Dem_00983] [SWS_Dem_01180]
[SRS_Diag_04115]	The optional parameter DTCSettingControlOption Record as part of UDS service ControlDTCSetting shall be limited to GroupOf DTC	[SWS_Dem_00080] [SWS_Dem_00626]

Requirement	Description	Satisfied by
[SRS_Diag_04117]	Configurable behavior for DTC deletion	[SWS_Dem_00343] [SWS_Dem_00620] [SWS_Dem_00670] [SWS_Dem_00679] [SWS_Dem_00680] [SWS_Dem_00879]
[SRS_Diag_04118]	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_01186]
[SRS_Diag_04119]	Handle the execution of diagnostic services according to the assigned diagnostic session	[SWS_Dem_00999]
[SRS_Diag_04120]	Support a predefined AddressAndLengthFormat Identifier	[SWS_Dem_00999]
[SRS_Diag_04121]	Provide the handling of service DynamicallyDefine DataIdentifier according to ISO 14229-1	[SWS_Dem_00999]
[SRS_Diag_04122]	No description	[SWS_Dem_00665] [SWS_Dem_00666]
[SRS_Diag_04124]	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]	Event debounce counter shall be configurable	[SWS_Dem_00625] [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] [SWS_Dem_01004]
[SRS_Diag_04126]	Configurable suppression of events	[SWS_Dem_00687] [SWS_Dem_00915] [SWS_Dem_01080] [SWS_Dem_01100] [SWS_Dem_01101] [SWS_Dem_01102] [SWS_Dem_01103] [SWS_Dem_01104] [SWS_Dem_01105] [SWS_Dem_01106] [SWS_Dem_01107] [SWS_Dem_01108] [SWS_Dem_01109] [SWS_Dem_01110] [SWS_Dem_01111] [SWS_Dem_01302]

Requirement	Description	Satisfied by
[SRS_Diag_04127]	Configurable record numbers and trigger options for DTCSnapshot Records and DTCExtended DataRecords	[SWS_Dem_00002] [SWS_Dem_00261] [SWS_Dem_00334] [SWS_Dem_00337] [SWS_Dem_00461] [SWS_Dem_00468] [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] [SWS_Dem_01068] [SWS_Dem_01069] [SWS_Dem_01070] [SWS_Dem_01071] [SWS_Dem_01081] [SWS_Dem_01082] [SWS_Dem_01083]
[SRS_Diag_04128]	No description	[SWS_Dem_00606] [SWS_Dem_00831] [SWS_Dem_00832] [SWS_Dem_00833] [SWS_Dem_00834] [SWS_Dem_00836] [SWS_Dem_00838]
[SRS_Diag_04129]	Provide OBD-specific configuration capabilities	[SWS_Dem_00752] [SWS_Dem_00932] [SWS_Dem_01220] [SWS_Dem_01221] [SWS_Dem_01222] [SWS_Dem_01223] [SWS_Dem_01224] [SWS_Dem_01225] [SWS_Dem_01234] [SWS_Dem_01235]
[SRS_Diag_04131]	Consistent event management mechanisms	[SWS_Dem_01063] [SWS_Dem_01064]
[SRS_Diag_04133]	Aging for event memory entries	[SWS_Dem_00019] [SWS_Dem_00442] [SWS_Dem_00489] [SWS_Dem_00490] [SWS_Dem_00492] [SWS_Dem_00493] [SWS_Dem_00498] [SWS_Dem_00698] [SWS_Dem_00824] [SWS_Dem_00985] [SWS_Dem_01054] [SWS_Dem_01185] [SWS_Dem_01214] [SWS_Dem_01215] [SWS_Dem_01219] [SWS_Dem_01221] [SWS_Dem_01223]
[SRS_Diag_04135]	Support UDS service $38 (RequestFileTransfer)	[SWS_Dem_00999]
[SRS_Diag_04136]	Configurable ”confirmed” threshold	[SWS_Dem_00999]
[SRS_Diag_04137]	Definition of replacement failure	[SWS_Dem_01085] [SWS_Dem_01086] [SWS_Dem_01087] [SWS_Dem_01088] [SWS_Dem_01089] [SWS_Dem_01090] [SWS_Dem_01091]
[SRS_Diag_04139]	Support subfunction 0x42 of UDS service 0x19	[SWS_Dem_00999]

Requirement	Description	Satisfied by
[SRS_Diag_04141]	No description	[SWS_Dem_00314] [SWS_Dem_00351] [SWS_Dem_00748] [SWS_Dem_01137] [SWS_Dem_01138] [SWS_Dem_01139] [SWS_Dem_01140] [SWS_Dem_01141] [SWS_Dem_01142] [SWS_Dem_01143] [SWS_Dem_01144] [SWS_Dem_01145] [SWS_Dem_01146] [SWS_Dem_01147] [SWS_Dem_01148] [SWS_Dem_01149] [SWS_Dem_01150] [SWS_Dem_01151] [SWS_Dem_01152] [SWS_Dem_01153] [SWS_Dem_01154] [SWS_Dem_01155] [SWS_Dem_01156] [SWS_Dem_01157] [SWS_Dem_01158] [SWS_Dem_01159] [SWS_Dem_01160] [SWS_Dem_01161] [SWS_Dem_01162] [SWS_Dem_01163] [SWS_Dem_01164] [SWS_Dem_01165] [SWS_Dem_01166] [SWS_Dem_01168] [SWS_Dem_01169] [SWS_Dem_01170] [SWS_Dem_01171] [SWS_Dem_01172] [SWS_Dem_01173] [SWS_Dem_01174] [SWS_Dem_01175] [SWS_Dem_01176] [SWS_Dem_01177] [SWS_Dem_01178] [SWS_Dem_01179] [SWS_Dem_01181] [SWS_Dem_01182] [SWS_Dem_01183]
[SRS_Diag_04142]	No description	[SWS_Dem_01007] [SWS_Dem_01017] [SWS_Dem_01032] [SWS_Dem_01114] [SWS_Dem_01115] [SWS_Dem_01116] [SWS_Dem_01117] [SWS_Dem_01118] [SWS_Dem_01119] [SWS_Dem_01120] [SWS_Dem_01121] [SWS_Dem_01122] [SWS_Dem_01123] [SWS_Dem_01124] [SWS_Dem_01125] [SWS_Dem_01126] [SWS_Dem_01127] [SWS_Dem_01128] [SWS_Dem_01129] [SWS_Dem_01130] [SWS_Dem_01131] [SWS_Dem_01132] [SWS_Dem_01133] [SWS_Dem_01134] [SWS_Dem_01135] [SWS_Dem_01136] [SWS_Dem_01195] [SWS_Dem_01197] [SWS_Dem_01211] [SWS_Dem_01218] [SWS_Dem_01226] [SWS_Dem_01227] [SWS_Dem_01228] [SWS_Dem_01229] [SWS_Dem_01230] [SWS_Dem_01231] [SWS_Dem_01232] [SWS_Dem_01239]
[SRS_Diag_04143]	The Default Error Tracer shall provide an interface to receive runtime error reports	[SWS_Dem_00999]
[SRS_Diag_04144]	The Default Error Tracer shall provide an interface to receive transient fault reports	[SWS_Dem_00999]

Requirement	Description	Satisfied by
[SRS_Diag_04145]	The Default Error Tracer shall forward received runtime error reports to configured integrator code	[SWS_Dem_00999]
[SRS_Diag_04146]	The Default Error Tracer shall forward received transient fault reports to configured integrator code	[SWS_Dem_00999]
[SRS_Diag_04148]	Provide capabilities to inform applications about diagnostic data changes	[SWS_Dem_00259]
[SRS_Diag_04150]	Support the primary fault memory defined by ISO 14229-1	[SWS_Dem_00242] [SWS_Dem_00243] [SWS_Dem_01202] [SWS_Dem_01203] [SWS_Dem_01206]
[SRS_Diag_04151]	Event status handling	[SWS_Dem_01208] [SWS_Dem_01209]
[SRS_Diag_04154]	No description	[SWS_Dem_01307] [SWS_Dem_91025]
[SRS_Diag_04155]	Notify applications and BSW modules about updates of event related data	[SWS_Dem_00475] [SWS_Dem_00618] [SWS_Dem_00619] [SWS_Dem_01003]
[SRS_Diag_04156]	Support DTCFunctional Unit	[SWS_Dem_00594]
[SRS_Diag_04158]	No description	[SWS_Dem_00243]
[SRS_Diag_04160]	ResponseOnEvent according to ISO 14229-1	[SWS_Dem_00562]
[SRS_Diag_04161]	Provide support for the ASMIP algorithm	[SWS_Dem_01242] [SWS_Dem_01243] [SWS_Dem_01244] [SWS_Dem_01245] [SWS_Dem_01246]
[SRS_Diag_04162]	Parallel fault memory access	[SWS_Dem_01251] [SWS_Dem_01252] [SWS_Dem_01263]
[SRS_Diag_04164]	Independent event memories for multiple diagnostic server instances (virtual ECUs)	[SWS_Dem_01247]
[SRS_Diag_04165]	Triggering of multiple events upon a master event is reported	[SWS_Dem_01250] [SWS_Dem_CONSTR_6115]
[SRS_Diag_04167]	Conversation preemption/abortion	[SWS_Dem_01042]
[SRS_Diag_04178]	Support operation cycles according to ISO 14229-1	[SWS_Dem_00019] [SWS_Dem_00338] [SWS_Dem_00481] [SWS_Dem_00482] [SWS_Dem_00483] [SWS_Dem_00484] [SWS_Dem_00577] [SWS_Dem_00673] [SWS_Dem_00693] [SWS_Dem_00698] [SWS_Dem_00773] [SWS_Dem_00826] [SWS_Dem_00853] [SWS_Dem_00854] [SWS_Dem_00968] [SWS_Dem_00985] [SWS_Dem_01078]
[SRS_Diag_04179]	Provide interfaces for diagnostic monitors.	[SWS_Dem_00331] [SWS_Dem_00756] [SWS_Dem_00759]
[SRS_Diag_04181]	No description	[SWS_Dem_00600] [SWS_Dem_00608] [SWS_Dem_00765] [SWS_Dem_00850]

Requirement	Description	Satisfied by
[SRS_Diag_04185]	Notify applications about the clearing of an event	[SWS_Dem_01240] [SWS_Dem_01241]
[SRS_Diag_04189]	Support a fine grained configuration for Snapshot Records and Extended DataRecords	[SWS_Dem_00469] [SWS_Dem_00779] [SWS_Dem_00821] [SWS_Dem_00995] [SWS_Dem_01216]
[SRS_Diag_04190]	Usage of internal data elements in Snapshot Records and Extended DataRecords	[SWS_Dem_00469] [SWS_Dem_00470] [SWS_Dem_00471] [SWS_Dem_00472] [SWS_Dem_00473] [SWS_Dem_00592] [SWS_Dem_00643] [SWS_Dem_00673] [SWS_Dem_00775] [SWS_Dem_00779] [SWS_Dem_00817] [SWS_Dem_00818] [SWS_Dem_00819] [SWS_Dem_00820] [SWS_Dem_00821] [SWS_Dem_00822] [SWS_Dem_00984] [SWS_Dem_01043] [SWS_Dem_01044] [SWS_Dem_01045] [SWS_Dem_01084]
[SRS_Diag_04191]	No description	[SWS_Dem_00515] [SWS_Dem_00516] [SWS_Dem_00667] [SWS_Dem_00668] [SWS_Dem_00669] [SWS_Dem_01295]
[SRS_Diag_04194]	ClearDTC shall be accessible for applications	[SWS_Dem_00515] [SWS_Dem_00516] [SWS_Dem_00570] [SWS_Dem_00571] [SWS_Dem_00572] [SWS_Dem_00573] [SWS_Dem_00661] [SWS_Dem_00664] [SWS_Dem_00667] [SWS_Dem_00668] [SWS_Dem_00669] [SWS_Dem_01295]
[SRS_Diag_04195]	Chronological reporting order of the DTCs located in the configured event memory	[SWS_Dem_00161] [SWS_Dem_00219] [SWS_Dem_00221] [SWS_Dem_00410] [SWS_Dem_00411] [SWS_Dem_00412] [SWS_Dem_00412] [SWS_Dem_00477] [SWS_Dem_00695] [SWS_Dem_00696] [SWS_Dem_00787]
[SRS_Diag_04201]	Support a configuration to assign specific events to a customer specific DTC	[SWS_Dem_00231] [SWS_Dem_00269]
[SRS_Diag_04202]	Report DTCs getting active to the error logging module/system	[SWS_Dem_00633] [SWS_Dem_00634] [SWS_Dem_00635] [SWS_Dem_00992] [SWS_Dem_00993] [SWS_Dem_01297] [SWS_Dem_01298]
[SRS_Diag_04204]	Provide the current status of each warning indicator.	[SWS_Dem_00074] [SWS_Dem_00650]
[SRS_Diag_04205]	Support of Snapshot Records	[SWS_Dem_00208] [SWS_Dem_01190]
[SRS_Diag_04213]	Support the mirror fault memory defined by ISO 14229-1	[SWS_Dem_01205] [SWS_Dem_01206]
[SRS_Diag_04214]	Support the user defined fault memories defined by ISO 14229-1	[SWS_Dem_01202] [SWS_Dem_01203] [SWS_Dem_01217]

Requirement	Description	Satisfied by
[SRS_Mem_08549]	The NVRAM manager shall provide functionality to automatically initialize RAM data blocks after a software update	[SWS_Dem_00578]
[SWS_BSW_00050]	Check parameters passed to Initialization functions	[SWS_Dem_00173]
[SWS_BSW_00212]	NULL pointer checking	[SWS_Dem_00173]

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.

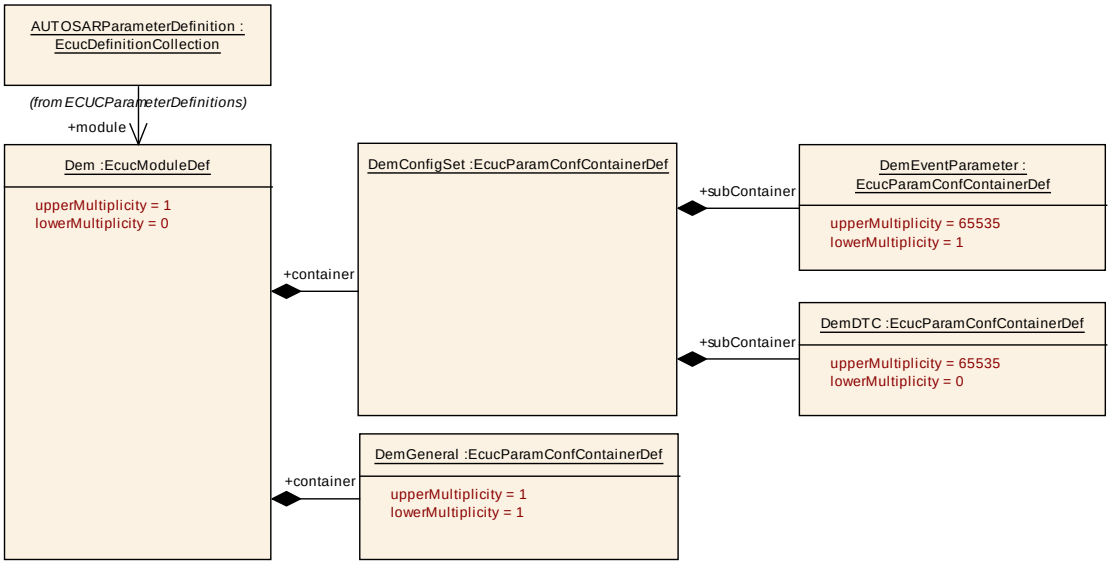


Figure 7.1: Top-level view of Dem configuration

7.1 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 shall initialize the internal states of the Dem module necessary to process events and reset debounce counters reported by SW-C or BSW modules by using Dem_SetEventStatus and Dem_ResetEventDebounceStatus. ⌋*(SRS_BSW_00101)*

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_SetEventStatus (refer to [SWS_Dem_00854], [SWS_Dem_00851], [SWS_Dem_00167]).

The function Dem_Init (refer also to [SWS_Dem_00340]) 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.

7.2 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 to trigger the initalization of a diagnostic monitor (refer also to chapter 7.7.3 and [SWS_Dem_00573]). ⌋*(SRS_BSW_00457)*

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 is used to specify the related port or c-callback per event.

[SWS_Dem_00679] ⌈ The API Dem_SetOperationCycleState 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] ⌈ The Dem_ClearDTC API 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 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_EnableDTCSetting 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)*

[SWS_Dem_01113] ⌈ The API Dem_SetStorageCondition shall trigger the callback function InitMonitorForEvent of the related event(s) in case a storage condition of the event(s) is changed to fulfilled and thus all storage conditions of the event(s) are fulfilled. Furthermore the callback shall only be called, if the event did

report FAILED or PASSED while the storage condition was disabled. The `InitMonitorReason` parameter shall be set to `DEM_INIT_MONITOR_REENABLED`. ⌋ *(SRS_Diag_04095)*

The following figures show two examples:

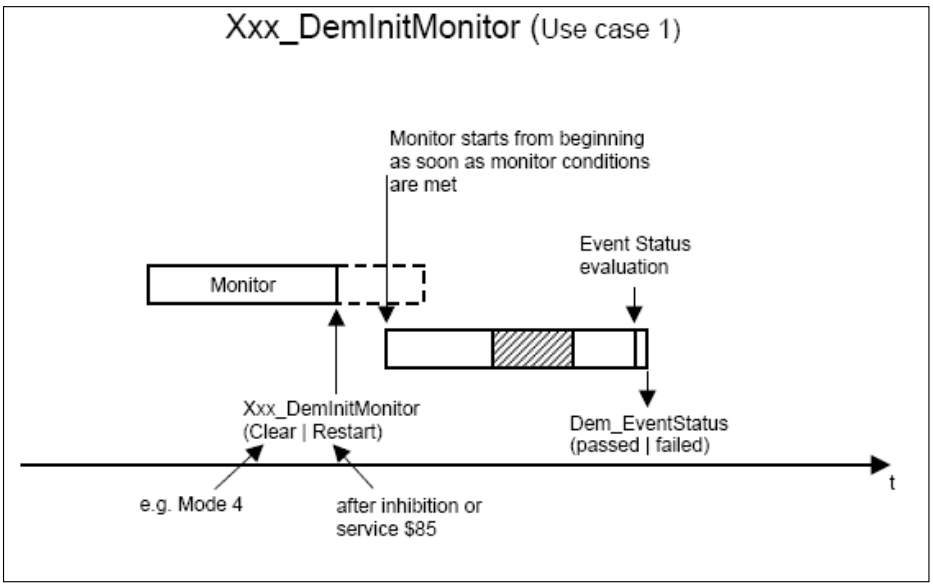


Figure 7.2: Use-case of the interface InitMonitorForEvent for a specific event

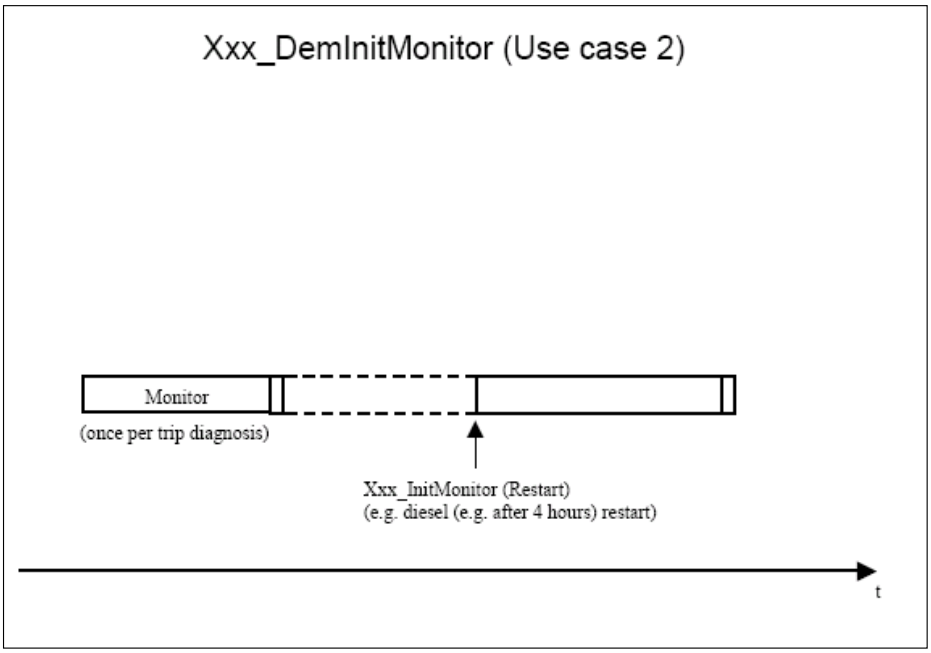


Figure 7.3: Use-case of the interface InitMonitorForEvent for a specific event

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

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[13] request).

[SWS_Dem_01046] ⌈ In case multiple triggers for `InitMonitorForEvent` occur simultaneously, the InitMforE shall only be called once. The reason shall be selected by following priority: `DEM_INIT_MONITOR_CLEAR` (highest priority), `DEM_INIT_MONITOR_RESTART`, `DEM_INIT_MONITOR_REENABLED` (lowest priority) (refer to `Dem_InitMonitorReasonType`). ⌋*(SRS_BSW_00387)*

7.3 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 7.3.5). 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 uses an internal monitor status to store the status of reported events. To the `Dcm` the UDS status is 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.

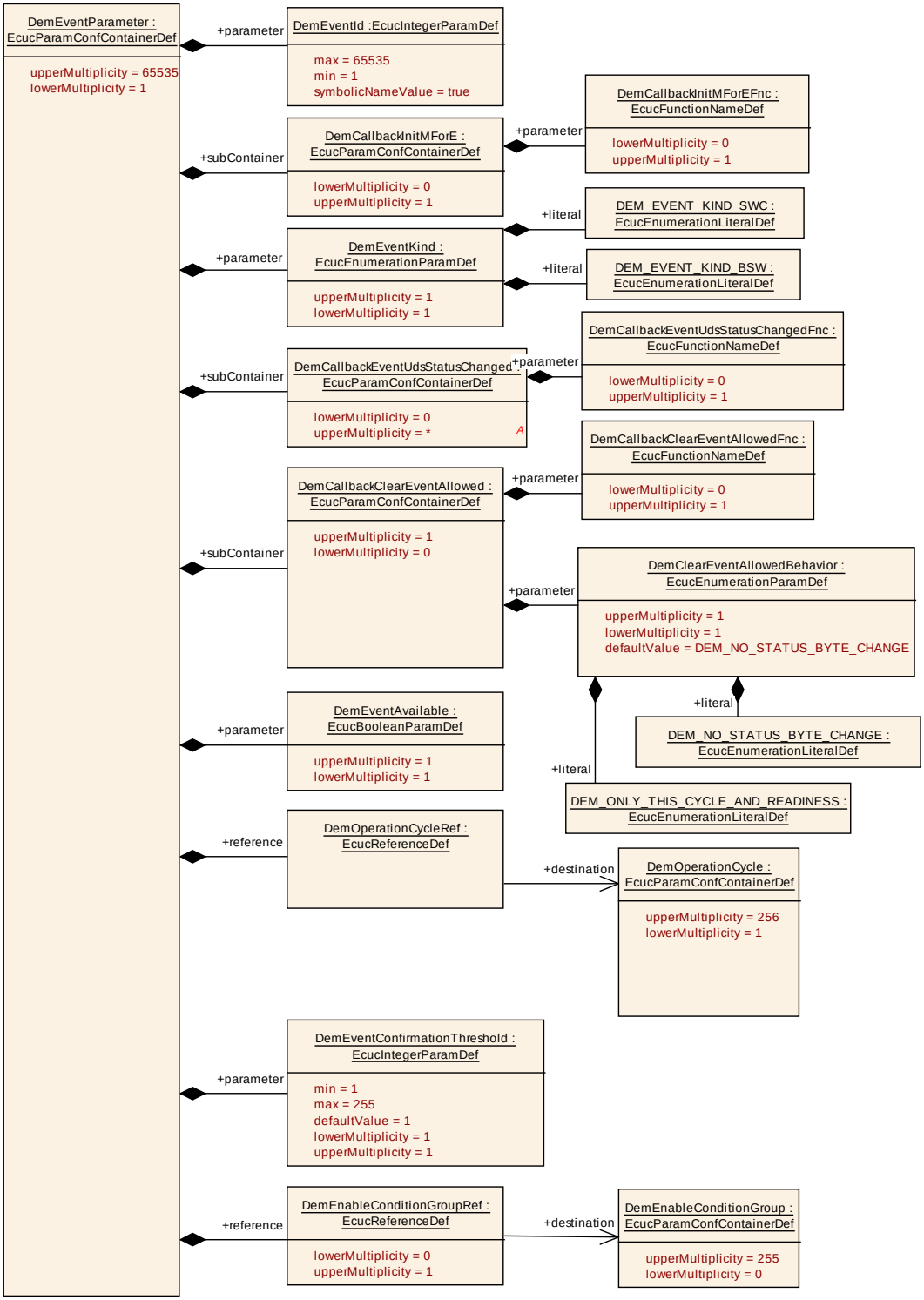


Figure 7.4: Event parameter configuration (part 1)

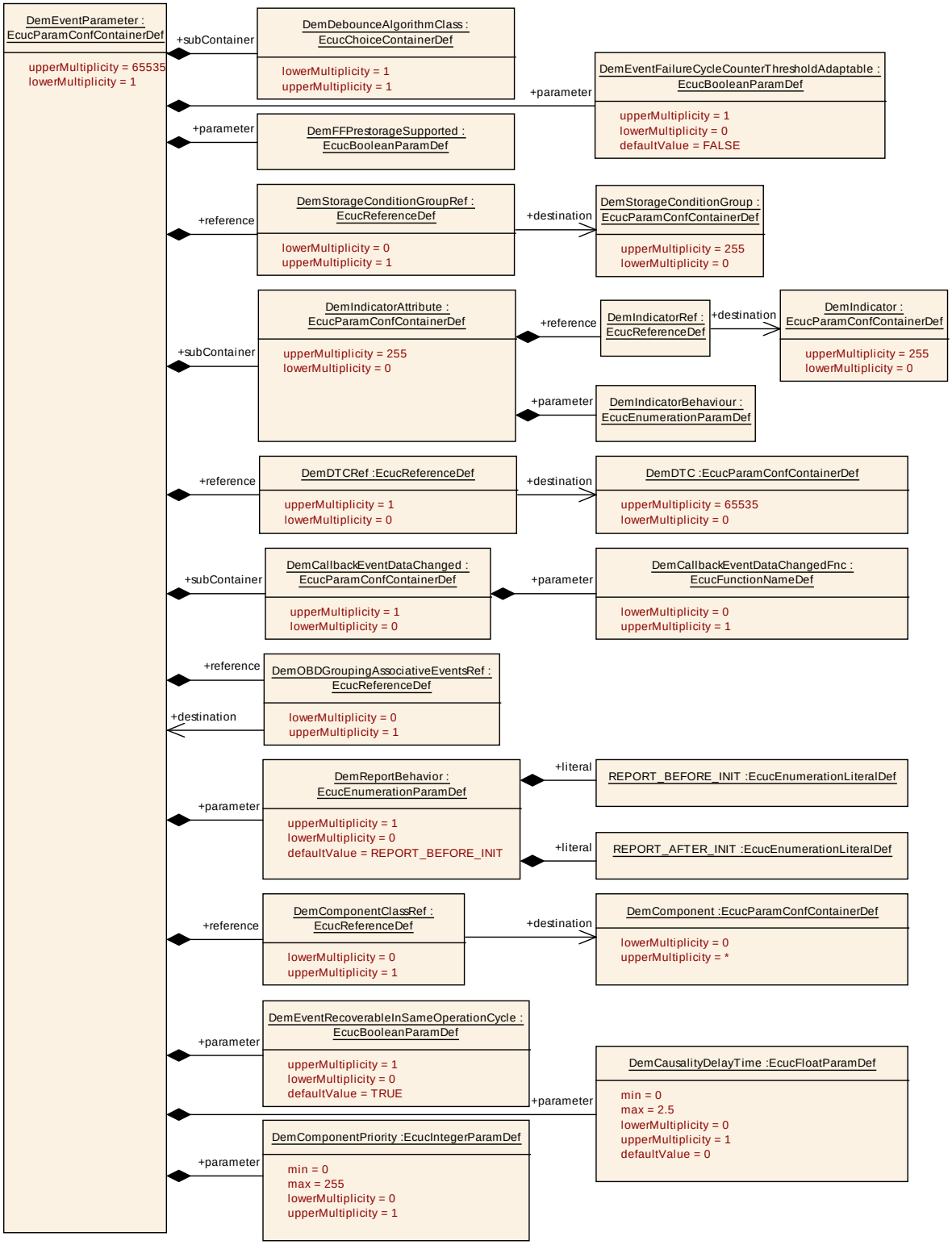


Figure 7.5: Event parameter configuration (part 2)

7.3.1 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 DemDTCPriority 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)*

7.3.2 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 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 status bit 0 transition from 0 to 1, if the related event is already stored in the event memory and the UDS status bit 3 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]). ⌋*(SRS_Diag_04125)*

7.3.3 Event kind

There are two different types of events:

- BSW-related events (reported via C-API - Dem_SetEventStatus)
- 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 7.8).

7.3.4 Event destination

The configuration parameter DemMemoryDestinationRef (refer to DemDTCAttributes) defines the dedicated storage location(s) of the event and its related data (refer to chapter 7.7.7).

The "permanent event memory" assignment is implicitly derived from the related DTC kind (refer to chapter 7.4.1). 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 7.9.5.8). 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, user defined, permanent and mirror memory).

[SWS_Dem_CONSTR_6104] Limitations on DemMemoryDestinationRef ⌈ If DemMirrorMemory is configured as DemMemoryDestinationRef, another DemMemoryDestinationRef on the same event of either DemPrimaryMemory or DemUserDefinedMemory shall be configured as a prerequisite. The same event shall not be configured two destinations if one is not DemMirrorMemory. ⌋*()*

[SWS_Dem_CONSTR_6114] Limitations on DemMemoryDestinationRef ⌈ A DTC can only reference the event memories via DemMemoryDestinationRef to the event memories of the same DemEventMemorySet. The scenario that a DTC references event memories via DemMemoryDestinationRef on different DemEventMemorySet is not supported. ⌋*()*

7.3.5 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.

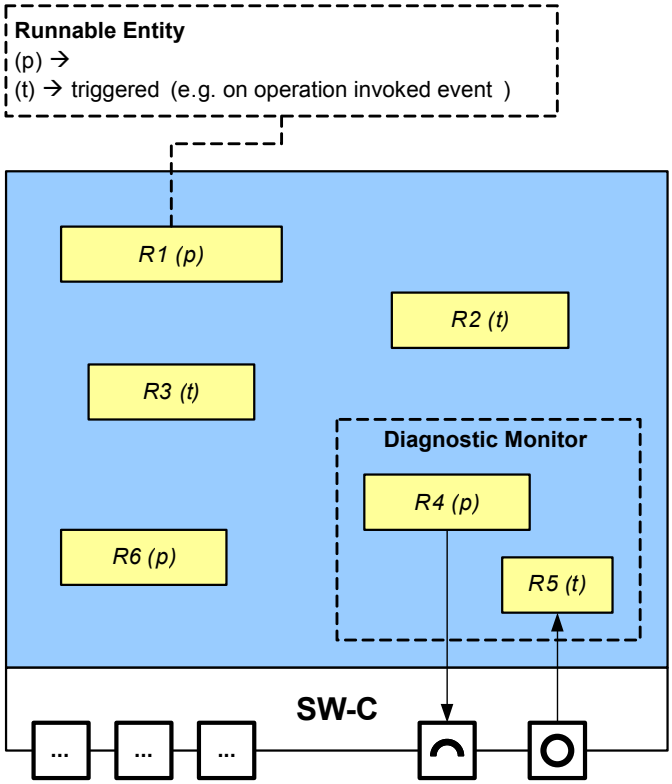


Figure 7.6: Example for a monitor embedded within a SW-C

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 7.7.3), the reporting API is called whenever the code with the functional check is executed.

7.3.6 Event dependencies

The priority of events within the assigned `DemComponent` and furthermore the dependency relation between the `DemComponent`s is used for filtering the storage of error reports to the failure memory.

[SWS_Dem_01126] ⌈ Whenever an event reports FAILED, it shall be considered as CONSECUTIVE FAULT, if any other event with higher priority at the same `DemComponent` already is FAILED. ⌋*(SRS_Diag_04142)*

[SWS_Dem_01127] ⌈ Whenever an event reports FAILED, it shall be considered as CONSECUTIVE FAULT, if any parent `DemComponent` is FAILED. ⌋*(SRS_Diag_04142)*

[SWS_Dem_01128] ⌈ The Dem shall allow to ignore the priority of events of a DemComponentIgnoresPriority). In this case, the event shall only be considered as CONSECUTIVE FAULT if any parent component is FAILED (other events FAILED status at the same component shall be ignored).] ⌋*(SRS_Diag_04142)*

[SWS_Dem_01129] ⌈ If a reported failure is not considered as consecutive fault, it shall be considered as CAUSAL FAULT. Causal faults shall be handled normally. ⌋ *(SRS_Diag_04142)*

[SWS_Dem_01218] ⌈ If a reported failure is not considered as consecutive fault and it has configured a DemCausalityDelayTime, it shall be considered as PRELIMINARY-CAUSAL FAULT. Starting from the point in time where the report of the event occurs, until the DemCausalityDelayTime is elapsed, the event can be reconsidered as being a consecutive fault. If another failure is reported during this time which has higher priority at the same DemComponent or at any parent DemComponent, the event shall be reconsidered as consecutive fault. After the time is elapsed, the failure will not be reconsidered. ⌋*(SRS_Diag_04142)*

[SWS_Dem_01130] ⌈ A failure which is considered as consecutive fault shall not be stored to the failure memory. (The handling shall be analogous to storage condition not fulfilled). ⌋*(SRS_Diag_04142)*

[SWS_Dem_01211] ⌈ A report of DEM_EVENT_STATUS_FDC_THRESHOLD_REACHED (e.g. either via interface call or when reaching the configured threshold in debounce algorithm) which is considered as consecutive report of a Failed event shall not be stored to the failure memory. The handling shall be analogous to storage condition not fulfilled. ⌋*(SRS_Diag_04142)*

Note: Other reports of DEM_EVENT_STATUS_FDC_THRESHOLD_REACHED shall not be considered as causal.

[SWS_Dem_01131] ⌈ The Dem shall provide the interface Dem_GetComponentFailed, which allows querying a DemComponents FAILED status. ⌋*(SRS_Diag_04142)*

[SWS_Dem_01132] ⌈ If DemTriggerFiMReports (refer to DemGeneral) is enabled, the Dem shall notify the FiM module [8] on each change of DemComponent failed status by calling the function DemTriggerOnComponentStatus. ⌋*(SRS_Diag_04142)*

[SWS_Dem_01133] ⌈ The Dem shall allow defining a callback per DemComponent (DemComponentFailedCallbackFnc; DemTriggerOnComponentStatus), which shall be called on a change of the components FAILED status. ⌋*(SRS_Diag_04142)*

7.3.7 Component availability

[SWS_Dem_01134] ⌈ The Dem shall support the availability of DemComponent. Components which are not available shall be treated as if they are not included in the system (e.g. Dem_GetComponentFailed will return E_NOT_OK). ⌋*(SRS_Diag_04142)*

[SWS_Dem_01135] ⌈ The interface `Dem_SetComponentAvailable` shall be available to set the availability state of a component. ⌋*(SRS_Diag_04142)*

[SWS_Dem_01136] ⌈ With setting a `DemComponent` to not available, all assigned events shall also be set as not available. ⌋*(SRS_Diag_04142)*

[SWS_Dem_01239] ⌈ After startup all `DemComponent`s shall be available. ⌋*(SRS_Diag_04142)*

[SWS_Dem_01226] ⌈ If a `DemComponent` is set to 'not available' via `Dem_SetComponentAvailable`, the Dem shall treat all the dependent components (children) as if they are set as 'not available'. ⌋*(SRS_Diag_04142)*

[SWS_Dem_01227] ⌈ If a `DemComponent` is set to 'not available' via `Dem_SetComponentAvailable`, the Dem shall set all assigned events to 'not available', including the events of all child nodes. The behavior for the events will be analogous to setting each event individually to 'not available' (e.g. setting the event status). ⌋*(SRS_Diag_04142)*

[SWS_Dem_01228] ⌈ If a `DemComponent` is set to 'not available' via `Dem_SetComponentAvailable` and any individual event assigned to it (see [SWS_Dem_01227]) is already failed, the individual event will be kept as 'available'. `Dem_SetComponentAvailable` still shall return E_OK. ⌋*(SRS_Diag_04142)*

[SWS_Dem_01229] ⌈ If a `DemComponent` is set to available via `Dem_SetComponentAvailable`, all assigned events are also set to 'available', including the events of all child nodes (if those nodes are not still set to 'not available'). The behavior for the individual events will be analogous to setting them individually to 'available' (e.g. triggering InitMonitorForEvent). ⌋*(SRS_Diag_04142)*

[SWS_Dem_01231] ⌈ If the function `Dem_SetEventAvailable` is called for an event to set the event to 'available', the function shall return E_NOT_OK, in case its related node is current 'not available'. ⌋*(SRS_Diag_04142)*

[SWS_Dem_01232] ⌈ If an event is set to available using `Dem_SetEventAvailable` and its component is 'not available', `Dem_SetEventAvailable` shall return E_NOT_OK.() ⌋*(SRS_Diag_04142)*

7.4 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 7.11.2).

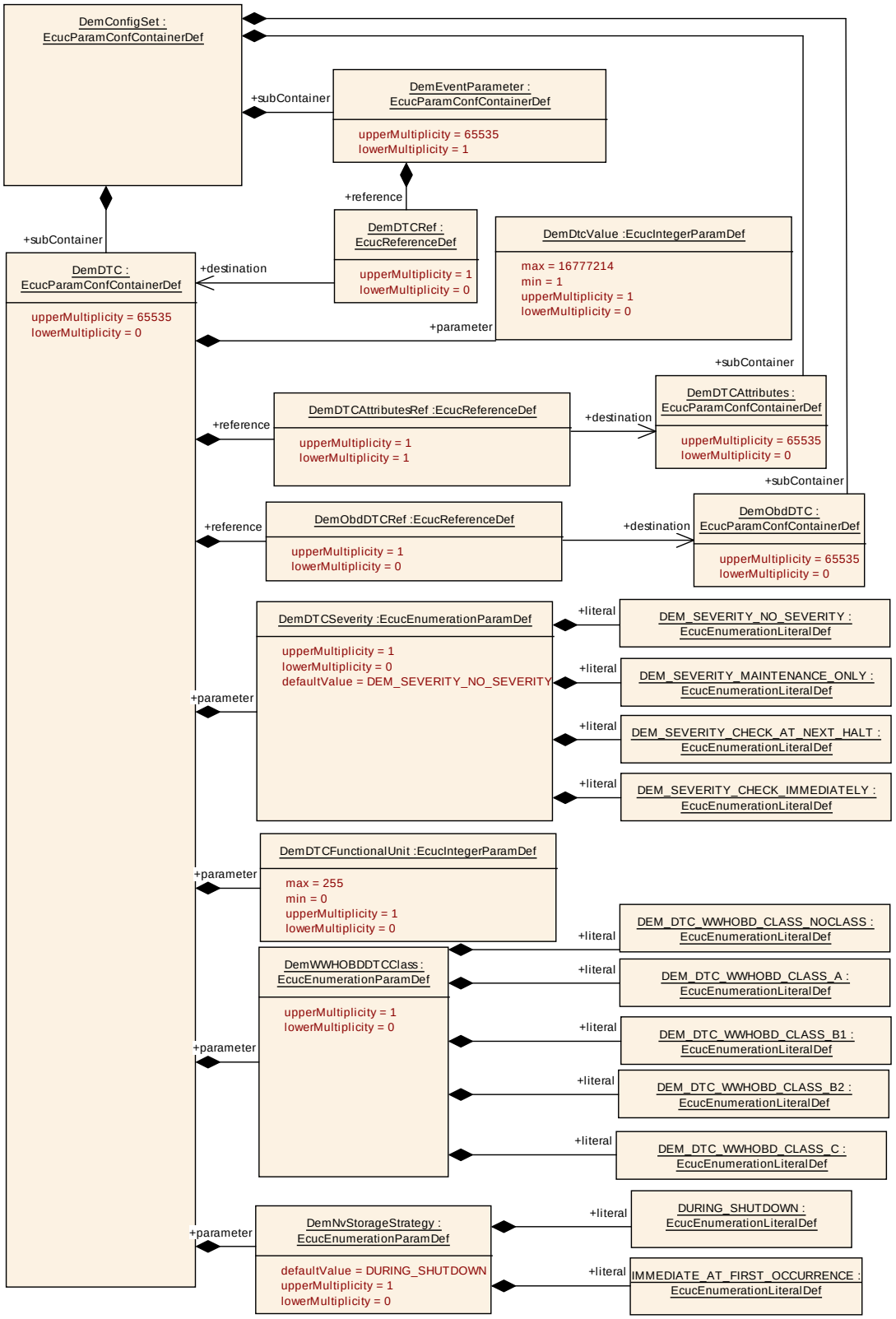


Figure 7.7: DTC configuration

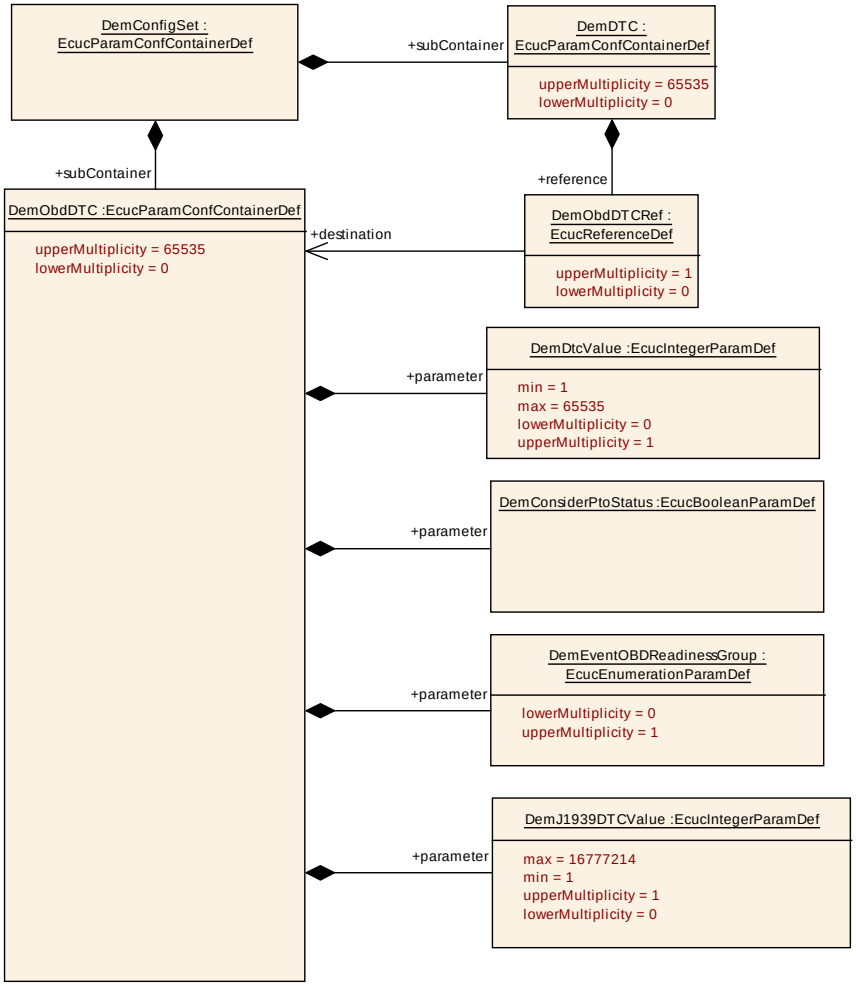


Figure 7.8: OBD-DTC configuration

7.4.1 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 8.6.1.9) 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_SetDTCFilter`), as well as for the diagnostic service ControlDTCSetting (0x85, refer to `Dem_DisableDTCSetting` and `Dem_EnableDTCSetting`).

7.4.2 DTC format

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

- ISO-14229-1[2]
- SAE J2012 OBD DTC (aka 2-byte DTC) [14]
- SAE J1939-73[15]
- ISO 11992-4[16]
- SAE J2012 WWH-OBD DTC (aka 3-byte DTC) [14]

⌋*()*

The configuration parameter DemTypeOfDTCSupported is used to specify one of the supported DTC formats defined in [SWS_Dem_00013] of the ECU (refer to Dem_GetTranslationType). This is required to determine the DTC format value to be reported for ISO-14229-1[2] 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_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 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)*

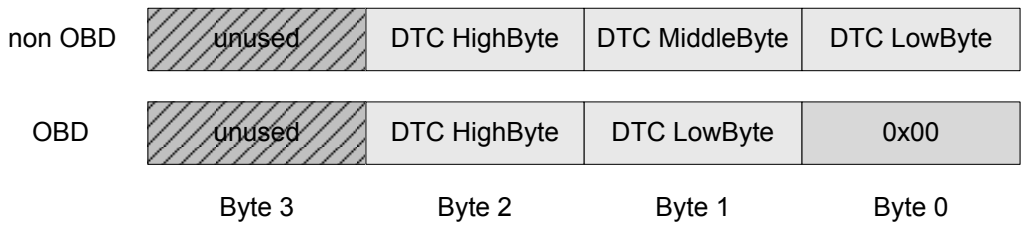


Figure 7.9: DTC Byte Order

[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 [15] for details.

[SWS_Dem_01180] ⌈ 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 WWH-OBD DTC format there are only three bytes (HighByte, MiddleByte and LowByte) used. The Dem services shall report these DTCs as a uint32 with byte 1 = LowByte, byte 2 = HighByte, byte 0 is used as FTB (as of SAE J2012-DA) [14]. ⌋*(SRS_Diag_04113)*

[SWS_Dem_CONSTR_06146] Dependency for DemDtcValue ⌈ The OBD DTC DemDtcValue shall only be present if DemOBDSupport is set to DEM_OBD_MASTER_ECU or DEM_OBD_PRIMARY_ECU. ⌋*()*

[SWS_Dem_00269] ⌈ The function Dem_GetDTCOfEvent shall get the DTC which is mapped to EventId by the Dem configuration. ⌋*(SRS_Diag_04201)*

7.4.3 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). When requesting operations on DTC groups (like ClearDTC and Disable/Enable DTC), the DTCGroup is selected by the DTC value.

[SWS_Dem_01059] ⌈ The DemGroupOfDTC shall represent the values of the DTC Group boundaries. ⌋*(SRS_Diag_04065)* Note: A group with the global boundaries 0x000100 and 0xfffe00 does not need to be configured, as the group ALL is automatically defined.

[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_ClearDTC), as well as for the diagnostic service ControlDTCSetting (0x85, refer to Dem_DisableDTCSetting and Dem_EnableDTCSetting).

The following DTC groups are provided:

- 'all DTCs' DTC group (mandatory, fixed value = 0xFFFFFF)
- Emission related DTCs for WWH-OBD (0xFFFF33)

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

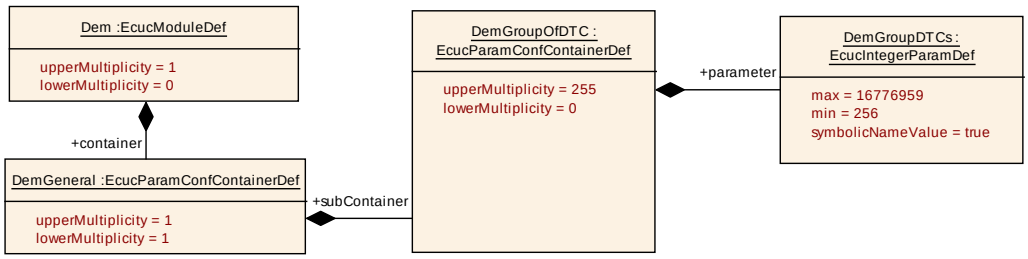


Figure 7.10: DTC group configuration

Taking into consideration that 0xFFFFFF 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 –> 0x3FFF00	DTC1 –> 0x3FFF04 DTC2 –> 0x3FFF06 . . . DTCn –> 0x3FFF0E
body DTC group –> 0x7FFF0F	... 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.

7.4.4 DTC severity

The DTC severity is used to provide information regarding the importance of the specific events according to ISO-14229-1[2], Annex D.3 “DTC severity and class definition”. The `DemDTCSeverity` is only available for UDS DTCs.

[SWS_Dem_01291] ⌈ If the API `Dem_GetSeverityOfDTC` or `Dem_GetNextFilteredDTCAndSeverity` is called and for the selected DTC a severity value in `DemDTCSeverity` is configured, the `Dem` shall set the value configured in `DemDTCSeverity` as `DTCSeverity`. ⌋*(SRS_Diag_04071)*

[SWS_Dem_01292] ⌈ If the API `Dem_GetSeverityOfDTC` or `Dem_GetNextFilteredDTCAndSeverity` is called and for the selected DTC no severity value in `DemDTCSeverity` is configured, the `Dem` shall set `DEM_SEVERITY_NO_SEVERITY` as `DTCSeverity`. ⌋*(SRS_Diag_04071)*

7.4.5 Functional unit

[SWS_Dem_01293] ⌈ If the `API Dem_GetFunctionalUnitOfDTC` is called and for the selected DTC a functional unit value in `DemDTCFunctionalUnit` is configured,

the Dem shall set the value configured in DemDTCFunctionalUnit in the out parameter DTCFunctionalUnit. ⌋(*SRS_Diag_04071*)

[SWS_Dem_01294] ⌈ If the API Dem_GetFunctionalUnitOfDTC is called and for the selected DTC no functional unit in DemDTCFunctionalUnit is configured, the Dem shall set a value of 0x00 in the out parameter DTCFunctionalUnit. ⌋ (*SRS_Diag_04071*)

7.4.6 DTC significance

There are two different significance levels of DTCs:

- 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 DemDTCSignificance in DemDTCAttributes) and can be mapped as a data element (refer to DEM_SIGNIFICANCE).

7.4.7 Suppress DTC output

This chapter describes the dynamic suppression of events/ DTCs during runtime via API call. A suppressed DTC behaves in a way that it is not visible for the tester but can be processed continuously by the diagnostic monitor.

An exemplary use-case could be that the decision to hide a DTC in a certain market is done by the SW-C at runtime.

The configuration parameter DemSuppressionSupport (refer to DemGeneral) controls the availability of the DTC suppression functionality

[SWS_Dem_00915] ⌈ The Dem shall suppress a DTC if all related events of this DTC are not available (refer to chapter 7.4.8) ⌋(*SRS_Diag_04126*)

If there is a one to one relationship between DTC and event, the DTC is suppressed if the related event is not available. If the DTC is a combined DTC, the DTC is suppressed if all combined events are not available.

Concerning output and query functions, suppressed DTCs are treated as if they are not configured in the system.

[SWS_Dem_01100] Behavior of DTC query functions for suppressed DTCs ⌈ In case any of the following APIs is called

- Dem_GetStatusOfDTC

- Dem_GetSeverityOfDTC
- Dem_GetFunctionalUnitOfDTC
- Dem_ClearDTC

on a single selected DTC which is suppressed, the Dem shall return DEM_WRONG_DTC. ⌋(*SRS_Diag_04126*)

[SWS_Dem_01101] ⌈ A suppressed DTC shall not be visible for the following Dcm query-functions; therefore the following functions shall treat the DTC as if filter is not matching:

- Dem_DcmGetDTCOfOBDFreezeFrame
- Dem_DcmReadDataOfOBDFreezeFrame
- Dem_DcmGetInfoTypeValue08
- Dem_DcmGetInfoTypeValue0B
- Dem_DcmReadDataOfPID01
- Dem_DcmGetDTRData
- Dem_GetDTCByOccurrenceTime
- Dem_GetNextFilteredDTC
- Dem_GetNextFilteredDTCAndFDC
- Dem_GetNextFilteredDTCAndSeverity
- Dem_GetNextFilteredRecord
- Dem_J1939DcmClearDTC
- Dem_J1939DcmFirstDTCwithLampStatus
- Dem_J1939DcmGetNextDTCwithLampStatus
- Dem_J1939DcmGetNextFreezeFrame
- Dem_J1939DcmSetDTCFilter
- Dem_J1939DcmGetNumberOfFilteredDTC
- Dem_J1939DcmGetNextFilteredDTC
- Dem_J1939DcmSetRatioFilter
- Dem_J1939DcmGetNextFilteredRatio
- Dem_J1939DcmReadDiagnosticReadiness1
- Dem_J1939DcmReadDiagnosticReadiness2
- Dem_J1939DcmReadDiagnosticReadiness3

- Dem_SelectFreezeFrameData
- Dem_SetDTCFilter
- Dem_SelectExtendedDataRecord

⌋(*SRS_Diag_04126*)

Note: This means, the (external) reporting (e.g. DTC number, UDS status, DTC extended data records, DTC statistical data, IUMPR, ...) of the DTC is suppressed.

[SWS_Dem_01102] ⌈ DTC suppression shall not stop event processing of the corresponding DTC. ⌋(*SRS_Diag_04126*)

Note: This means that e.g. event qualification, update of environmental data and functional degradation is still processed within the Dem.

Note: If DTC/freeze frame should not be stored in the fault memory storage conditions should be used in addition.

[SWS_Dem_01307] Functionality of Dem_GetDTCSuppression ⌈ If Dem_GetDTCSuppression is called, the Dem shall return the DTC suppression status in the output parameter SuppressionStatus. A value of TRUE means that the DTC was suppressed by a call of Dem_SetDTCSuppression with SuppressionStatus = TRUE, a value of FALSE means, that the DTC is currently not suppressed. ⌋(*SRS_Diag_04154*)

Also there are various reasons, why a DTC is not processed and thus is not visible for diagnostics, the suppression state itself is only changed by Dem_SetDTCSuppression. Dem_GetDTCSuppression only evaluates this suppression state.

7.4.8 Availability of events (visibility and computation)

This chapter describes a method to mark events as not available without removing them from the actual configuration. An event which is not available is treated as if it was not included in the Dem configuration.

An exemplary use-case could be that a control unit and its software support different kind of hardware to be controlled. Hardware variants vary in certain parts being available or not available. Still the same control unit and software shall be applied.

1. One calibration set is generated per hardware variant. The application engineers set up a post-build configuration to make sure the software behaves correctly for the hardware. For certain parts not included in a hardware variant, this means that related Events need to be shut down, suppressed. These events shall behave completely neutral, never set their LastFailed, not cause limp-home modes, not be visible to garage services.

   => ECUC parameter DemEventAvailable

2. One calibration covers multiple hardware variants. During production, or even at run-time when hardware is changed (by the garage) some coding bits are written to the control unit, or some network message is received, to change the behavior of the software in accordance with the changes in hardware. As in 1a, certain Events need to be suppressed.

   => Interface Dem_SetEventAvailable

[SWS_Dem_00687] ⌈ The Dem shall provide the functionality to treat events as if they were not configured as DemEventParameter. The configuration parameter DemAvailabilitySupport is used to enable the functionality (including API Dem_SetEventAvailable). ⌋*(SRS_Diag_04126)*

[SWS_Dem_01103] ⌈ Unavailable events shall not be considered for computation of service $01 PID $41. ⌋*(SRS_Diag_04126)*

[SWS_Dem_01104] ⌈ IUMPR ratios referring to an unavailable event shall neither be computed nor reported. ⌋*(SRS_Diag_04126)*

[SWS_Dem_01105] ⌈ DTRs referring to the event shall neither be computed nor reported (Dem_SetDTR). ⌋*(SRS_Diag_04126)*

[SWS_Dem_01106] ⌈ The Dem shall not provide the function Dem_SetEventAvailable in VARIANT-POST-BUILD. Rational: In VARIANT-POST-BUILD the initial values are not valid before Dem_Init (postbuild initialization time). ⌋*(SRS_Diag_04126)*

[SWS_Dem_01107] ⌈ The value of the configuration parameter DemEventAvailable shall be used as initial value for availability of an event and may be changed dynamically via API Dem_SetEventAvailable. ⌋*(SRS_Diag_04126)*

Note: Dem_SetEventAvailable is available before Dem_Init to “pre-configure” the Dem in advance of the application start.

Note: If an events availability is changed by Dem_SetEventAvailable and the ECU is powered down and up again. The desired status has to be set again by using Dem_SetEventAvailable.

[SWS_Dem_01108] ⌈ If an event is set to unavailable, the corresponding event shall be treated as if it is not configured in the system (e.g. Dem_SetEventStatus and Dem_GetEventUdsStatus shall return E_NOT_OK). The following APIs are affected:

- Dem_SetEventStatus
- Dem_GetEventUdsStatus
- Dem_ResetEventDebounceStatus
- Dem_ResetEventStatus
- Dem_PrestoreFreezeFrame
- Dem_ClearPrestoredFreezeFrame

- Dem_GetDebouncingOfEvent
- Dem_GetDTCOfEvent
- Dem_GetFaultDetectionCounter
- Dem_GetEventFreezeFrameDataEx
- Dem_GetEventExtendedDataRecordEx
- Dem_ClearDTC
- Dem_DcmGetDTRData
- Dem_RepIUMPRFaultDetect
- Dem_RepIUMPRDenLock
- Dem_RepIUMPRDenRelease
- Dem_SetWIRStatus
- Dem_DltGetMostRecentFreezeFrameRecordData
- Dem_DltGetAllExtendedDataRecords

⌋(*SRS_Diag_04126*)

Note: The availability of an event may also impact the suppression of a DTC (see [SWS_Dem_00915]).

[SWS_Dem_01109] ⌈ When the API Dem_SetEventAvailable is called with AvailableStatus = FALSE, the Dem shall return E_NOT_OK if:

- for the event an event memory entry already exists, or
- any of the event status flags 'testFailed', 'pending', 'confirmed' or 'warningIndicatorRequested' is set

⌋(*SRS_Diag_04126*)

[SWS_Dem_01302] Asynchronous behavior of Dem_SetEventAvailable ⌈ The Dem shall process a call of the Dem_SetEventAvailable asynchronously. This means that the final result is available at a later point in time. ⌋(*SRS_Diag_04126*)

A later point in time is meant to be implementation specific, it could be the next main function and after Dem_SetEventAvailable has returned.

[SWS_Dem_01110] ⌈ If Dem_SetEventAvailable is called with AvailableStatus == 'false', the Dem shall set the UDS status shall be set to 0x00 for this event. ⌋ (*SRS_Diag_04126*)

Note: If the UDS status changes, the corresponding callbacks are called to properly provide the information about a changed status to the users.

[SWS_Dem_01111] ⌈ If `Dem_SetEventAvailable` is called with `AvailableStatus` == 'true', the Dem shall set the UDS status shall be set to 0x50 for this event. ⌋ *(SRS_Diag_04126)*

[SWS_Dem_01230] ⌈ If the function `Dem_SetEventAvailable` is called to set an event to 'available', the `Dem` shall trigger the event initialization using `DEM_INIT_MONITOR_REENABLED` as `InitMonitorReason`. ⌋*(SRS_Diag_04142)*

7.5 Monitored Component Definition

A monitored component is a part of the system, which is relevant for being checked for proper operation. Malfunctions of the component need to be reported and documented. For this, usually one or multiple monitorings are defined and assigned to the component.

Furthermore the status of such components is relevant and used for influencing the system behavior (status of components may be used in `Fim`) and error documentation (filtering of consecutive failure entries).

The `Dem` allows to explicitly define the monitored components as `DemComponent`, defining the relation between events and components and finally allows to define the relation in-between the components.

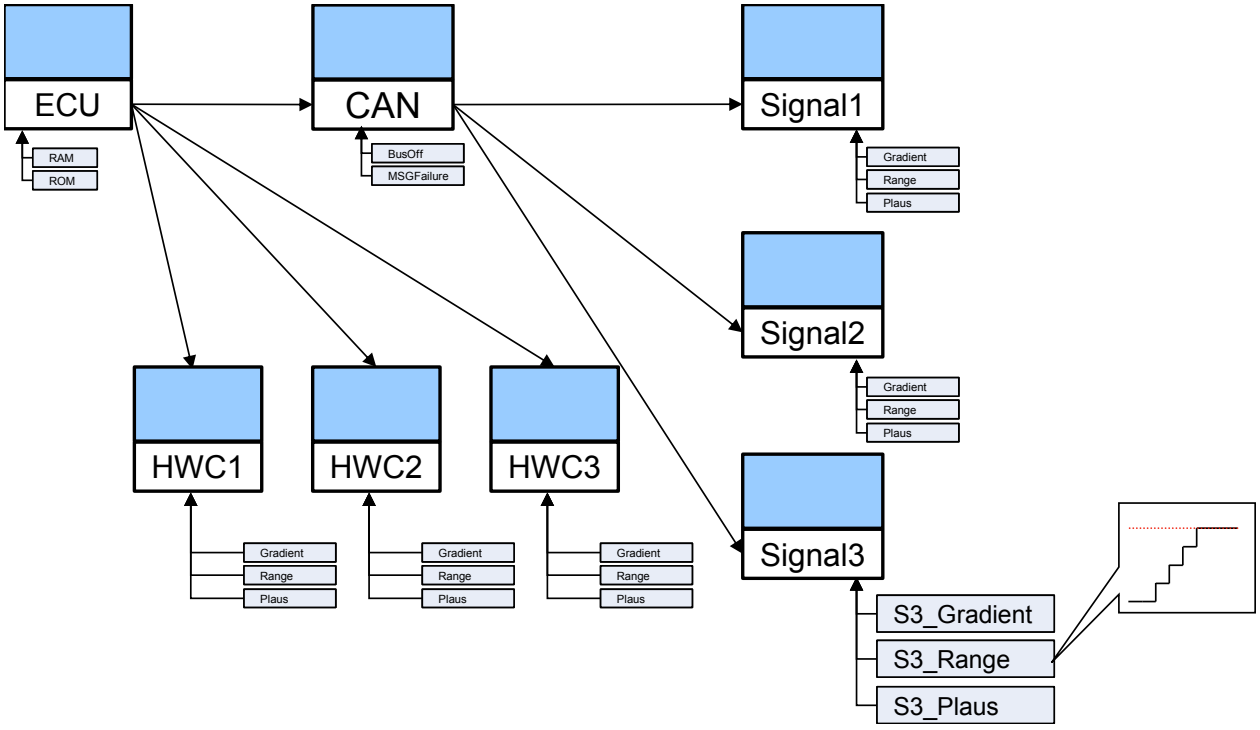


Figure 7.11: Event dependencies

7.5.1 Definition of components and dependencies

[SWS_Dem_01118] ⌈ The Dem shall represent monitored components as DemComponent with a unique identifier, which is used in interfaces to identify the component. ⌋ *(SRS_Diag_04142)*

[SWS_Dem_01119] ⌈ Each DemComponent shall have a boolean status FAILED, which represents the information whether the component has errors or not. ⌋ *(SRS_Diag_04142)*

[SWS_Dem_01120] ⌈ The Dem shall allow defining relations between DemComponent. The relations are directed, which means one component is depending on another component (DemImmediateChildComponentRef). Any component which has a directed relation (also over multiple components) is considered as parent component. ⌋ *(SRS_Diag_04142)*

Example: a -> b -> c -> d
a and b are parents of c
c and d are depending on b

[SWS_Dem_CONSTR_6106] ⌈ Only directed acyclic graph structures are supported for the dependencies of DemComponent. ⌋*()*

[SWS_Dem_01121] ⌈ The Dem shall allow to assign events to DemComponents. ⌋ *(SRS_Diag_04142)*

[SWS_Dem_CONSTR_6107] ⌈ Events may be assigned to exactly one DemComponent for which the monitoring is testing the error conditions. Multiple events may be assigned to the same component. ⌋*()*

[SWS_Dem_01122] ⌈ The Dem shall allow sorting the events assigned to a DemComponent by priority. The priority is defined per event as DemComponentPriority. ⌋ *(SRS_Diag_04142)*

[SWS_Dem_01123] ⌈ On reporting an error (DEM_MONITOR_STATUS_TF) for an event, the assigned DemComponent shall be set as FAILED in the context of the event report. ⌋*(SRS_Diag_04142)*

[SWS_Dem_01124] ⌈ On setting a DemComponent to FAILED, all dependent components shall be set as FAILED. ⌋*(SRS_Diag_04142)*

[SWS_Dem_01125] ⌈ A DemComponent shall be set to NOT FAILED, when allassigned events are NOT FAILED (DEM_MONITOR_STATUS_TF) and all parent components are NOT FAILED. ⌋*(SRS_Diag_04142)*

7.6 Operation cycle management

The Dem module uses different operation cycles (ref. to ISO-14229-1[2]).

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 cycle state changes via the API Dem_SetOperationCycleState.

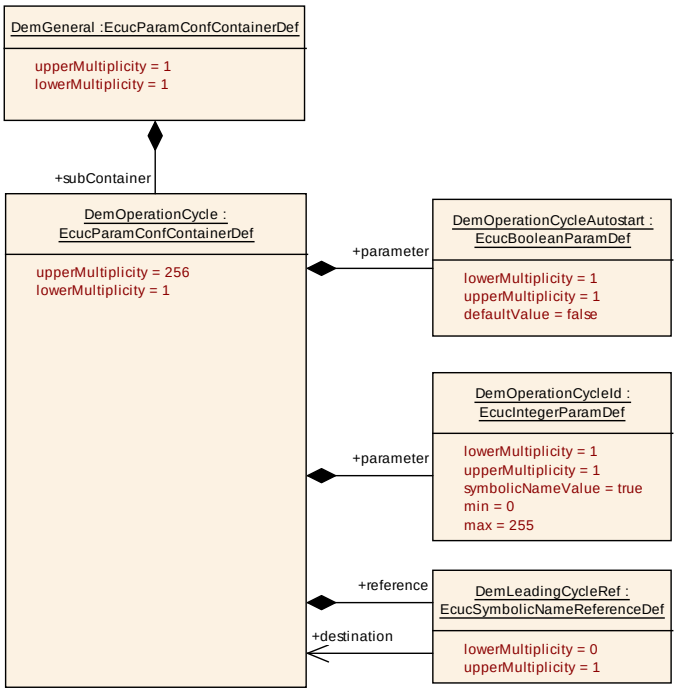


Figure 7.12: Operation cycle configuration

[SWS_Dem_00577] ⌈ The Dem module shall provide the configuration parameter DemOperationCycleStatusStorage to define if the operation cycle state shall be available over the power cycle (stored non-volatile) or not. ⌋*(SRS_Diag_04178)*

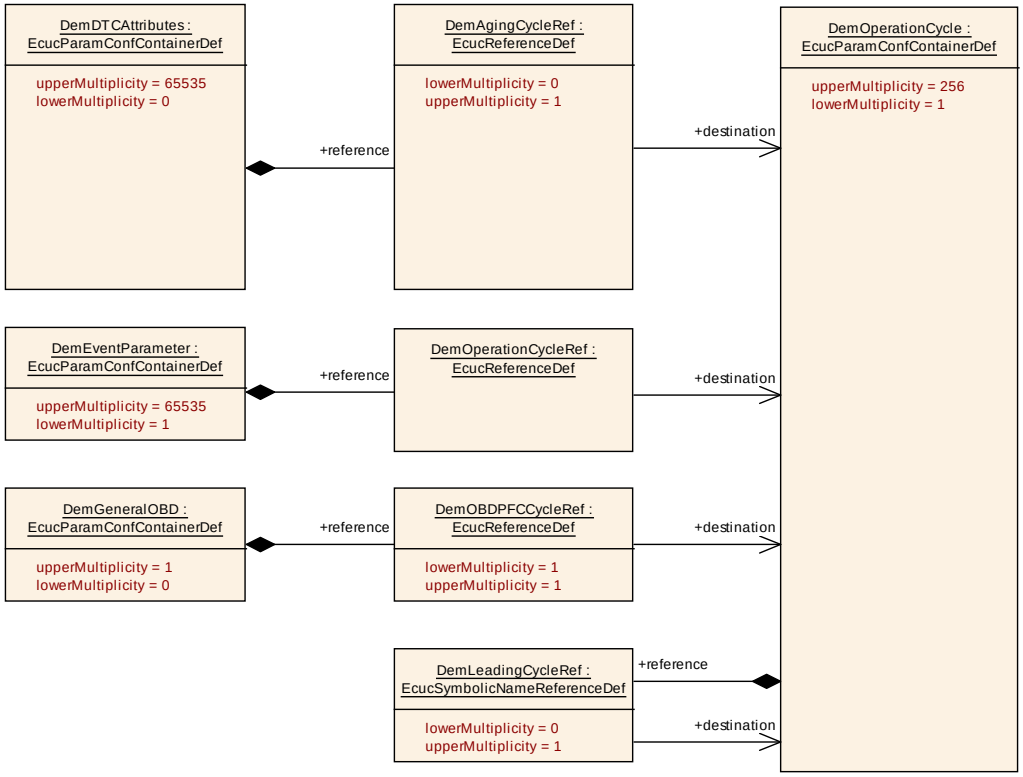


Figure 7.13: Operation and aging cycle assignment

[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. ⌋*(SRS_Diag_04178)*

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.

[SWS_Dem_00481] ⌈ If an operation cycle has started, all status reports from SW-Cs and BSW modules via `Dem_SetEventStatus` for those events, being assigned to this cycle, shall be accepted by the Dem from this point in time on. ⌋*(SRS_Diag_04178)*

[SWS_Dem_00482] ⌈ If an operation cycle has ended, all status reports from `Dem_SetEventStatus` for events assigned to this cycle shall be ignored and the `UDS status byte` shall remain unchanged. ⌋*(SRS_Diag_04178)*

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

[SWS_Dem_01221] ⌈ The `Dem` allows definition of dependent operation cycles using `DemLeadingCycleRef` parameter. Whenever the parameter is set, the `operation cycle` is considered as a dependent `operation cycle`. Whenever the parameter is not set, the `operation cycle` is considered as "normal" `operation cycle`. ⌋ *(SRS_Diag_04129, SRS_Diag_04133)*

[SWS_Dem_01222] ⌈ If the operation cycle passed to Dem_SetCycleQualified is not configured as dependent operation cycle, the Det error DEM_E_WRONG_CONFIGURATION shall be reported. ⌋(*SRS_Diag_04129*)

[SWS_Dem_01223] ⌈ If the operation cycle passed to Dem_SetOperationCycleState is configured as dependent operation cycle or not configured at all, the Det error DEM_E_WRONG_CONFIGURATION shall be reported. ⌋(*SRS_Diag_04129, SRS_Diag_04133*)

[SWS_Dem_01224] ⌈ When setting the state of an operation cycle using Dem_SetOperationCycleState to START (or RESTART), all operation cycles which are depending on the selected operation cycle shall also be started (restarted), but only if they are qualified. ⌋(*SRS_Diag_04129*)

Note: Setting a dependent operation cycle as qualified will not start the operation cycle, even if the leading operation cycle has been started before.

[SWS_Dem_01225] ⌈ When setting the state of an operation cycle using Dem_SetOperationCycleState to END, all operation cycle which are depending on the selected operation cycle shall also be ended, but only if they have been started. With ending a dependent operation cycle, its operation cycle qualification is reset. ⌋ (*SRS_Diag_04129*)

[SWS_Dem_01234] ⌈ The API Dem_GetCycleQualified shall provide the qualification state of the requested operation cycle in the out parameter "isQualified". ⌋ (*SRS_Diag_04129*)

[SWS_Dem_01235] ⌈ If the operation cycle passed to Dem_GetCycleQualified is not configured at all, the Det error DEM_E_WRONG_CONFIGURATION shall be reported. ⌋(*SRS_Diag_04129*)

[SWS_Dem_01220] ⌈ If DemOBDDelayedDCYConfirmedAndMIL is set to TRUE, the behavior of events linked to the OBD driving cycle (DemOBDDrivingCycleRef) shall be changed:
If the operation cycle is not qualified, status reports shall be processed, except update of the confirmed status. At the moment the OBD driving cycle is set to qualified, the Dem shall execute computations required to reach the confirmed states of events collected during the "not qualified" phase of the cycle. ⌋(*SRS_Diag_04001, 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 to set the Dem specific operation cycle state (started / ended). ⌋(*SRS_Diag_04178*)

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.

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_04178)*

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_04178)*

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_01078] ⌈ If DemOperationCycleStatusStorage is set to FALSE, a call of Dem_Shutdown shall end all operation cycles which are not stored in non-volatile memory. ⌋*(SRS_Diag_04178)*

7.6.1 Operation Cycle Counters

7.6.1.1 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. ⌋*(SRS_Diag_04190)*

[SWS_Dem_00771] ⌈ If internal debounce counter reach DemDebounceCounterFailedThreshold (latest UDS 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 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_04178)*

[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)*

7.6.1.2 Cyles since first failed

The counter Cycles 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 'Cycles 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, SRS_Diag_04190)*

[SWS_Dem_00776] ⌈ If internal debounce counter reach DemDebounceCounterFailedThreshold (latest UDS 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)*

7.6.1.3 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_04189, SRS_Diag_04190)*

[SWS_Dem_00780] ⌈ If internal debounce counter reach DemDebounceCounterFailedThreshold (latest UDS 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_04105*)

[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 status bit 1 is set to 1. ⌋(*SRS_Diag_04105*)

[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*)

7.7 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 UDS 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 (user defined, mirror, permanent). ⌋ (*SRS_Diag_04066*)

[SWS_Dem_01247] ⌈ Configured via the container DemEventMemorySet, the Dem supports multiple independent event memories. Each API call with DemClient as parameter, shall be applicable only on the configured event memory (primary, mirror or user defined) for this client. ⌋(*SRS_Diag_04164*)

[SWS_Dem_CONSTR_06118] Unique DTC values within a single event memory ⌈ The DemDtcValue shall be unique within all DTCs referencing the same event memory. ⌋()

[SWS_Dem_CONSTR_06119] Unique OBD DTC values within an ECU ⌈ The DemDtcValue shall be unique within all DTCs referencing the same event memory. ⌋()

The Dem can be configured to support more than one DemEventMemorySet. This allows the Dem to store DTCs and event related data of more than one diagnostic server. Each DemEventMemorySet is treated as independent entity. Changes in one DemEventMemorySet affect only the local addressed DemEventMemorySet and are not visible in another DemEventMemorySet.

As a result of this, operations such as ClearDTC on one event memory only effects the addressed DemEventMemorySet. In other words, a ClearDTC operation has no impact on DTCs or event related data in a different DemEventMemorySet.

[SWS_Dem_01217] ⌈ In order to calculate the user defined memory ID from `Dem_DTCOriginType`, the Dem shall substract a value of 0x0100 from the obtained value if the value is in the range of [0x0100, 0x01FF]. ⌋*(SRS_Diag_04214)*

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

[SWS_Dem_00162] ⌈ The `Dem` shall provide configuration parameters to adapt the fault memory size to the ECU memory space available (refer to configuration parameters `DemMaxNumberEventEntryPrimary`, `DemMaxNumberEventEntryMirror` or `DemMaxNumberEventEntryUserDefined`). ⌋*(SRS_Diag_04066)*

[SWS_Dem_00329] ⌈ For storing to non-volatile memory the Dem shall use the NVRAM Manager (refer to chapter 7.11.5). ⌋*(SRS_Diag_04077)*

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

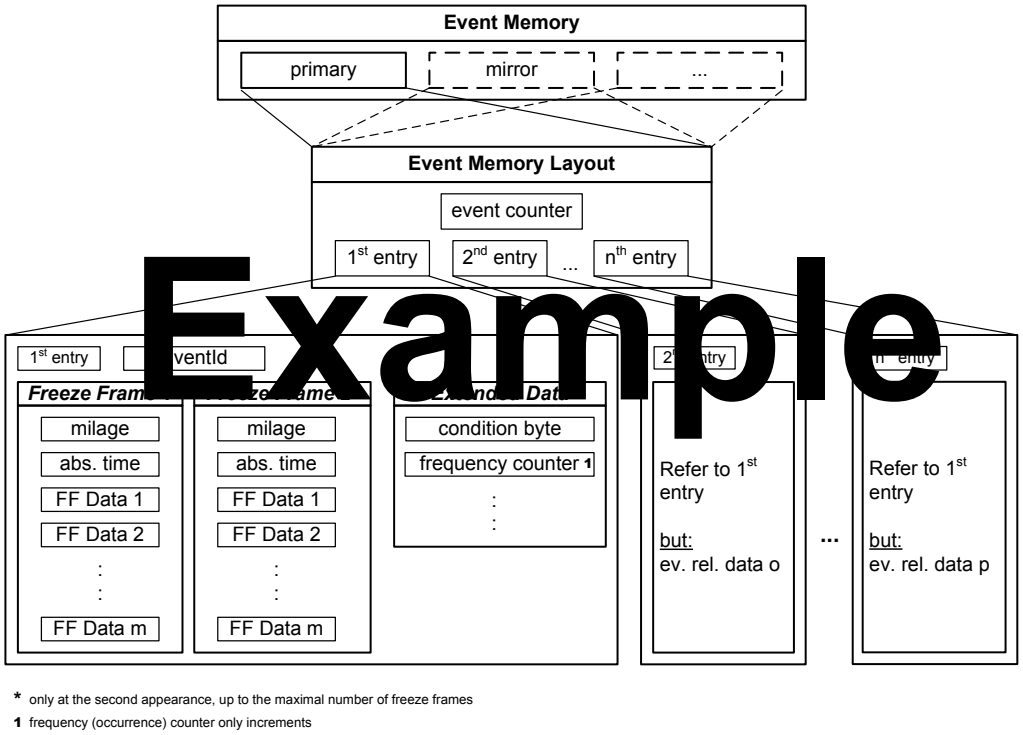


Figure 7.14: Example of a logical Dem `event memory` layout

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 7.7.2).

7.7.1 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 Dem_SetEventStatus). ⌋*(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).

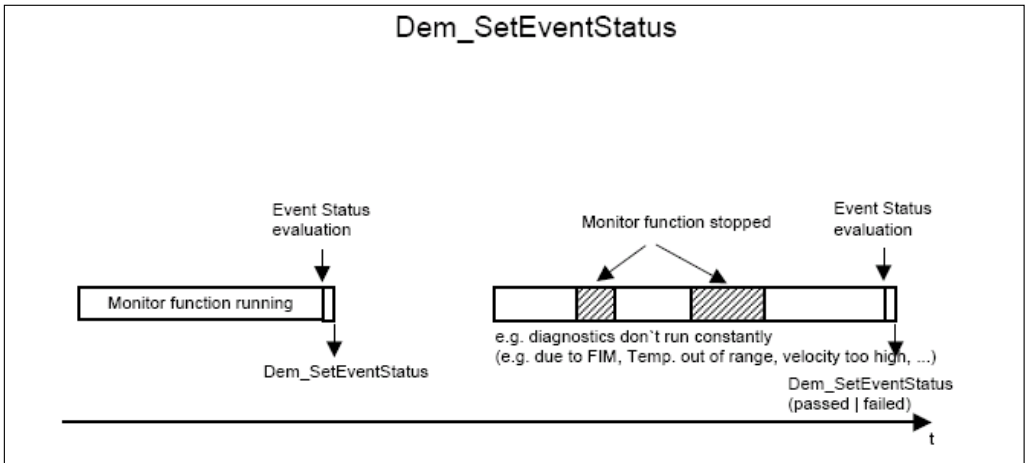


Figure 7.15: Example for using Dem_SetEventStatus

[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 Dem_ResetEventStatus). ⌋*(SRS_Diag_04179)*

Note: Dem_ResetEventStatus does not mean an event status report (like done by Dem_SetEventStatus). After the call, the monitor 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] Behavior of Dem_ResetEventStatus ⌈ The function Dem_ResetEventStatus shall set the event status bit 0 (TestFailed) to 0 and reset the Dem-internal debounce algorithm to initial values if configured. The Dem shall execute this action asynchronously at a later point in time. ⌋*(SRS_Diag_04067)*

A later point in time means that the exact moment is implementation specific. This could be the next main function and after Dem_ResetEventStatus has returned. Also a possible call of the status change callback is triggered asynchronously.

Note: The function Dem_ResetEventStatus does not change the monitor status and UDS status bit 6 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 status bit 6 is set to 0). ⌋*(SRS_Diag_04067)*

[SWS_Dem_00051] ⌈ If the API Dem_GetEventUdsStatus is called, the Dem shall provide the current event status byte of a specific event. ⌋*(SRS_Diag_04067)*

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

[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. ⌋ *(SRS_Diag_04068)*

The Dem supports a configuration per event to forbid to reset the event in the same operation cycle.

[SWS_Dem_01208] ⌈ If the configuration parameter DemEventRecoverableInSameOperationCycle is set to FALSE, a report of PASSED (directly via Dem_SetEventStatus or indirectly after Debouncing) will be ignored if the event status byte has the "TestFailedThisOperationCycle" flag is set to 1. ⌋*(SRS_Diag_04151)*

Note: Debouncing is done independently from the value DemEventRecoverableInSameOperationCycle. The FDC can reach -128 independently from the setting of DemEventRecoverableInSameOperationCycle.

[SWS_Dem_01209] ⌈ The Dem shall perform the configured debouncing independently from the setting of the configuration parameter DemEventRecoverableInSameOperationCycle. I.e. if DemEventRecoverableInSameOperationCycle is set to FALSE and the monitor for this event reports PREPASSED the Dem shall still process the debouncing (e.g. decrement the counter). ⌋*(SRS_Diag_04151)*

[SWS_Dem_01250] ⌈ If DemMultiEventTriggering is configured and any of the APIs:

- Dem_SetEventStatus
- Dem_ResetEventStatus
- Dem_PrestoreFreezeFrame
- Dem_ClearPrestoredFreezeFrame
- Dem_ResetEventDebounceStatus

is called with an EventId that is referenced by DemMultiEventTriggeringMasterEventRef of a DemMultiEventTriggering, for each DemMultiEventTriggeringSlaveEventRef of this DemMultiEventTriggering the Dem shall trigger

the same API calls with identical parameters only the EventId parameter shall be replaced with the event referenced from DemMultiEventTriggeringSlaveEventRef. ⌋(*SRS_Diag_04165*)

[SWS_Dem_CONSTR_6115] ⌈ The Dem does not support calls of

- Dem_SetEventStatus
- Dem_ResetEventStatus
- Dem_PrestoreFreezeFrame
- Dem_ClearPrestoredFreezeFrame
- Dem_ResetEventDebounceStatus

with an EventId that is referenced by any of the DemMultiEventTriggeringSlaveEventRef in container DemMultiEventTriggering. These events are exclusively used for internal triggering by calling these APIs for the master event (DemMultiEventTriggeringMasterEventRef). The behavior of the Dem is undefined if any of those APIs are called in this situation. ⌋(*SRS_Diag_04165*)

7.7.1.1 Status bit support

[SWS_Dem_00006] ⌈ The Dem module shall by default implement the current set of all UDS status bits according to the definition in ISO-14229-1[2] for each diagnostic event. ⌋(*SRS_Diag_04067*)

Note: For this specification, the UDS status byte of ISO-14229-1[2] for events is represented by the Dem data type Dem_UdsStatusByteType (defined in chapter 8.6.1.20). If particular UDS status bits do not need to be supported (refer also to [SWS_Dem_00060]), such optimizations or limitations may be implemented vendor-specific.

[SWS_Dem_01276] ⌈ If the API Dem_GetEventUdsStatus is called, the event from the parameter EventId exists and is available according to chapter subsection 7.4.8, the Dem shall copy the UDS status byte into the out parameter UdsStatusByte and return E_OK. ⌋(*SRS_Diag_04067*)

[SWS_Dem_01277] ⌈ If the API Dem_GetEventUdsStatus is called, the event from the parameter EventId does not exists or is unavailable according to chapter subsection 7.4.8, the Dem shall return E_NOT_OK. ⌋(*SRS_Diag_04067*)

7.7.1.2 Monitor status and UDS status update

Status bit processing is an essential part of the Dem functionality. SWCs or BSW modules frequently report monitor results via the API Dem_SetEventStatus. A call of these functions will trigger the status bit processing of the Dem.

[SWS_Dem_01278] ⌈ The Dem shall provide a monitor status per event to store current monitor state information. The monitor status contains the information as defined in Dem_MonitorStatusType. ⌋*(SRS_Diag_04067)*

[SWS_Dem_01279] ⌈ The Dem shall perform the counter based event debouncing synchronously within the context of the calling Dem_SetEventStatus function. ⌋ *(SRS_Diag_04068)*

[SWS_Dem_01280] ⌈ Upon call of Dem_SetEventStatus and after debouncing, the Dem shall process the monitor status in the same manner as the UDS status:

- Bit0 (TestFailed) according to [SWS_Dem_00386]
- Bit1 (TestNotCompleteThisOperationCycle) according to [SWS_Dem_00394].

The monitor status byte processing shall be done synchronously within the context of the calling Dem_SetEventStatus function. ⌋*(SRS_Diag_04067)*

[SWS_Dem_01281] ⌈ If the parameter DemGeneralCallbackMonitorStatusChangedFnc is configured, the Dem shall call this configured C function with the signature of <Module>_DemTriggerOnMonitorStatus. ⌋*(SRS_Diag_04067)*

[SWS_Dem_01282] ⌈ Upon processing the monitor status, the Dem shall call the monitor status changed callback functions:

- <Module>_DemTriggerOnMonitorStatus

whenever the monitor status has changed. ⌋*(SRS_Diag_04067)*

[SWS_Dem_01283] ⌈ Upon processing the monitor status for reported event with DemEventKind = DEM_EVENT_KIND_SWC, the Dem shall call the monitor status changed callback C/S Interface CallbackMonitorStatusChange whenever the monitor status has changed and a C/S interface is configured for this event. ⌋*(SRS_Diag_04067)*

[SWS_Dem_CONSTR_06120] Dependency for DemGeneralCallbackMonitorStatusChangedFnc ⌈ The DemGeneralCallbackMonitorStatusChangedFnc shall only be present if DemGeneralInterfaceSupport is set to TRUE." ⌋*()*

Limitation: The usage of monitorStatusChange via RTE is limited to SWC-monitorings. All basic software monitorings cannot be covered with this callback via RTE. To get complete statusChange information of all monitorings in a SW-C, the eventStatusChange callbacks should be used.

The CallbackMonitorStatusChange and GeneralCallbackMonitorStatusChanged via RTE is furthermore limited to monitorings, which are exclusively changing its status in known contexts. E.g. if a monitor status is changed from CDD (e.g. Dem_SetEventAvailable via C-function call), the monitor status change callback via RTE must not be used.

Rational: AUTOSAR RTE requires that calls from a BSW to a SW-C via RTE are done in a known call context. For the synchronous monitor status changed callbacks, this is only possible for events reported from a SW-C. For events reported from a BSW

module it is not possible to provide the calling context to the RTE. As a result of this, the Dem provides the C/S interface CallbackMonitorStatusChange only for events that are reported from a SW-C. For events reported from BSW modules, no such C/S interface is provided. Furthermore, the interface GeneralCallbackMonitorStatusChanged is not provided as C/S via RTE, as this callback could only be called for events reported from a SW-C and not for events reported from BSW modules. This would obviously lead to confusion.

If it is required to have a general synchronous reaction in a SW-C on monitor status changed callbacks, the synchronous reaction could be handled within the scope of a CDD using <Module>_DemGeneralTriggerOnMonitorStatus or <Module>_DemTriggerOnMonitorStatus.

[SWS_Dem_CONSTR_6116] Limited use of monitor status change callbacks to events reported from SW-Cs only ⌈ If Dem_SetEventAvailable is called from a Cdd or BSW module, the corresponding monitor status changed callback can only be used as C-function, but not via RTE interface. ⌋*()*

[SWS_Dem_01284] ⌈ When processing a clear DTC command (e.g. via Dem_ClearDTC), the Dem shall set the monitor status assigned to the cleared DTCs to 0x04. ⌋*(SRS_Diag_04067)*

[SWS_Dem_01285] ⌈ The Dem shall process the UDS status asynchronously. This means it is calculated outside the context of the calling Dem_SetEventStatus function. ⌋*(SRS_Diag_04067)*

[SWS_Dem_01286] ⌈ If the API Dem_SetEventStatus is called and the Dem has synchronously processed the reported monitor status according to [SWS_Dem_01280] and the corresponding UDS status (see [SWS_Dem_01285]) cannot be processed, the Dem shall report the development error DEM_E_UDS_STATUS_PROCESSING_FAILED. ⌋*(SRS_Diag_04067)*

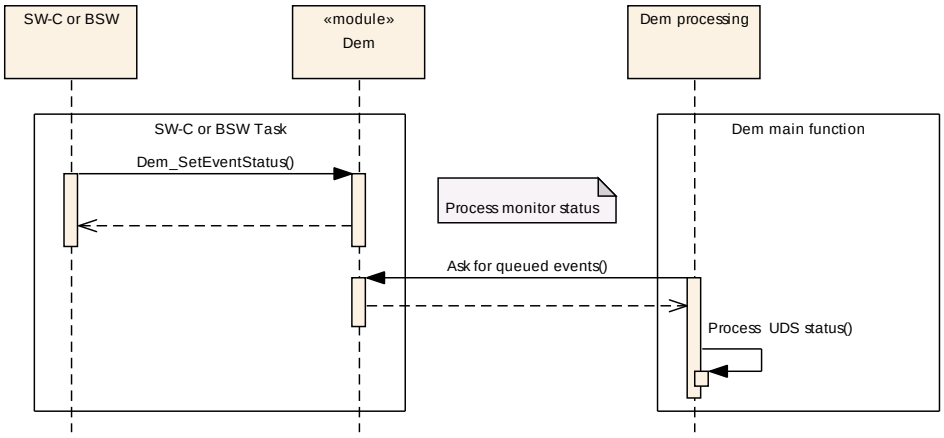


Figure 7.16: Synchronous and asynchronous event processing

Note: Refer to chapter 7.7.2 for a detailed description about the event memory management.

[SWS_Dem_01064] ⌈ The mirror memory is not relevant for calculation of the UDS status. ⌋*(SRS_Diag_04131)*

[SWS_Dem_01287] ⌈ If the API Dem_GetMonitorStatus is called, the Dem shall copy the current monitor status into the out parameter MonitorStatus and return E_OK. ⌋*(SRS_Diag_04067)*

[SWS_Dem_01288] ⌈ If the API Dem_GetMonitorStatus is called with an invalid EventId the Dem shall return E_NOT_OK. ⌋*()*

7.7.1.3 Status bit transitions

This section describes the behavior of the individual status bits of the UDS status byte (refer to ISO-14229-1[2], UDS status byte - bit transitions).

[SWS_Dem_00385] ⌈ After a clear command has been applied to a specific DTC (refer to chapter 7.7.2.2) 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_04067)*

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 status bit 0 according to figure 7.17. ⌋*(SRS_Diag_04067)*

[SWS_Dem_00387] ⌈ The Dem module shall support the configuration parameter DemStatusBitStorageTestFailed (refer to DemGeneral) to determine whether the information for UDS status bit 0 is stored volatile or non-volatile. ⌋*(SRS_Diag_04067)*

[SWS_Dem_00388] ⌈ If the configuration parameter DemStatusBitStorageTestFailed is set to False, the Dem module shall not retain the information for UDS status bit 0 over power cycles (volatile) ⌋*(SRS_Diag_04067)*

[SWS_Dem_CONSTR_6113] Configuration of the test failed status bit storage ⌈ For WWH-OBD ECU the DemStatusBitStorageTestFailed shall be set to True. ⌋*()*

[SWS_Dem_00525] ⌈ If the configuration parameter DemStatusBitStorageTestFailed is set to True, the Dem module shall retain the information for UDS status bit 0 over power cycles (non-volatile). ⌋*(SRS_Diag_04067)*

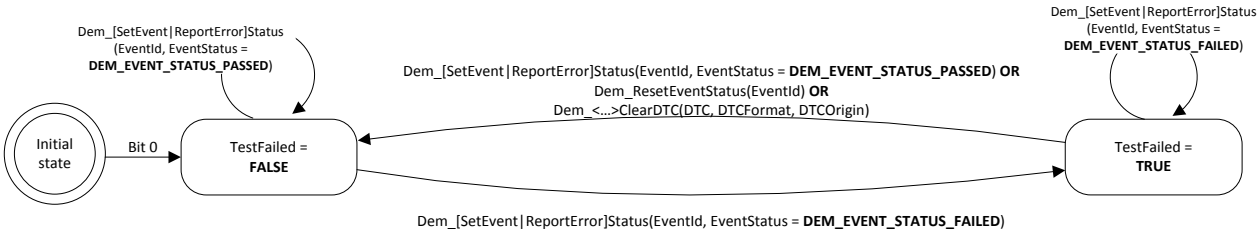


Figure 7.17: UDS status bit 0 TestFailed logic

[SWS_Dem_00389] ⌈ The Dem module shall implement the status bit transition for UDS status bit 1 according to figure 7.18. ⌋*(SRS_Diag_04067)*

Note: The information for event 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).

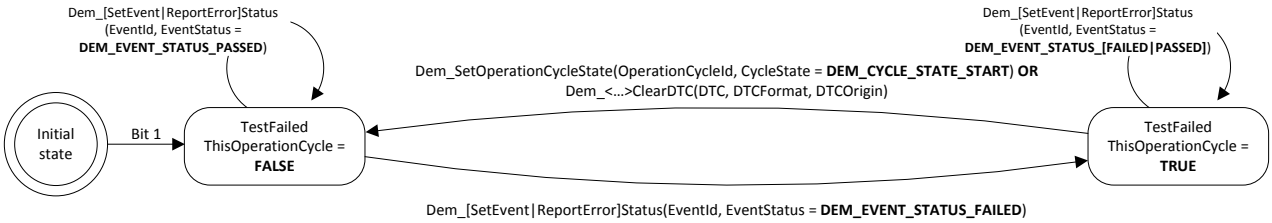


Figure 7.18: UDS status bit 1 TestFailedThisOperationCycle logic

[SWS_Dem_00390] ⌈ The Dem module shall implement the status bit transition for UDS status bit 2 according to figure 7.19. ⌋*(SRS_Diag_04067)*

[SWS_Dem_01183] ⌈ The information for UDS status bit 2 needs to be stored non-volatile ⌋*(SRS_Diag_04141)*

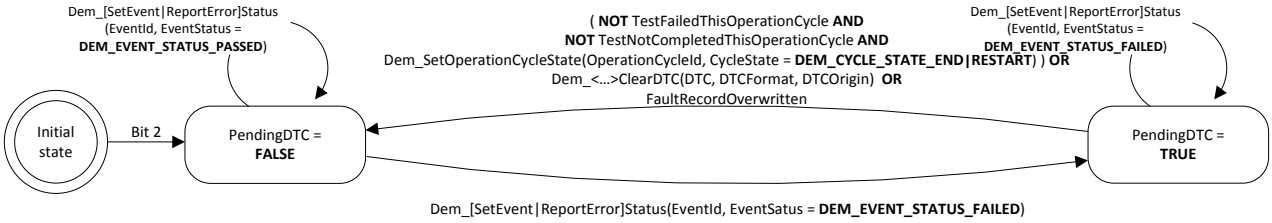


Figure 7.19: UDS status bit 2 PendingDTC logic

[SWS_Dem_00391] ⌈ The Dem module shall implement the status bit transition for UDS status bit 3 according to figure 7.20. ⌋*(SRS_Diag_04067)*

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

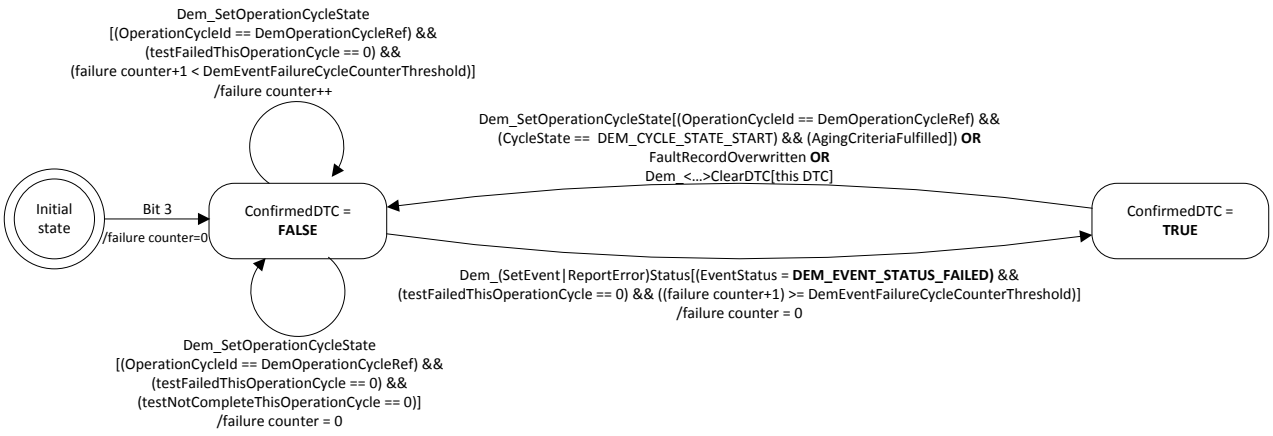


Figure 7.20: UDS status bit 3 ConfirmedDTC logic

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 status bit 4 according to figure 7.21. ⌋*(SRS_Diag_04067)*

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

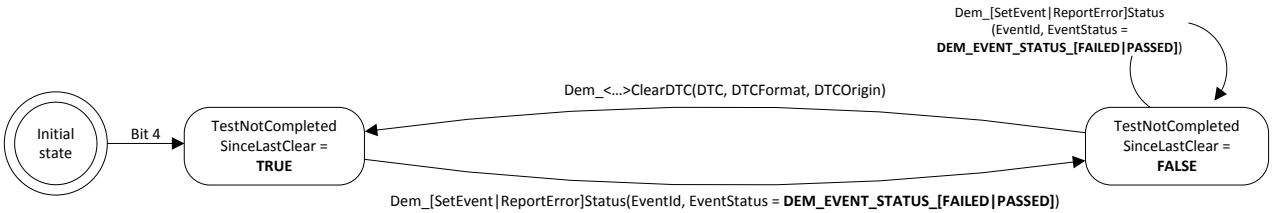


Figure 7.21: UDS status bit 4 TestNotCompletedSinceLastClear logic

[SWS_Dem_00393] ⌈ The Dem module shall implement the status bit transition for UDS status bit 5 according to figure 7.22 . ⌋*(SRS_Diag_04067)*

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

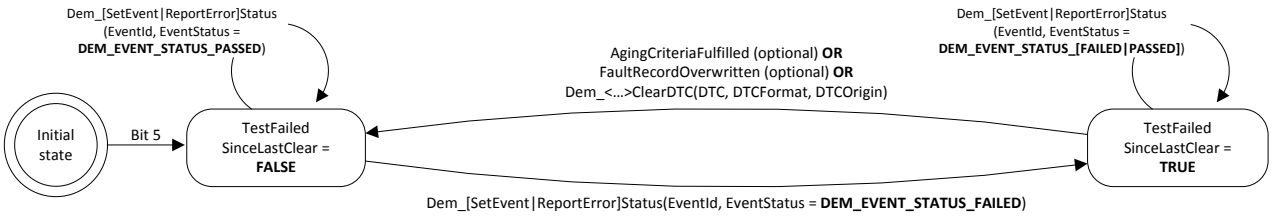


Figure 7.22: UDS status bit 5 TestFailedSinceLastClear logic

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 status bit 6 according tofigure 7.23. ⌋*(SRS_Diag_04067)*

Note: The information for UDS 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).

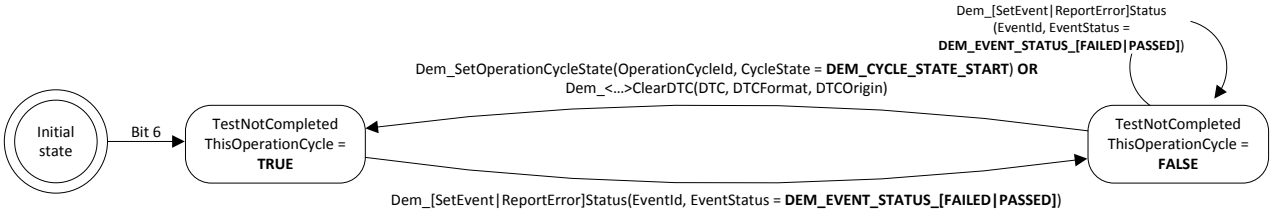


Figure 7.23: UDS status bit 6 TestNotCompleteThisOperationCycle logic

[SWS_Dem_00395] ⌈ The Dem module shall implement the status bit transition for UDS status bit 7 (WarningIndicatorRequested) according to figure 7.24. ⌋ *(SRS_Diag_04067)*

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

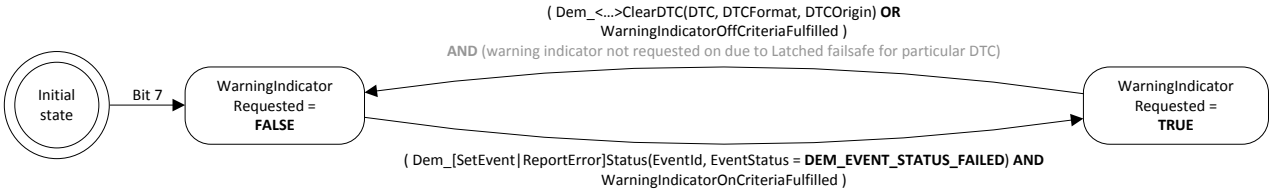


Figure 7.24: UDS status bit 7 WarningIndicatorRequested logic

Note: The “WarningIndicatorOffCriteriaFulfilled” and the “WarningIndicatorOnCriteriaFulfilled” condition are specified in chapter 7.7.9.

Note: ISO-14229-1[2] additionally specifies “warning indicator not requested on due to latched failsafe for particular DTC” as condition. This has to be ensured by the monitor.

7.7.1.4 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 event status byte.

As the `UDS 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: 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.

7.7.1.5 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, 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 EventUdsStatusChanged on each UDS 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 DemCallbackEventUdsStatusChanged (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 7.10.

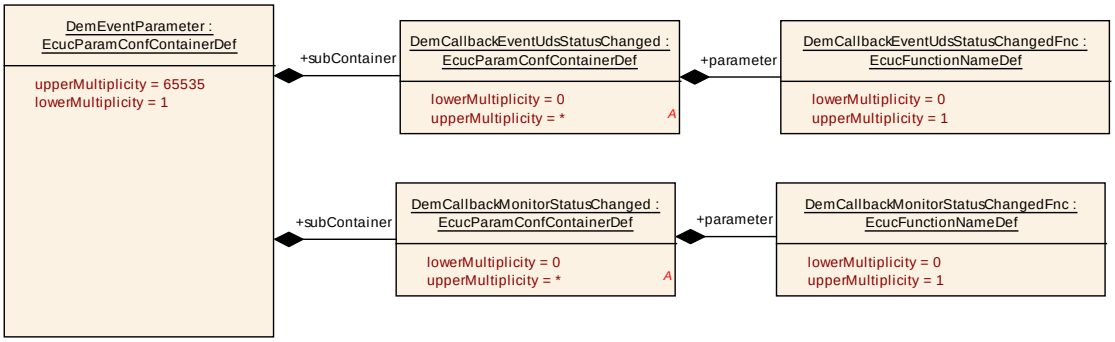


Figure 7.25: EventStatusChanged callback configuration

[SWS_Dem_00284] ⌈ The Dem module shall trigger the callback function configured in DemCallbackDTCStatusChanged on every DemDTC status change. ⌋ *(SRS_BSW_00457)*

[SWS_Dem_00986] ⌈ The Dem module shall trigger the callback function configured in DemCallbackOBDDTCStatusChanged on every DemObdDTC status change. ⌋ *(SRS_BSW_00457)*

[SWS_Dem_00987] ⌈ The Dem module shall trigger the callback function configured in DemCallbackJ1939DTCStatusChanged on every J1939 DTC status change. ⌋ *(SRS_BSW_00457)*

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 EventUdsStatusChanged may only be different to the result of DTCStatusChanged in case of event combination (refer to chapter 7.7.5).

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.

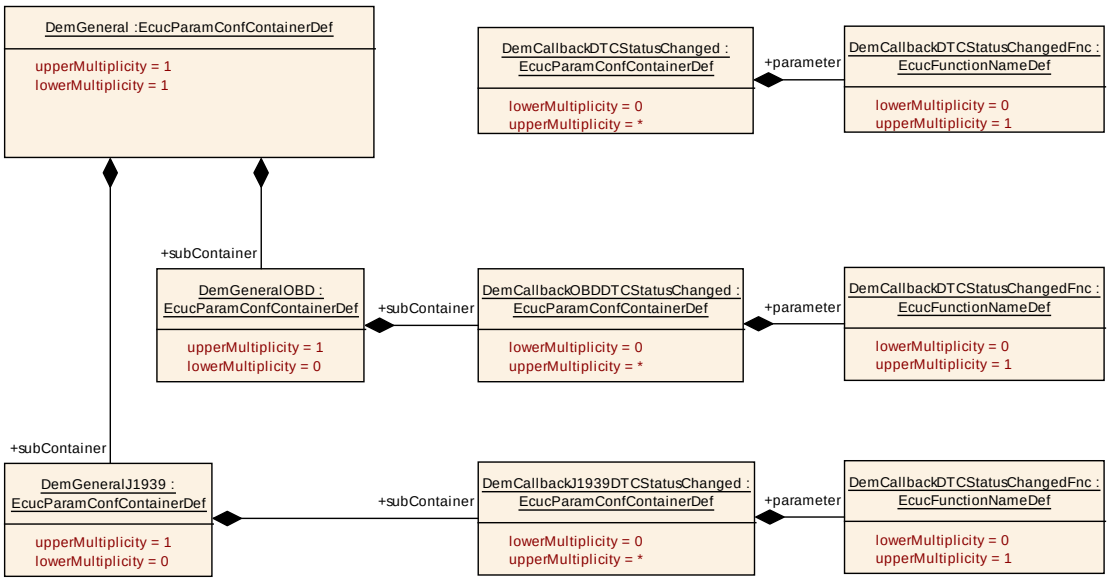


Figure 7.26: DTCStatusChanged callback configuration

7.7.2 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.

[SWS_Dem_01199] ⌈ The Dem module shall support the configuration parameter DemMemoryDestinationRef to define the destination memory/memories for individual DTCs. When using the Dem interfaces, Dem_DTCOriginType selects the destination memories. ⌋*(SRS_Diag_04066)*

Note: OBD relevant DTCs can be stored in primary or user defined event memory.

[SWS_Dem_01207] ⌈ The Dem module shall support the configuration parameter DemMemoryDestinationRef to define the destination memory/memories for individual DTCs. When using the Dem interfaces, Dem_DTCOriginType select the destination memories. ⌋*(SRS_Diag_04066)*

Note: OBD relevant DTCs can be stored in primary or user defined failure memory.

[SWS_Dem_01063] ⌈ The user defined memory shall have the same behavior as the primary memory(event retention, event prioritization, aging, displacement). ⌋ *(SRS_Diag_04131)*

Note: The mirror memory may have a different behaviour which is project specific and is not described in this document.

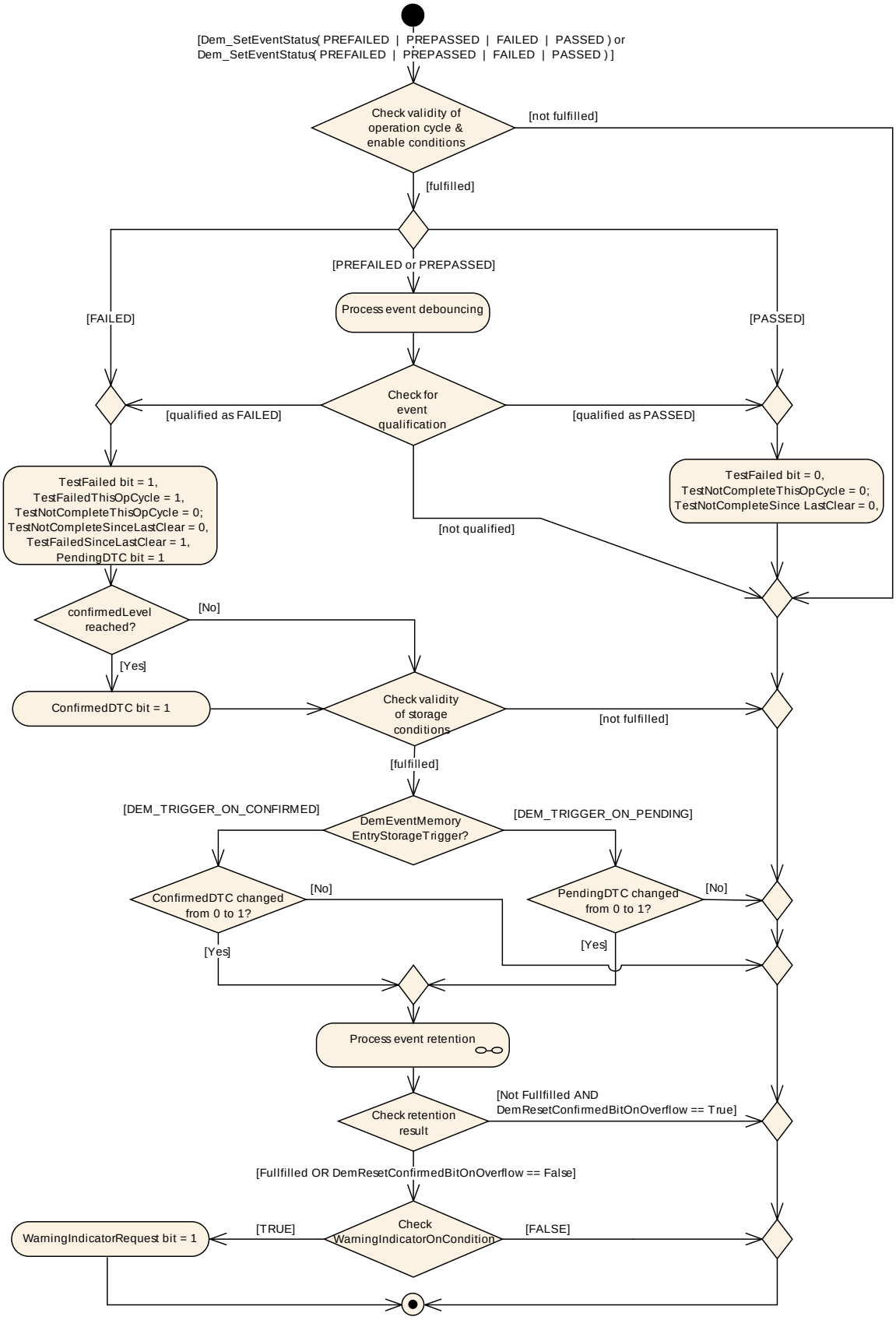


Figure 7.27: General diagnostic event storage processing

7.7.2.1 Event retention

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

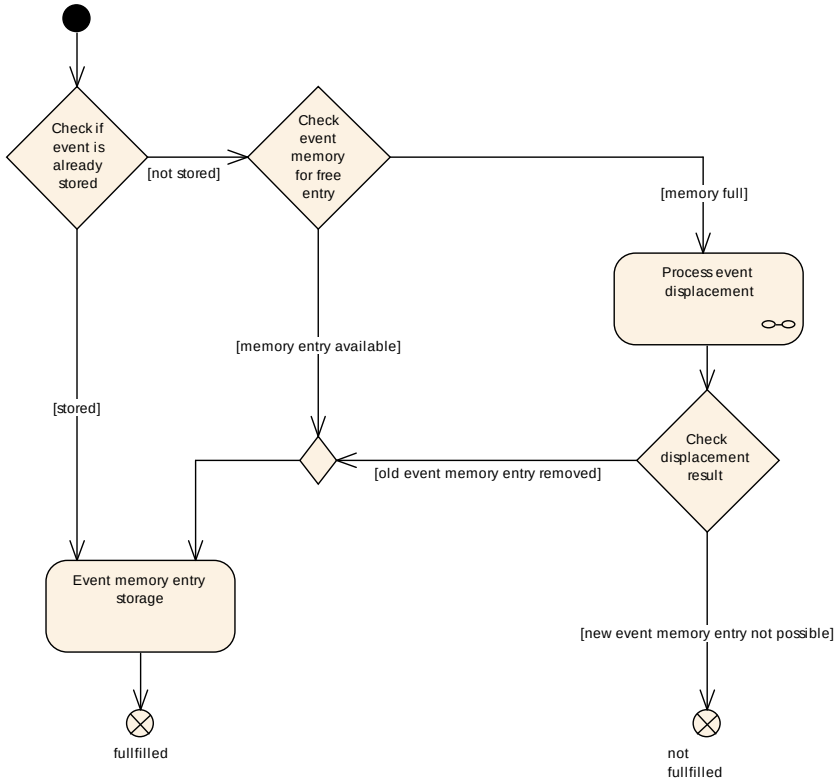


Figure 7.28: General diagnostic event retention processing

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 `event memory` (e.g. to have most recent update FreezeFrames).

[SWS_Dem_00783] ⌈ If an event

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

the Dem module shall try to allocate according figure 7.28 an `event memory entry` in its configured `event memory` (refer to `DemMemoryDestinationRef`). ⌋ *(SRS_Diag_04105)*

[SWS_Dem_00784] ⌈ If an event

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

the Dem module shall try to allocate according figure 7.28 an event memory entry in its configured event memory (refer to DemMemoryDestinationRef). ⌋ *(SRS_Diag_04105)*

[SWS_Dem_00922] ⌈ If an event

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

the Dem module shall try to allocate according figure 7.28 an event memory entry in its configured event memory (refer to DemMemoryDestinationRef). ⌋ *(SRS_Diag_04105)*

[SWS_Dem_00785] ⌈ If an event

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

the Dem module shall try to allocate according figure Figure 7.28 an event memory entry in its configured event memory (refer to DemMemoryDestinationRef). ⌋ *(SRS_Diag_04105)*

[SWS_Dem_00923] ⌈ If an event

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

the Dem module shall try to allocate according Figure 7.28 an event memory entry in its configured event memory (refer to DemMemoryDestinationRef). ⌋ *(SRS_Diag_04105)*

[SWS_Dem_00786] ⌈ If an event

1. uses Dem-internal debouncing (either counterbased or timebased) and
2. the Dem-internal debounce counter is incremented and reaches DemCounterBasedFdcThresholdStorageValue or DemTimeBasedFdcThresholdStorageValue and
3. the configuration parameter DemEventMemoryEntryStorageTrigger is set to DEM_TRIGGER_ON_FDC_THRESHOLD and
4. no event event memory exists,

the Dem module shall try to allocate according Figure 7.28 an event memory entry in its configured event memory (refer to DemMemoryDestinationRef). ⌋ *(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 7.7.7 “Event related data”).

[SWS_Dem_CONSTR_06155] Dependency on DemTimeBasedFdcThresholdStorageValue ⌈ DemTimeBasedFdcThresholdStorageValue shall only be present if DemFreezeFrameRecordTrigger is set to DEM_TRIGGER_ON_FDC_THRESHOLD or DemExtendedDataRecordTrigger is set to DEM_TRIGGER_ON_FDC_THRESHOLD or DemEventMemoryEntryStorageTrigger is set to DEM_TRIGGER_ON_FDC_THRESHOLD. ⌋*()*

7.7.2.2 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 Clear/Reset emission-related diagnostic information). Therefore, the Dem offers separate APIs for different users:

- Dem_ClearDTC
- Dem_J1939DcmClearDTC (for J1939 to the J1939Dcm [7], refer to chapter 7.10.2).

[SWS_Dem_01203] ⌈ If the Dem module is requested to clear diagnostic information and the selected DTC is set to DTC group ‘all DTCs’ and the selected DTCOrigin is set to DEM_DTC_ORIGIN_PRIMARY_MEMORY or DEM_DTC_ORIGIN_USERDEFINED_MEMORY_<NAME>, the Dem shall reset all event and DTC status bytes and clear event related data assigned to the requested DTCOrigin. ⌋*(SRS_Diag_04150, SRS_Diag_04214)*

[SWS_Dem_01205] ⌈ If the Dem module is requested to clear diagnostic information and the selected DTC is set to DTC group 'all DTCs' and the selected DTCOrigin is set to DEM_DTC_ORIGIN_MIRROR_MEMORY, the Dem shall clear only the event entries assigned to the mirror event memory. It does not affect UDS status. ⌋(*SRS_Diag_04213*)

[SWS_Dem_01206] ⌈ The Dem shall call the callback DemClearEventAllowed only if the selected DTCOrigin for the clear operation is the primary or user defined event memory. ⌋(*SRS_Diag_04150, SRS_Diag_04213*)

The clearing process can be started by different clients. Whenever one of these interfaces starts 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_01240] ⌈ If DemClearDtcNotificationTime is configured to START, the Dem shall inform the application about the start of a clear DTC operation. If the DemClearDtcNotificationFnc is configured, the Dem shall shall call this configured function otherwise the Dem shall call the R-Port ClearDtcNotification. ⌋ (*SRS_Diag_04185*)

[SWS_Dem_01241] ⌈ If DemClearDtcNotificationTime is configured to FINISH, the Dem shall inform the application about the end of a clear DTC operation. The Dem shall consider the configuration parameter DemClearDTCBehavior which defines the clearing process of diagnostic information including clearing completion. If the DemClearDtcNotificationFnc is configured, the Dem shall call this configured function otherwise the Dem shall call the R-Port ClearDtcNotification. ⌋(*SRS_Diag_04185*)

[SWS_Dem_00661] ⌈ The first client requesting call to clear diagnostic information shall lock the clearing process for all other clients. ⌋(*SRS_Diag_04065, SRS_Diag_04194*)

[SWS_Dem_00664] ⌈ If the clearing process is locked, the Dem shall return DEM_CLEAR_BUSY for any subsequent call of Dem_ClearDTC from a different client. ⌋(*SRS_Diag_04065, SRS_Diag_04194*)

[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_04167*)

[SWS_Dem_00670] ⌈ If the configuration parameter DemClearDTCLimitation is set to DEM_ONLY_CLEAR_ALL_DTCS, the APIs Dem_ClearDTC are called with a selected DTC group different to DEM_DTC_GROUP_ALL_DTCS the Dem shall return DEM_WRONG_DTC. ⌋(*SRS_Diag_04117*)

[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 E_OK after the volatile memory is cleared. ⌋(*SRS_Diag_04065, SRS_Diag_04194*)

[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 E_OK after

the volatile memory is cleared and clearing of the non-volatile memory is triggered. ⌋ *(SRS_Diag_04065, SRS_Diag_04194)*

[SWS_Dem_00572] ⌈ 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 E_OK after the volatile memory and the non-volatile memory is cleared. ⌋*(SRS_Diag_04065, SRS_Diag_04194)*

Note: The Dem implementation is responsible for consistency of the persistant data. (See SWS_NvM_00698: Contents of RAM buffer passed to NvM_WriteBlock may not change while the NvM processes the job). How to ensure data consistency (UDS status byte, freeze frame data and all other related information) is implementation specific (depends on the implemented NvM strategy). Options include, but are not restricted to, additional RAM mirrors and queuing of changes until the RAM area can be modified again.

Note: If the Dcm module receives the return type DEM_CLEAR_OK of the API Dem_ClearDTC, the Dcm module sends the positive response (refer to [10]).

[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 if the clearing of the non-volatile memory fails. ⌋ *(SRS_Diag_04065)*

[SWS_Dem_01202] Defintion of a failed clear DTC operation ⌈ If the Dem module is requested to clear diagnostic information, the Dem shall return DEM_CLEAR_FAILED:

- in case of clearing a single DTC at least one DemClearEventAllowed callback assigned to the event of the DTC returns FALSE.
- in case of clearing a DTC group or all DTCs, the individual DTCs shall be cleared like described in clearing single DTC. The resulting return value shall be DEM_CLEAR_FAILED if the clearing of all individual DTCs returned DEM_CLEAR_FAILED.

⌋*(SRS_Diag_04150, SRS_Diag_04214)*

[SWS_Dem_00573] ⌈ If DemTriggerMonitorInitBeforeClearOk is set to TRUE, the Dem_ClearDTC shall not return E_OK until the events intialization is triggered (callback function InitMonitorForEvent is called). If DemTriggerMonitorInitBeforeClearOk is set to FALSE, DEM_CLEAR_OK is not considering Event initialization (InitMonitorForEvent). ⌋*(SRS_Diag_04065, SRS_Diag_04194)*

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 DemClearEventAllowed for each event it is configured for, before clearing this event. ⌋ *(SRS_Diag_04191, SRS_Diag_04194)*

[SWS_Dem_00667] ⌈ If the DTC is not configured to use event combination, the out-parameter Allowed of the callback DemClearEventAllowed returns "false" and the return value is equal to E_OK, the Dem shall not clear the respective event memory entry and Dem-internal data values. ⌋*(SRS_Diag_04191, SRS_Diag_04194)*

[SWS_Dem_01295] ⌈ If the DTC is configured to use event combination, if only one DemClearEventAllowed callback has the out-parameter Allowed set to FALSE and returns E_OK, the Dem shall not clear the event status, respective event memory entry and Dem-internal data values. ⌋*(SRS_Diag_04191, SRS_Diag_04194, SRS_Diag_04073)*

[SWS_Dem_00668] ⌈ If the out-parameter 'Allowed' of the callback DemClearEventAllowed returns "false" and the configuration parameter DemClearEventAllowedBehavior is set to DEM_NO_STATUS_BYTE_CHANGE, the related UDS status byte shall not be modified. ⌋*(SRS_Diag_04191, SRS_Diag_04194)*

[SWS_Dem_00669] ⌈ If the out-parameter Allowed of the callback DemClearEventAllowed returns "false" and the configuration parameter DemClearEventAllowedBehavior is set to DEM_ONLY_THIS_CYCLE_AND_READINESS, the related UDS status bits 1 (TestFailedThisOperationCycle), 4 (TestNotCompletedSinceLastClear), 5 (TestFailedSinceLastClear), and 6 (TestNotCompletedThisOperationCycle) shall be reset. ⌋*(SRS_Diag_04191, SRS_Diag_04194)*

[SWS_Dem_00516] ⌈ If the return value of the callback DemClearEventAllowed is equal to E_NOT_OK, the return value shall not be changed and event-deletion is allowed. ⌋*(SRS_Diag_04191, SRS_Diag_04194)*

The Dem supports events that are not assigned to any event memory. This is the case, if the optional parameter DemDTCRef is not configured in DemEventParameter. Events are cleared, when the event memory assigned to them is cleared. In case there is no assigned event memory, these events would never be cleared. For this case the Dem provides the configuration parameter DemClearEventsWithoutDTCEventMemoryRef.

[SWS_Dem_01249] ⌈ If the event memory referenced by DemClearEventsWithoutDTCEventMemoryRef is cleared via any of the Dem_ClearDTC APIs, the Dem shall clear also all the events that are not assigned to any event memory. ⌋*()*

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

7.7.2.3 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 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 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).

7.7.2.4 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 7.29):

1. Priority (refer [SWS_Dem_00403])
2. Active/Passive status (configurable, only for DEM_DISPLACEMENT_FULL) (refer [SWS_Dem_00404])
3. Occurrence (refer [SWS_Dem_00405])

⌋*(SRS_Diag_04118)*

Note: Priority needs to have Fuel system/Misfire over the rest (US/EU5) and Class A before B1,B2, C (Euro VI).

[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 7.7.1.4). ⌋ *(SRS_Diag_04118)*

[SWS_Dem_00405] ⌈ If displacement strategy is DEM_DISPLACEMENT_FULL or DEM_DISPLACEMENT_PRIO_OCC, the Dem module shall displace older stored events in the chronological order prior to newer stored events (refer [SWS_Dem_00412] and [SWS_Dem_00787]). ⌋*(SRS_Diag_04118)*

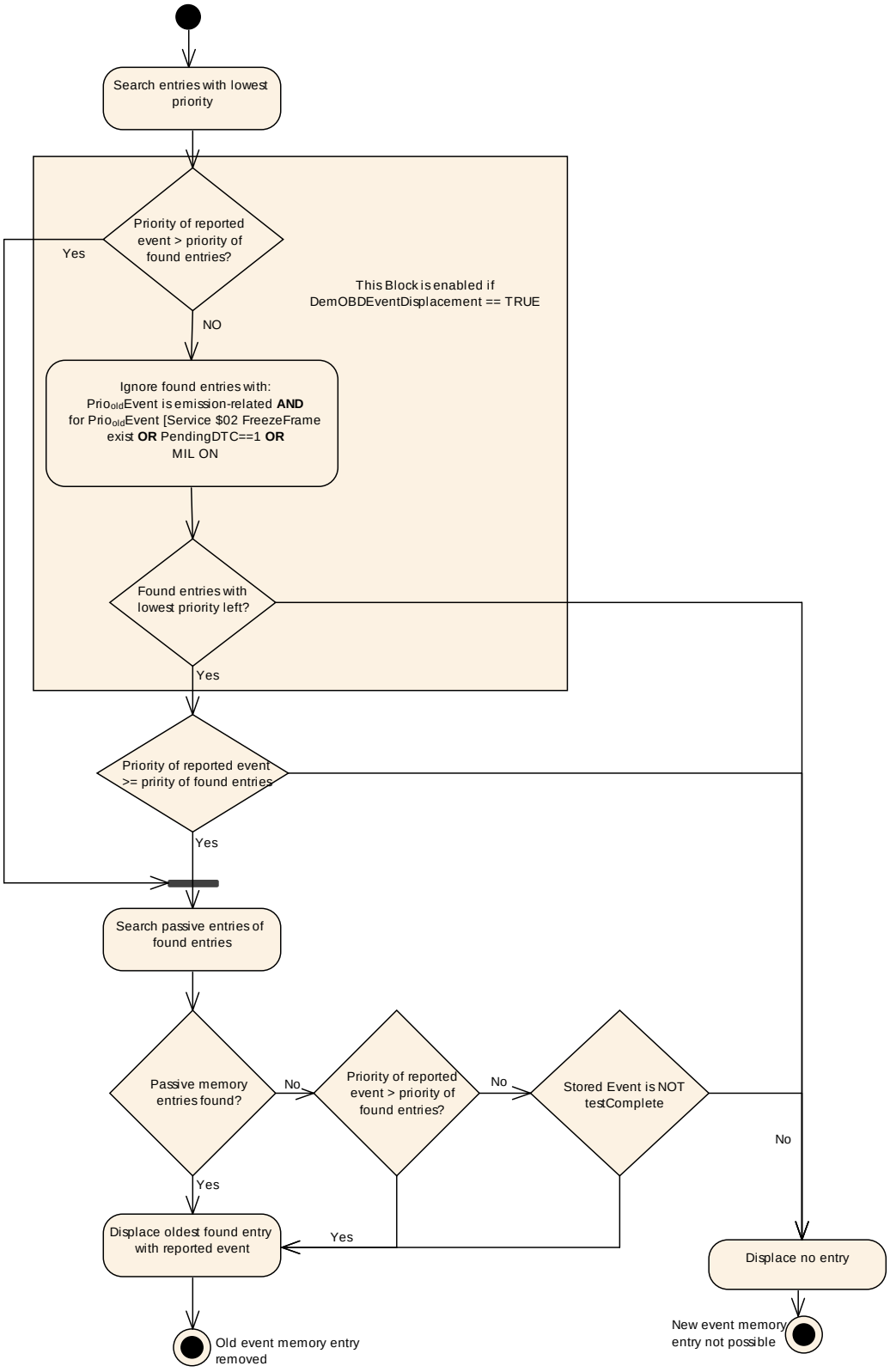


Figure 7.29: Combined displacement criteria processing

[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 is reported and an event memory entry with the same priority exists and the existing event is not tested in this operation cycle (TestNotCompletedThisOperationCycle bit == 1), the Dem shall displace the existing event with the reported event. ⌋*(SRS_Diag_04118, SRS_Diag_04071)*

[SWS_Dem_00693] ⌈ If an event is reported and an event memory entry with the same priority exists and the existing event is tested in this operation cycle (TestNotCompletedThisOperationCycle bit == 0), the Dem shall discard the reported event. ⌋*(SRS_Diag_04178)*

[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:
Event*old* is emission related
AND
(Event*old* triggers MIL indicator
OR (Event*old* holds legislative Freeze Frame **AND** has equal or higher priority)
OR Pending-DTC==1) ⌋*(SRS_Diag_04195)*

[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 7.29). ⌋*(SRS_Diag_04195)*

[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 and UDS status bit 3 to 0 if the configuration parameter DemResetConfirmedBitOnOverflow (refer to DemGeneral) is set to true. ⌋*(SRS_Diag_04067, SRS_Diag_04118)*

[SWS_Dem_01186] ⌈ If an event memory entry was removed during displacement, the UDS status bit 5 shall be reset to 0 in case of DemStatusBitHandlingTestFailedSinceLastClear is set to DEM_STATUS_BIT_AGING_AND_DISPLACEMENT and DemResetConfirmedBitOnOverflow is set to true. ⌋*(SRS_Diag_04067, SRS_Diag_04118)*

7.7.2.5 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_GetNextFilteredDTC), if:

- the DTCStatus parameter has the“pending DTC” or “confirmed DTC” bit or both bits set

- all other bits of the DTCStatus parameter are set to false
- DemResetConfirmedBitOnOverflow is set to true

⌋(*SRS_Diag_04195*)

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 UDS status mask parameters, additional resources may be necessary.

[SWS_Dem_00787] ⌈ If an 'stored event' gets re-qualified as failed (UDS 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 by setting the particular event as most recent event memory entry. ⌋(*SRS_Diag_04195*)

[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_04195*)

[SWS_Dem_00412] ⌈ If a new event get stored in the event memory, the Dem shall update the chronological order by setting the new event as most recent event memory entry. ⌋(*SRS_Diag_04195, SRS_Diag_04195*)

Note: If there are multiple 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 entries is reported or displaced first..

7.7.3 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 7.2).

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.

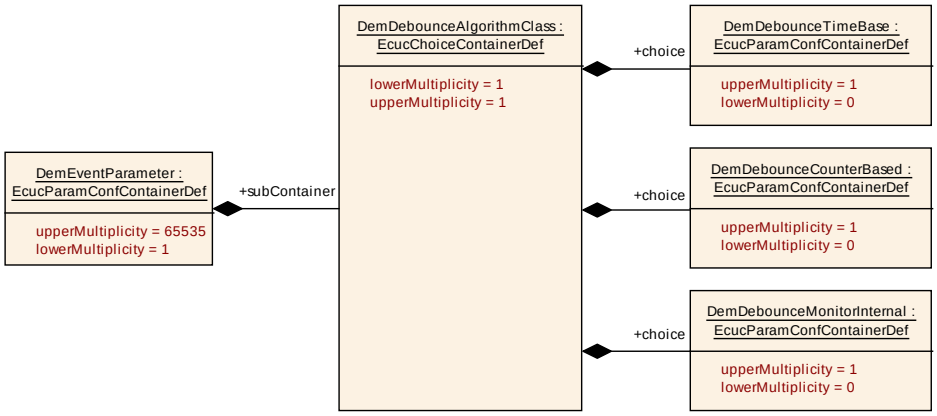


Figure 7.30: Event-specific debounce algorithms

7.7.3.1 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 internal `debounce counter` is defined as sint16. This does not imply any explicit implementation.

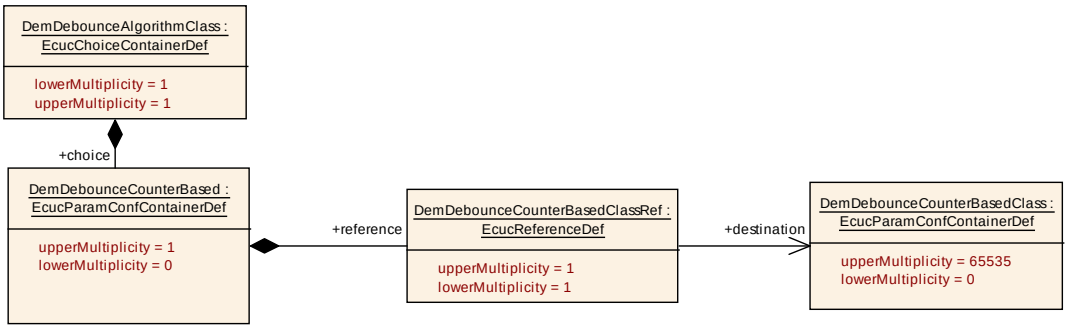


Figure 7.31: Counter based debounce algorithm

[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. ⌋ *(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. ⌋ *(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_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_04067, SRS_Diag_04068)*

[SWS_Dem_00422] ⌈ The Dem module shall provide the configuration parameter DemDebounceCounterJumpDown for enabling the jump-down behavior. ⌋ *(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_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_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 DemDebounceCounter-

JumpUpValue before performing the pre-failed debounce event ([SWS_Dem_00418]).
⌋*(SRS_Diag_04068)*

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

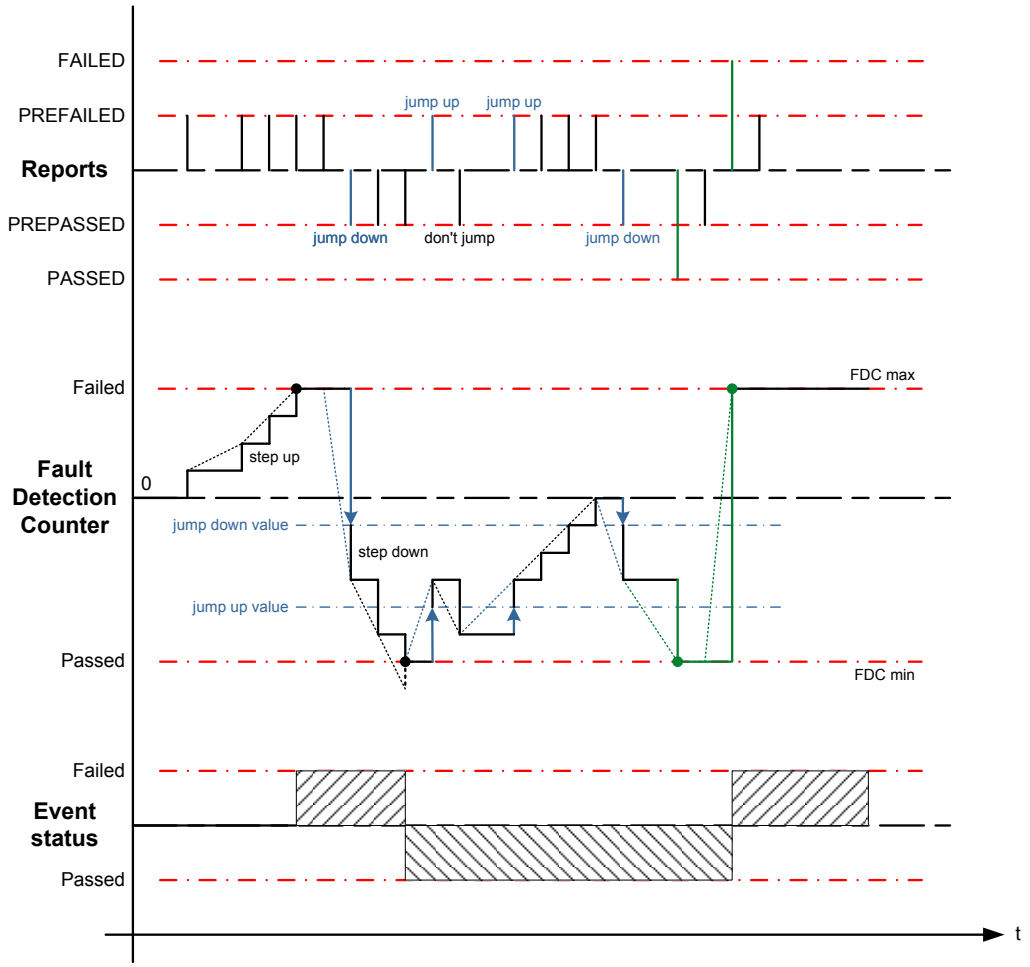


Figure 7.32: Example of counter based debouncing (including jump behaviour)

Note: In figure 7.32, 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_STATUS_FREEZE` is covered in chapter 7.7.6. The `DemDebounceBehavior` `DEM_DEBOUNCE_RESET` is covered by [SWS_Dem_00654] and [SWS_Dem_00677]. This is shown in figure 7.33.

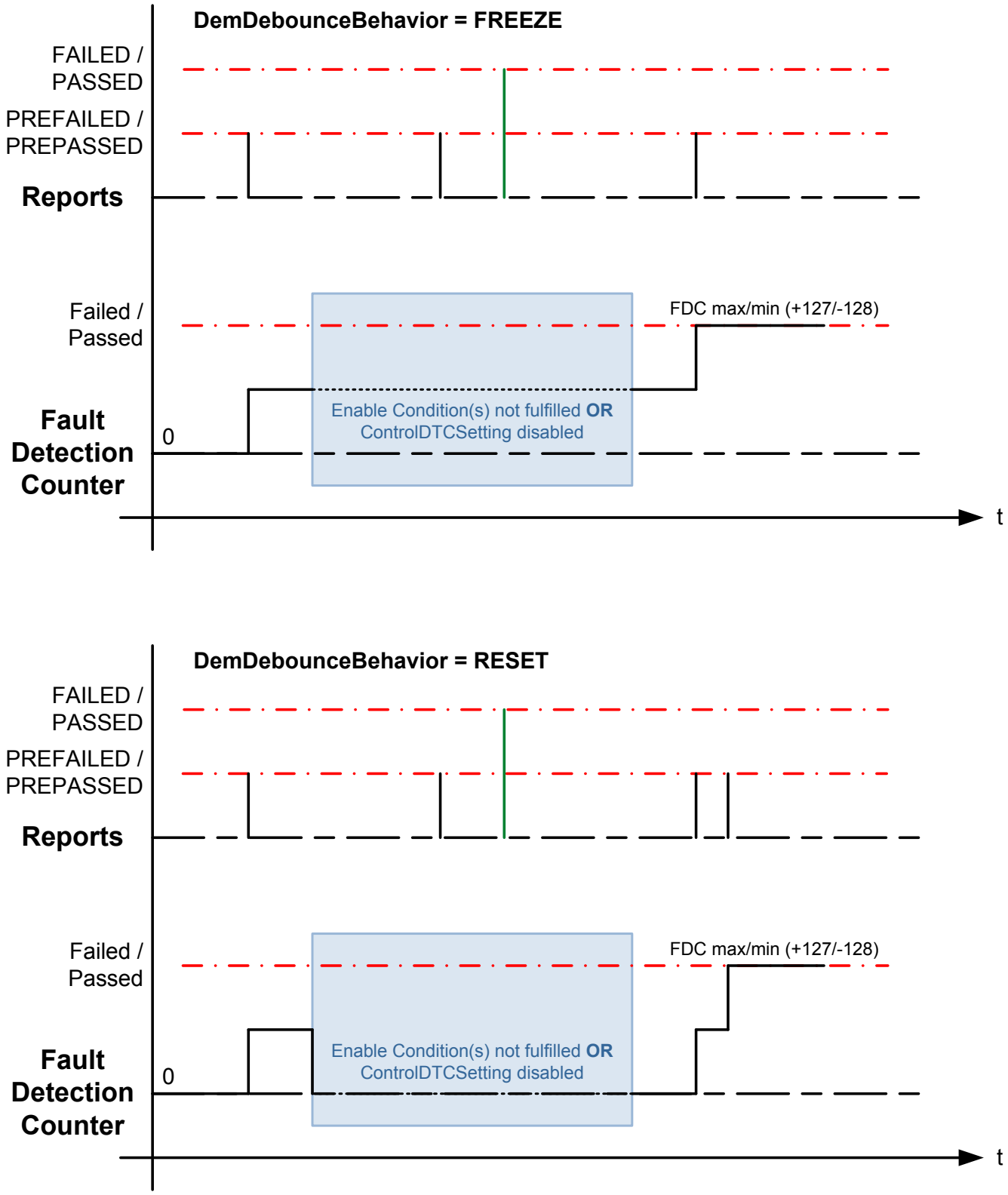


Figure 7.33: Counter based debouncing behavior on enable conditions

[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_SetEventStatus 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)*

[SWS_Dem_CONSTR_06151] Dependency on DemCounterBasedFdcThresholdStorageValue ⌈ The configuration parameter DemCounterBasedFdcThresholdStorageValue shall only be present if DemFreezeFrameRecordTrigger is set to DEM_TRIGGER_ON_FDC_THRESHOLD or DemExtendedDataRecordTrigger is set to DEM_TRIGGER_ON_FDC_THRESHOLD or DemEventMemoryEntryStorageTrigger is set to DEM_TRIGGER_ON_FDC_THRESHOLD. ⌋*()*

[SWS_Dem_CONSTR_06152] Dependency on DemDebounceCounterJumpDownValue ⌈ DemDebounceCounterJumpDownValue shall only be present if DemDebounceCounterJumpDown is set to TRUE. ⌋*()*

[SWS_Dem_CONSTR_06153] Dependency on DemDebounceCounterJumpUpValue ⌈ DemDebounceCounterJumpUpValue shall only be present if DemDebounceCounterJumpUp is set to TRUE. ⌋*()*

[SWS_Dem_CONSTR_06154] Dependency on DemDebounceCounterStorage ⌈ DemDebounceCounterStorage shall only be present if DemOperationCycleStatusStorage is set to TRUE. ⌋*()*

7.7.3.2 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.

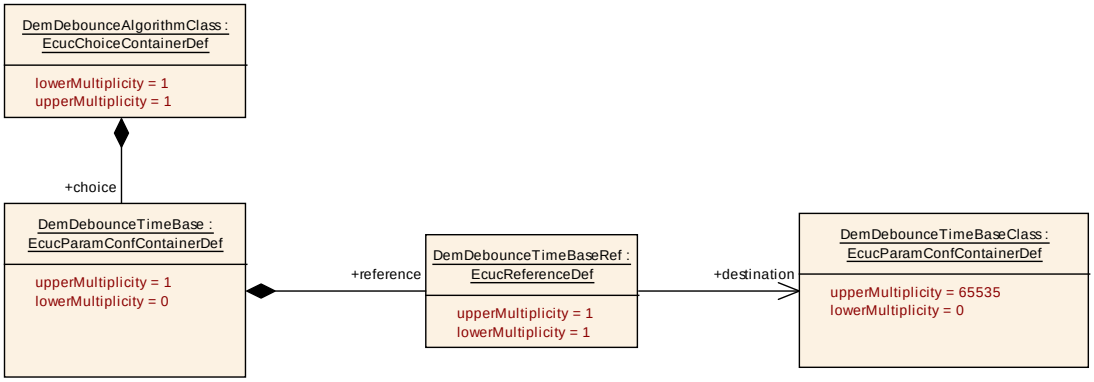


Figure 7.34: Time based debounce algorithm

[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 7.35). ⌋*(SRS_Diag_04068)*

[SWS_Dem_00428] ⌈ The Dem module shall start the internal debounce timer to qualify the reported event as failed when the monitor reports DEM_EVENT_STATUS_PREFAILED. ⌋*(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)*

Note: The Dem module provides the configuration parameter DemDebounceTimeFailedThreshold used to define the event-specific delay indicating the failed status (active).

[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_04067, 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. ⌋*(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_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_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 thresh-

old being the passed criteria (refer to DemDebounceTimePassedThreshold). ⌋ *(SRS_Diag_04068)*

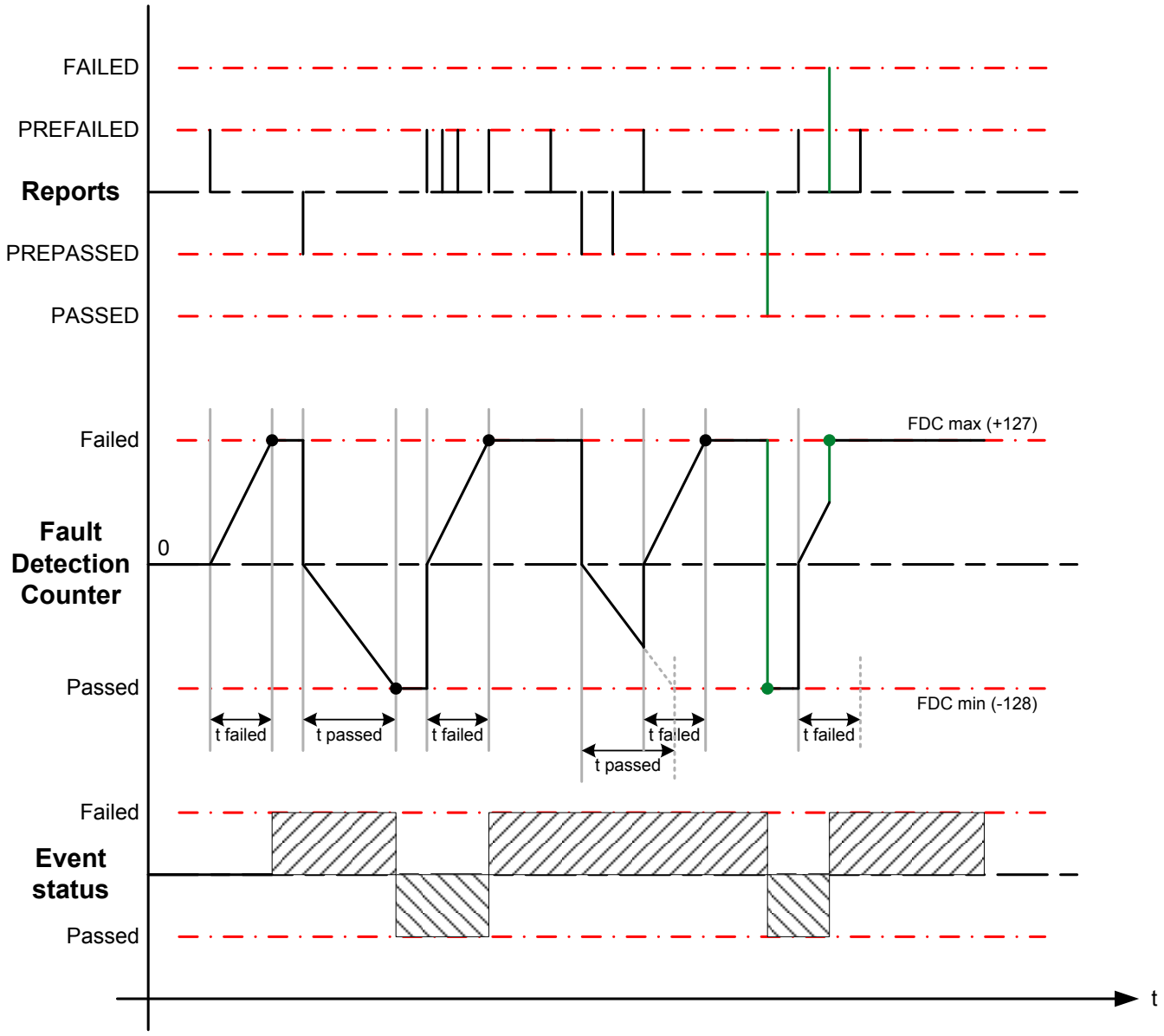


Figure 7.35: Example of time based debouncing

[SWS_Dem_00685] ⌈ If the API Dem_ResetEventDebounceStatus is called with DEM_DEBOUNCE_STATUS_FREEZE, it shall freeze the related internal debounce timer. ⌋*(SRS_Diag_04068)*

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_STATUS_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_STATUS_FREEZE, the Dem module shall freeze the internal debounce 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].

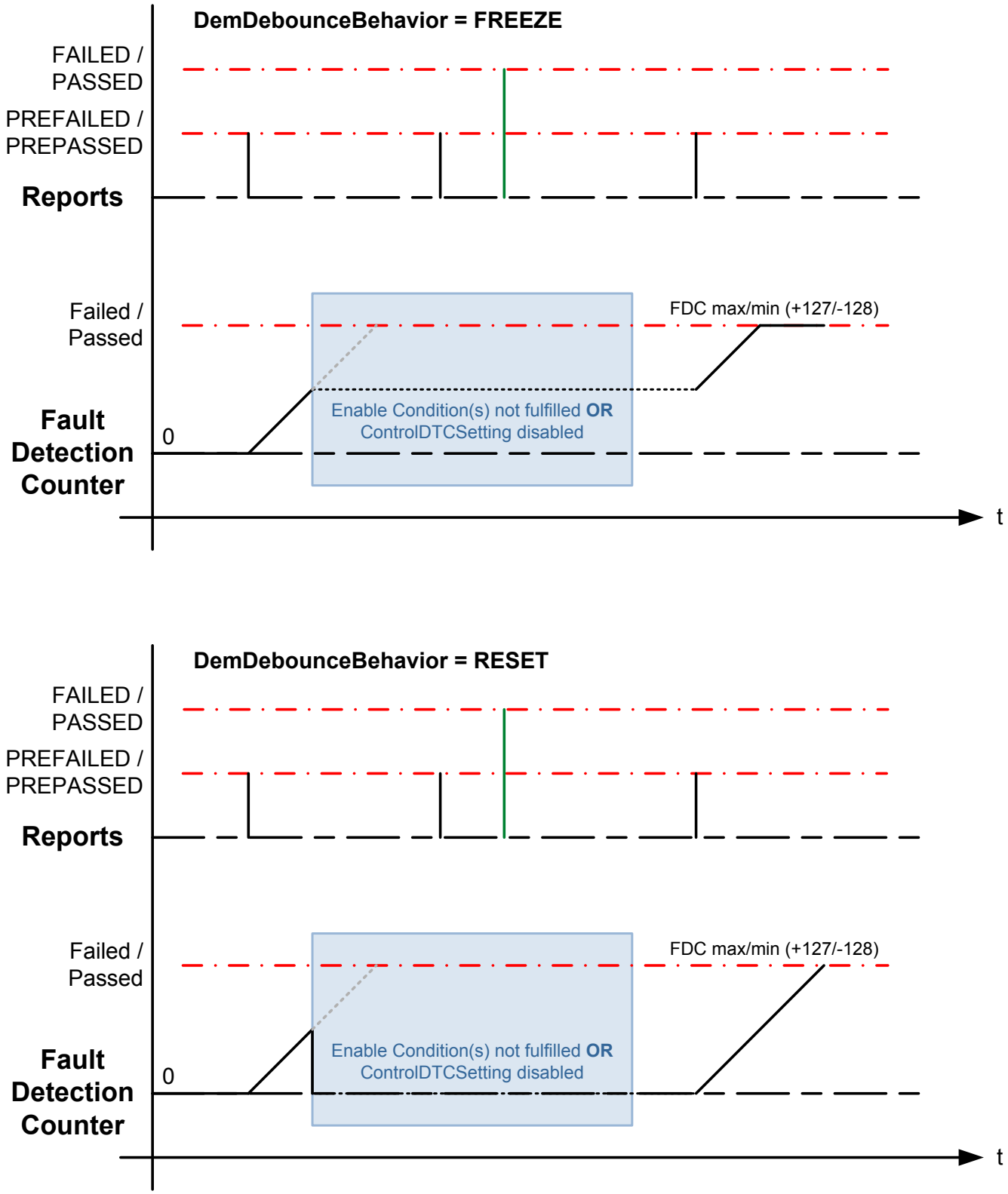


Figure 7.36: Time based debouncing behavior on enable conditions

7.7.3.3 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.

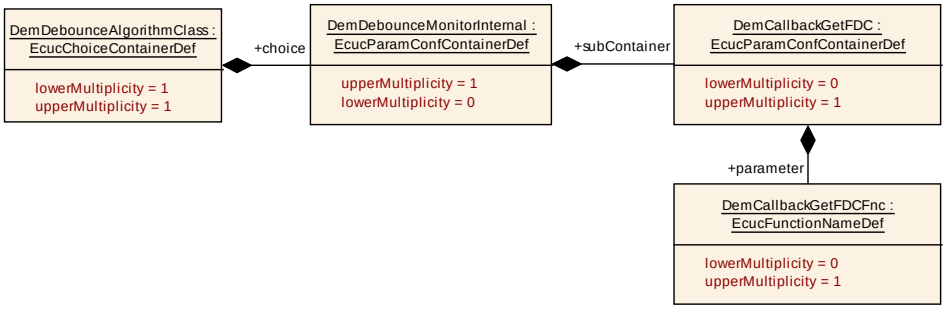


Figure 7.37: Monitor internal debounce algorithm

7.7.3.4 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_04068)*

[SWS_Dem_00343] ⌈ After receiving a command for clearing the event memory (refer to chapter 7.7.2.2), 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 is called with DEM_DEBOUNCE_STATUS_RESET, it shall reset the related fault detection counter. ⌋ *(SRS_Diag_04068)*

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 7.6). ⌋*(SRS_Diag_04068)*

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 chap-

ter 7.7.6) 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_DisableDTCSetting` (refer to chapter 7.11.2.6) shall reset the related fault detection counter(s).⌋*(SRS_Diag_04125)*

7.7.3.5 Fault detection counter retrieval

[SWS_Dem_00204] ⌈ The event-specific fault detection counter shall be accessible by using the API `Dem_GetFaultDetectionCounter`. ⌋*(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 7.7.3.1 and chapter 7.7.3.2).
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 retrieve the current value of the fault detection counter for the requested event by using the method defined in `DemCallbackGetFDC` container. ⌋*(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 monitor internal debouncing is used and the container `DemCallbackGetFDC` is configured and the configured callback-function 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 7.2).

[SWS_Dem_00671] ⌈ If monitor internal debouncing is used and the container `DemCallbackGetFDC` is not configured for a given event, the `API Dem_GetFaultDetectionCounter` shall return DEM_E_NO_FDC_AVAILABLE. ⌋ *(SRS_Diag_04010)*

7.7.3.6 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_00792] 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)*

7.7.4 Fault confirmation

After reporting, a fault and entering the pending status, the fault confirmation process begins within the Dem (refer to Figure 7.27 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 DemEventConfirmationThreshold per event defining the maximum number of tested and failed cycles, before the event becomes “confirmed”, i.e. enters the confirmed state. ⌋*()*

[SWS_Dem_00529] ⌈ The configuration parameter DemOperationCycleRef (refer to DemEventParameter) shall be used for calculating the trip counter according ISO14229-1[2] (see figure 7.20). ⌋*(SRS_Diag_04067)*

The healing confirmation process (described in chapter 7.7.8) 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 status bit 3 is set during the qualification to failed (according to [SWS_Dem_00391]).

[SWS_Dem_01242] ⌈ The Dem shall support calls of Dem_SetEventFailureCycleCounterThreshold after Dem_PreInit. ⌋ *(SRS_Diag_04161)*

[SWS_Dem_01243] ⌈ A call of the API Dem_SetEventFailureCycleCounterThreshold shall set a new failure confirmation threshold. ⌋*(SRS_Diag_04161)*

Note: A new failure confirmation threshold value will overwrite the default value given by DemEventConfirmationThreshold).

[SWS_Dem_01244] ⌈ If Dem_SetEventFailureCycleCounterThreshold is called and the corresponding event is already stored in event memory and the new failure threshold is less or equal to the current failure counter, then the event will get confirmed. This implies, that the event related data will also be updated according their defined triggers. ⌋*(SRS_Diag_04161)*

[SWS_Dem_01245] ⌈ If Dem_SetEventFailureCycleCounterThreshold is called and the corresponding event is already stored in event memory and already confirmed, the event memory entry and confirmation status will remain, even if the new threshold is greater than the failure counter. Also after new error status reports, the confirmation status will remain. ⌋*(SRS_Diag_04161)*

[SWS_Dem_01246] ⌈ A DemIndicatorAttribute with a configured DemMILIndicatorRef, shall use the DemEventConfirmationThreshold instead of the DemIndicatorFailureCycleCounterThreshold to calculate the indicator On Criteria (WarningIndicatorOnCriteriaFulfilled). ⌋*(SRS_Diag_04161)*

7.7.4.1 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. ⌋*(SRS_Diag_04001)*

Note: One event is only allowed to be referenced to only one group of associate events. A referenced 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_04178)*

7.7.5 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 DemEventCombination-Support 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 DemEventCombination-Support 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

⌋*(SRS_Diag_04073)*

[SWS_Dem_00024] ⌈ In case DemEventCombinationSupport is set to DEM_EVCOMB_ONRETRIEVAL or DEM_EVCOMB_ONSTORAGE, the combination of multiple events to a DTC is defined 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	*Monitor status*	*UDS 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	S4|S5|S6	DTC[2]	FF[74]	
Event[5]	S5	S4|S5|S6	DTC[2]	FF[77]	
Event[6]	S6	S4|S5|S6	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 UDS status byte according to Table 7.2. ⌋*(SRS_Diag_04073)*

UDS status bit description		*Combined UDS 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	$CbDTC_{Bit4}$ = ($Event[1]_{Bit4}$ | $Event[2]_{Bit4}$ | ... |$Event[n]_{Bit4}$) & ! $CbDTC_{Bit5}$
5	TestFailedSince LastClear	$CbDTC_{Bit5}$ = $Event[1]_{Bit5}$ | $Event[2]_{Bit5}$ | ... |$Event[n]_{Bit5}$
6	TestNotCompleted ThisOpera-tionCycle	$CbDTC_{Bit6}$ = ($Event[1]_{Bit6}$ | $Event[2]_{Bit6}$ | ... |$Event[n]_{Bit6}$) & ! $CbDTC_{Bit1}$CbDTC
7	WarningIndicator Requested	$CbDTC_{Bit7}$ = $Event[1]_{Bit7}$ | $Event[2]_{Bit7}$ | ... |$Event[n]_{Bit7}$

Table 7.2: Calculation of UDS 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]) ⌋*(SRS_Diag_04073)*

[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]. ⌋*(SRS_Diag_04073)*

[SWS_Dem_00672] ⌈ The fault detection counter of the combined DTC shall be the maximum fault detection counter value of the sub-events. ⌋*(SRS_Diag_04073)*

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

[SWS_Dem_CONSTR_6103] ⌈ In case the event combination is disabled, it is not allowed to reference from multiple events to the same dtc. ⌋*()*

7.7.5.1 Combination On Storage

The following section describes the behavior of the Dem module in case 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). ⌋*(SRS_Diag_04073)*

[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. ⌋*(SRS_Diag_04073)*

[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]. ⌋*(SRS_Diag_04073)*

[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]) ⌋*(SRS_Diag_04073)*

[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, SRS_Diag_04133)*

[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] and [SWS_Dem_01186]. ⌋ *(SRS_Diag_04118)*

7.7.5.2 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_04067)*

[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. ⌋*(SRS_Diag_04073)*

[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]). ⌋*(SRS_Diag_04073)*

7.7.6 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_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 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 Dem_SetEventStatus). ⌋*(SRS_Diag_04095)*

[SWS_Dem_00449] ⌈ If one enable condition is not fulfilled, all of Dem_SetEventStatus calls for those events being assigned to this condition shall be ignored (no change of event 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 and BSW modules via Dem_SetEventStatus for those events being assigned to these conditions shall be accepted by the Dem from this point in time on. ⌋*(SRS_Diag_04095)*

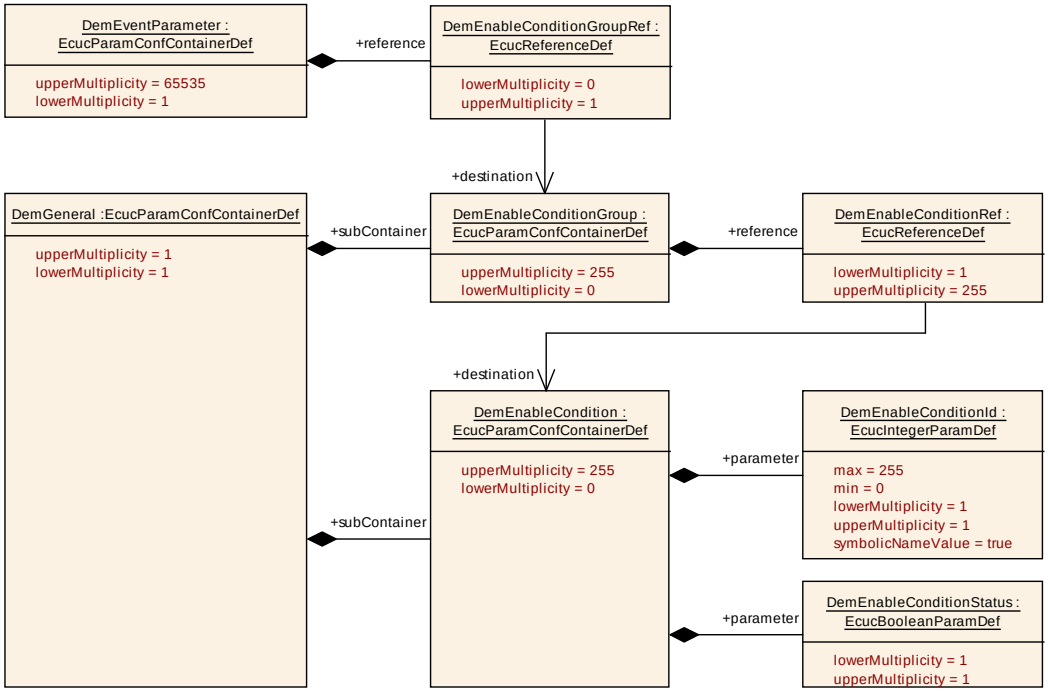


Figure 7.38: Enable condition assignment configuration

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` 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 status bit 0` changes from 0 to 1). ⌋*(SRS_Diag_04095)*

[SWS_Dem_01210] ⌈ If the `Dem` module is requested to support storage conditions, the `Dem` module shall check the assigned storage conditions after the event reports `DEM_EVENT_STATUS_FDC_THRESHOLD_REACHED` (e.g. either via interface call or when reaching the configured threshold in debounce algorithm). ⌋*(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)*

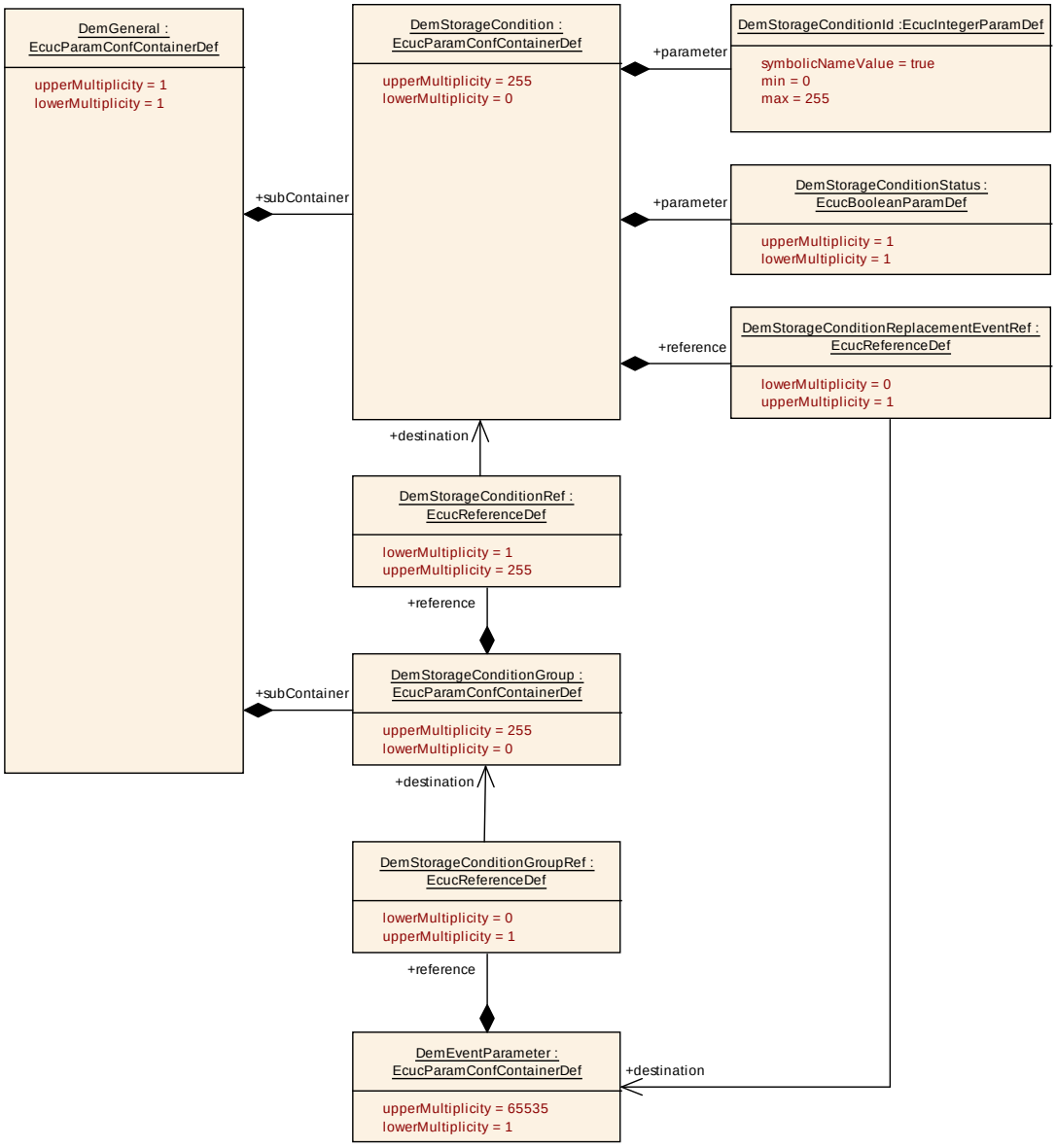


Figure 7.39: Storage condition assignment configuration

[SWS_Dem_01085] ⌈ For each storage condition a replacement event may be defined (see `DemStorageConditionReplacementEventRef`). This replacement event will be used as a replacement failure info. ⌋*(SRS_Diag_04137)*

[SWS_Dem_01086] ⌈ When a storage condition is deactivated, the replacement event shall be set (status FAILED) as soon as a failure is reported and filtered by the storage condition (storage condition not fulfilled). Note: As long as the storage condition is not

fulfilled and no event assigned to the storage condition reports FAILED, the replacement event shall not be set. ⌋(*SRS_Diag_04137*)

[SWS_Dem_01087] ⌈ The replacement error has an EventStatus (`Dem_UdsStatusByteType`) as every other event. ⌋(*SRS_Diag_04137*)

[SWS_Dem_01088] ⌈ The status/behavior of the filtered event (filtered by the storage-condition) is not affected by the replacement event (meaning: the failure which is filtered by the storage condition behaves identical in case of having a replacement event or in case of NOT having a replacement event configured for the storage condition). ⌋ (*SRS_Diag_04137*)

[SWS_Dem_01089] ⌈ The replacement event is reset (status PASSED), when the storage condition is fulfilled (on enabling the storage condition). ⌋(*SRS_Diag_04137*)

[SWS_Dem_01090] ⌈ constraints for the replacement event:

- must not be mapped to an storage condition
- must not be configured to use debouncing
- must not have an InitMForE callback/ clearEventAllowed
- must not be configured for enableConditions
- must be configured as `DEM_EVENT_KIND_BSW`

⌋(*SRS_Diag_04137*)

[SWS_Dem_01091] ⌈ Dem_SetEventStatus shall ignore the replacement event. ⌋ (*SRS_Diag_04137*)

7.7.7 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. ISO-14229-1[2] 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 7.7.7.4). ⌋(*SRS_Diag_04074*)

Note: The presence of a Confirmed UDS 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 7.7.7.4).

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.

7.7.7.1 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 `DemPidClass`).

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 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 DemCounterBasedFdcThresholdStorageValue or DemTimeBasedFdcThresholdStorageValue) is reached (by a positive increment), but at most once per operation cycle. ⌋*(SRS_Diag_04127)*

[SWS_Dem_01068] ⌈ If the DemFreezeFrameRecordTrigger is set to DEM_TRIGGER_ON_FDC_THRESHOLD and the event reports DEM_EVENT_STATUS_FDC_THRESHOLD_REACHED (monitor-internal debounced event), the FreezeFrame shall be captured (refer [SWS_Dem_00797]), but at most once per operation cycle. ⌋*(SRS_Diag_04127)*

[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 UDS 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 UDS 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 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 configuration parameter DemEnvironmentDataCapture (refer to DemGeneral) is set to DEM_CAPTURE_ASYNCHRONOUS_TO_REPORTING, event-specific freeze frame data shall be captured latest in the next cycle of the Dem_MainFunction. ⌋*(SRS_Diag_04127)*

[SWS_Dem_00805] ⌈ If the configuration parameter DemEnvironmentDataCapture is set to DEM_CAPTURE_SYNCHRONOUS_TO_REPORTING and DemDebounceTimeBasedSupport is set to 'false', event-specific freeze frame data shall be captured within the reporting function (i.e. in the context of Dem_SetEventStatus). ⌋ *(SRS_Diag_04127)*

Note: [SWS_Dem_00461] and [SWS_Dem_00805] does not apply if a pre-stored freeze frame is available, see also [SWS_Dem_00464].

[SWS_Dem_01081] ⌈ If the configuration parameter DemEnvironmentDataCapture is set to DEM_CAPTURE_ASYNCHRONOUS_TO_REPORTING, event-specific extended data shall be captured latest in the next cycle of the Dem_MainFunction. ⌋*(SRS_Diag_04127)*

[SWS_Dem_01082] ⌈ If the configuration parameter DemEnvironmentDataCapture is set to DEM_CAPTURE_SYNCHRONOUS_TO_REPORTING, event-specific extended data shall be captured within the context of Dem_SetEventStatus. ⌋ *(SRS_Diag_04127)*

If DemEnvironmentDataCapture is equal to DEM_CAPTURE_SYNCHRONOUS_TO_REPORTING, a queue could be used to store the environment data until the data is transferred to the event memory. To not loose any reporting, the queue size needs to be configured to store all reporting between two cycles of Dem_MainFunction.

[SWS_Dem_01083] ⌈ If DemEnvironmentDataCapture is equal to DEM_CAPTURE_SYNCHRONOUS_TO_REPORTING, the Dem module shall queue the environmental data from the reporting (see [SWS_Dem_00805] and [SWS_Dem_01082]) till the next call of the Dem_MainFunction. The Dem shall at minimum queue the configured size (refer DemMaxNumberEventEntryEventBuffer). In case the queue is full Dem_SetEventStatus shall return E_NOT_OK. ⌋*(SRS_Diag_04127)*

[SWS_Dem_CONSTR_06121] Dependency for DemMaxNumberEventEntryEventBuffer ⌈ The DemMaxNumberEventEntryEventBuffer shall only be present if DemEnvironmentDataCapture is set to DEM_CAPTURE_SYNCHRONOUS_TO_REPORTING. ⌋*()*

[SWS_Dem_CONSTR_06122] Dependency for DemOccurrenceCounterProcessing ⌈ The DemOccurrenceCounterProcessing shall only be present if DemEnvironmentDataCapture is set to DEM_CAPTURE_SYNCHRONOUS_TO_REPORTING. ⌋ *()*

[SWS_Dem_CONSTR_06123] Dependency for DemOperationCycleStatusStorage ⌈ The DemOperationCycleStatusStorage shall only be present if DemOBDSupport is set to DEM_OBD_MASTER_ECU or DEM_OBD_PRIMARY_ECU. ⌋*()*

[SWS_Dem_CONSTR_06124] Dependency for DemPTOSupport ⌈ DemPTOSupport shall only be present if DemOBDSupport is set to DEM_OBD_MASTER_ECU or DEM_OBD_PRIMARY_ECU. ⌋()

[SWS_Dem_CONSTR_06125] Dependency for DemAgingCycleCounterThreshold ⌈ DemAgingCycleCounterThreshold shall only be present if DemAgingAllowed is set to TRUE. ⌋()

[SWS_Dem_CONSTR_06126] Dependency for DemAgingCycleCounterThresholdForTFSLC ⌈ DemAgingCycleCounterThresholdForTFSLC shall only be present if DemStatusBitHandlingTestFailedSinceLastClear is set to DEM_STATUS_BIT_AGING_AND_DISPLACEMENT. ⌋()

[SWS_Dem_CONSTR_06127] Dependency for DemMaxNumberFreezeFrameRecords ⌈ DemMaxNumberFreezeFrameRecords shall only be present if DemTypeOfFreezeFrameRecordNumeration is set to DEM_FF_RECNUM_CALCULATED. ⌋()

[SWS_Dem_CONSTR_06128] Dependency for DemAgingCycleRef ⌈ DemAgingCycleRef shall only be present if DemAgingAllowed is set to TRUE. ⌋()

[SWS_Dem_CONSTR_06129] Dependency for DemFreezeFrameRecNumClassRef ⌈ DemFreezeFrameRecNumClassRef shall only be present if DemTypeOfFreezeFrameRecordNumeration is set to DEM_FF_RECNUM_CONFIGURED. ⌋()

[SWS_Dem_CONSTR_06130] Dependency for DemReportBehavior ⌈ DemReportBehavior shall only be present if DemEventKind is set to DEM_EVENT_KIND_SWC. ⌋()

[SWS_Dem_CONSTR_06131] Dependency for DemOBDGroupingAssociativeEventsRef ⌈ DemOBDGroupingAssociativeEventsRef shall only be present if DemOBDSupport is set to DEM_OBD_MASTER_ECU or DEM_OBD_PRIMARY_ECU. ⌋()

[SWS_Dem_CONSTR_06132] Dependency for DemOBDCentralizedPID21Handling ⌈ DemOBDCentralizedPID21Handling shall only be present if DemOBDSupport is set to DEM_OBD_MASTER_ECU or DEM_OBD_PRIMARY_ECU. ⌋()

[SWS_Dem_CONSTR_06133] Dependency for DemOBDCentralizedPID31Handling ⌈ DemOBDCentralizedPID31Handling shall only be present if DemOBDSupport is set to DEM_OBD_MASTER_ECU or DEM_OBD_PRIMARY_ECU. ⌋()

[SWS_Dem_CONSTR_06134] Dependency for DemOBDCompliancy ⌈ DemOBDCompliancy shall only be present if DemOBDSupport is set to DEM_OBD_MASTER_ECU or DEM_OBD_PRIMARY_ECU. ⌋()

[SWS_Dem_CONSTR_06135] Dependency for DemOBDEngineType ⌈ DemOBDEngineType shall only be present if DemOBDSupport is set to DEM_OBD_MASTER_ECU or DEM_OBD_PRIMARY_ECU. ⌋()

[SWS_Dem_CONSTR_06136] Dependency for DemOBDEventDisplacement ⌈ DemOBDEventDisplacement shall only be present if DemOBDSupport is set to DEM_OBD_MASTER_ECU or DEM_OBD_PRIMARY_ECU. ⌋()

[SWS_Dem_CONSTR_06137] Dependency for DemOBDInputAcceleratorPedalInformation ⌈ DemOBDInputAcceleratorPedalInformation shall only be present if DemOBDSupport is set to DEM_OBD_MASTER_ECU or DEM_OBD_PRIMARY_ECU. ⌋*()*

[SWS_Dem_CONSTR_06138] Dependency for DemOBDInputAmbientPressure ⌈ DemOBDInputAmbientPressure shall only be present if DemOBDSupport is set to DEM_OBD_MASTER_ECU or DEM_OBD_PRIMARY_ECU. ⌋*()*

[SWS_Dem_CONSTR_06139] Dependency for DemOBDInputAmbientTemperature ⌈ DemOBDInputAmbientTemperature shall only be present if DemOBDSupport is set to DEM_OBD_MASTER_ECU or DEM_OBD_PRIMARY_ECU. ⌋*()*

[SWS_Dem_CONSTR_06140] Dependency for DemOBDInputDistanceInformation ⌈ DemOBDInputDistanceInformation shall only be present if DemOBDSupport is set to DEM_OBD_MASTER_ECU or DEM_OBD_PRIMARY_ECU. ⌋*()*

[SWS_Dem_CONSTR_06141] Dependency for DemOBDInputEngineSpeed ⌈ DemOBDInputEngineSpeed shall only be present if DemOBDSupport is set to DEM_OBD_MASTER_ECU or DEM_OBD_PRIMARY_ECU. ⌋*()*

[SWS_Dem_CONSTR_06142] Dependency for DemOBDInputEngineTemperature ⌈ DemOBDInputEngineTemperature shall only be present if DemOBDSupport is set to DEM_OBD_MASTER_ECU or DEM_OBD_PRIMARY_ECU. ⌋*()*

[SWS_Dem_CONSTR_06143] Dependency for DemOBDInputProgrammingEvent ⌈ DemOBDInputProgrammingEvent shall only be present if DemOBDSupport is set to DEM_OBD_MASTER_ECU or DEM_OBD_PRIMARY_ECU. ⌋*()*

[SWS_Dem_CONSTR_06144] Dependency for DemOBDInputVehicleSpeed ⌈ DemOBDInputVehicleSpeed shall only be present if DemOBDSupport is set to DEM_OBD_MASTER_ECU or DEM_OBD_PRIMARY_ECU. ⌋*()*

[SWS_Dem_CONSTR_06145] Dependency for DemConsiderPtoStatus ⌈ DemConsiderPtoStatus shall only be present if DemOBDSupport is set to DEM_OBD_MASTER_ECU or DEM_OBD_PRIMARY_ECU. ⌋*()*

[SWS_Dem_CONSTR_06156] Dependency on DemFreezeFrameRecordTrigger ⌈ DemFreezeFrameRecordTrigger shall only be present if DemTypeOfFreezeFrameRecordNumeration is set to DEM_FF_RECNUM_CONFIGURED. ⌋*()*

[SWS_Dem_00261] ⌈ The Dem module shall use the C-callback DemRead respectively the operation ReadData of the interface DataServices_{Data} to collect all configured data elements of the respective freeze frame. ⌋*(SRS_Diag_04127)*

[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, report the runtime error DEM_E_NODATAAVAILABLE to the Det and continue 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.

7.7.7.2 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. ⌋(*SRS_Diag_04127*)

Note: If DemMaxNumberPrestoredFF 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 and Dem_ClearPrestoredFreezeFrame. Otherwise they are not provided. ⌋(*SRS_Diag_04127*)

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

Note: If a request to prestore a freeze frame (Dem_PrestoreFreezeFrame) is interrupted by reporting an event status (Dem_SetEventStatus) or a call to Dem_ClearPrestoredFreezeFrame of the same EventId the detailed behavior is implementation specific. The freeze frame which is requested to be prestored might be used immediately or might be stored and used with next upcoming event report or vice versa.

[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] ⌈ A pre-stored freeze frame shall be released after it has been stored in the event memory or discarded. ⌋*(SRS_Diag_04074)*

Note : A pre-stored freeze frame is considered to be stored if event retention is successful. A pre-stored freeze frame is discarded if the event retention failed.

[SWS_Dem_00191] ⌈ If no pre-stored freeze frame is available, the Dem module shall behave according to chapter 7.7.7.1 Storage of freeze frame data. ⌋ *(SRS_Diag_04074)*

Note: The captured data while using pre-stored freeze frames can differ from the data, which is collected using the UDS 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.

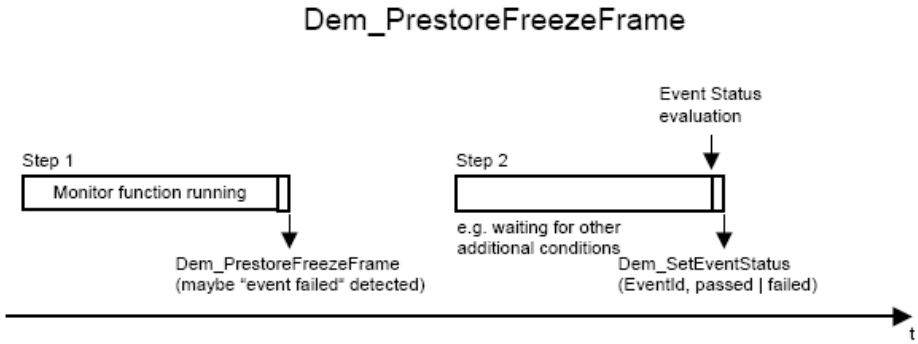


Figure 7.40: Example to use `Dem_PrestoreFreezeFrame` to prestore freeze frame data

[SWS_Dem_00050] ⌈ The Dem module shall provide the API `Dem_ClearPrestoredFreezeFrame` to release the pre-stored freeze frame for the specific event. ⌋*(SRS_Diag_04074)*

[SWS_Dem_00465] ⌈ If an event gets qualified as passed (`UDS 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)*

7.7.7.3 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 DemCounterBasedFdcThresholdStorageValue or DemTimeBasedFdcThresholdStorageValue) is reached (by a positive increment), but at most once per operation cycle. ⌋*(SRS_Diag_04127)*

[SWS_Dem_01069] ⌈ If the DemExtendedDataRecordTrigger is set to DEM_TRIGGER_ON_FDC_THRESHOLD and the event reports DEM_EVENT_STATUS_FDC_THRESHOLD_REACHED (monitor-internal debounced event), the ExtendedDataRecord shall be captured (refer [SWS_Dem_00810]), but at most once per operation cycle. ⌋*(SRS_Diag_04127)*

[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 UDS 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 UDS 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 UDS status bit 3 is set (changing from 0 to 1). ⌋*(SRS_Diag_04127)*

[SWS_Dem_01070] ⌈ If the DemExtendedDataRecordTrigger is set to DEM_TRIGGER_ON_PASSED, the ExtendedDataRecord shall be captured with the change of Testfailed UDS status bit 0 is reset (changing from 1 to 0). If no entry exists, on passed report no new entry shall be created. ⌋*(SRS_Diag_04127)*

[SWS_Dem_01071] ⌈ If the DemExtendedDataRecordTrigger is set to DEM_TRIGGER_ON_MIRROR, the ExtendedDataRecord shall be captured with the transfer of the memory entry to the mirror memory. ⌋*(SRS_Diag_04127)*

[SWS_Dem_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_{Data} to collect all configured data elements which are not typed as internal data elements (refer to DemInternalDataElementClass) of the respective extended data record. ⌋*(SRS_Diag_04074)*

[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. ⌋ *(SRS_Diag_04127)*

7.7.7.4 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.

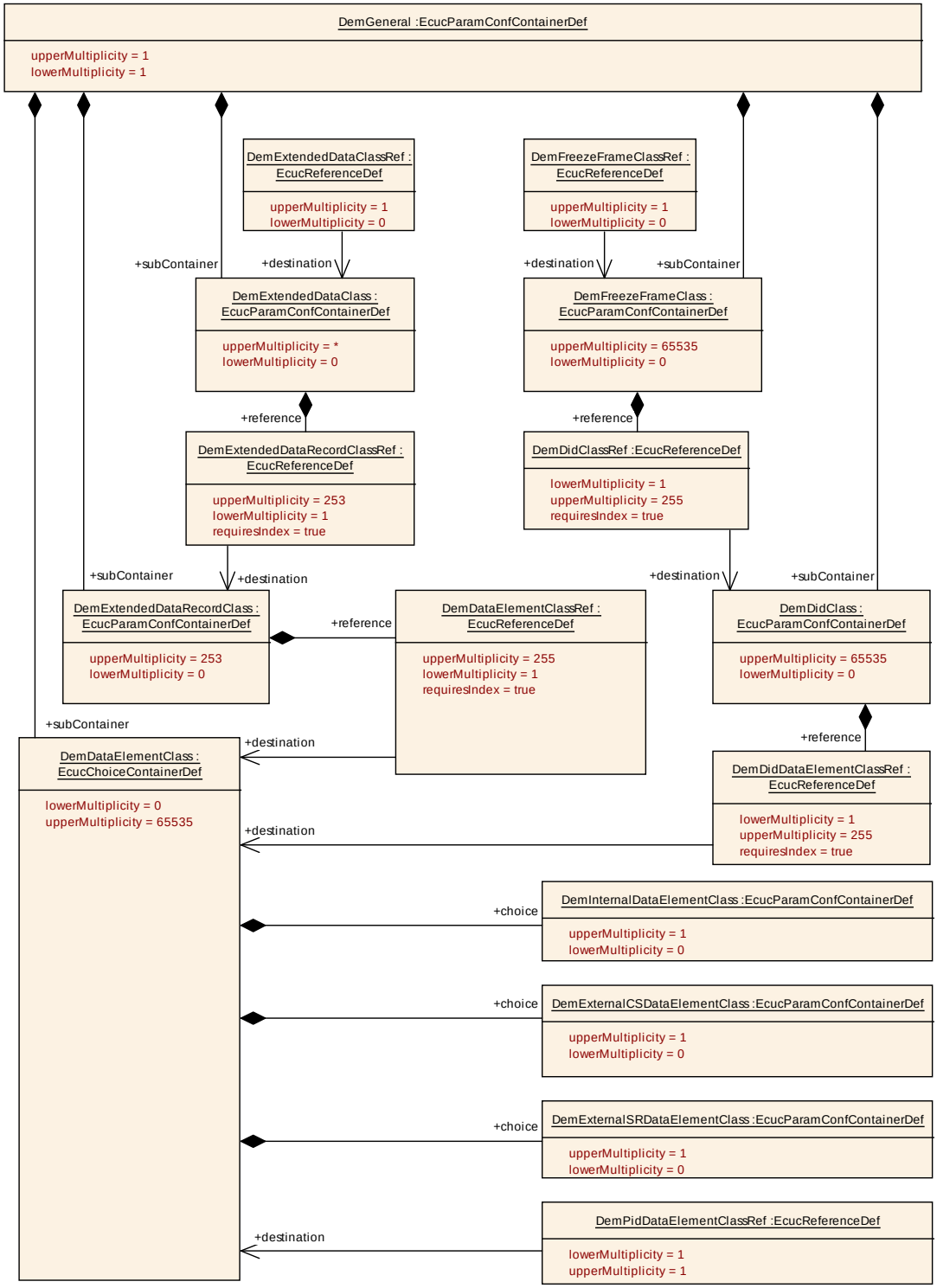


Figure 7.41: Event related data configuration

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_<Data>) or C-callback (refer to ReadDataElement) is generated based on the configuration `DemExternalCSDataElementClass` and `DemExternalSR-`

DataElementClass. For each internal data element, the respective Dem-internal value is mapped.

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

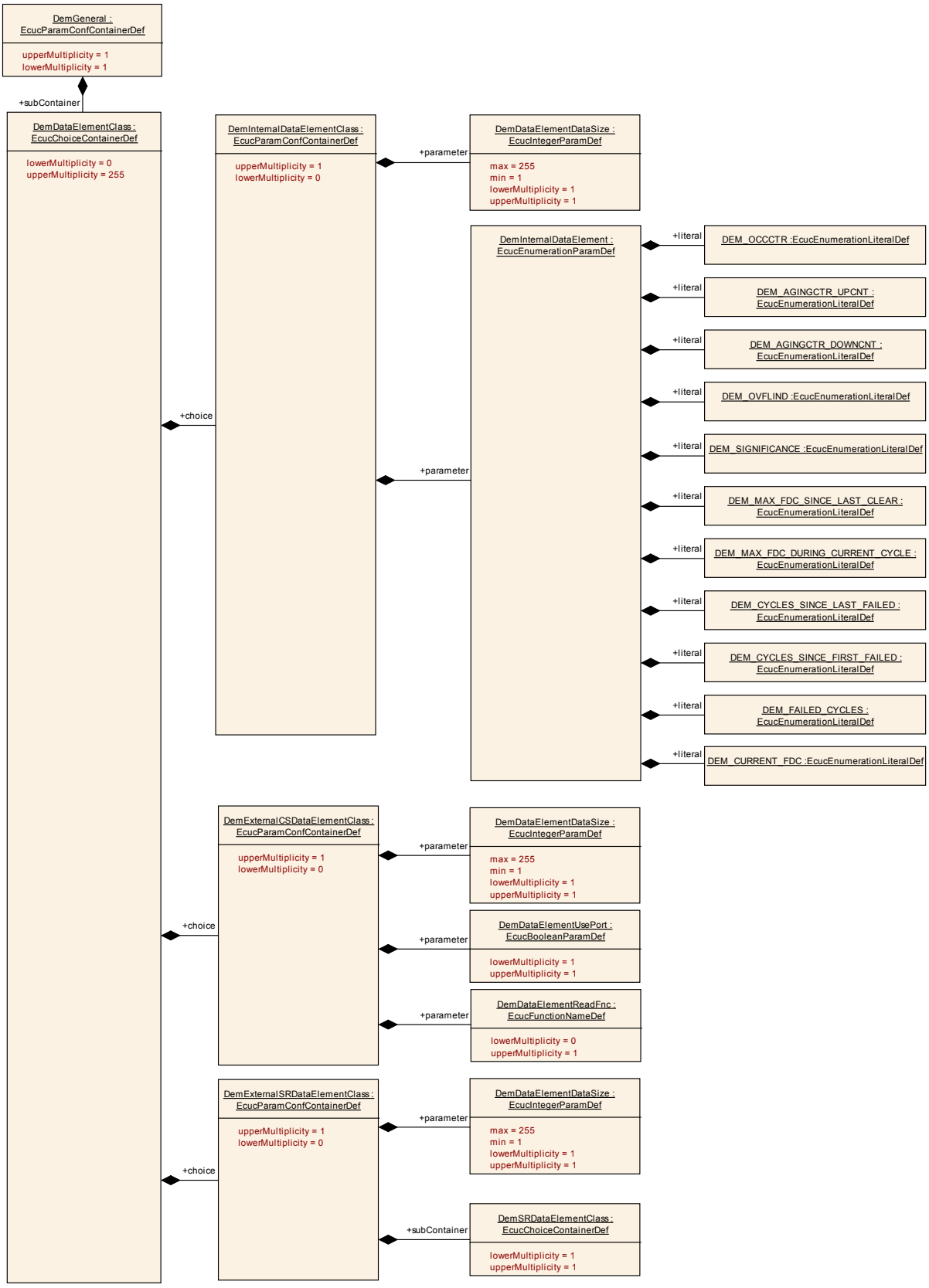


Figure 7.42: Data element configuration

[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_04189, SRS_Diag_04190)*

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_00817] ⌈ Internal data elements (refer to DemInternalDataElementClass) shall not be stored, but the current value shall be used instead. ⌋ *(SRS_Diag_04127, SRS_Diag_04190)*

Note: Further customer-specific Dem-internal data elements (according to [SWS_Dem_00470]) can have different semantics (e.g. Dem-internal data elements can freeze a value or provide a current value)

[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 7.3.2) shall be mapped to the respective data element. ⌋*(SRS_Diag_04127, SRS_Diag_04190)*

[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 7.6.1) shall be mapped to the respective data element (based on [SWS_Dem_00643] or [SWS_Dem_00673]). ⌋ *(SRS_Diag_04190)*

[SWS_Dem_00643] ⌈ If 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[2], Annex D). ⌋ *(SRS_Diag_04068, SRS_Diag_04190)*

[SWS_Dem_00673] ⌈ If 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[2], Annex D). ⌋ *(SRS_Diag_04178, SRS_Diag_04190)*

[SWS_Dem_01043] ⌈ If DemAgingAllowed is set to ‘false’ the internal data element DEM_AGINGCTR_DOWNCNT shall be DemAgingCycleCounterThreshold if configured or ‘255’. ⌋*(SRS_Diag_04190)*

[SWS_Dem_01044] ⌈ If Dem DemAgingAllowed is set to ‘false’ the internal data element DEM_AGINGCTR_UPCNT shall be ‘0’. ⌋*(SRS_Diag_04190)*

[SWS_Dem_01219] ⌈ If the configuration parameter DemInternalDataElement is set to DEM_AGINGCTR_UPCNT_FIRST_ACTIVE, the value shall be calculated analogous to DEM_AGINGCTR_UPCNT. The difference is, that in case the current aging counter has a value of 0 and the event status bits TestFailedThisOperationCycle and TestNotCompletedThisOperationCycle are set to False, a value of 1 is reported upon reading the Dem internal aging counter. ⌋*(SRS_Diag_04133)*

[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 7.7.2.3) shall be mapped to the respective data element as boolean (0 = False, 1 = True). ⌋*(SRS_Diag_04190)*

[SWS_Dem_00592] ⌈ If the configuration parameter DemInternalDataElement is set to DEM_SIGNIFICANCE, then the (static) Dem-internal value of the DTC significance (refer to chapter 7.4.6) shall be mapped to the respective data element with 0 = OCCURRENCE and 1 = FAULT. ⌋*(SRS_Diag_04190)*

[SWS_Dem_01084] ⌈ If the configuration parameter DemInternalDataElement is set to DEM_CURRENT_FDC, then the Dem-internal value of the current fault detection counter (refer to subsubsection 7.7.3.3 and subsubsection 7.7.3.4) shall be mapped to the respective data element with the following value-translation:
[0..127]->[0x00..0x7F] and [-1..-128]->[0xFF..0x80]. ⌋*(SRS_Diag_04190)*

[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 7.7.3.6) shall be mapped to the respective data element. ⌋*(SRS_Diag_04068, SRS_Diag_04190)*

[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 7.7.3.6) shall be mapped to the respective data element. ⌋*(SRS_Diag_04127, SRS_Diag_04190)*

[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 7.6.1.1) shall be mapped to the respective data element. ⌋*(SRS_Diag_04127, SRS_Diag_04190)*

[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 7.6.1.2) shall be mapped to the respective data element. ⌋*(SRS_Diag_04189, SRS_Diag_04190)*

[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 7.6.1.3) shall be mapped to the respective data element. ⌋*(SRS_Diag_04190)*

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. ⌋*(SRS_Diag_04190)*

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 7.7.3) shall be mapped to the respective data element. ⌋*(SRS_Diag_04190)*

[SWS_Dem_00918] ⌈ The Dem 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 DemDataElementEndianness is configured to OPAQUE. ⌋*(SRS_Diag_04074, SRS_Com_02041)*

[SWS_Dem_00919] ⌈ The Dem shall interpret opaque data as uint8[n] and shall always map it to an n-bytes sized signal. ⌋*(SRS_Diag_04074, SRS_Com_02041)*

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

[SWS_Dem_00920] ⌈ The Dem shall extend the endianness conversion defined in [17] Chapter 2.4 to signed data types. ⌋*(SRS_Diag_04074, SRS_Com_02041)*

Note: In [17] Chapter 2.4 defines the endianness conversion for unsigned data types.

[SWS_Dem_01216] ⌈ To serialize the required signed- and unsigned integer AUTOSAR data types into the FreezeFrame/ExtendedRecord the target endianness configured in DemDataElementEndianness shall be considered. If DemDataElementEndianness is not present, the DemDataElementDefaultEndianness shall be used instead. ⌋*(SRS_Diag_04189)*

7.7.7.5 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 chapter 8.4.3.8).

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 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' (extended data or freeze frame data) of an event memory entry is added or updated AND notifications on data changes are configured via DemCallbackEventDataChanged, the Dem shall trigger these configured event-specific notifications as well as the general notification GeneralCallbackEventDataChanged.

The datachanged-callbacks shall be triggered in case an event is reported as:

- DEM_EVENT_STATUS_PASSED

- DEM_EVENT_STATUS_PREPASSED
- DEM_EVENT_STATUS_FDC_THRESHOLD_REACHED

⌋(*SRS_Diag_04155*)

Note: In case of:

- deletion
- displacement (refer to chapter 7.7.2.4)
- aging (refer to chapter 7.7.8)

the Dem doesn't trigger the datachanged-callbacks.

[SWS_Dem_01062] ⌈ The functions Dem_GetEventExtendedDataRecordEx and Dem_GetEventFreezeFrameDataEx shall consider only primary and user defined memory. ⌋(*SRS_Diag_04074*)

Note: The origin of the Event can be derived from the EventId (DemMemoryDestinationRef).

[SWS_Dem_00479] ⌈ The function Dem_GetEventFreezeFrameDataEx 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_01194] ⌈ The function Dem_GetEventFreezeFrameDataEx shall return the WWH-OBD freeze frame when called with RecordNumber 0x00. If WWH-OBD is not supported, the function shall return DEM_NO_SUCH_ELEMENT instead (refer to DemOBDSupport). ⌋(*SRS_Diag_04024*)

[SWS_Dem_00991] ⌈ The format of the data in the destination buffer (DestBuffer) of the function Dem_GetEventFreezeFrameDataEx 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*)

[SWS_Dem_00477] ⌈ The function Dem_GetEventExtendedDataRecordEx shall report the data of the extended data record of the requested diagnostic event. ⌋(*SRS_Diag_04195*)

[SWS_Dem_00989] ⌈ The format of the data in the destination buffer (DestBuffer) of the function Dem_GetEventExtendedDataRecordEx is raw hexadecimal values and contains no header-information like RecordNumber. ⌋(*SRS_Diag_04074*)

Note: The Dcm uses the function Dem_GetNextExtendedDataRecord instead of the function Dem_GetEventExtendedDataRecordEx. The Dlt uses the function Dem_DltGetAllExtendedDataRecords instead.

[SWS_Dem_00995] ⌈ If the interfaces `Dem_GetEventFreezeFrameDataEx` and `Dem_GetEventExtendedDataRecordEx` are called in the context of `General-CallbackEventDataChanged` or `CallbackEventDataChanged`, the data of the triggering event shall be retrievable (return value E_NOT_OK is not allowed). ⌋ *(SRS_Diag_04074, SRS_Diag_04189)*

[SWS_Dem_00996] ⌈ `Dem_GetEventFreezeFrameDataEx` shall return E_NOT_OK in case the requested FreezeFrame data is currently not accessible (e.g. in case of asynchronous preempted data retrieval from application). ⌋ *(SRS_Diag_04074)*

[SWS_Dem_00997] ⌈ `Dem_GetEventExtendedDataRecordEx` shall return E_NOT_OK if the requested data is currently not accessible (e.g.in case of asynchronous preempted data retrieval from application). ⌋ *(SRS_Diag_04074)*

7.7.8 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".

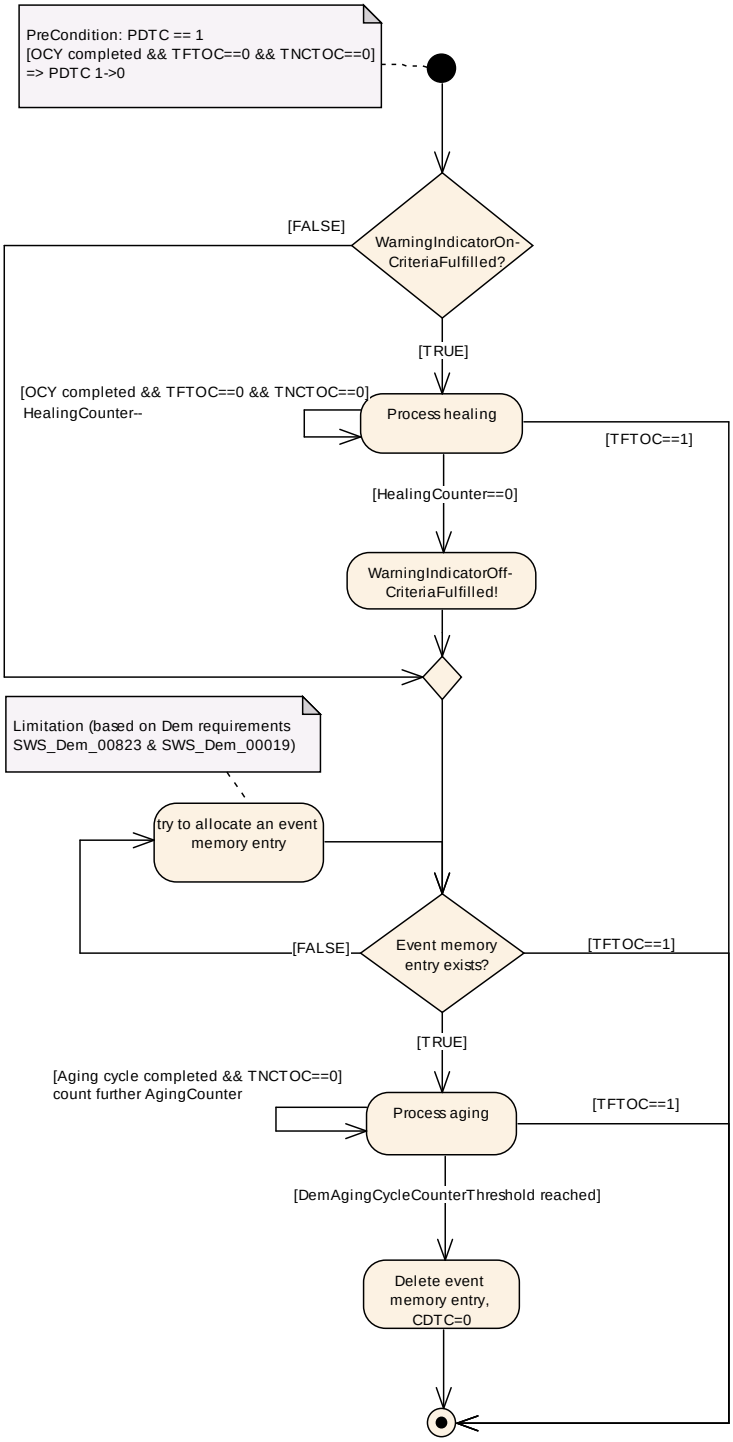


Figure 7.43: General diagnostic event deletion processing

[SWS_Dem_00698] ⌈ The process of `aging` (counting of `aging counter`) starts when healing is completed (WarningIndicatorRequested bit == 0 , refer to Figure 7.43). ⌋*(SRS_Diag_04133, SRS_Diag_04178)*

[SWS_Dem_00019] ⌈ The Dem module shall support at least an `aging counter` for each `event memory entry`. ⌋*(SRS_Diag_04133, SRS_Diag_04068, SRS_Diag_04178)*

[SWS_Dem_00985] ⌈ The aging counter shall be calculated based on the referenced aging/operation cycle (refer to configuration parameter DemAgingCycleRef), if aging is enabled (refer to DemAgingAllowed) for this event. ⌋*(SRS_Diag_04133, SRS_Diag_04068, SRS_Diag_04178)*

[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.⌋ *(SRS_Diag_04133)*

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 7.7.7.3 Storage of extended data).

[SWS_Dem_00493] ⌈ DemAgingCycleCounterThreshold or DemAgingCycleCounterThresholdForTFSLC (depending on which value is higher) defines the number of completed aging cycles, after which the event memory entry shall be deleted(aged) from the event memory. ⌋*(SRS_Diag_04133)*

Note: For completed aging cycles definition see [SWS_Dem_00489]

[SWS_Dem_00823] ⌈ If configuration parameter DemResetConfirmedBitOnOverflow is set to false and in case an event has UDS status bit 3 set and gets qualified as passed and is not stored in an event memory entry the Dem module shall try to allocate an event memory entry to get an aging counter. ⌋ *(SRS_Diag_04067)*

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_00823] or by [SWS_Dem_00824]).

[SWS_Dem_00824] ⌈ If configuration parameter DemResetConfirmedBitOnOverflow is set to false and an event memory entry aging occurs the Dem module shall check for other events having UDS status bit 3 set to 1 and UDS status bit 0 set to 0. If such an event is found, the Dem shall allocate an event memory location to get an aging counter. ⌋*(SRS_Diag_04133)*

Note: The priorizitation which event is choosen is implementation specific.

[SWS_Dem_00498] ⌈ Upon event aging counter reach threshold DemAgingCycleCounterThreshold, the UDS status bit 3 shall be set to 0. ⌋ *(SRS_Diag_04133)*

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_04195)*

[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. ⌋*(SRS_Diag_04133)*

Note: Aging is independent of the UDS status bit 3 and therefore independent of the fault confirmation (refer to chapter 7.7.4).

[SWS_Dem_01054] ⌈ Upon event aging counter reach threshold DemAgingCycleCounterThresholdForTFSLC, the UDS status bit 5 (TestFailedSinceLastClear) shall be set to 0 if DemStatusBitHandlingTestFailedSinceLastClear is set to DEM_STATUS_BIT_AGING_AND_DISPLACEMENT.

In case parameter DemAgingCycleCounterThresholdForTFSLC is not configured, TestFailedSinceLastClear will not be aged. ⌋*(SRS_Diag_04133)*

[SWS_Dem_01075] ⌈ Upon event aging, the Dem shall remove the event related Snapshot data (Freeze frame) and extended data records from the event memory. ⌋*(SRS_Diag_04065)*

[SWS_Dem_01185] ⌈ In case there is no memory location available or occupied by an individual event, the UDS status bit TestFailedSinceLastClear of this event will not be aged. ⌋*(SRS_Diag_04065, SRS_Diag_04133)*

[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 7.6 for the handling of the operation cycle.

[SWS_Dem_00490] ⌈ If the configuration parameter DemAgingRequiresTestedCycle is set to False, the Dem module shall process (count further) the aging counter value, if the respective aging cycle ends/restarts. ⌋*(SRS_Diag_04133)*

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 the configuration parameter DemAgingRequiresTestedCycle is 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 is set to zero. ⌋*(SRS_Diag_04178)*

[SWS_Dem_01214] ⌈ If the configuration parameter DemAgingRequiresNotFailedCycle is set to true, the aging counter shall only be processed in an operation cycle without a test failed report. ⌋*(SRS_Diag_04133)*

[SWS_Dem_01215] ⌈ If the configuration parameter DemAgingRequiresNotFailedCycle is set to false, the aging counter behavior shall not be influenced. ⌋*(SRS_Diag_04133)*

7.7.9 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 7.43).

[SWS_Dem_01056] ⌈ The Dem shall process healing only on passed events using the current operation cycle DemOperationCycleRef. ⌋*()*

7.7.9.1 Warning indicator handling

The detailed configuration of the warning indicator handling within the Dem is shown in Figure 7.44.

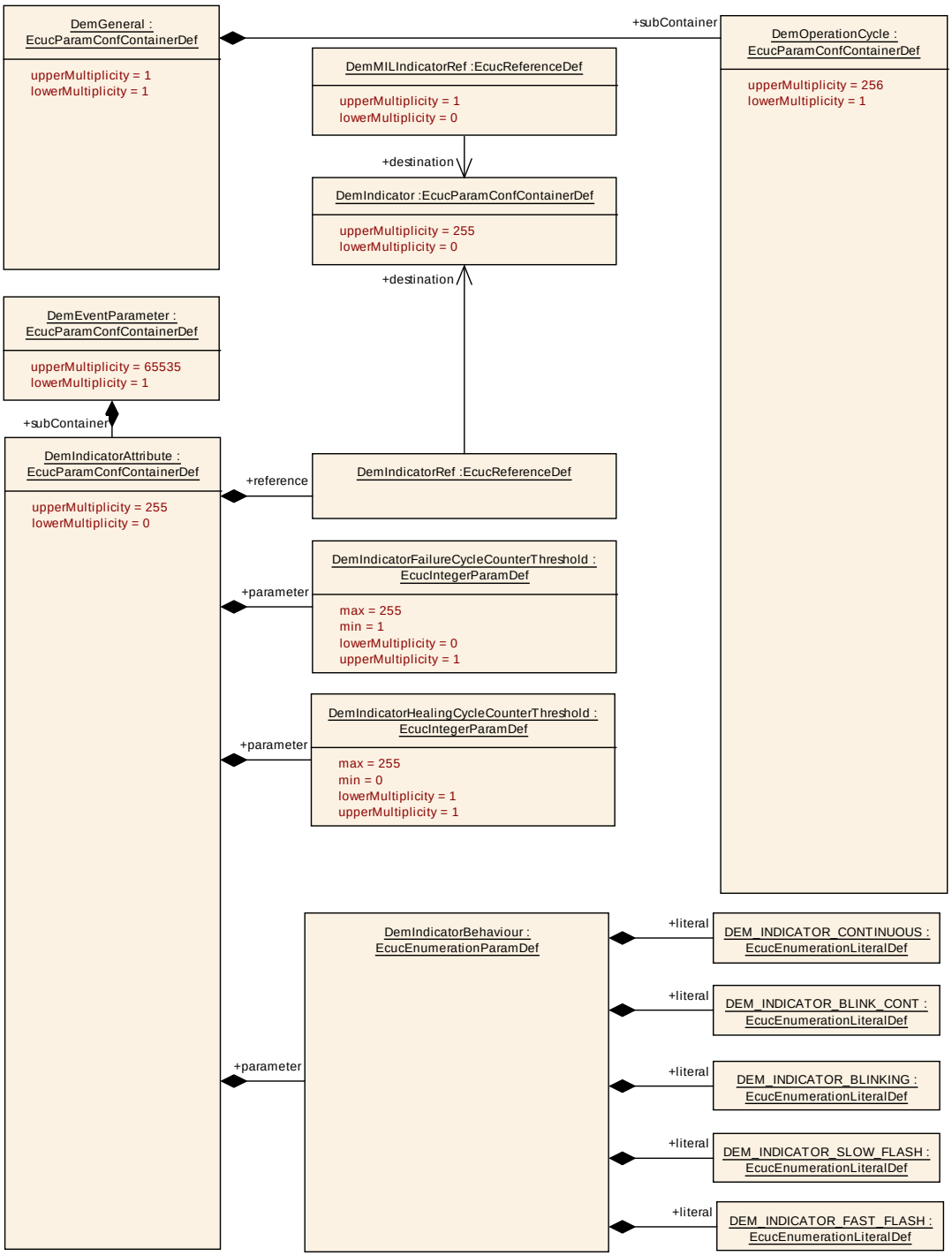


Figure 7.44: Warning indicator configuration

The Dem module supports event specific counters to activate and deactivate indicators. These counters 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).

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 status bit 7 (WarningIndicator). ⌋ *(SRS_Diag_04067)*

[SWS_Dem_00501] ⌈ The condition WarningIndicatorOnCriteriaFulfilled shall be fulfilled if at least one respective event indicator failure counter has reached its DemIndicatorFailureCycleCounterThreshold of tested and failed DemOperationCycles. ⌋ *(SRS_Diag_04069)*

[SWS_Dem_00503] ⌈ If an indicator is configured (via DemIndicatorAttribute) the Dem module shall set at the end of an operation cycle the UDS status bit 7 (WarningIndicatorRequested)to 0, if the following conditions are fulfilled:

- at least one DemIndicatorHealingCycleCounterThreshold is unequal to 0
- all respective events indicator healing counters have reached their DemIndicatorHealingCycleCounterThreshold of tested and passed healing DemOperationCycles (equals WarningIndicatorOffCriteriaFulfilled)
- WIRbit is not enabled by calling the API Dem_SetWIRStatus

⌋ *(SRS_Diag_04069)*

[SWS_Dem_01233] ⌈ If an indicator is configured (via DemIndicatorAttribute) the Dem module shall set when the Event is reported/qualified to PASSED the UDS status bit 7 (WarningIndicatorRequested)to 0, if the following conditions are fulfilled:

- all DemIndicatorHealingCycleCounterThreshold are equal to 0
- WIRbit is not enabled by calling the API Dem_SetWIRStatus

⌋ *(SRS_Diag_04069)*

7.7.9.2 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 ISO-14229-1[2] 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 EventUdsStatusChanged).

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_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.

[SWS_Dem_01303] Asynchronous behavior of Dem_SetWirStatus ⌈ The Dem shall process a call of the Dem_SetWIRStatus asynchronously. This means that the final result is available at a later point in time. ⌋*(SRS_Diag_04069)*.

A later point in time is meant to be implementation specific, it could be the next main function and after Dem_SetWIRStatus has returned.

7.7.9.3 Handling of the warning indicator lamp (MIL)

[SWS_Dem_00546] ⌈ For OBD-relevant ECUs, the Dem module shall provide the configuration parameter DemMILIndicatorRef to indicate that the configured indicator controls the MIL activation and deactivation. ⌋*(SRS_Diag_04110)*

[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. ⌋*(SRS_Diag_04069)*

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 DemEventConfirmationThreshold and DemOperationCycleRef) has to be consistent 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)*

7.7.9.4 Notification and Set of the warning indicator status

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

7.8 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 [12] for further details). For these errors (only a small number compared to application specific events), some additional aspects apply:

- Errors can be detected at startup 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)

Dem_SetEventStatus 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_SetEventStatus is necessary to process the reported fault within the main function of the Dem module.

Dem_SetEventStatus can be 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 initialization of the Dem module, the API Dem_SetEventStatus 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_01212] ⌈ During initialization of the Dem module, the Dem shall support debouncing by calling API Dem_SetEventStatus (with EventStatus set to DEM_EVENT_STATUS_PREPASSED or DEM_EVENT_STATUS_PREFAILED). ⌋ *(SRS_BSW_00339)*

[SWS_Dem_01213] ⌈ The Dem-internal debouncing (counters and timers) shall be initialized by calling Dem_PreInit. ⌋*(SRS_Diag_04068)*

[SWS_Dem_00167] ⌈ The Dem module shall provide a buffer mechanism to queue events which are reported before Dem_Init via Dem_SetEventStatus as qualified as Passed or Failed or reaching the qualification (debouncing). Reporting of DEM_EVENT_STATUS_FDC_THRESHOLD_REACHED shall not be queued. ⌋ *(SRS_BSW_00339)*

[SWS_Dem_00207] ⌈ The size of the queue of the function Dem_SetEventStatus which is used while Dem is not initialized is configurable by the configuration parameter DemBswErrorBufferSize. ⌋*(SRS_BSW_00339)*

[SWS_Dem_01079] ⌈ The Dem shall determine the queue size depending on the number of events that can be reported before Dem_Init, i.e., those events with undefined DemReportBehavior or DemReportBehavior = REPORT_BEFORE_INIT. In case DemBswErrorBufferSize is configured, it overwrites the calculated size of the queue. ⌋*(SRS_BSW_00339)*

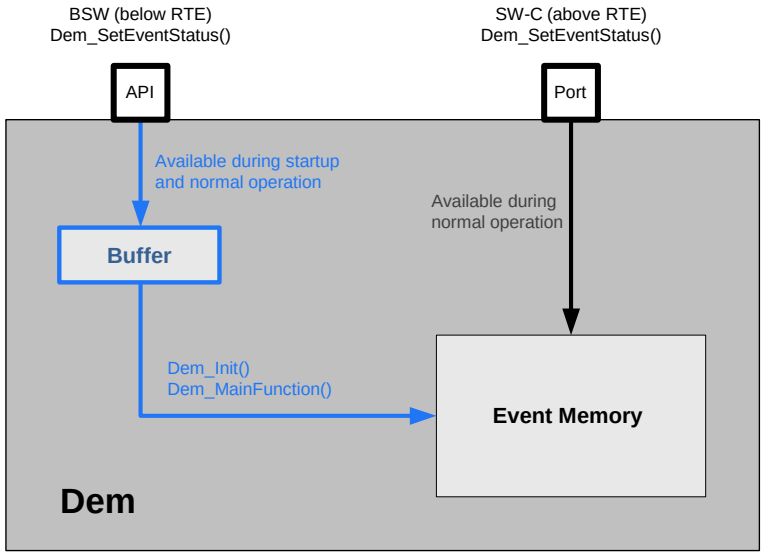


Figure 7.45: Dem_SetEventStatus buffering behavior

[SWS_Dem_00851] ⌈ All events which are reported via `Dem_SetEventStatus` after `Dem_PreInit` and before `Dem_Init` shall consider the state of its assigned operation cycles as started (refer to [SWS_Dem_00481]). ⌋*(SRS_BSW_00339)*

[SWS_Dem_01289] ⌈ All events which are reported via `Dem_SetEventStatus` after `Dem_PreInit` and before `Dem_Init` shall not trigger any monitor status change callback. ⌋*(SRS_BSW_00339)*

[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. ⌋*(SRS_Diag_04178)*

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

7.9 OBD-specific functionality

7.9.1 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), in WWH-OBD [3] referenced as VOBD
- Primary ECU (several per vehicle)
- Dependent / Secondary ECUs (several per vehicle)

From the Basic Software point of view Dependent / Secondary ECUs doesn't need any specific OBD functionality. In Dependent / Secondary ECUs are always related to a Master or a Primary ECU. In Dependent / 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 Dependent / 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 Dependent / Secondary ECU:

Functionality	***Master ECU***	***Primary ECU***	***Dependent / 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
Readiness per vehicle (PID $90)	Yes	No	No
Readiness per ECU (PID $91)	Yes	Yes	No
Vehicle OBD counters (PID $93)	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
Continuous-MI counter	Yes	No	No
Cumulative Continuous-MI counter	Yes	No	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.

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, Continuous-MI, Short-MI, On-demand MI,...)
- 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 [13] and [18]).

[SWS_Dem_01248] ⌈ The configuration parameter `DemOBDEventMemorySetRef` shall be used to reference to the one and only `DemEventMemorySet` that contains `OBD` relevant information. All `OBD` operations of the `Dem` shall be only executed on this `DemEventMemorySet`. ⌋*(SRS_Diag_04001)*

[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
- Qualified OBD Driving cycle
- Permanent fault code - driving cycle (PFC cycle)

The information about the actual state (started / ended) can be accessed via the API `Dem_GetOperationCycleState`.

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.

Qualified OBD Driving cycle
The Qualified OBD Driving cycle is a legally prescribed cycle and has the following definition: For combustion engines (no hybrid): A driving cycle starts with “ignition on“ and ends with the next time “ignition on“ if the engine has reached the first engine start conditions for at least 2 seconds (+/-a sec.) in between. Engine stops which are not caused by the driver or ECU don’t end the driving cycle.

For hybrids and engines with shutoff strategies, the driving cycle is defined as the time between “ignition on“ and the next time “ignition on“ after the driving cycle is qualified.

If a driving cycle is qualified every “ignition off > ignition on - EVENT“ results in the beginning of a new driving cycle. This new driving shall be qualified. A driving cycle is qualified when the engine speed has exceeded the minimum speed for engine start conditions for at least 2 seconds (+/-a sec.) or condition ready for driving for at least 2 seconds (+/-a sec.)

Permanent fault code - driving cycle (PFC cycle)
(refer to chapter 7.9.5.8)

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.

7.9.2 PIDs provided by Dem

[SWS_Dem_00293] ⌈ For the following PIDs the data layout shall be compliant to SAE J1979DA [19].

- PID $01 monitor status since DTCs cleared (4 byte)
- PID $02[1] DTC that caused required freeze frame storage (2 byte)
- PID $1C OBD requirements to which vehicle or engine is certified (1 byte)

[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])

- PID $21 distance traveled while MIL is activated (2 byte)
- PID $30 number of warm-ups (WUC) since DTCs cleared (1 byte)
- PID $31 distance traveled since DTCs cleared (2 byte)
- PID $41 monitor status this driving cycle (4 byte)
- PID $4D engine run time while MIL is activated (2 byte)
- PID $4E engine run time since DTCs cleared (2 byte)
- PID $91 ECU OBD System Information (5 byte)

⌋()

Note: PID $90 and $93 are calculated in the application of the VOBD. Primary ECUs only provides interfaces to retrieve the related data.

[SWS_Dem_CONSTR_06150] Dependency on container DemPidClass ⌈ The container DemPidClass and aggregated sub-container shall only be present if DemOBDSupport is set to DEM_OBD_MASTER_ECU or DEM_OBD_PRIMARY_ECU. ⌋*()*

[SWS_Dem_00351] ⌈ The Dem module shall compute and provide the number of confirmed faults (PID $01, Byte A). ⌋*(SRS_Diag_04141)*

[SWS_Dem_01137] ⌈ For WWH-OBD PID$01 shall not provide the number of confirmed faults. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01138] ⌈ The Dem module shall compute and provide the ECU MIL status (PID $01, Byte A). ⌋*(SRS_Diag_04141)*

[SWS_Dem_01139] ⌈ For WWH-OBD PID$01 shall not provide the ECU MIL status. ⌋ *(SRS_Diag_04141)*

[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 [13], e.g. OBD, OBDII, JOBD etc. The value PID1Cvalue to return is in configuration parameter DemOBDCompliancy defined. ⌋ *(SRS_Diag_04141)*

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

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

A Primary ECU is not allowed to calculate information 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 information and will provide information about mileage / time for PID $31 / $4D / $4E to the Primary ECU via the bussystem.

On receiving the information 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 information for PID $31 / $4D / $4E to the scan tool. ⌋*()*

Primary ECUs are not allowed report informations for PID $31 / $4D / $4E to the scan tool.

On receiving the information 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_00346] ⌈ The Dem module shall use PID $0D (refer to chapter 7.11.8) to calculate PID $21 and PID $31. ⌋*()*

[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. ⌋*()*

[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 [20]). ⌋*()*

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

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

The Dem module provides the configuration switch DemPTOSupport 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 for OBD systems, to assign individual events to one specific readiness group. ⌋*()*

According to SAEJ1979 [18], the group AirCondition Component shall not be supported anymore. However, it is still included in ISO 15031-5 [13]. 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).

PID $21 handling

PID $21 is either reported only by the OBD Master ECU, or a synchronized valued by all OBD ECUs (centralized PID handling).

If the configuration switch DemOBDCentralizedPID21Handling is enabled:

A Software component of the OBD Master ECU could retrieve the current PID $21 value via the interface GetDataOfPID21 from Dem module with the goal to distribute the value to all other Primary ECUs in the vehicle.

A Software component of the OBD Primary ECU receiving the information for PID $21 could forward this information to the Dem module via the interface SetDataOfPID21 to synchronize the value with the OBD Master ECU.

[SWS_Dem_01095] ⌈ On invocation of Dem_SetDataOfPID21 the internal PID $21 value shall be updated. ⌋*()*

Note: AUTOSAR defines that OBD Primary ECUs do not calculate the PID $21 value by its own.

[SWS_Dem_01096] ⌈ An OBD Master ECU shall calculate the PID $21 value by its own (considering [SWS_Dem_00346]). ⌋*()*

[SWS_Dem_01097] ⌈ On invocation of Dem_GetDataOfPID21 the internal calculated PID $21 value shall be returned. ⌋*()*

Note: AUTOSAR defines that OBD Primary ECUs do not calculate the PID $21 value by its own.

[SWS_Dem_01098] ⌈ On invocation of Dem_DcmReadDataOfPID21 the Dem shall return the current value of PID $21. ⌋*()*

[SWS_Dem_01099] ⌈ If Dem_DcmReadDataOfPID21 is called before Dem_SetDataOfPID21, the Dem shall return 0xFFFF as PID $21 value. ⌋*()*

7.9.3 Readiness status

[SWS_Dem_00354] ⌈ The Dem module shall compute for PID $01 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 7.4.8) shall be ignored for this computation. ⌋*()*

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

[SWS_Dem_CONSTR_06147] Dependency for DemEventOBDReadinessGroup ⌈ DemEventOBDReadinessGroup shall only be present if DemOBDSupport is set to DEM_OBD_MASTER_ECU. ⌋*()*

[SWS_Dem_00355] ⌈ The Dem module shall compute for PID $41 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 7.4.8) shall be ignored for this computation. ⌋*()*

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 7.4.8) shall be ignored for this computation. ⌋*()*

[SWS_Dem_00348] ⌈ The Dem module shall provide the disabling of events (refer to `Dem_SetEventDisabled`). OBD Events Suppression (refer to chapter 7.4.8) shall be ignored for this computation. ⌋*()*

[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`. ⌋*()*

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 7.4.8

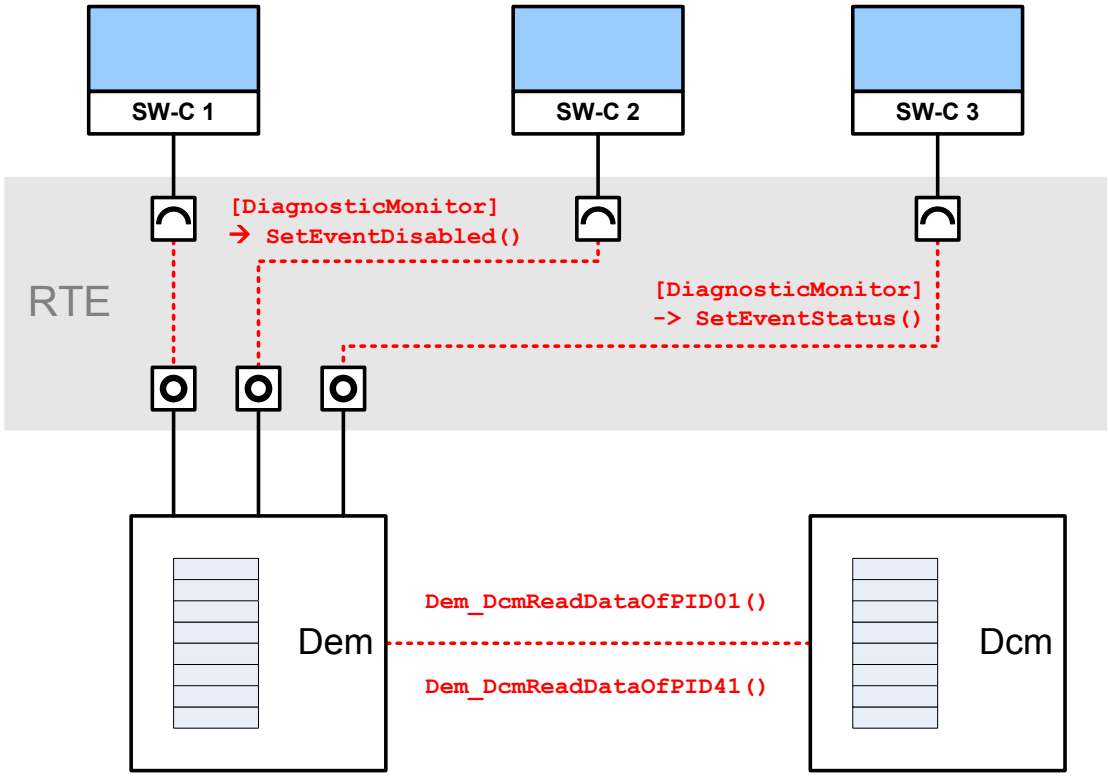


Figure 7.46: Dem calculates PID$01 and PID$41 data based on specific port operations

7.9.4 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. ⌋*()*

[SWS_Dem_00710] ⌈ The numerator shall be calculated in the ECU (Master and Primary). ⌋*()*

[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 to give a software component the possibility to get the General Denominator status information. ⌋*()*

The Master ECU (software component) will provide the IUMPR-Cycle Flag (included in the General Denominator signal) via the bus system.

[SWS_Dem_00712] ⌈ In master and optionally primary OBD ECUs, the Dem shall increment ECU global internal general denominator depending on the denominator DEM_IUMPR_GENERAL_OBDCOND, if available. Depending on the system definition the global general denominator (and therefore the DEM_IUMPR_GENERAL_OBDCOND) may not be needed in some or even all primary ECUs. In such cases the DEM_IUMPR_GENERAL_OBDCOND is never set, and the general denominator is stuck at 0 (and reporting of the InfoType $08 / $0B is disabled in the Dcm). ⌋*()*

[SWS_Dem_01236] ⌈ The Dem shall increment monitor internal denominators depending on the DEM_IUMPR_GENERAL_INDIVIDUAL_DENOMINATOR, and additional conditions if configured in DemIUMPRDenGroup. ⌋*()*

[SWS_Dem_00714] ⌈ The Dem shall provide an API Dem_SetIUMPRDenCondition 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. For gasoline engines, the Info Type $08 is used and for diesel engines the Info Type $0B is used (refer to [13] and [18]).

[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`. ⌋*()*

[SWS_Dem_00357] ⌈ If the `DemOBDEngineType` is set to `DEM_IGNITION_SPARK` the Dem module shall provide the API `Dem_DcmGetInfoTypeValue08` for InfoType $08 IUMPR data. ⌋*()*

Note: The service `Dem_DcmGetInfoTypeValue08` will 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 `DemOBDEngineType` is set to `DEM_IGNITION_COMPRESSION` the Dem module shall provide the API `Dem_DcmGetInfoTypeValue0B` for Info Type $0B IUMPR data. ⌋*()*

Note: The API `Dem_DcmGetInfoTypeValue0B` will 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 DemRatioId). 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 OK status, 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. ⌋*()*

[SWS_Dem_00296] ⌈ The Dem module shall provide the API Dem_RepIUMPRFaultDetect for the asymmetric monitor to report that a malfunction could have been found. ⌋*()*

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_01188] ⌈ The Dem shall increment the ratio-individual IUMPR-numerator at maximum of one time per OBD driving cycle. ⌋*()*

[SWS_Dem_00361] ⌈ The Dem module shall provide the configuration parameter DemRatioKind, to indicate per RatioId if the numerator is calculated based on the TESTED-status or the API call. ⌋*()*

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 to offer several conditions to be applied to the denominator per RatioId. ⌋*()*

[SWS_Dem_00362] ⌈ The Dem module shall provide the APIs for locking (refer to Dem_RepIUMPRDenLock) and unlocking (refer to Dem_RepIUMPRDenRelease) of the denominator under special conditions if the DemIUMPRDenGroup is configured as DEM_IUMPR_DEN_PHYS_API. ⌋*()*

[SWS_Dem_00297] ⌈ The Dem shall provide the API Dem_RepIUMPRDenLock to IUMPR-relevant SWC, to control the denominator specific to the respective RatioId.⌋*()*

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 to IUMPRrelevant SW-C, to control the denominator specific to the respective RatioId.⌋*()*

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 to read out the status of a particular condition. ⌋*(SRS_Diag_04001)*

[SWS_Dem_00717] ⌈ The Dem shall provide the API Dem_SetIUMPRDenCondition to set the status of a particular condition (mainly in depending ECUs). ⌋*()*

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.

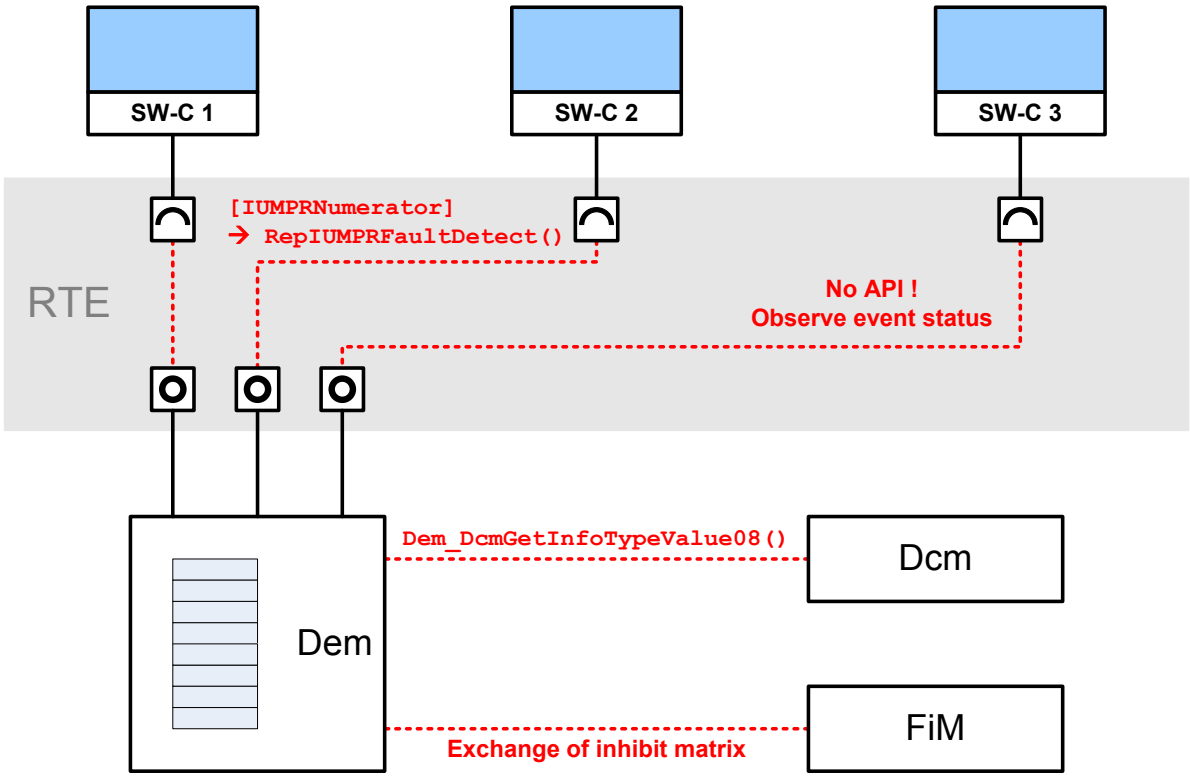


Figure 7.47: Dem calculates IUMPR data based on specific interface operations

7.9.5 OBD for light duty (OBD2)

Together with the Dcm, this Dem SWS provides standardized AUTOSAR interfaces to support the OBD services $01 - $0A defined in SAE J1979 [18]. 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.)

[SWS_Dem_CONSTR_06148] Dependency on container DemRation ⌈ The container `DemRatio` shall only be available if `DemOBDSupport` is set to `DEM_OBD_MASTER_ECU`. ⌋*()*

7.9.5.1 Service $01 Read Current Powertrain Diagnostic Data

With Service $01, the current value of any PID (except PID $02) can be read. For each PID calculated by the Dem, a dedicated API function Dem_DcmReadDataOfPID<NN> as described in chapter 7.9.2 is provided.

7.9.5.2 Service $02 Read Powertrain Freeze Frame Data

In order to retrieve relevant data 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). ⌋*()*

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.

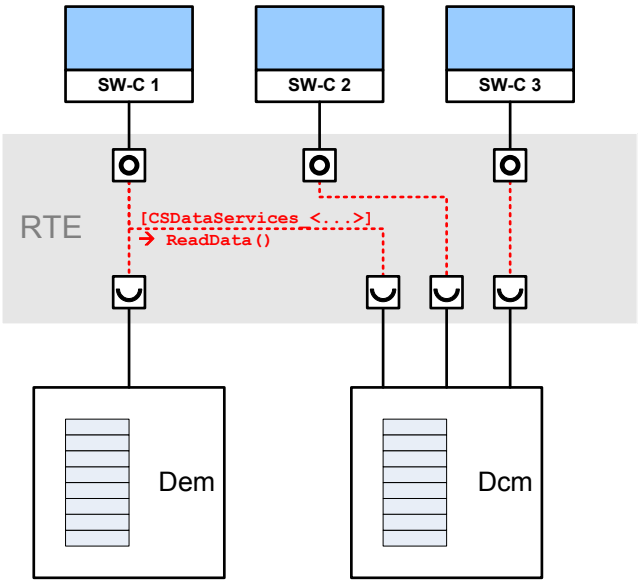


Figure 7.48: Dem and Dcm module requests PID data elements of SW-C via ReadData operation

[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 Dem_DcmReadDataOfOBDFreezeFrame). ⌋*()*

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. ⌋()

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

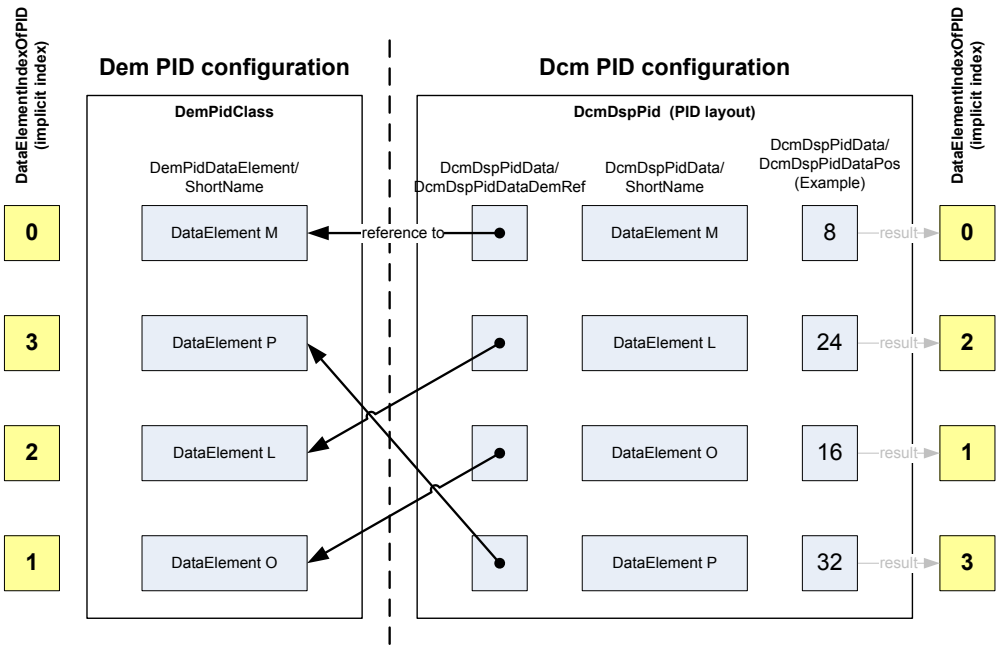


Figure 7.49: PID layout configuration within Dcm (master) and Dem

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

7.9.5.3 Service $03 - Read Emission-Related Powertrain Diagnostic Trouble Codes

The diagnostic service provides the OBD DTCs where the `UDS status bit 3` is currently set. Therefore the interfaces described in chapter 8.3.4.1 will be used by `Dcm`.

7.9.5.4 Service $04 - Clear Reset Emission-Related Diagnostic Information

[SWS_Dem_00718] ⌈ In Master, Primary and `Dependent / Secondary ECUs` executing service $04 shall clear all emission-related diagnostic information in primary and user defined (event) memory (if configured) as defined in SAE J1979 [18]. ⌋()

[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 all user defined event memories (if configured) in Master, Primary and Dependent / Secondary ECUs. ⌋()

7.9.5.5 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 ISO 15031-5 [13] and SAE J1979 [18]) 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 Autosar 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 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 UDS 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_00751] ⌈ The Dem shall provide data structures and functionality to receive, store and report Service$06 test results persistently. ⌋*(SRS_Diag_04001)*

[SWS_Dem_00752] ⌈ The DemDtrId is assigned by the Dem during the BSW configuration step. ⌋*(SRS_Diag_04129)*

Note: The Dem refers to a BSW configuration DemDtr which associates

- a DemDtrId 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, an EventId can only be referenced by at maximum one DTR. ⌋*(SRS_Diag_04001)*

[SWS_Dem_00754] ⌈ The Dem shall use the DemDtrUpdateKind for the evaluation and processing of reported DTR-values:

- DEM_DTR_UPDATE_ALWAYS: State of the event is not considered (default, and used if DemDtrEventRef is not configured).
- DEM_DTR_UPDATE_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.

⌋*()*

[SWS_Dem_00755] ⌈ The Dem shall only support a linear mapping, i.e. only support values for the coefficients b, c, and f. ⌋*()*

Note: ASAM formula coefficients: $(ax^2 + bx + c) / (dx^22 + ex + f)$ 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 DemDtrId, the test value, the minimum and maximum limit, and a state indicating whether the monitor provides a valid minimum limit, maximum limit, or both, or whether the reported values shall be reset to zero. ⌋ *(SRS_Diag_04179)*

[SWS_Dem_00757] ⌈ If DemDtrEventRef references a DemEventParameter, the Dem shall process the reported DTR values only if all the enable/storage conditions of the referenced event are fulfilled, independent of DemDtrUpdateKind. If any of the enable/storage conditions of the referenced event are not fulfilled, then the Dem shall ignore the reported DTR values. ⌋*()*

Note the example with DEM_DTR_UPDATE_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 (=converted 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). If the parameter TestResult passed in Dem_SetDTR is inside/outside the thresholds given by UpperLimit and LowerLimit, Testvalue of Dem_DcmGetDTRData shall also be inside/outside Upplimvalue and Lowlimvalue. If rounding effects lead to an unintended change of the meaning, the thresholds shall be shifted by one increment accordingly to restore the result. ⌋ *(SRS_Diag_04179)*

[SWS_Dem_00760] Computing available OBD MID values ⌈ If Dem_DcmGetAvailableOBDMIDs is called with an "availablity Obdmid" value of 0x00, 0x20, 0x40, 0x60, 0x80, 0xA0, 0xC0 or 0xE0 and the Dem has configured DemDtrMid, the Dem shall compute and provide the “Supported-OBDMID” information in the OUT parameter Obdmidvalue and return E_OK. The computed Obdmids parameter is according to ISO 15031-5[13] and sets a bit for each supported Obdmid in the request range (starting with Obdmid+1) and having the last bit of the Obdmidvalue set, if any subsequent range has configured Obdmids. ⌋*()*

[SWS_Dem_01301] Behavior on invalid Obdmid value ⌈ If Dem_DcmGetAvailableOBDMIDs is called with an invalid Obdmid value which is different to 0x00, 0x20, 0x40, 0x60, 0x80, 0xA0, 0xC0 or 0xE0, the Dem shall report the Det development error DEM_E_INVALID_OBDMID. ⌋*()*

[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 / TID-index whereas the TID-index loops from 0 to number-of-TIDs minus one. ⌋ *(SRS_Diag_04001)*

[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. ⌋*()*

[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_ClearDTC with:

DTC = DEM_DTC_GROUP_ALL_DTCS

DTCFormat = DEM_DTC_FORMAT_OBD

DTCOrigin = DEM_DTC_ORIGIN_PRIMARY_MEMORY

⌋*()*

[SWS_Dem_CONSTR_06149] Dependency on container DemDtr ⌈ The container DemDtr shall only be available if DemOBDSupport is set to DEM_OBD_MASTER_ECU or DEM_OBD_PRIMARY_ECU. ⌋*()*

7.9.5.6 Service $07 - Read emission-related diagnostic trouble codes detected during during current or last completed driving cycle

The diagnostic service provides the OBD DTCs where the DTC Status bit 2 (PendingDTC) is currently set. Therefore the interfaces described in chapter 8.3.4.1 will be used by Dcm.

7.9.5.7 Service $0A - Read Emission-Related Diagnostic Trouble Codes with Permanent Status

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.

[SWS_Dem_00300] ⌈ The Dem shall be able to handle Permanent DTCs according to regulations (refer to [21]) within the specific event memory type "permanent" (refer to chapter 7.3.4). ⌋*()*

[SWS_Dem_00590] ⌈ Events that have been confirmed and activate the MIL (refer to OBDrelevant DTCs, chapter 7.4.1), shall be stored robustly against ClearDTC or powerfail (for details confer [21]). ⌋*()*

[SWS_Dem_00301] ⌈ The Dem shall provide the ability to access the non-volatile stored Permanent DTC by filtering for permanent DTCs to the Dcm (refer to Dem_SetDTCFilter and Dem_GetNextFilteredDTC). ⌋*()*

Note: For service $0A, the Dcm uses the DTCOrigin DEM_DTC_ORIGIN_PERMANENT_MEMORY.

[SWS_Dem_01076] ⌈ New permanent faults of the current driving cycle shall be reported to the Dcm service $0A request latest after the next OBD driving cycle starts. ⌋ *(SRS_Diag_04001)*

[SWS_Dem_01077] ⌈ New permanent faults shall be stored in non-volatile memory (Next Driving Cycle after WriteAll or after positive Callback from NvM) latest at ECU shutdown. ⌋*(SRS_Diag_04001)*

In a Master ECU the interface Dem_GetCycleQualified can be used to read the current state of the PFC cycle, providing the operation cycle id of the DemOBDPFC-CycleRef. A value of TRUE in isQualified indicates that during the current OBD driving cycle the conditions for the PFC cycle have been met.

Note: A SWC on the MIL Master Ecu forwards the information "Minimum trip conditions satisfied" (PFC cycle) via Bussystem to the primary ECUs.

Note: A software component in the Master ECU optionally polls the API Dem_GetCycleQualified and send the information via the bussystem.

In a primary ECU the interface Dem_GetCycleQualified can be used to get informed that the current OBD driving cycle has met the criteria for the PFC cycle by a software component.

7.9.5.8 MIL Handling

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.

Therefore in primary ECUs the function Dem_GetIndicatorStatus can be used to poll the MIL indicator status and send the information to the “MIL Master functionality“ (e.g. located in the master ECU).

7.9.6 WWH-OBD

7.9.6.1 DTC WWHOBD class

[SWS_Dem_01140] ⌈ The Dem shall provide a class value per DTC (to characterize the impact of a malfunction OBD system’s monitoring capability) according to ISO 14229-1 [1], Annex D.3 “DTCSeverityMask and DTC class information bit definitions” (refer to Dem_GetSeverityOfDTC and Dem_GetNextFilteredDTCAndSeverity), only if configured for at least one DTC. ⌋*(SRS_Diag_04141)*

[SWS_Dem_CONSTR_6109] ⌈ The DTC class is only available for ISO 14229-1 [1] DTCs. It is configurable per DTC optionally (refer to DemWWHOBDDTCClass). ⌋*()*

[SWS_Dem_CONSTR_6110] ⌈ The WWH-OBD DTC priority shall be according table Table 7.4. ⌋*()*

WWH-OBD DTC class	*Priority*
A	> B1 DTC
B1	> B2 DTC
B2	> C DTC
C	> no OBD DTC

Table 7.4: WWH-OBD DTC priority

Note : ISO 27145-3 [22]defines the following DTC Classes utilizing the severity levels: No Class, Class A, Class B1, Class B2 and Class C.

7.9.6.2 Continuous-MI counter

The counter Continuous-MI counter is representing the number of engine hours during which the MIL was continuously commanded to be on.

[SWS_Dem_01141] ⌈ If Activation Mode 4 becomes active the counter shall be reset to 0. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01142] ⌈ The counter shall increment for each engine operating hour during which the event is detected and the Continuous-MI is commanded to be on. ⌋ *(SRS_Diag_04141)*

[SWS_Dem_01143] ⌈ The counter shall halt its value when the Continuous-MI is no longer commanded to be on. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01144] ⌈ The counter shall continue its incrementation if the Continuous-MI is again commanded on within three operation sequences. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01145] ⌈ The counter shall reset to zero and start incrementing its value if the Continuous-MI is again commanded on after more than three operation sequences. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01146] ⌈The counter shall reset to zero if the Continuous-MI is not commanded on after more than 40 warm-up cycles or after more than 200 engine operating hours or upon a ClearDTC request. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01147] ⌈ The counter shall be reported as a two byte value. If any count operation occurs which would cause a counter to roll over past 0x00FF then the count value shall instead be maintained at 0x00FF. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01148] ⌈ The counter shall be stored non-volatile. ⌋*(SRS_Diag_04141)*

7.9.6.3 Cumulative Continuous-MI counter (Master ECU)

The counter `Cumulative Continuous-MI counter` is representing the number of engine hours during which MIL has been continuously commanded to be on during its lifetime. The counter shall only be implemented in the `VOBD`.

[SWS_Dem_01149] ⌈ The counter shall increment for each engine operating hour during which the event is detected and the Continuous-MI is commanded to be on, ⌋ *(SRS_Diag_04141)*

[SWS_Dem_01150] ⌈ The counter shall halt its value when the Continuous-MI is no longer commanded to be on, ⌋*(SRS_Diag_04141)*

[SWS_Dem_01151] ⌈ The counter shall never be reset. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01152] ⌈ The counter shall be reported as a two byte value. If any count operation occurs which would cause a counter to roll over past 0x00FF then the count value shall instead be maintained at 0x00FF. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01153] ⌈ The counter shall be stored non-volatile. ⌋*(SRS_Diag_04141)*

7.9.6.4 Class B1 counter

The `Class B1 counter` is representing the number of engine hours during which a Class B1 has been Confirmed and TestFailed. The counter can be implemented as global, i.e. one counter per ECU or local, i.e. one counter per B1 Event. The `Dem` module supports only the global `B1 counter`.

[SWS_Dem_01154] ⌈ The B1 counter shall increment for each 1 hour engine operating hour during which at least one Class B1 event is detected as Confirmed and TestFailed. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01155] ⌈ The B1 counter shall latch its value when no Class B1 event is no longer detected as Confirmed and TestFailed or after a ClearDTC request. ⌋ *(SRS_Diag_04141)*

The B1 counter continue its incrementation if a Class B1 event is again detected as Confirmed and TestFailed.

[SWS_Dem_01156] ⌈ The B1 counter shall reset to zero after three consecutive operating sequences where no Class B1 event have been detected. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01157] ⌈ If B1 Counter exceeds 200 engine operating hours and no Class B1 event have been detected the counter shall be set to 190. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01158] ⌈ The B1 counter shall be reported as a two byte value. If any count operation occurs which would cause a counter to roll over past 0x00FF then the count value shall instead be latched at 0x00FF. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01159] ⌈ The B1 counter shall be stored non-volatile. ⌋ *(SRS_Diag_04141)*

7.9.6.5 Activation Mode

Activation Mode Escalation

[SWS_Dem_01160] ⌈ If the B1 counter exceeds 200 engine operating hours the `Activation Mode 4` shall be active. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01161] ⌈ If at least one Class A event is Confirmed and TestFailed the `Activation Mode 4` shall be active. ⌋*(SRS_Diag_04141)*

Activation Mode Deescalation/ Healing

[SWS_Dem_01162] ⌈ If Activation Mode 4 is active and no Activation Mode 4 relevant condition is present anymore, then the Activation Mode 3 shall be activated. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01163] ⌈ If Activation Mode 4 has been degraded to Activation Mode 3 and an Activation Mode 4 relevant event is confirmed the Activation Mode 4 shall be reactivated if the configured value of EventFailureCycleCounterThreshold is reached. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01164] ⌈ If Activation Mode 3 is active and no Activation Mode 3 and Activation Mode 4 relevant condition is present, then the Activation Mode 1 shall be set. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01165] ⌈ If an Class C event is Confirmed and TestFailed and no Activation Mode 3 or Activation Mode 4 is active, the Activation Mode 2 shall be set to active. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01166] ⌈ The function Dem_GetIndicatorStatus shall return the current "indicator status"(refer to Table 7.5) of those events referenced by the DemMILIndicatorRef. The conditions for the activation modes are defined in Table 7.5. ⌋*(SRS_Diag_04141)*

Condition	*Activation Mode*	*Indicator Status*
No WWH-OBD event active	1	DEM_INDICATOR_OFF
Class C event active	2	DEM_INDICATOR_ON_DEMAND
Class B event active and B1 counter < 200h	3	DEM_INDICATOR_SHORT
Class A event active or B1 counter >= 200h	4	DEM_INDICATOR_CONTINUOUS

Table 7.5: Mapping of conditions to the activation mode

7.9.6.6 Freeze Frame 0x00

The Freeze Frame shall provide the operating conditions of the vehicle at the time of malfunction detection.

[SWS_Dem_01170] ⌈ The legislative Freeze Frame 0x00 shall be stored on transition of UDS status bit 2 from 0 to 1 (via DemFreezeFrameRecordTrigger set to DEM_TRIGGER_ON_PENDING). ⌋*(SRS_Diag_04141)*

[SWS_Dem_01171] ⌈ The update behavior if the Freeze Frame 0x00 is updated on transition of UDS status bit 3 from 0 to 1, can be configured using DemFreezeFrameRecordUpdate. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01172] ⌈ The storage/reporting of freeze frame information associated with a Class A malfunction shall take precedence over a Class B1 malfunction which

shall take precedence over a Class B2 malfunction and likewise for information associated with a Class C malfunction. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01173] ⌈ The first malfunction detected shall take precedence over the most recent malfunction unless the most recent malfunction is of a higher class. ⌋ *(SRS_Diag_04141)*

[SWS_Dem_01174] ⌈ The Freeze Frame 0x00 shall be erased if the event was removed from fault memory due to aging. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01175] ⌈ The legislative freeze frame 0x00 is configured analogues to the non-emission related freeze frames described in chapter 7.7.7.1. ⌋*(SRS_Diag_04141)*

7.9.6.7 Extended Data Record 0x90

Extended Data Record 0x90 is not supported due to a single B1 counter only supported in Dem.

7.9.6.8 Aging

[SWS_Dem_01176] ⌈ Additional to event aging described in chapter 7.7.8 the event shall be aged after 200 hours of engine operation while the aging conditions are met. ⌋*(SRS_Diag_04141)*

[SWS_Dem_CONSTR_6111] ⌈ An OBD related DTC shall have an aging counter threshold of 40. ⌋*()*

[SWS_Dem_CONSTR_6112] ⌈ An OBD related DTC shall have the Warm-Up cycle as aging cycle. ⌋*()*

7.9.6.9 Service $19 42 - Read WWH-OBD DTC By Mask Record

This diagnostic service returns the OBD related DTCs that will match the given DTC Status mask and DTC Severity mask.

[SWS_Dem_01177] ⌈ The interface Dem_GetDTCSeverityAvailabilityMask shall provide the DtcSeverityAvailabilityMask to the DCM. ⌋*(SRS_Diag_04141)*

Note: The DTCSeverityAvailabilityMask can be derived from the configured DTC severity and the WWH-OBD DTC class.

7.9.6.10 Service $14 FFFF33 - Clear Emission Related DTCs

[SWS_Dem_01178] ⌈ In addition to chapter 7.7.2.2 the Dem shall clear/reset the following values:

- Activation Mode
- Readiness (PID $01, PID $41 and PID $90)
- Continuous-MI Counter
- Highest ECU B1 Counter
- B1 Counter
- Legislative Freeze Frame

⌋(*SRS_Diag_04141*)

[SWS_Dem_01179] ⌈ Service $14 FFFF33 shall clear all DTCs (WWH-OBD DTCs and UDS DTCs) ⌋(*SRS_Diag_04141*)

7.10 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*)

Note: The Dem is designed to distinguish 'Service only DTCs' and 'all other DTCs' assigning DTCs to certain event memories. In order to work correctly, with the J1939Dcm APIs 'Service only DTCs' need to be assigned to the user defined memory. All other DTCs need to be assigned to primary memory.

7.10.1 Read DTC

[SWS_Dem_00855] ⌈ The function Dem_J1939DcmSetDTCFilter 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. ⌋(*SRS_Diag_04112*)

[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 7.6. ⌋()

SetFilter Parameter	*UDS Status filter*	*further filter criteria*	*DM representation*	
			AllDTCs	*EmissionDTCs*

DEM_J1939DTC_ACTIVE	(ConfirmedDTC == 1 AND TestFailed == 1) OR MIL_ON	n/a	DM01	DM12
DEM_J1939DTC_PREVIOUSLY_ACTIVE	ConfirmedDTC == 1 AND TestFailed == 0 AND MIL_OFF	n/a	DM02	DM23
DEM_J1939DTC_PENDING	PendingDTC == 1	n/a	DM27	DM06
DEM_J1939DTC_PERMANENT	n/a	permanent memory entry available	n/a	DM28
DEM_J1939DTC_CURRENTLY_ACTIVE	TestFailed == 1	n/a	DM35	n/a

Table 7.6: Types of errors which can be detected by the Dem module

Each reading of DTC returns a composite lamp status of the following lamps:

a) Malfunction Indicator Lamp

b) Red Stop Lamp

c) Amber Warning Lamp

d) Protect Lamp

7.10.1.1 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)*

7.10.1.2 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)*

7.10.1.3 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. ⌋*(SRS_Diag_04110)*

[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. ⌋ *(SRS_Diag_04110)*

7.10.1.4 Composite Protect Lamp Status

[SWS_Dem_00870] ⌈ The composite "Protect Lamp" shall be set to "Lamp Off" in case the Indicator referenced by DemProtectLampIndicatorRef has an IndicatorStatus DEM_INDICATOR_OFF. All other IndicatorStatus states than DEM_INDICATOR_OFF shall set the composite Protect Lamp to "Lamp On". ⌋*(SRS_Diag_04110)*

[SWS_Dem_00871] ⌈ The composite "Flash Protect Lamp" shall be set to "Unavailable / Do Not Flash" (0x03) in case the Indicator referenced by DemProtectLampIndicatorRef 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 DemProtectLampIndicatorRef 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 DemProtectLampIndicatorRef has an IndicatorStatus DEM_INDICATOR_FAST_FLASH. ⌋*(SRS_Diag_04110)*

7.10.1.5 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 the corresponding primary event memory of the requested ClientId by the function call of Dem_J1939DcmSetDTCFilter.

- In case the function has calculated the total number of matching DTCs, the return value shall be E_OK.
- In case the search needs to be interrupted due to internal implementations or limitations, the return value can be set to DEM_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 7.6 for the node 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_NO_SUCH_ELEMENT. The out parameters need not to be valid in this case. ⌋ *(SRS_Diag_04112)*

[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 occur-

rence counter. The return value shall be E_OK. In case the search needs to be interrupted due to internal implementations or limitations, the return value can be set to DEM_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)*

7.10.2 Clear DTCs

For definition of locking the clearing process, refer to chapter 7.7.2.2.

[SWS_Dem_00878] ⌈ The Dem module shall provide the Dem_J1939DcmClearDTC to the J1939Dcm [7] 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 DM03 and DM11 of SAE J1939-73 [15] are requested, all active (DM11) or previously active (DM03) 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_00515].

7.10.3 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_J1939DcmFirstDTCwithLampStatus.

[SWS_Dem_00880] ⌈ Each call to Dem_J1939DcmFirstDTCwithLampStatus shall set the internal counter to the first event having a J1939DTC for this particular “ClientId” assiged. ⌋*(SRS_Diag_04113)*

[SWS_Dem_00881] ⌈ Each call to Dem_J1939DcmGetNextDTCwithLampStatus shall search within the event memory (set by the function Dem_J1939DcmFirstDTCwithLampStatus) for the next event having a J1939DTC assigned. In case no more events are available that have a J1939DTC assigend, the function return value shall be DEM_NO_SUCH_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 E_OK. In case the search needs to be interrupted due to internal implementations or limitations, the return value can be set to DEM_PENDING. The out parameter needs not to be valid in this case. The next call of Dem_J1939DcmGetNextDTCwithLampStatus 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_04113*)

7.10.3.1 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 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*)

7.10.3.2 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 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. ⌋(*SRS_Diag_04110*)

[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 DemIndi-

catorBehaviour 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 DemIndicatorBehaviour of the Indicator referenced by DemRedStopIndicatorRef is set to DEM_INDICATOR_FAST_FLASH. ⌋()

7.10.3.3 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 status bit 7 (WarningIndicator) is active (set to 1), otherwise it shall be set to “Lamp Off”. ⌋(*SRS_Diag_04110*)

[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. ⌋(*SRS_Diag_04110*)

[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*)

7.10.3.4 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 DemProtectLampIndicatorRef and the UDS 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 DemProtectLampIndicatorRef 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 DemIndi-

catorBehaviour of the Indicator referenced by DemProtectLampIndicatorRef 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 DemIndicatorBehaviour of the Indicator referenced by DemProtectLampIndicatorRef is set to DEM_INDICATOR_FAST_FLASH. ⌋(*SRS_Diag_04110*)

7.10.4 FreezeFrame

[SWS_Dem_00899] ⌈ The function Dem_J1939DcmSetFreezeFrameFilter 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 event memory defined by the filter criteria (FreezeFrameKind and ClientId) of Dem_J1939DcmSetFreezeFrameFilter. In case no more (Expanded-)FreezeFrames are matching the filter criteria, the function shall return DEM_NO_SUCH_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 DEM_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_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 E_OK. In case the occurrence counter is above +126 (0x7F), the returned value shall be set to +126 (0x7F).

⌋(*SRS_Diag_04111*)

7.10.4.1 SPNs in ExpandedFreezeFrame

[SWS_Dem_00904] ⌈ Each call to Dem_J1939DcmGetNextSPNInFreezeFrame shall step to the next SPN of the ExpandedFreezeFrame for this DemEventMemory. In case no more SPN of the ExpandedFreezeFrame definition is available, the function return value shall be DEM_NO_SUCH_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_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 E_OK. In case the occurrence counter is above +126 (0x7F), the returned value shall be set to +126 (0x7F).

⌋(*SRS_Diag_04111*)

7.10.5 Diagnostic Readiness

The Dem module needs to calculate internally the (Non-)Continuously Monitored Systems support and status for “Diagnostic Readiness 1” (DM05), “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. ⌋(*SRS_Diag_04113*)

Dem_J1939DcmReadDiagnosticReadiness1 reports the diagnostics information that relates to diagnostic readiness according DM05.

[SWS_Dem_00909] ⌈ A call of Dem_J1939DcmReadDiagnosticReadiness1 shall report a response message based on the J1939-73 [15] DM05 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).

⌋(*SRS_Diag_04113*)

Dem_J1939DcmReadDiagnosticReadiness2 reports the diagnostic information relevant to a second PGN conveying diagnostic readiness according to DM21 (see also DM05).

[SWS_Dem_00910] ⌈ A call of Dem_J1939DcmReadDiagnosticReadiness2 shall report a response message based on the J1939-73 [15] 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).

⌋*(SRS_Diag_04112)*

Dem_J1939DcmReadDiagnosticReadiness3 reports the diagnostic information conveying the pending status of OBD system monitors for the current drive cycle according to DM26 (See also DM05 and DM21).

[SWS_Dem_00911] ⌈ A call of Dem_J1939DcmReadDiagnosticReadiness3 shall report a response message based on the J1939-73 [15] 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 J1939-73 [15] 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).

⌋*(SRS_Diag_04112)*

7.10.6 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 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). ⌋*(SRS_Diag_04112)*

[SWS_Dem_00913] ⌈ Each call of the function Dem_J1939DcmGetNextFilteredRatio shall skip to the next valid SPN within the event memory 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 E_OK.
- In case the calculation needs to be interrupted due to internal implementations or limitations, the return value can be set to DEM_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_NO_SUCH_ELEMENT. The out parameter needs not to be valid in this case.

⌋*(SRS_Diag_04113)*

7.11 Interaction with other Software Modules

7.11.1 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 5.1).

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”. ⌋ *(SRS_BSW_00300)*

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 `GeneralCallbackEventUdsStatusChanged`, `GeneralDiagnosticInfo` and `CallbackEventUdsStatusChanged`), 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.

7.11.2 Interaction with Diagnostic Clients

The `Dem` module provides interfaces to diagnostic clients. A diagnostic client is a `BSW` module, which is related to processing diagnostic requests. The most common diagnostic client is the `Dcm`. There are further used cases, where a diagnostic client is a `SW-C` interacting with the `Dem` via the `RTE` interfaces. A diagnostic client accesses the `Dem` module in order to delegate 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 [10, 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. ⌋*(SRS_Diag_04010)*

[SWS_Dem_00828] ⌈ If the status of a DTC changes, the Dem shall call the configured DemCallbackDTCStatusChanged function ⌋*(SRS_Diag_04010)*

Note: The function Dcm_DemTriggerOnDTCStatus (SWS_Dcm_00614) uses the same prototype as DemTriggerOnDTCStatus. This configuration parameter is for simplicity (so the generic parameter DemCallbackDTCStatusChangedFnc needs not to be used).

The ClientID is also used to filter for specific contents (like UDS instances , J1939 nodes, OBD,..).

7.11.2.1 Parallel event memory access

There are multiple clients that can request access to the fault memory. To allow parallel access to the clients, the Dem uses the concept of a client ID. The client IDs are configured in the container DemClient. Each client which requests access to the fault memory of the Dem has an assigned unique client ID. Examples for these clients are:

- Parallel OBD and UDS communication, where each protocol is handled by an own Dcm client
- Multiple Dcm instances in virtual ECUs, where Dcm instances are independent from each other
- SWCs or complex device driver, as proxy in OBD systems for secondary ECUs
- Clearing DTC by different clients

The Dem supports the client id as parameter ‘ClientId’ in various APIs to:

- Access fault memory information
- Read event related data
- Clear DTC and fault memory
- Control fault memory behavior

[SWS_Dem_01251] ⌈ The Dem shall ensure that clients with a different client Id can call all APIs with ‘ClientId’ as parameter in parallel. An API call from one client shall be processed independently from calls of the same API from a client with a different ID.

E.g. if one client is setting a filter to iterate over extended data records, another client can set a different filter and iterate the extended data records. ⌋(*SRS_Diag_04162*)

[SWS_Dem_01252] ⌈ The Dem shall accept in all APIs with 'ClientId' as parameter only a ClientId value matching to any of the configured DemClientId. If a not configured ClientId value is passed, the Dem shall report the Det error DEM_E_WRONG_CONFIGURATION. ⌋(*SRS_Diag_04162*)

Note: A client is a BSW/SW-C module or a CDD that is processing diagnostic requests or is used to support a distributed diagnostic environment such as events from a secondary ECU that are handled on a primary ECU. The client id concept is explicitly not designed to provide any arbitrary BSW module access to the fault memory, especially if the modules purpose is different than processing diagnostic requests. The design focuses on Dcm for OBD and UDS processing, as well as remote event communication modules for distributed OBD systems.

For each DemClient with the parameter DemClientUsesRte set to TRUE, the Dem will provide the C/S interfaces ClearDTC and EvMemOverflowIndication.

7.11.2.2 Accessing diagnostic fault memory

The Dem provides a range of APIs controlling and accessing event memory and its behavior or configuration and states. Many of those APIs follow a 'select and request' mechanism. Initially a client selects a DTC or a group of DTCs and then calls APIs working on the active selection. A selection is client local and remains until a new selection is made.

[SWS_Dem_01253] ⌈ If the API Dem_SelectDTC is called, the Dem shall use the selected DTC or group of DTC in the requested DTCOrigin and DTCFormat as target for the following API calls:

- Dem_ClearDTC
- Dem_DisableDTCRecordUpdate
- Dem_EnableDTCRecordUpdate
- Dem_GetDTCSelectionResult
- Dem_GetFunctionalUnitOfDTC
- Dem_GetSeverityOfDTC
- Dem_GetStatusOfDTC
- Dem_SelectFreezeFrameData
- Dem_SelectExtendedDataRecord
- Dem_SetDTCSuppression
- Dem_GetDTCSuppression.

⌋(*SRS_Diag_04010*)

[SWS_Dem_01306] Behavior if one client calling Dem_SelectDTC before the previous operation finished ⌈ If Dem_SelectDTC is called by one client and the same client is calling Dem_SelectDTC again and the Dem is currently still processing the previous operation based on Dem_SelectDTC according to [SWS_Dem_01253], the Dem shall return DEM_BUSY. ⌋(*SRS_Diag_04010*)

[SWS_Dem_01305] Behavior if one client starting a new operation while the previous one is still running ⌈ In case any of the Dem_SelectDTC related APIs according to [SWS_Dem_01253] is called by one client and the same client has already started another Dem_SelectDTC related operation according to [SWS_Dem_01253] and this different operation is not completed, the Dem shall return DEM_BUSY. ⌋ (*SRS_Diag_04010*)

[SWS_Dem_01304] Life cycle of Dem_Select dependent operations ⌈ The Dem shall consider an operation based on Dem_SelectDTC according to [SWS_Dem_01253] to be finished independently if the client has requested the E_OK result from the Dem or not. ⌋(*SRS_Diag_04010*)

A client calling Dem_SelectDTC while another operation of the same client is still running, would indicate that the client is no longer interested in the result of the running operation. The result can be dropped and new select can be processed after Dem is ready.

[SWS_Dem_01198] ⌈ For addressing all DTCs, the Dem shall provide the symbol DEM_DTC_GROUP_ALL_DTCS which is selecting all configured DTCs (representation is 0xFFFFFF). ⌋(*SRS_Diag_04065*)

[SWS_Dem_01254] ⌈ The API Dem_SelectDTC triggers the Dem internal DTC selection process on the event memory assigned to the ClientId. The result of the selection shall be reflected in the return value of the API calls following the Dem_SelectDTC call, see [SWS_Dem_01253]. ⌋(*SRS_Diag_04010*)

[SWS_Dem_01255] ⌈ If any of the APIs requiring a Dem_SelectDTC (see [SWS_Dem_01253]) is called without a prior call to Dem_SelectDTC, the Dem shall report the Det error DEM_E_WRONG_CONDITION. () ⌋(*SRS_BSW_00369*)

Note: The Dem_SelectDTC selects internally a DTC or group of DTC. In case a client is interested only in the result of the selection without a further action behind, the API Dem_GetDTCSelectionResult can be used. One use case of this API is the NRC handling for the UDS service 0x14 ClearDTC. The Dcm needs to perform further NRC relevant verifications between selecting and deleting the DTC.

[SWS_Dem_01261] API Behaviour on undefined DTC or format ⌈ If any of the APIs requiring a Dem_SelectDTC according to [SWS_Dem_01253] is called while the selection with Dem_SelectDTC is invalid, then the Dem shall return DEM_WRONG_DTC. A selection is invalid in case the provided DTC does not specify a valid DTC or GroupOfDTCs in the corresponding DTCFormat. The selection also is invalid, if Dem_SelectDTC was not called before. ⌋(*SRS_BSW_00369*)

[SWS_Dem_01299] API behavior without selected DTC ⌈ If any of the APIs requiring a Dem_SelectDTC according to [SWS_Dem_01253] is called without a prior call of Dem_SelectDTC for this client, the Dem shall report the Det DEM_E_WRONG_CONDITION. ⌋*(SRS_BSW_00369)*

[SWS_Dem_01262] API Behaviour on undefined DTCOrigin ⌈ If any of the APIs requiring a Dem_SelectDTC (see [SWS_Dem_01253]) is called on a not existing selected DTCOrigin, the Dem shall return DEM_WRONG_DTCORIGIN on this API. ⌋ *()*.

[SWS_Dem_01256] ⌈ The API Dem_GetDTCSelectionResult shall return E_OK, if the selection of Dem_SelectDTC was successful and the DTC or group of DTC are ready for further processing. ⌋*(SRS_Diag_04010)*

Note: The call of Dem_GetDTCSelectionResult after a Dem_SelectDTC is optional. APIs depending on Dem_SelectDTC have the same behaviour with or without a call of Dem_GetDTCSelectionResult.

[SWS_Dem_01257] ⌈ If the API Dem_GetDTCSelectionResult is called and the DTC or group of DTC from Dem_SelectDTC is invalid, the Dem shall return DEM_WRONG_DTC. ⌋*(SRS_Diag_04010)*

[SWS_Dem_01258] ⌈ If the API Dem_GetDTCSelectionResult is called and the Origin provided to Dem_SelectDTC is invalid, the Dem shall return DEM_WRONG_DTCORIGIN. ⌋*(SRS_Diag_04010)*

[SWS_Dem_01296] Definition of Dem_GetDTCSelectionResultForClearDTC functionality ⌈ The API Dem_GetDTCSelectionResultForClearDTC shall behave exactly as Dem_GetDTCSelectionResult with the only difference that, DEM_WRONG_DTC is returned in case of a single DTC was selected and DemClearDTCLimitation is set to DEM_ONLY_CLEAR_ALL_DTCS for that client. ⌋ *(SRS_Diag_04010)*

7.11.2.3 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 API Dem_GetTranslationType shall return the translation format of the DemEventMemorySet configured in DemTypeOfDTCSupported. ⌋ *(SRS_Diag_04201)*

[SWS_Dem_00060] ⌈ The function Dem_GetDTCStatusAvailabilityMask shall provide the UDS status availability mask (refer to DemDtcStatusAvailabilityMask in DemGeneral), that means the UDS status information (according to ISO-14229-1[2]) supported by the Dem module with respect to the tester client with UDS service 0x19. ClientID shall not be evaluated for determining the availability mask. ⌋ *(SRS_Diag_04067, SRS_Diag_04010)*

[SWS_Dem_01181] ⌈ The Dem module shall be able to return the DTC severity availability mask (refer to Dem_GetDTCSeverityAvailabilityMask) in accordance to ISO 14229-1 [1]. ⌋*(SRS_Diag_04141)*

[SWS_Dem_01182] ⌈ The function Dem_GetDTCSeverityAvailabilityMask shall calculate the DTC severity availability mask in accordance to ISO 14229-1 [1] from the severity configuration of each supported DTC. ⌋*(SRS_Diag_04141)*

[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_GetStatusOfDTC shall copy the status of the selected DTC to the parameter DTCStatus. The DTCStatus format is according to ISO-14229-1[2]. ⌋*(SRS_Diag_04067)*

[SWS_Dem_01259] ⌈ If the API Dem_GetStatusOfDTC is called on a selected DTC that does not exist in the selected format or if the selected DTC is a group of DTC or no DTC was selected, the Dem shall return DEM_WRONG_DTC. ⌋*()*

[SWS_Dem_01260] ⌈ If the API Dem_GetStatusOfDTC is called on a not existing selected DTCOrigin, the Dem shall return DEM_WRONG_DTCORIGIN. ⌋*()*

[SWS_Dem_01275] ⌈ If the API Dem_GetStatusOfDTC is called on a selected DTC that does not have an assigned DTC status, the Dem shall return DEM_NO_SUCH_ELEMENT. ⌋*()*

Note: In mirror event memory the DTCs does not necessarily have a status assigned.

It is possible, that a DTC with different states exists depending on the location. If the mirror 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 mirror memory with its latest status).

[SWS_Dem_00057] ⌈ The function Dem_SetDTCFilter shall set the filter criteria for the client with the same client ID for subsequent calls of::

- Dem_GetNumberOfFilteredDTC
- Dem_GetNextFilteredDTC
- Dem_GetNextFilteredDTCAndFDC
- Dem_GetNextFilteredDTCAndSeverity.

The filter criteria attributes shall be used until the next call of Dem_SetDTCFilter or Dem initialization. ⌋*(SRS_Diag_04057)*

Note: Calls to Dem_SetDTCFilter with different ClientID can be called in parallel (see [SWS_Dem_01251]) without interfering each other. This way multiple clients can access the fault memory data in parallel.

In figure 7.50 a situation is shown, where two clients (e.g. an Dcm for OBD and an Dcm for UDS) are reading the number of filtered DTCs in parallel.

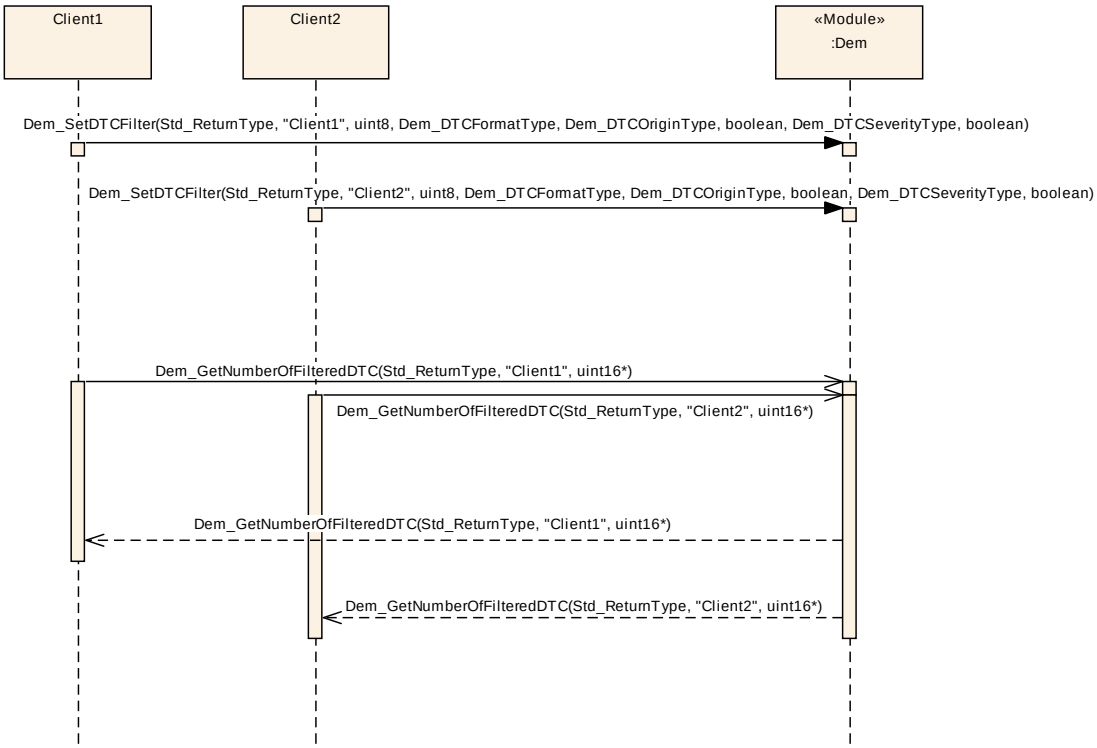


Figure 7.50: Multiple clients requesting event memory information in parallel

[SWS_Dem_01300] API behavior without set DTC filter ⌈ If any of the APIs requiring a Dem_SetDTCFilter according to [SWS_Dem_00057] is called without a prior call of Dem_SetDTCFilter for this client, the Dem shall report the Det DEM_E_WRONG_CONDITION. ⌋*(SRS_BSW_00369)*

[SWS_Dem_01264] ⌈ IF Dem_SetDTCFilter is called with an invalid DTCOrigin, the Dem shall return E_NOT_OK. ⌋*()*

[SWS_Dem_01265] ⌈ IF Dem_SetDTCFilter is called with an invalid DTCStatus-Mask, the Dem shall return E_NOT_OK. ⌋*()*

[SWS_Dem_01266] ⌈ In case of calling any subsequent function of Dem_SetDTCFilter (see [SWS_Dem_00057]) after the initial Dem_SetDTCFilter did not return E_OK, the Dem shall shall return E_NOT_OK on this subsequent function call. ⌋*()*

[SWS_Dem_01263] ⌈ A call of Dem_SetDTCFilter sets the requested filter on the event memory assigned to ClientID of the call. ⌋*(SRS_Diag_04162)*

[SWS_Dem_00649] ⌈ Each call of Dem_SetDTCFilter shall lead to a reset of the sequence for that client. ⌋*(SRS_Diag_04057)*

[SWS_Dem_01058] ⌈ The function Dem_SetDTCFilter shall ignore unsupported bits (refer to configuration parameter DemDtcStatusAvailabilityMask) retrieved in DTCStatusMask. ⌋*()*

The parameter DTCFormat of Dem_SetDTCFilter is relevant to decide about specific protocol characteristics like OBD vs. UDS vs. J1939. The DTCOrigin only references the fault memory location.

[SWS_Dem_01066] ⌈ The Det error DEM_E_WRONG_CONFIGURATION shall be reported if the function Dem_SetDTCFilter is called with a value of the parameter DTCFormat that is not supported per configuration (e.g. if DTCFormat = DEM_DTC_FORMAT_OBD is passed, but OBD is not supported per configuration). ⌋ *(SRS_Diag_04057)*

[SWS_Dem_01067] ⌈ The Det error DEM_E_WRONG_CONFIGURATION shall be reported if the function Dem_SetDTCFilter is called with a value of the parameter DTCOrigin that is not supported per configuration (e.g. if DTCOrigin "DEM_DTC_ORIGIN_MIRROR_MEMORY" is passed, but no mirror memory is configured). ⌋*()*

[SWS_Dem_00061] ⌈ The function Dem_GetNumberOfFilteredDTC shall get the number of DTCs matching the filter criteria defined by the function call of Dem_SetDTCFilter [SWS_Dem_00057]. ⌋*()*

[SWS_Dem_01267] ⌈ If any of the APIs requesting a Dem_SetDTCFilter (see [SWS_Dem_00057]) is called prior of calling Dem_SetDTCFilter a Det error DEM_E_WRONG_CONDITION shall be reported for this API call. ⌋*()*

[SWS_Dem_00216] ⌈ With each call to the function Dem_GetNextFilteredDTC the Dem module shall return the next DTC and its associated status matching the filter criteria defined by the function call of Dem_SetDTCFilter [SWS_Dem_00057]. ⌋ *(SRS_Diag_04010)*

The Dcm calls the function Dem_GetNextFilteredDTC continuously until the return value of the function is DEM_NO_SUCH_ELEMENT, to receive all DTCs matching the filter criteria.

[SWS_Dem_00653] ⌈

The API Dem_GetNextFilteredDTC shall not return the value DEM_PENDING when DTCFormat = DEM_DTC_FORMAT_OBD (indicating emission-related services). ⌋*()*

Rationale: The API Dem_GetNextFilteredDTC 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_GetNextFilteredDTCAndFDC 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_SetDTCFilter (refer to [SWS_Dem_00057]). ⌋*(SRS_Diag_04010)*

[SWS_Dem_00513] ⌈ If the callback-function Dem_GetFaultDetectionCounter returns other than E_OK or is not configured for a DTC filtered by Dem_GetNextFilteredDTCAndFDC, this FDC value shall be returned as 0. ⌋ *(SRS_Diag_04010)*

The Dcm calls the function Dem_GetNextFilteredDTCAndFDC continuously until the return value of the function is DEM_NO_SUCH_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 Dem_GetFaultDetectionCounter).

[SWS_Dem_00287] ⌈ With each call to the function Dem_GetNextFilteredDTCAndSeverity the Dem module shall return the next DTC and its associated fault severity matching the filter criteria defined by the function call Dem_SetDTCFilter (refer to [SWS_Dem_00057]). ⌋*(SRS_Diag_04010)*

The Dcm calls the function Dem_GetNextFilteredDTCAndSeverity continuously until the return value of the function is DEM_NO_SUCH_ELEMENT, to receive all DTCs matching the filter criteria.

For information about the functions Dem_GetSeverityOfDTC and Dem_GetFunctionalUnitOfDTC, refer to chapter 7.4.

[SWS_Dem_00595] ⌈ The function Dem_SetFreezeFrameRecordFilter 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_GetNextFilteredRecord. The filter criteria attributes shall be used until the next call of Dem_SetFreezeFrameRecordFilter or Dem initialization. ⌋ *(SRS_Diag_04074)*

[SWS_Dem_00650] ⌈ Each call of Dem_SetFreezeFrameRecordFilter shall lead to a reset of the sequence. ⌋*(SRS_Diag_04204)*

[SWS_Dem_00210] ⌈ The function Dem_SetFreezeFrameRecordFilter 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_GetNextFilteredRecord 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_SetFreezeFrameRecordFilter. ⌋*(SRS_Diag_04074)*

The client calls the function Dem_GetNextFilteredRecord continuously until the return value of the function is DEM_NO_SUCH_ELEMENT, to receive all records matching the filter criteria.

[SWS_Dem_00219] ⌈ The function Dem_GetDTCByOccurrenceTime shall provide the capability to get one DTC stored in the primary event memory of the addressed DemEventMemorySet according to the API parameter DTCRequest. ⌋ *(SRS_Diag_04195)*

[SWS_Dem_00221] ⌈ The API Dem_GetDTCByOccurrenceTime shall return DEM_NO_SUCH_ELEMENT, if no DTC is matching the requested occurrence time in the parameter DTCRequest. ⌋*(SRS_Diag_04195)*

7.11.2.4 Access event related data

This section defines the APIs, to get access to the event related data (refer to chapter 7.7.7.1 and chapter 7.7.7.3) stored with the DTCs in the event memory of the Dem via diagnostics. Furthermore, access to WWH-OBD relevant PIDs stored in a freeze frame is made available. Details concerning freeze frame handling can be found in ISO-14229-1[2] and ISO 27145-3[22].

Example for the mapping of a PID to a DID: PID $04 –> DID 0xF404

Note: The data returned include the DTCSnapshotRecordNumberOfIdentifiers (Num of DIDs) as defined by ISO-14229-1[2] for the response message to service 0x19 0x05.

[SWS_Dem_01268] ⌈ If the API Dem_SelectFreezeFrameData is called, the Dem shall use the DTC selected by Dem_SelectDTC and the RecordNumber as a target for the following API calls:

- Dem_GetNextFreezeFrameData
- Dem_GetSizeOfFreezeFrameSelection

⌋*(SRS_Diag_04074)*

[SWS_Dem_01269] ⌈ The API Dem_SelectFreezeFrameData triggers the Dem internal Freeze Frame selection process on the event memory assigned to the ClientId. The Dem shall provide the result of the selection process in the return value of the API calls following the Dem_SelectFreezeFrameData call, see [SWS_Dem_01268]. ⌋ *(SRS_Diag_04010)*

[SWS_Dem_01270] ⌈ If any of the APIs requiring a Dem_SelectFreezeFrameData (see [SWS_Dem_01268]) are called without a prior call to Dem_SelectFreezeFrameData, the Dem shall report the Det error DET_E_WRONG_CONDITION. ⌋*(SRS_BSW_00369)*

[SWS_Dem_01271] ⌈ If any of the APIs requiring a Dem_SelectFreezeFrameData is called without a prior call to Dem_SelectFreezeFrameData, the Dem shall report the Det error DEM_E_WRONG_CONDITION. ⌋*(SRS_BSW_00369)*

[SWS_Dem_00071] ⌈ The function Dem_GetNextFreezeFrameData shall copy the specific data identifiers (DIDs) of the freezeframe record selected by the API Dem_SelectFreezeFrameData for the specified ClientId to the destination buffer

provided by the DestBuffer parameter. The Dem shall transmit these data as a complete record with the format shown in Figure 7.51. ⌋(*SRS_Diag_04074*)

[SWS_Dem_00576] ⌈ If the RecordNumber selected by the API Dem_SelectFreezeFrameData is set to 0x00, Dem_GetNextFreezeFrameData shall provide the event/DTC specific WWH-OBD equivalent freeze frame record. ⌋ (*SRS_Diag_04074*)

Note: Retrieving record 0x00 with Dem_SelectFreezeFrameData and Dem_GetNextFreezeFrameData is only supported if the Dem module supports WWH-OBD (refer to DemOBDSupport)

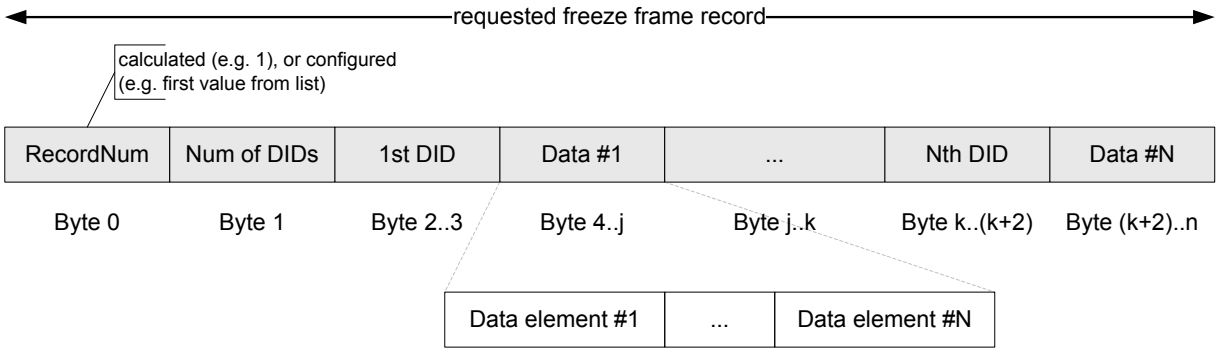


Figure 7.51: Buffer format used by Dem_GetNextFreezeFrameData

Note: The data returned include the DTCSnapshotRecordNumber (RecordNum) and DTCSnapshotRecordNumberOfIdentifiers (Num of DIDs) as defined by ISO-14229-1[2] 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 a single DTC is selected and the API Dem_SelectFreezeFrameData is called with a valid RecordNumber which is not stored, Dem_GetNextFreezeFrameData shall return E_OK and BufSize 0 (empty buffer). ⌋(*SRS_Diag_04074*)

[SWS_Dem_00074] ⌈ The function Dem_GetSizeOfFreezeFrameSelection shall return the size of the requested freeze frame record(s) by Dem_SelectFreezeFrameData for the specified ClientId, which represents the number of user data bytes, including any freeze frame headerinformation (according to the format defined in [SWS_Dem_00071]). ⌋(*SRS_Diag_04204*)

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_01272] ⌈ If the API Dem_SelectExtendedDataRecord is called, the Dem shall use the DTC selected by Dem_SelectDTC and the ExtendedDataNumber as a target for the following API calls:

- Dem_GetNextExtendedDataRecord

- Dem_GetSizeOfExtendedDataRecordSelection

⌋(*SRS_Diag_04074*)

[SWS_Dem_01273] ⌈ The API Dem_SelectExtendedDataRecord triggers the Dem internal extended data record selection process on the event memory assigned to the ClientId. The Dem shall provide the result of the selection process in the return value of the API calls following the Dem_SelectExtendedDataRecord, see [SWS_Dem_01272]. ⌋(*SRS_Diag_04010*)

[SWS_Dem_01274] ⌈ If any of the APIs requiring a Dem_SelectExtendedDataRecord (see [SWS_Dem_01272]) are called without a prior call to Dem_SelectExtendedDataRecord, the Dem shall report the Det error DEM_E_WRONG_CONDITION. ⌋(*SRS_BSW_00369*)

[SWS_Dem_00075] ⌈ The API Dem_GetNextExtendedDataRecord shall copy the complete data of the requested extended data record by Dem_SelectExtendedDataRecord API for the specified ClientId to the destination buffer (DestBuffer). The function shall transmit these data as a complete record with the format shown in figure 7.52. The extended data record number is placed as the first byte in the copied data. ⌋(*SRS_Diag_04074*)

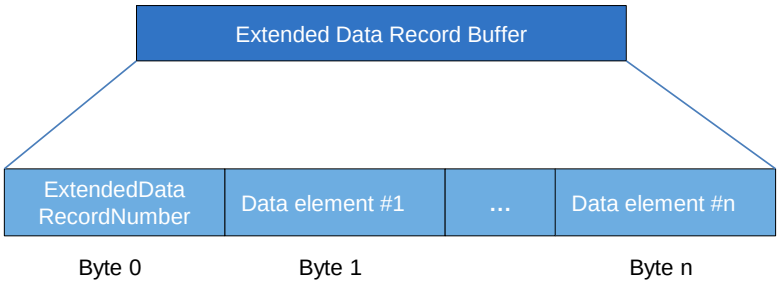


Figure 7.52: Buffer format used by Dem_GetNextExtendedDataRecord

[SWS_Dem_00631] ⌈ If the API Dem_SelectExtendedDataRecord is called with a valid DTC and a valid extended data record number which is not stored, the API shall return E_OK and Dem_GetNextExtendedDataRecord shall return BufSize 0 (empty buffer). ⌋(*SRS_Diag_04074*)

[SWS_Dem_00076] ⌈ The function Dem_GetSizeOfExtendedDataRecordSelection shall return the size of the requested extended data record(s) selected by Dem_SelectExtendedDataRecord, which represents the number of user data bytes stored in the extended data record, including any extended data record header-information (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_DisableDTCRecordUpdate, if the freeze frame or extended data of a specific DTC is about to be accessed by subsequent API calls. 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_DisableDTCRecordUpdate shall protect the event related data of the selected DTC within the selected DTCOrigin from updating or deleting, to allow a consistent read for the following subsequent APIcalls:

- Dem_SelectFreezeFrameData
  Dem_GetSizeOfFreezeFrameSelection and
  Dem_GetNextFreezeFrameData
- Dem_SelectExtendedDataRecord
  Dem_GetSizeOfExtendedDataRecordSelection and
  Dem_GetNextExtendedDataRecord.

⌋*()*

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. Dem-internal data elements).

Note: The function Dem_DisableDTCRecordUpdate does not affect the UDS status information update.

[SWS_Dem_00271] ⌈ The function Dem_EnableDTCRecordUpdate shall release the currently disabled DTC which has been protected by the function Dem_DisableDTCRecordUpdate, so that the data can be updated again. ⌋ *(SRS_Diag_04074)*

The function Dem_EnableDTCRecordUpdate is the counterpart to the function Dem_DisableDTCRecordUpdate.

The Dcm calls the function Dem_EnableDTCRecordUpdate after the freeze frame and extended data of the specific DTC were protected by the function Dem_DisableDTCRecordUpdate, after the access by subsequent APIcalls is finished.

[SWS_Dem_00648] ⌈ If development error detection is enabled and the function Dem_DisableDTCRecordUpdate is called while another DTC is locked by the same client, the Dem module shall report the Det DEM_E_WRONG_CONDITION (refer also to [SWS_Dem_00370]). ⌋*(SRS_Diag_04066)*

7.11.2.5 Clear diagnostic information

The Dem provides the functionality to clear DTCs to the Dcm and other diagnostic clients. The general DTC clearing process is described in chapter 7.7.2.2.

7.11.2.6 Control DTC setting

[SWS_Dem_01290] DTCs affected by Dem_DisableDTCSetting and Dem_EnableDTCSetting ⌈ Calls of the APIs Dem_DisableDTCSetting and Dem_EnableDTCSetting have effect on all events and UDS status bytes assigned the DemEventMemorySet referenced by the clientId of the call. DTCs not referenced from this DemEventMemorySet are not affected. ⌋*(SRS_Diag_04010)*

[SWS_Dem_00079] ⌈ If the function Dem_DisableDTCSetting is called, the Dem shall disable the storage of all events and UDS status byte updates. ⌋ *(SRS_Diag_04010)*

The function Dem_DisableDTCSetting 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_DisableDTCSetting / Dem_EnableDTCSetting) 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 status byte updates, as these are also suppressed during disabled DTC setting.

[SWS_Dem_00080] ⌈ If the function Dem_EnableDTCSetting is called, the Dem shall enable the storage of all events and UDS status byte. ⌋*(SRS_Diag_04115)*

[SWS_Dem_00626] ⌈ When DTC setting is disabled, all status reports from the API Dem_SetEventStatus for those events being assigned to this specific DTC group shall be ignored (no change of UDS 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 7.33 and Figure 7.36).

7.11.2.7 Asynchronous Dcm operations

Most of the APIs of the Dem/Dcm interface depend on NVRAM data. Therefore, an asynchronous processing of these APIs is realized by using the additional “DEM_PENDING” return value.

7.11.3 Interaction with J1939 Diagnostic Manager (J1939)

[SWS_Dem_00971] ⌈ In case the TestFailed bit of an event within an primary event memory changes and a J1939 DTC number is assigned to that event, the function J1939Dcm_DemTriggerOnDTCStatus shall be called for each `DemClient` referencing this `DemEventMemorySet` ⌋*(SRS_Diag_04112)*

7.11.4 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 the monitoring status change and the `DemComponent` information to the FiM.

[SWS_Dem_00029] ⌈ If `DemTriggerFiMReports` is set to True, the `Dem` module shall notify the FiM module (refer to [8]) on each change of the monitoring status (refer also to [SWS_Dem_00016]), by calling the function FiM_DemTriggerOnEventStatus (with same syntax as `<Module>_DemTriggerOnMonitorStatus`) in the context of `Dem_SetEventStatus`. ⌋*(SRS_Diag_04031)*

The Fim uses the function `Dem_GetMonitorStatus` for possible plausibility checks, rebuilding, etc. of inhibition relations.

[SWS_Dem_01189] ⌈ If `DemTriggerFiMReports` is set to True, , the `Dem` shall call Fim_DemInit during `Dem_Init` to trigger the initalization of the permissions inside the `Fim` (independent of trigger or polling mode). The Fim_DemInit shall not be called at any other point in time. ⌋*(SRS_Diag_04031)*

7.11.5 Interaction with NVRAM Manager (NvM)

Typically, the Dem module uses nonvolatile memory blocks (configurable in size by the NVRAM Manager [5]) to achieve permanent storage of UDS 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 implementation specific. 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` 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. ⌋*(SRS_Mem_08549)*

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_SetEventStatus`). 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_01237] ⌈ If a DTC is reported as failed (UDS status testfailed flag changes from 0->1) and gets newly stored to the event memory (no entry existed for this DTC) and this DTC is configured with `DemNvStorageStrategy` = `IMMEDIATE_AT_FIRST_OCCURRENCE`, the Dem shall trigger the immediate storage to NvM. ⌋*(SRS_Diag_04077)*

[SWS_Dem_01238] ⌈ If a DTC is configured with `DemNvStorageStrategy` = `DURING_SHUTDOWN`, any change of its data (DTC, UDS status, event memory entry) shall only be stored to NvM during shutdown. ⌋*(SRS_Diag_04077)*

[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). ⌋*(SRS_Diag_04077)*

Note: For `event memory entries`, which are stored immediately, it is necessary to ensure data consistency (e.g. with the `UDS 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).

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.

7.11.6 Interaction with Default Error Tracer (Det)

The interaction with the `Det` is described in chapter 7.13.

7.11.7 Interaction with Diagnostic Log & Trace (Dlt)

[SWS_Dem_00633] ⌈ The function Dem_DltGetMostRecentFreezeFrameRecordData shall return the data of the most recent freeze frame record of the requested diagnostic event (refer to `DemFreezeFrameClassRef`). Note: This does not include the OBD-II or J1939 freeze frame. ⌋*(SRS_Diag_04202)*

[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_04202)*

[SWS_Dem_00634] ⌈ If `DemTriggerDltReports` is enabled, the Dem module shall provide access on all extended data records (refer to `Dem_DltGetAllExtendedDataRecords`) to the Dlt module. ⌋*(SRS_Diag_04202)*

[SWS_Dem_01298] Dem_DltGetMostRecentFreezeFrameRecordData behavior if no snapshot record is available ⌈ If `Dem_DltGetMostRecentFreezeFrameRecordData` is called and for the provided EventId no UDS snapshot record is stored, the Dem shall return

DEM_NO_SUCH_ELEMENT. The value of the out parameters DestBuffer and BufSize are undefined in this case. ⌋*(SRS_Diag_04202)*

[SWS_Dem_00635] ⌈ The function `Dem_DltGetAllExtendedDataRecords` shall report the data of all extended data records of the requested diagnostic event. ⌋ *(SRS_Diag_04202)*

[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 header information like RecordNumber. The size of the buffer equals to the configuration settings of all respective data elements. ⌋*(SRS_Diag_04202)*

[SWS_Dem_01297] Dem_DltGetAllExtendedDataRecords behavior if no extended data record is available ⌈ If `Dem_DltGetAllExtendedDataRecords` is called and for the provided EventId no extended data record is stored, the Dem shall return DEM_NO_SUCH_ELEMENT. The value of the out parameters DestBuffer and BufSize are undefined in this case. ⌋*(SRS_Diag_04202)*

7.11.8 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 7.7.7).

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.

7.11.9 Scaling information on Service Interfaces

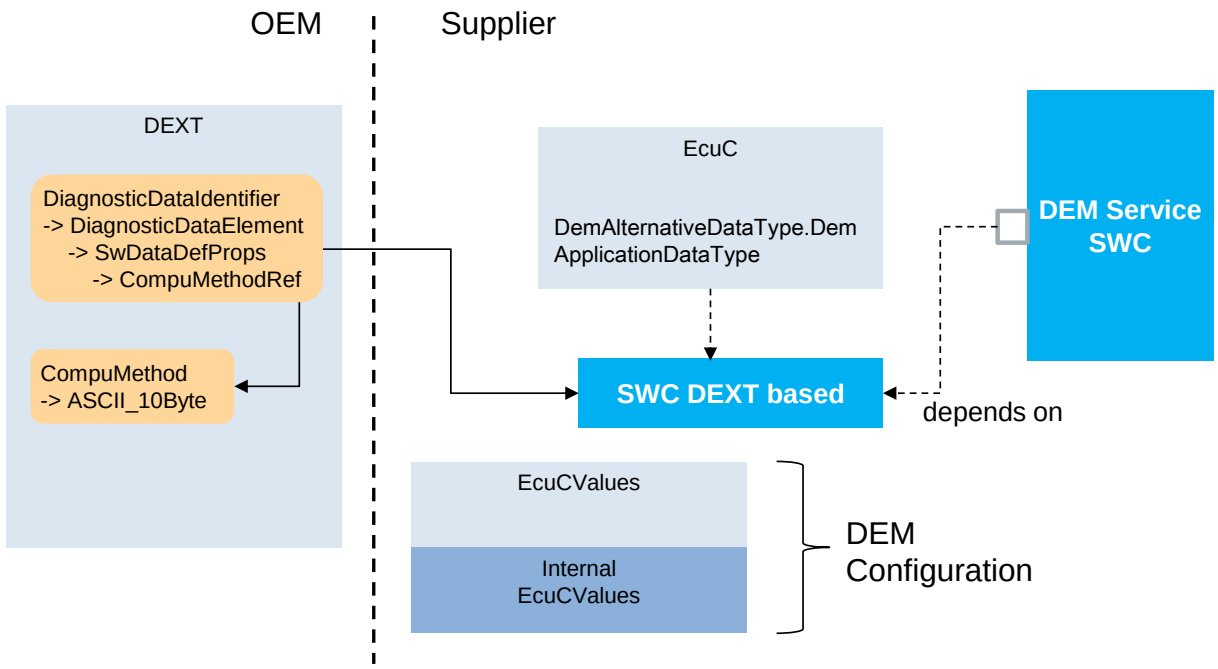
The SoftwareComponentTemplate [23] defines the meta-class SwDataDefProps which covers all properties of a particular data object under various aspects. This chapter describes two alternative workflows how the SwDataDefProps properties such as CompuMethod, DataContraints and Units can be used within the Dem Service SWC and make them there available per `DemExternalSRDataElementClass`.

[SWS_Dem_CONSTR_6117] ⌈ The aggregation of `DemTextTableMapping` at `DemAlternativeDataType` is only valid if the category of the CompuMethod of the DataType referenced by `DemApplicationDataType` has category set to TEXTTABLE or SCALE_LINEAR_AND_TEXTTABLE. ⌋*()*

Work flow 1

This work flow is the use of a EcucForeignReference inside the generated EcuC values. While importing the DEXT [24] information the `DemDiagnosticDataElementRef` is derived and holds a EcucForeignReference to a DiagnosticDataElement in the DEXT [24] file. This EcucForeignReference enables the access to all SwDataDefProps (BaseTyoe, CompuMethod, DatConstr, etc.) of the corresponding DiagnosticDataElement. The container `DemAlternativeDiagnosticDataElement` aggregates this EcucForeignReference. In the process step of generating the corresponding Service SWC all needed content will be copied directly based on the EcucForeignReference from DEXT [24] to the Service SWC. In this work flow the

existence of the DEXT file while the generation of the Service SWC is required.

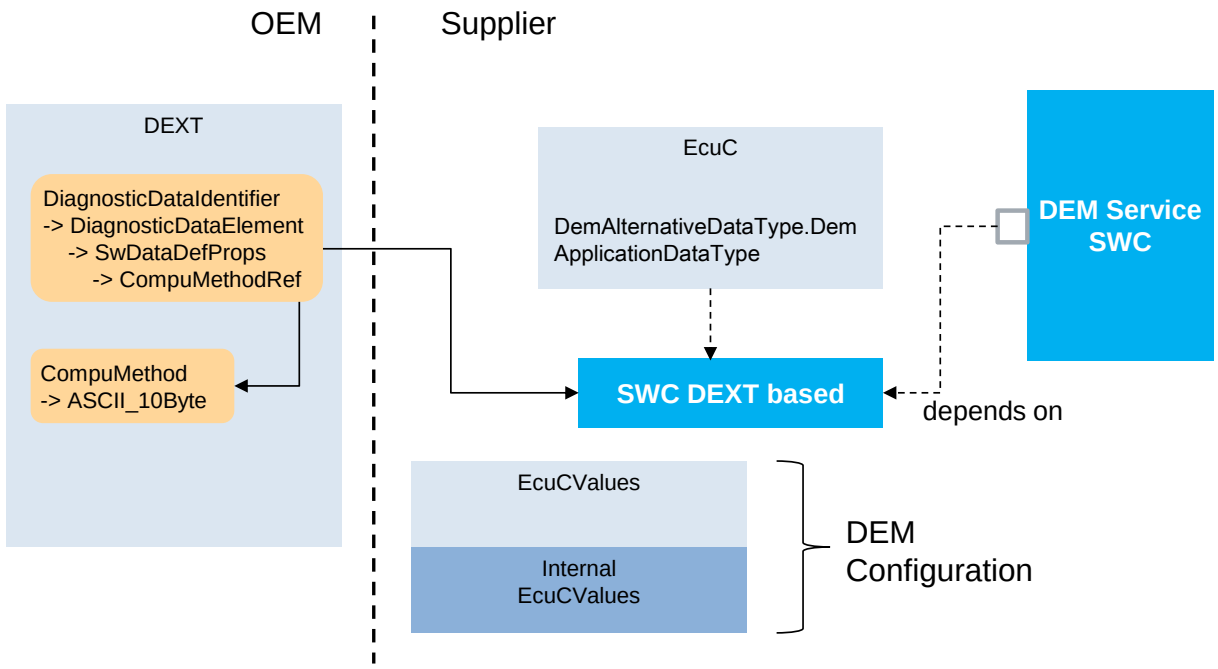


* The DEXT importer generates one SWC which contains all SwDataDefProps and the ECUC parameter (DemAlternativeDataType.DemApplicationDataType) refers to the SWC. Within the SWBuild processing this reference to SWC is taken and the required contents (CompuMethod / DataContraints / Units) are included into the Dem SWCD.

Figure 7.53: Workflow 1

Work flow 2

This work flow is that while importing the DEXT [24] information beside the EcuC values also a SWC fragment is generated. In this SWC fragment all needed SwDataDefProps are directly copied from the DEXT [24] file. Inside the generated EcuC values the EcuC Parameter `DemApplicationDataType` refers to the SWC fragment and enables the access to all SwDataDefProps (BaseTyoe, CompuMethod, DatConstr, etc.). In the process step of generating the corresponding Service SWC all needed content will be included based on the reference from `DemApplicationDataType` to the SWC fragment. In this work flow the existence of the DEXT [24] file while the generation of the Service SWC is not required.



* The DEXT importer generates one SWC which contains all SwDataDefProps and the ECUC parameter (DemAlternativeDataType.DemApplicationDataType) refers to the SWC. Within the SWBuild processing this reference to SWC is taken and the required contents (CompuMethod / DataContraints / Units) are included into the Dem SWCD.

Figure 7.54: Workflow 2

7.12 Version check

For details refer to the chapter 5.1.8 “Version Check” in SWS_BSWGeneral.

7.13 Error classification

7.13.1 Development Errors

[SWS_Dem_00173] ⌈ The errors shown in table 7.7 shall be detectable by the Dem module depending on its configuration (development / production mode). ⌋ *(SWS_BSW_00050, SWS_BSW_00212)*

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_WRONG_CONFIGURATION	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
Dem initialisation failed (refer to SWS_BSW_00151)	Development	DEM_E_INIT_FAILED	0x14
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
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
Dem_DcmGetAvailableOBDMIDs called with invalid OBDMID.	Development	DEM_E_INVALID_OBDMID	0x50

Table 7.7: Types of errors which can be detected by the Dem module

[SWS_Dem_00124] ⌈ If DemDevErrorDetect is set to TRUE and any Dem API excluding :

- Dem_SetEventStatus
- Dem_ResetEventStatus
- Dem_SetEventAvailable
- Dem_ResetEventDebounceStatus
- Dem_GetVersionInfo

is called before Dem has been 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_SetEventStatus 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_SetEventStatus`, `Dem_ResetEventDebounceStatus` and `Dem_GetVersionInfo`, after `Dem_Shutdown` has been called, the Dem module shall set the error code `DEM_E_UNINIT` until `Dem_Init` is called again. ⌋*(SRS_BSW_00337)*

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_GetNextFilteredDTCAndFDC` is called, after `Dem_SetDTCFilter` with `FilterForFaultDetectionCounter` = FALSE was called.

[SWS_Dem_00370] ⌈ If development error detection is enabled and a `Dem` function detects a development error, 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. ⌋*(SRS_Diag_04057)*

7.13.2 Runtime Errors

Type or error	*Relevance*	*Related error code*	*Value [hex]*
The UDS status corresponding to a changed monitor status could not be processed.	Runtime	`DEM_E_UDS_STATUS_PROCESSING_FAILED`	0x21
No valid data for data element available by SW-C or BSW call	Runtime	DEM_E_NODATAAVAILABLE	0x30

Table 7.8: Types of errors which can be detected by the Dem module

7.13.3 Transient Faults

There are no transient faults.

7.13.4 Production Errors

There are no production errors.

7.13.5 Extended Production Errors

There are no extended production errors.

7.14 Error detection

For details refer to the chapter 7.2.2.3 “Configuration” in SWS_BSWGeneral [6].

7.15 Error notification

For details refer to the chapter 7.2.2.4 “Reporting” in SWS_BSWGeneral [6].

7.16 Support for Debugging

For details refer to the chapter 7.1.18 “Debugging support” in SWS_BSWGeneral [6].

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.

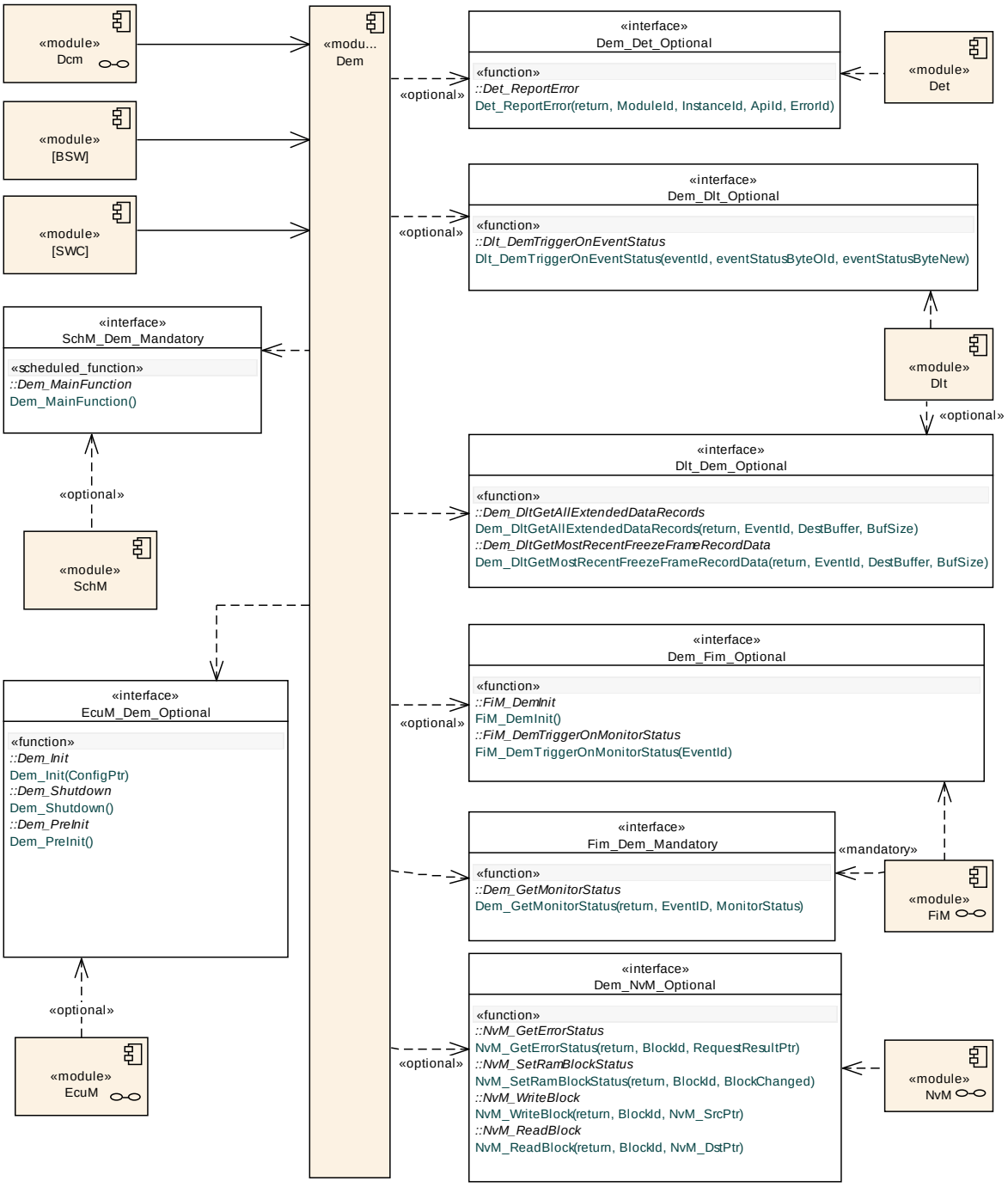


Figure 8.1: Overview of interfaces between the Dem and other BSW modules

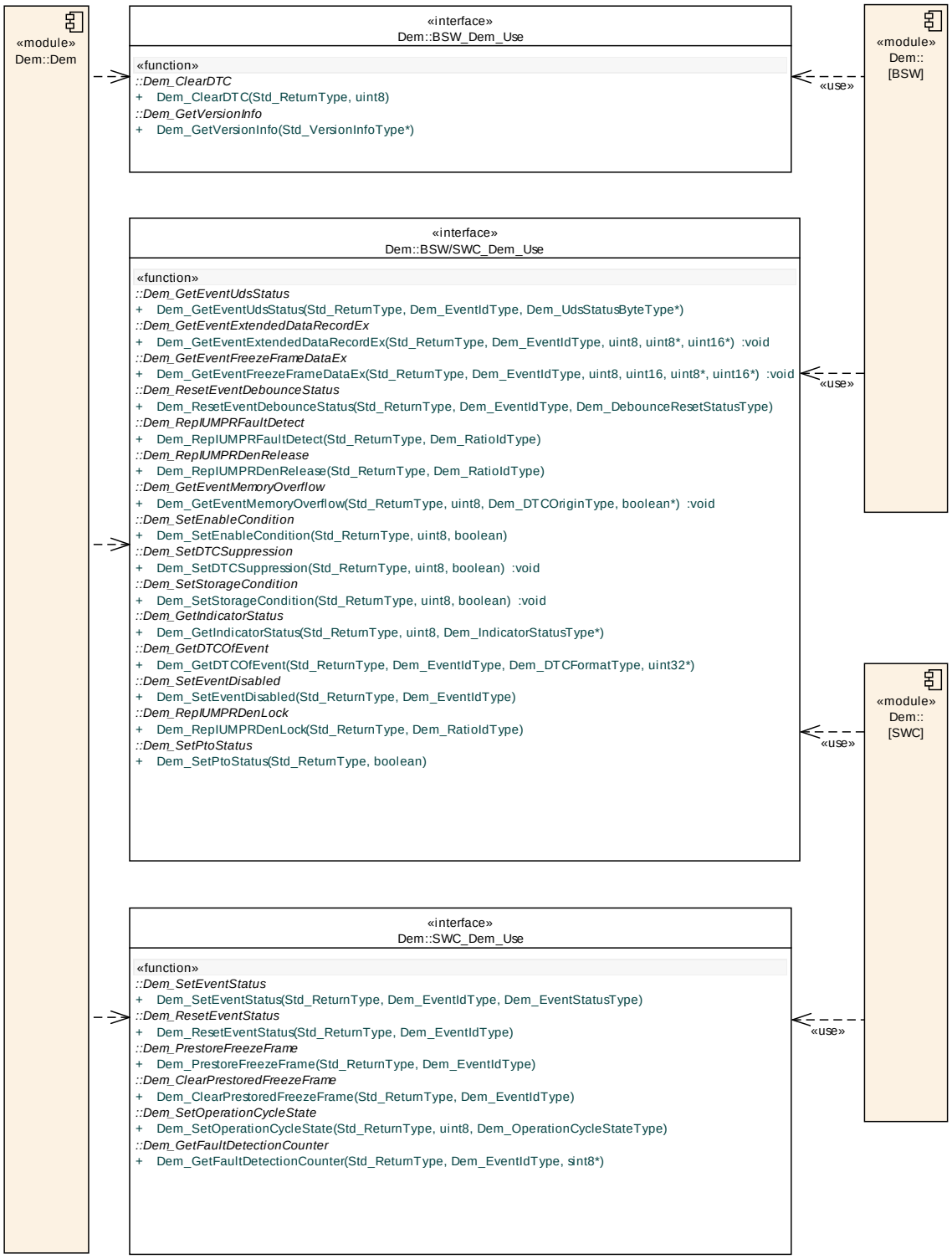


Figure 8.2: Overview of interfaces between the Dem and other BSW modules (in general)

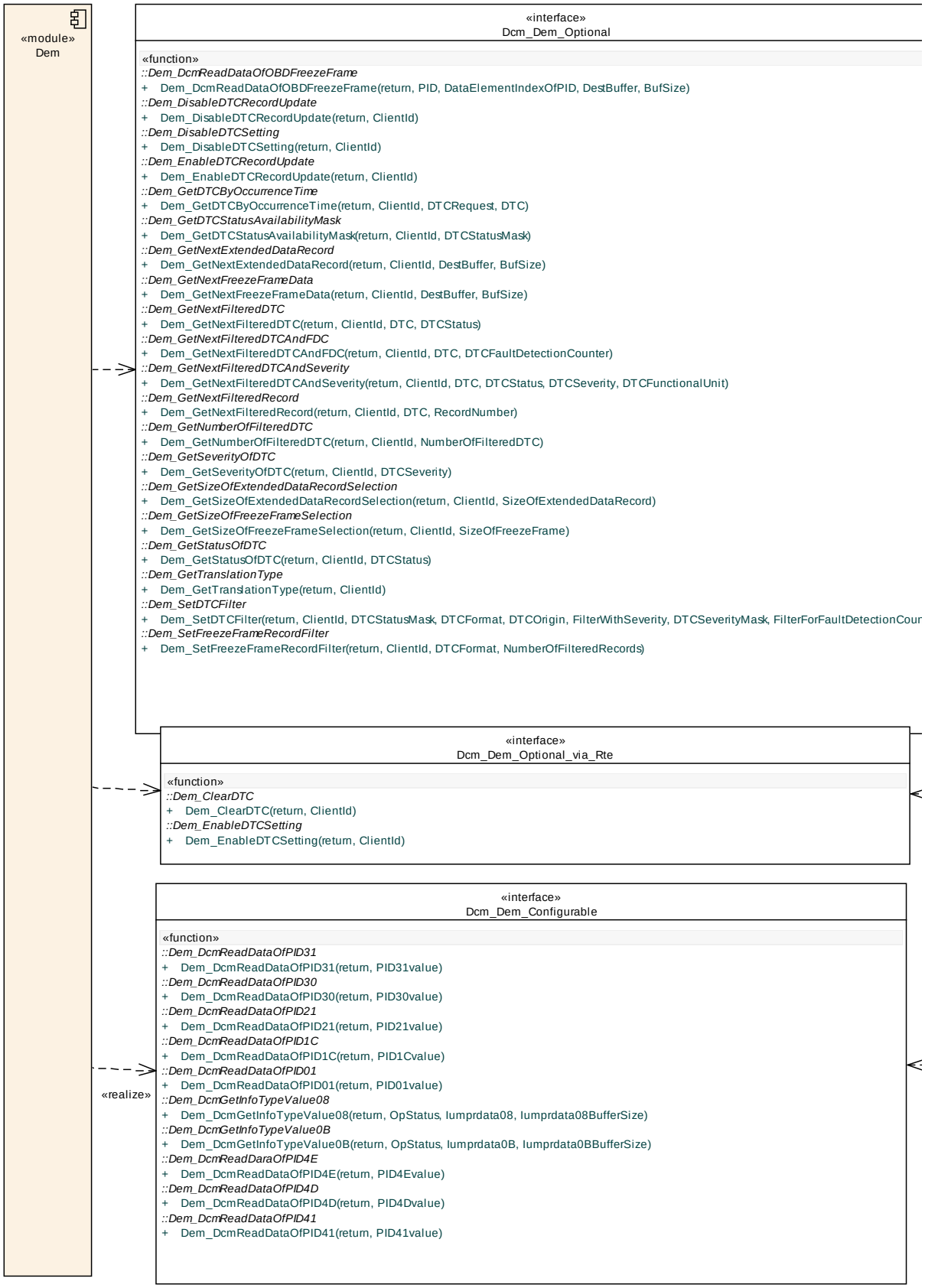


Figure 8.3: Overview of interfaces between the Dem and Dcm

8.1 Imported Types

This section lists all types included from other modules.

[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

⌋(*SRS_BSW_00301*)

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_ComponentIdType

[SWS_Dem_01114] ⌈

Name:	`Dem_ComponentIdType`		
Type:	`uint16`		
Range:	`1..65535`	–	Internal identifier of a monitored component. Remark: 0 is not a valid value
Description:	Identification of a DemComponent by assigned ComponentId. The ComponentId is automatically assigned by the Dem.		

Table 8.2: Dem_ComponentIdType

⌋(*SRS_Diag_04142*)

8.2.1.2 Dem_ConfigType

[SWS_Dem_00924] ⌈

Name:	`Dem_ConfigType`
Type:	`Structure`

Element:		`implementation specific`	–
Description:	This type of the external data structure shall contain the post build initialization data for the Dem.		

Table 8.3: Dem_ConfigType

⌋()

8.2.1.3 Dem_EventIdType

This data type is used in both, C-APIs and service interfaces. It's definition is available in chapter 8.6.1.6.

8.2.1.4 Dem_EventStatusType

This data type is used in both, C-APIs and service interfaces. It's definition is available in chapter 8.6.1.7.

8.2.1.5 Dem_DebouncingStateType

This data type is used in both, C-APIs and service interfaces. It's definition is available in chapter 8.6.1.3.

8.2.1.6 Dem_DebounceResetStatusType

This data type is used in both, C-APIs and service interfaces. It's definition is available in chapter 8.6.1.4.

8.2.1.7 Dem_UdsStatusByteType

This data type is used in both, C-APIs and service interfaces. It's definition is available in chapter 8.6.1.20.

8.2.1.8 Dem_OperationCycleStateType

This data type is used in both, C-APIs and service interfaces. It's definition is available in chapter 8.6.1.15.

8.2.1.9 Dem_IndicatorStatusType

This data type is used in both, C-APIs and service interfaces. It's definition is available in chapter 8.6.1.16.

8.2.1.10 Dem_MonitorStatusType

This data type is used in both, C-APIs and service interfaces. It's definition is available in chapter 8.6.1.14.

8.2.1.11 Dem_DTCKindType

This data type is used in both, C-APIs and service interfaces. It's definition is available in chapter 8.6.1.9.

8.2.1.12 Dem_DTCFormatType

This data type is used in both, C-APIs and service interfaces. It's definition is available in chapter 8.6.1.8.

8.2.1.13 Dem_DTCOriginType

This data type is used in both, C-APIs and service interfaces. It's definition is available in chapter 8.6.1.2.

8.2.1.14 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 8.4: Dem_DTCRequestType

⌋(*SRS_Diag_04102*)

8.2.1.15 Dem_DTCTranslationFormatType

[SWS_Dem_00936] ⌈

Name:	Dem_DTCTranslationFormatType		
Type:	uint8		
Range:	DEM_DTC_TRANSLATION_ISO15031_6	0x00	ISO15031-6 DTC format/SAE J2012-DA_DTCFormat_00 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
	DEM_DTC_TRANSLATION_J2012DA_FORMAT_04	0x04	SAE_J2012-DA_DTCFormat_04 DTC format
Description:	DTC translation format as defined in ISO14229-1.		

Table 8.5: Dem_DTCTranslationFormatType

⌋(*SRS_Diag_04067*)

8.2.1.16 Dem_DTCSeverityType

[SWS_Dem_00937] ⌈

Name:	Dem_DTCSeverityType		
Type:	uint8		
Range:	DEM_SEVERITY_NO_SEVERITY	0x00	No severity information available
	DEM_SEVERITY_WWHOBD_CLASS_NO_CLASS	0x01	No class information
	DEM_SEVERITY_WWHOBD_CLASS_A	0x02	WWH-OBD Class A
	DEM_SEVERITY_WWHOBD_CLASS_B1	0x04	WWH-OBD Class B1
	DEM_SEVERITY_WWHOBD_CLASS_B2	0x08	WWH-OBD Class B2
	DEM_SEVERITY_WWHOBD_CLASS_C	0x10	WWH-OBD Class C
	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:	Type definition of DTCSeverityMask / DTCSeverity byte containing the DTC severity and DTC class information according to ISO 14229-1 Annex D.3. The upper 3 bits (bit 7-5) are used to represent the DTC severity information. The lower 5 bits (bit 4-0) are used to represent the DTC class information.

Table 8.6: Dem_DTCSeverityType

⌋()

8.2.1.17 Dem_RatioIdType

This data type is used in both, C-APIs and service interfaces. It's definition is available in chapter 8.6.1.19.

8.2.1.18 Dem_DTRControlType

This data type is used in both, C-APIs and service interfaces. It's definition is available in chapter 8.6.1.5.

8.2.1.19 Dem_InitMonitorReasonType

This data type is used in both, C-APIs and service interfaces. It's definition is available in chapter 8.6.1.10.

8.2.1.20 Dem_IumprDenomCondIdType

This data type is used in both, C-APIs and service interfaces. It's definition is available in chapter 8.6.1.11.

8.2.1.21 Dem_IumprDenomCondStatusType

This data type is used in both, C-APIs and service interfaces. It's definition is available in chapter 8.6.1.12.

8.2.1.22 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 8.7: Dem_J1939DcmDTCStatusFilterType

⌋(*SRS_Diag_04112*)

8.2.1.23 Dem_J1939DcmSetClearFilterType

[SWS_Dem_00946] ⌈

Name:	Dem_J1939DcmSetClearFilterType		
Type:	uint8		
Range:	DEM_J1939DTC_CLEAR_ACTIVE	0	active DTCs
	DEM_J1939DTC_CLEAR_PREVIOUSLY_ACTIVE	1	previously active DTCs
	DEM_J1939DTC_CLEAR_ACTIVE_AND_PREVIOUSLY_ACTIVE	2	active and previously active DTCs
Description:	The type to distinguish which DTCs gets cleared		

Table 8.8: Dem_J1939DcmSetClearFilterType

⌋(*SRS_Diag_04112*)

8.2.1.24 Dem_J1939DcmSetFreezeFrameFilterType

[SWS_Dem_00947] ⌈

Name:	Dem_J1939DcmSetFreezeFrameFilterType		
Type:	uint8		
Range:	DEM_J1939DCM_FREEZEFRAME	0	FreezeFrame (DM04)
	DEM_J1939DCM_EXPANDED_FREEZEFRAME	1	ExpandedFreezeFrame (DM25)
	DEM_J1939DCM_SPNS_IN_EXPANDED_FREEZEFRAME	2	SPNs in ExpandedFreezeFrame (DM24)
Description:	The type to distinguish which DTCs gets cleared		

Table 8.9: Dem_J1939DcmSetFreezeFrameFilterType

⌋(*SRS_Diag_04010*)

8.2.1.25 Dem_J1939DcmLampStatusType

[SWS_Dem_00948] ⌈

Name:	`Dem_J1939DcmLampStatusType`		
Type:	`Structure`		
Element:	`uint8` `uint8`	`LampStatus` `FlashLampStatus`	lamp status flash lamp status
Description:	For details refer SAE J1939-73		

Table 8.10: Dem_J1939DcmLampStatusType

⌋(*SRS_Diag_04110*)

8.2.1.26 Dem_J1939DcmDiagnosticReadiness1Type

[SWS_Dem_00949] ⌈

Name:	`Dem_J1939DcmDiagnosticReadiness1Type`		
Type:	`Structure`		
Element:	`uint8`	`ActiveTroubleCodes`	Number of active DTCs
	`uint8`	`PreviouslyActive DiagnosticTrouble Codes`	Number of previously active DTCs
	`uint8`	`OBDCompliance`	OBD Compliance
	`uint8`	`Continuously MonitoredSystems Support_Status`	Identifies the continuously monitored system support and status
	`uint8`	`NonContinuously MonitoredSystems Support5`	Identifies the non-continuously monitored systems support (byte5)
	`uint8`	`NonContinuously MonitoredSystems Support6`	Identifies the non-continuously monitored systems support (byte6)
	`uint8`	`NonContinuously MonitoredSystems Status7`	Identifies the non-continuously monitored systems status (byte7)
	`uint8`	`NonContinuously MonitoredSystems Status8`	Identifies the non-continuously monitored systems status (byte8)
Description:	This structure represents all data elemets of the DM05 message. The encoding shall be done acording SAE J1939-73		

Table 8.11: Dem_J1939DcmDiagnosticReadiness1Type

⌋(*SRS_Diag_04113*)

8.2.1.27 Dem_J1939DcmDiagnosticReadiness2Type

[SWS_Dem_00950] ⌈

Name:	Dem_J1939DcmDiagnosticReadiness2Type		
Type:	Structure		
Element:	uint16	DistanceTraveled WhileMILis Activated	The kilometers accumulated while the MIL is activated
	uint16	DistanceSinceDTCs Cleared	Distance accumulated since emission related DTCs were cleared
	uint16	MinutesRunbyEngine WhileMILis Activated	Accumulated count (in minutes) while the MIL is activated (on)
	uint16	TimeSince DiagnosticTrouble CodesCleared	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 8.12: Dem_J1939DcmDiagnosticReadiness2Type

⌋(*SRS_Diag_04113*)

8.2.1.28 Dem_J1939DcmDiagnosticReadiness3Type

[SWS_Dem_00951] ⌈

Name:	Dem_J1939DcmDiagnosticReadiness3Type		
Type:	Structure		
Element:	uint16	TimeSinceEngine Start	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	Continuously MonitoredSystems EnableCompleted Status	Identifies the continuously monitored system enable/-completed support and status.
	uint8	NonContinuously MonitoredSystems EnableStatus5	Enable status of non-continuous monitors this monitoring cycle (byte5)
	uint8	NonContinuously MonitoredSystems EnableStatus6	Enable status of non-continuous monitors this monitoring cycle (byte6)
	uint8	NonContinuously MonitoredSystems7	Completion status of non-continuous monitors this monitoring cycle (byte7)

	uint8	NonContinuously MonitoredSystems8	Completion status of non-continuous monitors this monitoring cycle (byte8)
Description:	This structure represents all data elemets of the DM26 message. The encoding shall be done acording SAE J1939-73		

Table 8.13: Dem_J1939DcmDiagnosticReadiness3Type

⌋(*SRS_Diag_04113*)

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 (inout):	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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemVersionInfoApi)} == true)	

Table 8.14: Dem_GetVersionInfo

⌋(*SRS_BSW_00402*, *SRS_BSW_00407*)

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 (inout):	None
Parameters (out):	None
Return value:	None
Description:	Initializes the internal states necessary to process events reported by BSW-modules.

Table 8.15: 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 (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Initializes or reinitializes this module.	

Table 8.16: 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 (inout):	None
Parameters (out):	None
Return value:	None
Description:	Shuts down this module.

Table 8.17: Dem_Shutdown

⌋(*SRS_BSW_00336*)

8.3.3 Interface BSW modules / SW-Components <=> Dem

8.3.3.1 Dem_ClearDTC

[SWS_Dem_00665] ⌈

Service name:	Dem_ClearDTC	
Syntax:	`Std_ReturnType Dem_ClearDTC( uint8 ClientId )`	
Service ID[hex]:	0x23	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant for different ClientIds, non reentrant for the same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
Parameters (inout):	None	
Parameters (out):	None	

Return value:	Std_ReturnType	E_OK: DTC successfully cleared E_NOT_OK: No DTC selected DEM_WRONG_DTC: Selected DTC value in selected format does not exist or clearing is restricted by configuration to group of all DTCs only. DEM_WRONG_DTCORIGIN: Selected DTCOrigin does not exist DEM_CLEAR_FAILED: DTC clearing failed DEM_CLEAR_BUSY: Another client is currently clearing DTCs. The requested operation will not be started and the caller shall try again at a later moment. DEM_CLEAR_MEMORY_ERROR: An error occurred during erasing a memory location (e.g. if DemClearDTCBehavior is set to DEM_CLRRESP_NON-VOLATILE_FINISH and erasing of non-volatile-block failed). DEM_PENDING: Clearing the DTCs is currently in progress. The caller shall call this function again at a later moment. DEM_BUSY: A different Dem_SelectDTC dependent operation according to SWS_Dem_01253 of this client is currently in progress.
Description:	Clears single DTCs, as well as groups of DTCs.	

Table 8.18: Dem_ClearDTC

⌋(*SRS_Diag_04122*)

8.3.3.2 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 (inout):	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. API Availability: This API will be available only if ({ecuc(Dem/DemConfigSet/DemEventParameter.DemFFPrestorage-Supported)} == true)

Table 8.19: Dem_ClearPrestoredFreezeFrame

⌋(*SRS_Diag_04074*)

8.3.3.3 Dem_GetComponentFailed

[SWS_Dem_01115] ⌈

Service name:	Dem_GetComponentFailed	
Syntax:	`Std_ReturnType Dem_GetComponentFailed( Dem_ComponentIdType ComponentId, boolean* ComponentFailed )`	
Service ID[hex]:	0x2a	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	ComponentId	Identification of a DemComponent
Parameters (inout):	None	
Parameters (out):	ComponentFailed	TRUE: failed FALSE: not failed
Return value:	Std_ReturnType	E_OK: getting "ComponentFailed" was successful E_NOT_OK: getting "ComponentFailed" was not successful
Description:	Gets the failed status of a DemComponent.	

Table 8.20: Dem_GetComponentFailed

⌋(*SRS_Diag_04142*)

8.3.3.4 Dem_GetDTCSelectionResult

[SWS_Dem_91023] ⌈

Service name:	Dem_GetDTCSelectionResult
Syntax:	`Std_ReturnType Dem_GetDTCSelectionResult( uint8 ClientId )`
Service ID[hex]:	0xb8
Sync/Async:	Asynchronous

Reentrancy:	Reentrant for different ClientIds, non reentrant for the same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: The DTC select parameter check is successful and the requested DTC or group of DTC in the selected origin is selected for further operations. E_NOT_OK: No DTC selected DEM_WRONG_DTC: Selected DTC value in selected format does not exist DEM_WRONG_DTCORIGIN: Selected DTCOrigin does not exist DEM_PENDING: Checking the SelectDTC parameters is currently in progress. The caller shall call this function again later. DEM_BUSY: A different Dem_SelectDTC dependent operation according to SWS_Dem_01253 of this client is currently in progress.
Description:	Provides information if the last call to Dem_SelectDTC has selected a valid DTC or group of DTCs.	

Table 8.21: Dem_GetDTCSelectionResult

⌋()

8.3.3.5 Dem_GetDTCSelectionResultForClearDTC

[SWS_Dem_91020] ⌈

Service name:	Dem_GetDTCSelectionResultForClearDTC	
Syntax:	`Std_ReturnType Dem_GetDTCSelectionResultForClearDTC( uint8 ClientId )`	
Service ID[hex]:	0xbb	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant for different ClientIds, non reentrant for the same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
Parameters (inout):	None	
Parameters (out):	None	

Return value:	Std_ReturnType	E_OK: The DTC select parameter check is successful and the requested DTC or group of DTC in the selected origin is selected for further operations. DEM_WRONG_DTC: Selected DTC value in selected format does not exist or a single DTC was selected and Dem only supports to Clear all DTCs. DEM_WRONG_DTCORIGIN: Selected DTCOrigin does not exist. DEM_PENDING: Checking the SelectDTC parameters is currently in progress. The caller shall call this function again later. DEM_BUSY: A different Dem_SelectDTC dependent operation according to SWS_Dem_01253 of this client is currently in progress.
Description:	Provides information if the last call to Dem_SelectDTC has selected a valid DTC or group of DTCs, respecting the settings if the Dem shall clear only all DTCs.	

Table 8.22: Dem_GetDTCSelectionResultForClearDTC

⌋()

8.3.3.6 Dem_GetEventUdsStatus

[SWS_Dem_91008] ⌈

Service name:	Dem_GetEventUdsStatus	
Syntax:	`Std_ReturnType Dem_GetEventUdsStatus( Dem_EventIdType EventId, Dem_UdsStatusByteType* UDSStatusByte )`	
Service ID[hex]:	0xb6	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	EventId	Identification of an event by assigned EventId.
Parameters (inout):	None	
Parameters (out):	UDSStatusByte	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 UDS status byte assigned to the DTC for the event	

Table 8.23: Dem_GetEventUdsStatus

⌋(*SRS_Diag_04067*)

8.3.3.7 Dem_GetMonitorStatus

[SWS_Dem_91007] ⌈

Service name:	Dem_GetMonitorStatus	
Syntax:	`Std_ReturnType Dem_GetMonitorStatus( Dem_EventIdType EventID, Dem_MonitorStatusType* MonitorStatus )`	
Service ID[hex]:	0xb5	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	EventID	Identification of an event by assigned EventId.
Parameters (inout):	None	
Parameters (out):	MonitorStatus	Monitor status byte of the requested event. 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 monitor status was successful, E_NOT_OK: getting the monitor status failed (e.g. an invalid event id was provided).
Description:	Gets the current monitor status for an event.	

Table 8.24: Dem_GetMonitorStatus

⌋()

8.3.3.8 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 (inout):	None	
Parameters (out):	DebouncingState	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 8.25: Dem_GetDebouncingOfEvent

⌋(*SRS_Diag_04068*)

8.3.3.9 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 DTCFormat	Identification of an event by assigned EventId. Defines the output-format of the requested DTC value.
Parameters (inout):	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 8.26: Dem_GetDTCOfEvent

⌋(*SRS_Diag_04075*)

8.3.3.10 Dem_GetDTCSuppression

[SWS_Dem_91025] ⌈

Service name:	Dem_GetDTCSuppression

Syntax:	`Std_ReturnType Dem_GetDTCSuppression( uint8 ClientId, boolean* SuppressionStatus )`	
Service ID[hex]:	0xbc	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module
Parameters (inout):	None	
Parameters (out):	SuppressionStatus	Defines whether the respective DTC is suppressed (TRUE) or enabled (FALSE).
Return value:	Std_ReturnType	E_OK: Operation was successful. E_NOT_OK: Dem_SelectDTC was not called. DEM_WRONG_DTC: No valid DTC or DTC group selected. DEM_WRONG_DTCORIGIN: Wrong DTC origin selected. DEM_PENDING: The requested value is calculated asynchronously and currently not available. The caller can retry later.
Description:	Returns the suppression status of a specific DTC. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemSuppressionSupport)} == DEM_DTC_SUPPRESSION)	

Table 8.27: Dem_GetDTCSuppression

⌋(*SRS_Diag_04154*)

8.3.3.11 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 (inout):	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 8.28: Dem_GetFaultDetectionCounter

⌋()

8.3.3.12 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 (inout):	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 UDS status. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral/DemEventMemorySet/DemIndicator)} != NULL)	

Table 8.29: Dem_GetIndicatorStatus

⌋()

8.3.3.13 Dem_GetEventFreezeFrameDataEx

[SWS_Dem_01191] ⌈

Service name:	Dem_GetEventFreezeFrameDataEx

Syntax:	`Std_ReturnType Dem_GetEventFreezeFrameDataEx(`
	`Dem_EventIdType EventId,`
	`uint8 RecordNumber,`
	`uint16 DataId,`
	`uint8* DestBuffer,`
	`uint16* BufSize`
	`)`	

Service ID[hex]:	0x6e	
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 ISO14229-1. **0xFF means most recent freeze frame record is returned.** 0x00 is only supported if the Dem module supports WWH-OBD (refer to DemOBDSupport)
	DataId	This parameter specifies the DID (ISO14229-1) that shall be copied to the destination buffer.
Parameters (inout):	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 raw hexadecimal values and contains no header-information.
Return value:	Std_ReturnType	E_OK: Operation was successful E_NOT_OK: Operation could not be performed DEM_NO_SUCH_ELEMENT: The requested event data is not currently stored (but the request was valid) OR The requested record number is not supported by the event OR The requested DID is not supported by the freeze frame. DEM_BUFFER_TOO_SMALL: The provided buffer size is too small.
Description:	Gets the data of a freeze frame by event.	

Table 8.30: Dem_GetEventFreezeFrameDataEx

⌋()

8.3.3.14 Dem_GetEventExtendedDataRecordEx

[SWS_Dem_01190] ⌈

Service name:	Dem_GetEventExtendedDataRecordEx

Syntax:	`Std_ReturnType Dem_GetEventExtendedDataRecordEx( Dem_EventIdType EventId, uint8 RecordNumber, uint8* DestBuffer, uint16* BufSize )`	
Service ID[hex]:	0x6d	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	EventId RecordNumber	Identification of an event by assigned EventId. Identification of requested Extended data record. Valid values are between 0x01 and 0xEF as defined in ISO14229-1.
Parameters (inout):	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 E_NOT_OK: Operation could not be performed DEM_NO_SUCH_ELEMENT: The requested event data is not currently stored (but the request was valid) OR the requested record number is not supported by the event. DEM_BUFFER_TOO_SMALL: The provided buffer size is too small.
Description:	Gets the data of an extended data record by event.	

Table 8.31: Dem_GetEventExtendedDataRecordEx

⌋(*SRS_Diag_04205*)

8.3.3.15 Dem_GetEventMemoryOverflow

[SWS_Dem_00559] ⌈

Service name:	Dem_GetEventMemoryOverflow
Syntax:	`Std_ReturnType Dem_GetEventMemoryOverflow( uint8 ClientId, Dem_DTCOriginType DTCOrigin, boolean* OverflowIndication )`
Service ID[hex]:	0x32
Sync/Async:	Synchronous
Reentrancy:	Re-entrant for different ClientIDs, Non re-entrant for same ClientId.

Parameters (in):	ClientId DTCOrigin	DemClientID identifying the DemEventMemorySet indicating in which event memory the overflow has occurred. 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 (inout):	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 8.32: Dem_GetEventMemoryOverflow

⌋(*SRS_Diag_04093*)

8.3.3.16 Dem_GetNumberOfEventMemoryEntries

[SWS_Dem_00652] ⌈

Service name:	Dem_GetNumberOfEventMemoryEntries	
Syntax:	`Std_ReturnType Dem_GetNumberOfEventMemoryEntries( uint8 ClientId, Dem_DTCOriginType DTCOrigin, uint8* NumberOfEventMemoryEntries )`	
Service ID[hex]:	0x35	
Sync/Async:	Synchronous	
Reentrancy:	Re-entrant for different ClientIDs, Non re-entrant for same ClientId.	
Parameters (in):	ClientId DTCOrigin	DemClientID identifying the DemEventMemorySet to which the requested event memory belongs to. 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):	NumberOfEvent MemoryEntries	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 8.33: Dem_GetNumberOfEventMemoryEntries

⌋(*SRS_Diag_04109*)

8.3.3.17 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 DebounceReset Status	Identification of an event by assigned EventId. Freeze or reset the internal debounce counter/timer of the specified event.
Parameters (inout):	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 8.34: Dem_ResetEventDebounceStatus

⌋(*SRS_Diag_04105*)

8.3.3.18 Dem_ResetEventStatus

[SWS_Dem_00185] ⌈

Service name:	Dem_ResetEventStatus	
Syntax:	`Std_ReturnType Dem_ResetEventStatus( Dem_EventIdType EventId )`	
Service ID[hex]:	0x05	
Sync/Async:	Asynchronous	
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: Request to reset the event status was successful accepted. E_NOT_OK: Request to reset the 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 8.35: Dem_ResetEventStatus

⌋()

8.3.3.19 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 (inout):	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. API Availability: This API will be available only if ({ecuc(Dem/DemConfigSet/DemEventParameter.DemFFPrestorage-Supported)} == true)	

Table 8.36: Dem_PrestoreFreezeFrame

⌋(*SRS_Diag_04074*)

8.3.3.20 Dem_SelectDTC

[SWS_Dem_91016] ⌈

Service name:	Dem_SelectDTC
Syntax:	`Std_ReturnType Dem_SelectDTC( uint8 ClientId, uint32 DTC, Dem_DTCFormatType DTCFormat, Dem_DTCOriginType DTCOrigin )`
Service ID[hex]:	0xb7
Sync/Async:	Synchronous
Reentrancy:	Reentrant for different ClientIds, non reentrant for the same ClientId.

Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
	DTC	Defines the DTC in respective format that is selected. If the DTC fits to a DTC group number, the DTC group is selected.
	DTCFormat	Defines the input-format of the provided DTC value.
	DTCOrigin	The event memory of the requested DTC or group of DTC.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: DTC successfully selected. DEM_BUSY: Another Dem_SelectDTC or Dem_SelectDTC dependent operation of this client is currently in progress.
Description:	Selects a DTC or DTC group as target for further operations.	

Table 8.37: Dem_SelectDTC

⌋()

8.3.3.21 Dem_SetComponentAvailable

[SWS_Dem_01117] ⌈

Service name:	Dem_SetComponentAvailable	
Syntax:	`Std_ReturnType Dem_SetComponentAvailable( Dem_ComponentIdType ComponentId, boolean AvailableStatus )`	
Service ID[hex]:	0x2b	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	ComponentId	Identification of a DemComponent.
	AvailableStatus	This parameter specifies whether the respective Component shall be available (TRUE) or not (FALSE).
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Operation was successful E_NOT_OK: Operation failed
Description:	Set the availability of a specific DemComponent.	

Table 8.38: Dem_SetComponentAvailable

⌋(*SRS_Diag_04142*)

8.3.3.22 Dem_SetDTCSuppression

[SWS_Dem_01047] ⌈

Service name:	Dem_SetDTCSuppression	
Syntax:	`Std_ReturnType Dem_SetDTCSuppression( uint8 ClientId, boolean SuppressionStatus )`	
Service ID[hex]:	0x33	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
	SuppressionStatus	This parameter specifies whether the respective DTC shall be disabled (TRUE) or enabled (FALSE).
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: The status of the DTC is correctly provided in the DTCStatus parameter. E_NOT_OK: No DTC selected. DEM_WRONG_DTC: Selected DTC value in selected format does not exist. DEM_WRONG_DTCORIGIN: Selected DTCOrigin does not exist.
Description:	Set the suppression status of a specific DTC. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemSuppressionSupport)} == DEM_DTC_SUPPRESSION)	

Table 8.39: Dem_SetDTCSuppression

⌋()

8.3.3.23 Dem_SetEnableCondition

[SWS_Dem_00201] ⌈

Service name:	Dem_SetEnableCondition	
Syntax:	`Std_ReturnType Dem_SetEnableCondition( uint8 EnableConditionID, boolean ConditionFulfilled )`	
Service ID[hex]:	0x39	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant	
Parameters (in):	EnableConditionID	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 (inout):	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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral/DemEnableCondition)} != NULL)	

Table 8.40: Dem_SetEnableCondition

⌋(*SRS_Diag_04095*)

8.3.3.24 Dem_SetEventAvailable

[SWS_Dem_01080] ⌈

Service name:	Dem_SetEventAvailable	
Syntax:	`Std_ReturnType Dem_SetEventAvailable( Dem_EventIdType EventId, boolean AvailableStatus )`	
Service ID[hex]:	0x37	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant for different EventIds. Non reentrant for the same EventId.	
Parameters (in):	EventId AvailableStatus	Identification of an event by assigned EventId. This parameter specifies whether the respective Event shall be available (TRUE) or not (FALSE).
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Request to set the availability status was successful. E_NOT_OK: Request to set the availability status not accepted.
Description:	Set the available status of a specific Event.	

Table 8.41: Dem_SetEventAvailable

⌋(*SRS_Diag_04126*)

8.3.3.25 Dem_SetEventFailureCycleCounterThreshold

[SWS_Dem_91004] ⌈

Service name:	Dem_SetEventFailureCycleCounterThreshold	
Syntax:	`Std_ReturnType Dem_SetEventFailureCycleCounterThreshold( Dem_EventIdType EventId, uint8 FailureCycleCounterThreshold )`	
Service ID[hex]:	0x57	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant for different EventIds. Non reentrant for the same EventId.	
Parameters (in):	EventId FailureCycleCounterThreshold	Identification of an event by assigned EventId. Failure cycle counter threshold of event to be set.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Change of threshold was successful. E_NOT_OK: Threshold cannot be changed as DemEventFailureCycleCounterThresholdAdaptable is set to FALSE for this event.
Description:	Set the failure confirmation threshold of an event.	

Table 8.42: Dem_SetEventFailureCycleCounterThreshold

⌋()

8.3.3.26 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 EventStatus	Identification of an event by assigned EventId. Monitor test result
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: set of event status was successful E_NOT_OK: Event status setting or processing failed or could not be accepted.
Description:	Called by SW-Cs or BSW modules to report monitor status information to the Dem. BSW modules calling Dem_SetEventStatus can safely ignore the return value.	

Table 8.43: Dem_SetEventStatus

⌋()

8.3.3.27 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 CycleState	Identification of operation cycle, like power cycle, driving cycle. 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 8.44: Dem_SetOperationCycleState

⌋(*SRS_Diag_04076*)

8.3.3.28 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 (inout):	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 8.45: Dem_GetOperationCycleState

⌋()

8.3.3.29 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 ConditionFulfilled	This parameter identifies the storage condition. 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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral/DemStorageCondition)} != NULL)	

Table 8.46: Dem_SetStorageCondition

⌋()

8.3.3.30 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:	Asynchronous	
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: Request to set the WIR status was successful. E_NOT_OK: Request to set the WIR status was not 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 8.47: Dem_SetWIRStatus

⌋()

8.3.4 Interface Dcm <=> Dem

8.3.4.1 Access DTCs and Status Information

8.3.4.1.1 Dem_GetTranslationType

[SWS_Dem_00230] ⌈

Service name:	Dem_GetTranslationType	

Syntax:	`Dem_DTCTranslationFormatType Dem_GetTranslationType(`
	`uint8 ClientId`
	`)`	

Service ID[hex]:	0x3c	
Sync/Async:	Synchronous	
Reentrancy:	Re-entrant for different ClientIDs, Non re-entrant for same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
Parameters (inout):	None	

Parameters (out):	None	
Return value:	Dem_DTCTranslationFormatType	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 8.48: Dem_GetTranslationType

⌋(*SRS_Diag_04010*)

8.3.4.1.2 Dem_GetDTCStatusAvailabilityMask

[SWS_Dem_00213] ⌈

Service name:	Dem_GetDTCStatusAvailabilityMask	
Syntax:	`Std_ReturnType Dem_GetDTCStatusAvailabilityMask( uint8 ClientId, Dem_UdsStatusByteType* DTCStatusMask )`	
Service ID[hex]:	0x16	
Sync/Async:	Synchronous	
Reentrancy:	Re-entrant for different ClientIDs, Non re-entrant for same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
Parameters (inout):	None	
Parameters (out):	DTCStatusMask	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 8.49: Dem_GetDTCStatusAvailabilityMask

⌋(*SRS_Diag_04067*, *SRS_Diag_04010*)

8.3.4.1.3 Dem_GetStatusOfDTC

[SWS_Dem_00212] ⌈

Service name:	Dem_GetStatusOfDTC
Syntax:	`Std_ReturnType Dem_GetStatusOfDTC( uint8 ClientId, uint8* DTCStatus )`

Service ID[hex]:	0x15	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant for different ClientIds, non reentrant for the same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
Parameters (inout):	None	
Parameters (out):	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:	Std_ReturnType	E_OK: The status of the DTC is correctly provided in the DTCStatus parameter. E_NOT_OK: No DTC selected DEM_WRONG_DTC: Selected DTC value in selected format does not exist DEM_WRONG_DTCORIGIN: Selected DTCOrigin does not exist DEM_PENDING: Retrieving the DTC status is currently in progress. The caller shall call this function again at a later moment. DEM_NO_SUCH_ELEMENT - Selected DTC does not have an assigned DTC status. DEM_BUSY: A different Dem_SelectDTC dependent operation according to SWS_Dem_01253 of this client is currently in progress.
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_WRONG_DTC.	

Table 8.50: Dem_GetStatusOfDTC

⌋(*SRS_Diag_04066*, *SRS_Diag_04067*)

8.3.4.1.4 Dem_GetSeverityOfDTC

[SWS_Dem_00232] ⌈

Service name:	Dem_GetSeverityOfDTC	

Syntax:	`Std_ReturnType Dem_GetSeverityOfDTC(`
	`uint8 ClientId,`
	`Dem_DTCSeverityType* DTCSeverity`
	`)`	

Service ID[hex]:	0x0e	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant for different ClientIds, non reentrant for the same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.

Parameters (inout):	None	
Parameters (out):	DTCSeverity	This parameter contains the DTCSeverity according to ISO 14229-1.
Return value:	Std_ReturnType	E_OK: The DTC severity is correctly provided in the DTCSeverity parameter. E_NOT_OK: No DTC selected DEM_WRONG_DTC: Selected DTC value in selected format does not exist DEM_WRONG_DTCORIGIN: Selected DTCOrigin does not exist DEM_PENDING: Retrieving the DTC is currently in progress. The caller shall call this function again at a later moment. DEM_BUSY: A different Dem_SelectDTC dependent operation according to SWS_Dem_01253 of this client is currently in progress.
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 8.51: Dem_GetSeverityOfDTC

⌋(*SRS_Diag_04071*)

8.3.4.1.5 Dem_GetFunctionalUnitOfDTC

[SWS_Dem_00594] ⌈

Service name:	Dem_GetFunctionalUnitOfDTC	

Syntax:	`Std_ReturnType Dem_GetFunctionalUnitOfDTC(`
	`uint8 ClientId,`
	`uint8* DTCFunctionalUnit`
	`)`	

Service ID[hex]:	0x34	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant for different ClientIds, non reentrant for the same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
Parameters (inout):	None	
Parameters (out):	DTCFunctionalUnit	Functional unit value of this DTC

Return value:	Std_ReturnType	E_OK: The DTC functional unit is correctly provided in the DTCSeverity parameter. E_NOT_OK: No DTC selected DEM_WRONG_DTC: Selected DTC value in selected format does not exist DEM_WRONG_DTCORIGIN: Selected DTCOrigin does not exist DEM_PENDING: Retrieving the DTC functional unit is currently in progress. The caller shall call this function again at a later moment. DEM_BUSY: A different Dem_SelectDTC dependent operation according to SWS_Dem_01253 of this client is currently in progress.
Description:	Gets the functional unit of the requested DTC.	

Table 8.52: Dem_GetFunctionalUnitOfDTC

⌋(*SRS_Diag_04156*)

8.3.4.1.6 Dem_SetDTCFilter

[SWS_Dem_00208] ⌈

Service name:	Dem_SetDTCFilter	

Syntax:	`Std_ReturnType Dem_SetDTCFilter(`
	`uint8 ClientId,`
	`uint8 DTCStatusMask,`
	`Dem_DTCFormatType DTCFormat,`
	`Dem_DTCOriginType DTCOrigin,`
	`boolean FilterWithSeverity,`
	`Dem_DTCSeverityType DTCSeverityMask,`
	`boolean FilterForFaultDetectionCounter`
	`)`	

Service ID[hex]:	0x13	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different ClientIds, non reentrant for the same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
	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

	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_MIRROR_MEMORY" is passed, but no mirror 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	Contains the DTCSeverityMask according to ISO14229-1.
	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 (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_NOT_OK: Indicates a wrong DTCOrigin or DTCFormat
Description:	Sets the DTC Filter. The server shall perform a bit-wise logical AND-ing operation between the parameter DTCStatusMask and the current UDS 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 8.53: Dem_SetDTCFilter

⌋(*SRS_Diag_04205*)

8.3.4.1.7 Dem_GetNumberOfFilteredDTC

[SWS_Dem_00214] ⌈

Service name:	Dem_GetNumberOfFilteredDTC	
Syntax:	`Std_ReturnType Dem_GetNumberOfFilteredDTC( uint8 ClientId, uint16* NumberOfFilteredDTC )`	
Service ID[hex]:	0x17	
Sync/Async:	Asynchronous	
Reentrancy:	Re-entrant for different ClientIDs, Non re-entrant for same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
Parameters (inout):	None	
Parameters (out):	NumberOfFilteredDTC	The number of DTCs matching the defined status mask.
Return value:	Std_ReturnType	E_OK: Getting number of filtered DTCs was successful E_NOT_OK: No DTC filter set DEM_PENDING: The requested operation is currently in progress. The caller shall call this function again at a later moment.
Description:	Gets the number of a filtered DTC.	

Table 8.54: Dem_GetNumberOfFilteredDTC

⌋()

8.3.4.1.8 Dem_GetNextFilteredDTC

[SWS_Dem_00215] ⌈

Service name:	Dem_GetNextFilteredDTC	
Syntax:	`Std_ReturnType Dem_GetNextFilteredDTC( uint8 ClientId, uint32* DTC, uint8* DTCStatus )`	
Service ID[hex]:	0x18	
Sync/Async:	Synchronous or Asynchronous	
Reentrancy:	Re-entrant for different ClientIDs, Non re-entrant for same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
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.
Return value:	Std_ReturnType	E_OK: Returned next filtered element E_NOT_OK: No DTC filter set DEM_NO_SUCH_ELEMENT: No further element matching the filter criteria found DEM_PENDING: The requested operation is currently in progress. The caller shall call this function again at a later moment. Note that according to SWS_Dem_00653 this return value is not always allowed.
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 8.55: Dem_GetNextFilteredDTC

⌋()

8.3.4.1.9 Dem_GetNextFilteredDTCAndFDC

[SWS_Dem_00227] ⌈

Service name:	Dem_GetNextFilteredDTCAndFDC	
Syntax:	`Std_ReturnType Dem_GetNextFilteredDTCAndFDC( uint8 ClientId, uint32* DTC, sint8* DTCFaultDetectionCounter )`	
Service ID[hex]:	0x3b	
Sync/Async:	Asynchronous	
Reentrancy:	Re-entrant for different ClientIDs, Non re-entrant for same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
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.

	DTCFaultDetection Counter	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:	Std_ReturnType	E_OK: Returned next filtered element E_NOT_OK: No DTC filter set DEM_NO_SUCH_ELEMENT: No further element matching the filter criteria found DEM_PENDING: The requested operation is asynchronously processed is currently in progress. The caller shall call this function again at a later moment.
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 8.56: Dem_GetNextFilteredDTCAndFDC

⌋()

8.3.4.1.10 Dem_GetNextFilteredDTCAndSeverity

[SWS_Dem_00281] ⌈

Service name:	Dem_GetNextFilteredDTCAndSeverity	
Syntax:	`Std_ReturnType Dem_GetNextFilteredDTCAndSeverity( uint8 ClientId, uint32* DTC, uint8* DTCStatus, Dem_DTCSeverityType* DTCSeverity, uint8* DTCFunctionalUnit )`	
Service ID[hex]:	0x3d	
Sync/Async:	Asynchronous	
Reentrancy:	Re-entrant for different ClientIDs, Non re-entrant for same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
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.
	DTCFunctionalUnit	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:	Std_ReturnType	E_OK: Returned next filtered element E_NOT_OK: No DTC filter set DEM_NO_SUCH_ELEMENT: No further element matching the filter criteria found DEM_PENDING: The requested operation is currently in progress. The caller shall call this function again at a later moment.
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 8.57: Dem_GetNextFilteredDTCAndSeverity

⌋(*SRS_Diag_04010*)

8.3.4.1.11 Dem_SetFreezeFrameRecordFilter

[SWS_Dem_00209] ⌈

Service name:	Dem_SetFreezeFrameRecordFilter	
Syntax:	`Std_ReturnType Dem_SetFreezeFrameRecordFilter( uint8 ClientId, Dem_DTCFormatType DTCFormat, uint16* NumberOfFilteredRecords )`	
Service ID[hex]:	0x3f	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different ClientIds, non reentrant for the same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
	DTCFormat	Defines the output-format of the requested DTC values
Parameters (inout):	None	
Parameters (out):	NumberOfFiltered Records	Number of freeze frame records currently stored in the event memory.
Return value:	Std_ReturnType	Status of the operation to (re-)set a freeze frame record filter. E_OK: Filter is accepted, E_NOT_OK: Wrong filter selected
Description:	Sets a freeze frame record filter.	

Table 8.58: Dem_SetFreezeFrameRecordFilter

⌋()

8.3.4.1.12 Dem_GetNextFilteredRecord

[SWS_Dem_00224] ⌈

Service name:	Dem_GetNextFilteredRecord	
Syntax:	`Std_ReturnType Dem_GetNextFilteredRecord( uint8 ClientId, uint32* DTC, uint8* RecordNumber )`	
Service ID[hex]:	0x3a	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant for different ClientIds, non reentrant for the same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
Parameters (inout):	None	
Parameters (out):	DTC	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 E_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 E_OK this parameter does not contain valid data.
Return value:	Std_ReturnType	Status of the operation to retrieve a DTC and its associated snapshot record number from the Dem. E_OK: Returned next filtered element DEM_NO_SUCH_ELEMENT: No further element (matching the filter criteria) found DEM_PENDING: The requested value is calculated asynchronously and currently not available. The caller can retry later. Only used by asynchronous interfaces.
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 8.59: Dem_GetNextFilteredRecord

⌋()

8.3.4.1.13 Dem_GetDTCByOccurrenceTime

[SWS_Dem_00218] ⌈

Service name:	Dem_GetDTCByOccurrenceTime

Syntax:	`Std_ReturnType Dem_GetDTCByOccurrenceTime( uint8 ClientId, Dem_DTCRequestType DTCRequest, uint32* DTC )`	
Service ID[hex]:	0x19	
Sync/Async:	Synchronous	
Reentrancy:	Re-entrant for different ClientIDs, Non re-entrant for same ClientId.	
Parameters (in):	ClientId DTCRequest	Unique client id, assigned to the instance of the calling module. This parameter defines the request type of the DTC.
Parameters (inout):	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:	Std_ReturnType	E_OK: get of DTC was successful E_NOT_OK: the call was not successful DEM_NO_SUCH_ELEMENT: The requested element is not stored
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 8.60: Dem_GetDTCByOccurrenceTime

⌋(*SRS_Diag_04072*)

8.3.4.2 Access extended data records and FreezeFrame data

8.3.4.2.1 Dem_DisableDTCRecordUpdate

[SWS_Dem_00233] ⌈

Service name:	Dem_DisableDTCRecordUpdate	
Syntax:	`Std_ReturnType Dem_DisableDTCRecordUpdate( uint8 ClientId )`	
Service ID[hex]:	0x1a	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant for different ClientIds, non reentrant for the same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
Parameters (inout):	None	
Parameters (out):	None	

Return value:	Std_ReturnType	E_OK: Event memory update successfully disabled E_NOT_OK: No DTC selected DEM_WRONG_DTC: Selected DTC value in selected format does not exist or a group of DTC was selected DEM_E_WRONG_CONDITION: Required conditions for the respective API call are not fulfilled. DEM_WRONG_DTCORIGIN: Selected DTCOrigin does not exist DEM_PENDING: Disabling the DTC record update is currently in progress. The caller shall call this function again at a later moment. DEM_BUSY: A different Dem_SelectDTC dependent operation according to SWS_Dem_01253 of this client is currently in progress.
Description:	Disables the event memory update of a specific DTC (only one at one time).	

Table 8.61: Dem_DisableDTCRecordUpdate

⌋(*SRS_Diag_04095*)

8.3.4.2.2 Dem_EnableDTCRecordUpdate

[SWS_Dem_00234] ⌈

Service name:	Dem_EnableDTCRecordUpdate	
Syntax:	`Std_ReturnType Dem_EnableDTCRecordUpdate( uint8 ClientId )`	
Service ID[hex]:	0x1b	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant for different ClientIds, non reentrant for the same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: DTC record successfully updated. E_NOT_OK: No DTC selected. DEM_BUSY: A different Dem_SelectDTC dependent operation according to SWS_Dem_01253 of this client is currently in progress.
Description:	Enables the event memory update of the DTC disabled by Dem_DisableDTCRecordUpdate() before.	

Table 8.62: Dem_EnableDTCRecordUpdate

⌋(*SRS_Diag_04074*)

8.3.4.2.3 Dem_GetSizeOfExtendedDataRecordSelection

[SWS_Dem_00240] ⌈

Service name:	Dem_GetSizeOfExtendedDataRecordSelection	
Syntax:	`Std_ReturnType Dem_GetSizeOfExtendedDataRecordSelection( uint8 ClientId, uint16* SizeOfExtendedDataRecord )`	
Service ID[hex]:	0x21	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant for different ClientIds, non reentrant for the same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
Parameters (inout):	None	
Parameters (out):	SizeOfExtended DataRecord	Size of the requested extended data record(s) including record number. The format for a single ExtendedDataRecord is: {RecordNumber, data[1], ..., data[N]}
Return value:	Std_ReturnType	E_OK: Size returned successfully E_NOT_OK : selection function is not called. DEM_PENDING: The requested value is calculated asynchronously and currently not available. The caller can retry later. DEM_WRONG_DTC: DTC value not existing DEM_WRONG_DTCORIGIN: Wrong DTC origin DEM_NO_SUCH_ELEMENT: Record number is not supported by configuration and therefore invalid
Description:	Gets the size of Extended Data Record by DTC selected by the call of Dem_SelectExtendedDataRecord.	

Table 8.63: Dem_GetSizeOfExtendedDataRecordSelection

⌋(*SRS_Diag_04066*)

8.3.4.2.4 Dem_GetSizeOfFreezeFrameSelection

[SWS_Dem_00238] ⌈

Service name:	Dem_GetSizeOfFreezeFrameSelection	
Syntax:	`Std_ReturnType Dem_GetSizeOfFreezeFrameSelection( uint8 ClientId, uint16* SizeOfFreezeFrame )`	
Service ID[hex]:	0x1f	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant for different ClientIds, non reentrant for the same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
Parameters (inout):	None	

Parameters (out):	SizeOfFreezeFrame	Number of bytes in the requested freeze frame record.
Return value:	Std_ReturnType	E_OK: Size returned successfully E_NOT_OK : selection function is not called. DEM_PENDING: The requested value is calculated asynchronously and currently not available. The caller can retry later. DEM_WRONG_DTC: DTC value not existing DEM_WRONG_DTCORIGIN: Wrong DTC origin DEM_NO_SUCH_ELEMENT: Record number is not supported by configuration and therefore invalid
Description:	Gets the size of freeze frame data by DTC selected by the call of Dem_SelectFreezeFrameData.	

Table 8.64: Dem_GetSizeOfFreezeFrameSelection

⌋(*SRS_Diag_04066*)

8.3.4.2.5 Dem_GetNextExtendedDataRecord

[SWS_Dem_00239] ⌈

Service name:	Dem_GetNextExtendedDataRecord	
Syntax:	`Std_ReturnType Dem_GetNextExtendedDataRecord( uint8 ClientId, uint8* DestBuffer, uint16* BufSize )`	
Service ID[hex]:	0x20	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant for different ClientIds, non reentrant for the same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
Parameters (inout):	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: {ExtendedDataRecordNumber, data[0], data[1], ..., data[n]}

Return value:	Std_ReturnType	E_OK: Size and buffer successfully returned. E_NOT_OK : selection function is not called. DEM_BUFFER_TOO_SMALL: provided buffer size too small. DEM_PENDING: The requested value is calculated asynchronously and currently not available. The caller can retry later. DEM_WRONG_DTC: DTC value not existing DEM_WRONG_DTCORIGIN: Wrong DTC origin DEM_NO_SUCH_ELEMENT: Found no (further) element matching the filter criteria
Description:	Gets extended data record for the DTC selected by Dem_SelectExtendedDataRecord. The function stores the data in the provided DestBuffer.	

Table 8.65: Dem_GetNextExtendedDataRecord

⌋(*SRS_Diag_04066*)

8.3.4.2.6 Dem_GetNextFreezeFrameData

[SWS_Dem_00236] ⌈

Service name:	Dem_GetNextFreezeFrameData	
Syntax:	`Std_ReturnType Dem_GetNextFreezeFrameData( uint8 ClientId, uint8* DestBuffer, uint16* BufSize )`	
Service ID[hex]:	0x1d	
Sync/Async:	Asynchronous	
Reentrancy:	Reentrant for different ClientIds, non reentrant for the same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
Parameters (inout):	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:	Std_ReturnType	E_OK: Size and buffer successfully returned. DEM_BUFFER_TOO_SMALL: provided buffer size too small DEM_PENDING: The requested value is calculated asynchronously and currently not available. The caller can retry later. DEM_WRONG_DTC: DTC value not existing E_NOT_OK : selection function is not called. DEM_WRONG_DTCORIGIN: Wrong DTC origin DEM_NO_SUCH_ELEMENT: Found no (further) element matching the filter criteria
Description:	Gets freeze frame data by the DTC selected by Dem_SelectFreezeFrameData. The function stores the data in the provided DestBuffer.	

Table 8.66: Dem_GetNextFreezeFrameData

⌋(*SRS_Diag_04066*)

8.3.4.2.7 Dem_SelectExtendedDataRecord

[SWS_Dem_91017] ⌈

Service name:	Dem_SelectExtendedDataRecord	
Syntax:	`Std_ReturnType Dem_SelectExtendedDataRecord( uint8 ClientId, uint8 ExtendedDataNumber )`	
Service ID[hex]:	0xba	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different ClientIds, non reentrant for the same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
	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 (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Extended data record successfully selected. DEM_WRONG_DTC: Selected DTC value in selected format does not exist. DEM_WRONG_DTCORIGIN: Selected DTCOrigin does not exist. DEM_PENDING: Selecting the extended data record is currently in progress. The caller shall call this function again at a later moment. DEM_BUSY: A different Dem_SelectDTC dependent operation according to SWS_Dem_01253 of this client is currently in progress.

Description:	Sets the filter to be used by Dem_GetNextExtendedDataRecord and Dem_GetSizeOfExtendedDataRecordSelection.

Table 8.67: Dem_SelectExtendedDataRecord

⌋()

8.3.4.2.8 Dem_SelectFreezeFrameData

[SWS_Dem_91015] ⌈

Service name:	Dem_SelectFreezeFrameData	
Syntax:	`Std_ReturnType Dem_SelectFreezeFrameData( uint8 ClientId, uint8 RecordNumber )`	
Service ID[hex]:	0xb9	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different ClientIds, non reentrant for the same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
	RecordNumber	Unique identifier for a snapshot record as defined in ISO 14229-1. The value 0xFF is a placeholder referencing all snapshot records of the addressed DTC. The value 0x00 indicates the DTC-specific WWH-OBD snapshot record.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Freeze frame data successfully selected. DEM_WRONG_DTC: Selected DTC value in selected format does not exist. DEM_WRONG_DTCORIGIN: Selected DTCOrigin does not exist. DEM_PENDING: Selecting the freeze frame is currently in progress. The caller shall call this function again at a later moment. DEM_BUSY: A different Dem_SelectDTC dependent operation according to SWS_Dem_01253 of this client is currently in progress.
Description:	Sets the filter to be used by Dem_GetNextFreezeFrameData and Dem_GetSizeOfFreezeFrameSelection.	

Table 8.68: Dem_SelectFreezeFrameData

⌋()

8.3.4.3 DTC storage

8.3.4.3.1 Dem_DisableDTCSetting

[SWS_Dem_00242] ⌈

Service name:	Dem_DisableDTCSetting	
Syntax:	`Std_ReturnType Dem_DisableDTCSetting( uint8 ClientId )`	
Service ID[hex]:	0x24	
Sync/Async:	Asynchronous	
Reentrancy:	Re-entrant for different ClientIDs, Non re-entrant for same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Returned next filtered element DEM_PENDING: The requested operation is currently in progress. The caller shall call this function again at a later moment.
Description:	Disables the DTC setting for all DTCs assigned to the DemEventMemorySet of the addressed client.	

Table 8.69: Dem_DisableDTCSetting

⌋*(SRS_Diag_04150)*

8.3.4.3.2 Dem_EnableDTCSetting

[SWS_Dem_00243] ⌈

Service name:	Dem_EnableDTCSetting	
Syntax:	`Std_ReturnType Dem_EnableDTCSetting( uint8 ClientId )`	
Service ID[hex]:	0x25	
Sync/Async:	Asynchronous	
Reentrancy:	Re-entrant for different ClientIDs, Non re-entrant for same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: The operation was successful; DEM_PENDING: The started operation is currently in progress. The caller shall call this function again at a later moment.
Description:	(Re)-Enables the DTC setting for all DTCs assigned to the DemEventMemorySet of the addressed client.	

Table 8.70: Dem_EnableDTCSetting

⌋(*SRS_Diag_04095*, *SRS_Diag_04158*, *SRS_Diag_04150*)

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* Iumprdata08, uint8* Iumprdata08BufferSize )`	
Service ID[hex]:	0x6b	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	OpStatus	Only DCM_INITIAL will appear, because this API behaves synchronous.
Parameters (inout):	Iumprdata08BufferSize	The maximum number of data bytes that can be written to the Iumprdata08 Buffer.
Parameters (out):	Iumprdata08	Buffer containing the number of data elements (as defined in ISO-15031-5) and 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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.71: 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, uint8* Iumprdata0BBufferSize )`
Service ID[hex]:	0x6c

Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	OpStatus	Only DCM_INITIAL will appear, because this API behaves synchronous.
Parameters (inout):	lumprdata0BBuffer Size	The maximum number of data bytes that can be written to the lumprdata0B Buffer.
Parameters (out):	lumprdata0B	Buffer containing the number of data elements (as defined in ISO-15031-5) and 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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.72: 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 (inout):	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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.73: 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 (inout):	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 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 $1C computed by the Dem. API is needed in OBD-relevant ECUs only. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.74: Dem_DcmReadDataOfPID1C

⌋(*SRS_Diag_04082*)

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 (inout):	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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.75: 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 (inout):	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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.76: 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 (inout):	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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.77: Dem_DcmReadDataOfPID31

⌋(*SRS_Diag_04082*)

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 (inout):	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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.78: 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 (inout):	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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.79: Dem_DcmReadDataOfPID4D

⌋(SRS_Diag_04082)

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 (inout):	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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.80: Dem_DcmReadDataOfPID4E

⌋()

8.3.5.11 Dem_DcmReadDataOfPID91

[SWS_Dem_01187] ⌈

Service name:	Dem_DcmReadDataOfPID91
Syntax:	`Std_ReturnType Dem_DcmReadDataOfPID91( uint8* PID91value )`
Service ID[hex]:	0x6a
Sync/Async:	Synchronous
Reentrancy:	Non Reentrant
Parameters (in):	None
Parameters (inout):	None

Parameters (out):	PID91value	Buffer containing the contents of PID $91 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 $91 computed by the Dem. API is needed in OBD-relevant ECUs only. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.81: Dem_DcmReadDataOfPID91

⌋(*SRS_Diag_04082*)

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.
	DataElementIndexOfPID	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 (inout):	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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.82: Dem_DcmReadDataOfOBDFreezeFrame

⌋()

8.3.5.13 Dem_DcmGetDTCOfOBDFreezeFrame

[SWS_Dem_00624] ⌈

Service name:	Dem_DcmGetDTCOfOBDFreezeFrame	
Syntax:	`Std_ReturnType Dem_DcmGetDTCOfOBDFreezeFrame( uint8 FrameNumber, uint32* DTC, Dem_DTCFormatType DTCFormat )`	
Service ID[hex]:	0x53	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	FrameNumber DTCFormat	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. Output format of the DTC value.
Parameters (inout):	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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.83: Dem_DcmGetDTCOfOBDFreezeFrame

⌋(*SRS_Diag_04010*)

8.3.5.14 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 (inout):	None	
Parameters (out):	Obdmidvalue	Bit coded information on the support of OBDMIDs.
Return value:	Std_ReturnType	E_OK: Report of DTR result successful
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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.84: Dem_DcmGetAvailableOBDMIDs

⌋*(SRS_Diag_04082)*

8.3.5.15 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 (inout):	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 number of TIDs successful E_NOT_OK: get number of TIDs 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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)

Table 8.85: Dem_DcmGetNumTIDsOfOBDMID

⌋()

8.3.5.16 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 TIDindex	Identification of a DTR element by assigned DTRId. Index of the TID within the DEM. Runs from 0 to "numberOfTIDs" obtained in the call to Dem_DcmGetNumTIDsOfOBDMID()
Parameters (inout):	None	
Parameters (out):	TIDvalue UaSID Testvalue Lowlimvalue Upplimvalue	TID to be put on the tester reponse 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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.86: 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:	`Std_ReturnType Dem_J1939DcmSetDTCFilter( Dem_J1939DcmDTCStatusFilterType DTCStatusFilter, Dem_DTCKindType DTCKind, Dem_DTCOriginType DTCOrigin, uint8 ClientId, Dem_J1939DcmLampStatusType* LampStatus )`	
Service ID[hex]:	0x90	
Sync/Async:	Synchronous	
Reentrancy:	Re-entrant for different ClientIDs, Non re-entrant for same ClientId.	
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	Defines the functional group of DTCs to be reported (e.g. all DTC, OBD-relevant DTC)
	DTCOrigin	This parameter is used to select the source memory the DTCs shall be read/cleared from.
	ClientId	ClientId to address the J1939 event memory
Parameters (inout):	None	
Parameters (out):	LampStatus	E_OK: Operation successful DEM_NO_SUCH_ELEMENT: The requested element is not available DEM_PENDING: Operation successful and result pending. DEM_BUFFER_TOO_SMALL: The provided buffer is too small
Return value:	Std_ReturnType	E_OK: Operation successful E_NOT_OK: Filter could not be set
Description:	The function sets the DTC filter for a specific node and returns the composite lamp status of the filtered DTCs.	

Table 8.87: Dem_J1939DcmSetDTCFilter

⌋(*SRS_Diag_04112*)

8.3.6.1.2 Dem_J1939DcmGetNumberOfFilteredDTC

[SWS_Dem_00972] ⌈

Service name:	Dem_J1939DcmGetNumberOfFilteredDTC	
Syntax:	`Std_ReturnType Dem_J1939DcmGetNumberOfFilteredDTC( uint16* NumberOfFilteredDTC, uint8 ClientId )`	
Service ID[hex]:	0x91	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	ClientId	ClientId to address the J1939 event memory
Parameters (inout):	None	
Parameters (out):	NumberOfFilteredDTC	The number of DTCs matching the defined status mask.
Return value:	Std_ReturnType	E_OK: Operation successful DEM_NO_SUCH_ELEMENT: The requested element is not available DEM_PENDING: Operation successful and result pending. DEM_BUFFER_TOO_SMALL: The provided buffer is too small
Description:	Gets the number of currently filtered DTCs set by the function Dem_J1939DcmSetDTCFilter.	

Table 8.88: Dem_J1939DcmGetNumberOfFilteredDTC

⌋()

8.3.6.1.3 Dem_J1939DcmGetNextFilteredDTC

[SWS_Dem_00973] ⌈

Service name:	Dem_J1939DcmGetNextFilteredDTC	
Syntax:	`Std_ReturnType Dem_J1939DcmGetNextFilteredDTC( uint32* J1939DTC, uint8* OccurenceCounter, uint8 ClientId )`	
Service ID[hex]:	0x92	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	ClientId	ClientId to address the J1939 event memory
Parameters (inout):	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.

	OccurenceCounter	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:	Std_ReturnType	E_OK: Operation successful DEM_NO_SUCH_ELEMENT: The requested element is not available DEM_PENDING: Operation successful and result pending. DEM_BUFFER_TOO_SMALL: The provided buffer is too small
Description:	Gets the next filtered J1939 DTC.	

Table 8.89: Dem_J1939DcmGetNextFilteredDTC

⌋()

8.3.6.1.4 Dem_J1939DcmFirstDTCwithLampStatus

[SWS_Dem_00974] ⌈

Service name:	Dem_J1939DcmFirstDTCwithLampStatus	
Syntax:	`void Dem_J1939DcmFirstDTCwithLampStatus( uint8 ClientId )`	
Service ID[hex]:	0x93	
Sync/Async:	Synchronous	
Reentrancy:	Re-entrant for different ClientIDs, Non re-entrant for same ClientId.	
Parameters (in):	ClientId	ClientId to address the J1939 event memory
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	The function sets the filter to the first applicable DTC for the DM31 response for a specific node.	

Table 8.90: Dem_J1939DcmFirstDTCwithLampStatus

⌋(*SRS_Diag_04110*)

8.3.6.1.5 Dem_J1939DcmGetNextDTCwithLampStatus

[SWS_Dem_00975] ⌈

Service name:	Dem_J1939DcmGetNextDTCwithLampStatus

Syntax:	`Std_ReturnType Dem_J1939DcmGetNextDTCwithLampStatus( Dem_J1939DcmLampStatusType* LampStatus, uint32* J1939DTC, uint8* OccurenceCounter, uint8 ClientId )`	
Service ID[hex]:	0x94	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	ClientId	ClientId to address the J1939 event memory
Parameters (inout):	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	Receives the J1939DTC value. If the return value of the function is other than DEM_FILTERED_OK this parameter does not contain valid data.
	OccurenceCounter	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:	Std_ReturnType	E_OK: Operation successful DEM_NO_SUCH_ELEMENT: The requested element is not available DEM_PENDING: Operation successful and result pending. DEM_BUFFER_TOO_SMALL: The provided buffer is too small
Description:	Gets the next filtered J1939 DTC for DM31 including current LampStatus.	

Table 8.91: Dem_J1939DcmGetNextDTCwithLampStatus

⌋()

8.3.6.2 DTC storage

8.3.6.2.1 Dem_J1939DcmClearDTC

[SWS_Dem_00976] ⌈

Service name:	Dem_J1939DcmClearDTC
Syntax:	`Std_ReturnType Dem_J1939DcmClearDTC( Dem_J1939DcmSetClearFilterType DTCTypeFilter, Dem_DTCOriginType DTCOrigin, uint8 ClientId )`
Service ID[hex]:	0x95
Sync/Async:	Asynchronous
Reentrancy:	Re-entrant for different ClientIDs, Non re-entrant for same ClientId.

Parameters (in):	DTCTypeFilter DTCOrigin ClientId	Defines the type of DTCs to be cleared. This parameter is used to select the source memory the DTCs shall be read/cleared from. ClientId to address the J1939 event memory
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: DTC successfully cleared DEM_WRONG_DTC: Selected DTC value in selected format does not exist or clearing is restricted by configuration to group of all DTCs only. DEM_WRONG_DTCORIGIN: Selected DTCOrigin does not exist DEM_CLEAR_FAILED: DTC clearing failed DEM_CLEAR_BUSY: Another client is currently clearing DTCs. The requested operation will not be started and the caller shall try again at a later moment. DEM_CLEAR_MEMORY_ERROR: An error occurred during erasing a memory location (e.g. if DemClearDTCBehavior is set to DEM_CLRRESP_NON-VOLATILE_FINISH and erasing of non-volatile-block failed). DEM_PENDING: Clearing the DTCs is currently in progress. The caller shall call this function again at a later moment.
Description:	Clears the status of all event(s) related to the specified DTC(s), as well as all associated event memory entries for these event(s).	

Table 8.92: Dem_J1939DcmClearDTC

⌋(*SRS_Diag_04112*)

8.3.6.2.2 Dem_J1939DcmSetFreezeFrameFilter

[SWS_Dem_00977] ⌈

Service name:	Dem_J1939DcmSetFreezeFrameFilter	
Syntax:	`Std_ReturnType Dem_J1939DcmSetFreezeFrameFilter( Dem_J1939DcmSetFreezeFrameFilterType FreezeFrameKind, uint8 ClientId )`	
Service ID[hex]:	0x96	
Sync/Async:	Synchronous	
Reentrancy:	Re-entrant for different ClientIDs, Non re-entrant for same ClientId.	
Parameters (in):	FreezeFrameKind ClientId	The following types are available: DEM_J1939DCM_FREEZEFRAME DEM_J1939DCM_EXPANDED_FREEZEFRAME DEM_J1939DCM_SPNS_IN_EXPANDED_FREEZEFRAME ClientId to address the J1939 event memory
Parameters (inout):	None	

Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: Operation successful E_NOT_OK: Filter could not be set
Description:	The function sets the FreezeFrame filter for a specific node.	

Table 8.93: Dem_J1939DcmSetFreezeFrameFilter

⌋(*SRS_Diag_04112*)

8.3.6.2.3 Dem_J1939DcmGetNextFreezeFrame

[SWS_Dem_00978] ⌈

Service name:	Dem_J1939DcmGetNextFreezeFrame	
Syntax:	`Std_ReturnType Dem_J1939DcmGetNextFreezeFrame( uint32* J1939DTC, uint8* OccurenceCounter, uint8* DestBuffer, uint16* BufSize, uint8 ClientId )`	
Service ID[hex]:	0x97	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	ClientId	ClientId to address the J1939 event memory
Parameters (inout):	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.
	OccurenceCounter	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:	Std_ReturnType	E_OK: Operation successful DEM_NO_SUCH_ELEMENT: The requested element is not available DEM_PENDING: Operation successful and result pending. DEM_BUFFER_TOO_SMALL: The provided buffer is too small
Description:	Gets next freeze frame data. The function stores the data in the provided DestBuffer.	

Table 8.94: Dem_J1939DcmGetNextFreezeFrame

⌋(*SRS_Diag_04112*)

8.3.6.2.4 Dem_J1939DcmGetNextSPNInFreezeFrame

[SWS_Dem_00979] ⌈

Service name:	Dem_J1939DcmGetNextSPNInFreezeFrame	
Syntax:	`Std_ReturnType Dem_J1939DcmGetNextSPNInFreezeFrame( uint32* SPNSupported, uint8* SPNDataLength, uint8 ClientId )`	
Service ID[hex]:	0x98	
Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	ClientId	ClientId to address the J1939 event memory
Parameters (inout):	None	
Parameters (out):	SPNSupported SPNDataLength	This parameter contains the next SPN in the ExpandedFreezeFrame This parameter contains the corresponding dataLength of the SPN
Return value:	Std_ReturnType	E_OK: Operation successful DEM_NO_SUCH_ELEMENT: The requested element is not available DEM_PENDING: Operation successful and result pending. DEM_BUFFER_TOO_SMALL: The provided buffer is too small
Description:	Gets next SPN.	

Table 8.95: Dem_J1939DcmGetNextSPNInFreezeFrame

⌋(*SRS_Diag_04112*)

8.3.6.3 Reporting

8.3.6.3.1 Dem_J1939DcmSetRatioFilter

[SWS_Dem_00980] ⌈

Service name:	Dem_J1939DcmSetRatioFilter
Syntax:	`Std_ReturnType Dem_J1939DcmSetRatioFilter( uint16* IgnitionCycleCounter, uint16* OBDMonitoringConditionsEncountered, uint8 ClientId )`
Service ID[hex]:	0x99
Sync/Async:	Synchronous

Reentrancy:	Re-entrant for different ClientIDs, Non re-entrant for same ClientId.	
Parameters (in):	ClientId	ClientId to address the J1939 event memory
Parameters (inout):	None	
Parameters (out):	IgnitionCycle Counter OBDMonitoringConditionsEncountered	Ignition Cycle Counter OBD Monitoring Conditions Encountered
Return value:	Std_ReturnType	E_OK: Operation successful E_NOT_OK: Filter could not be set
Description:	The function sets the Ratio filter for a specific node and returns the corresponding Ignition Cycle Counter and General Denominator.	

Table 8.96: Dem_J1939DcmSetRatioFilter

⌋()

8.3.6.3.2 Dem_J1939DcmGetNextFilteredRatio

[SWS_Dem_00981] ⌈

Service name:	Dem_J1939DcmGetNextFilteredRatio	
Syntax:	`Std_ReturnType Dem_J1939DcmGetNextFilteredRatio( uint16* SPN, uint16* Numerator, uint16* Denominator, uint8 ClientId )`	
Service ID[hex]:	0x9a	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	ClientId	ClientId to address the J1939 event memory
Parameters (inout):	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:	Std_ReturnType	E_OK: Operation successful DEM_NO_SUCH_ELEMENT: The requested element is not available DEM_PENDING: Operation successful and result pending. DEM_BUFFER_TOO_SMALL: The provided buffer is too small
Description:	Gets the next filtered Ratio.	

Table 8.97: Dem_J1939DcmGetNextFilteredRatio

⌋(*SRS_Diag_04112*)

8.3.6.3.3 Dem_J1939DcmReadDiagnosticReadiness1

[SWS_Dem_00982] ⌈

Service name:	Dem_J1939DcmReadDiagnosticReadiness1	
Syntax:	`Std_ReturnType Dem_J1939DcmReadDiagnosticReadiness1( Dem_J1939DcmDiagnosticReadiness1Type* DataValue, uint8 ClientId )`	
Service ID[hex]:	0x9b	
Sync/Async:	Synchronous	
Reentrancy:	Re-entrant for different ClientIDs, Non re-entrant for same ClientId.	
Parameters (in):	ClientId	ClientId to address the J1939 event memory
Parameters (inout):	None	
Parameters (out):	DataValue	Buffer of 8 bytes containing the contents of Diagnostic Readiness 1 (DM05) computed by the Dem.
Return value:	Std_ReturnType	E_OK: Operation was successful E_NOT_OK: Operation failed
Description:	Service to report the value of Diagnostic Readiness 1 (DM05) computed by the Dem.	

Table 8.98: Dem_J1939DcmReadDiagnosticReadiness1

⌋(*SRS_Diag_04113*)

8.3.6.3.4 Dem_J1939DcmReadDiagnosticReadiness2

[SWS_Dem_00983] ⌈

Service name:	Dem_J1939DcmReadDiagnosticReadiness2
Syntax:	`Std_ReturnType Dem_J1939DcmReadDiagnosticReadiness2( Dem_J1939DcmDiagnosticReadiness2Type* DataValue, uint8 ClientId )`

Service ID[hex]:	0x9c	
Sync/Async:	Synchronous	
Reentrancy:	Re-entrant for different ClientIDs, Non re-entrant for same ClientId.	
Parameters (in):	ClientId	ClientId to address the J1939 event memory
Parameters (inout):	None	
Parameters (out):	DataValue	Buffer of 8 bytes containing the contents of Diagnostic Readiness 2 (DM21) computed by the Dem.
Return value:	Std_ReturnType	E_OK: Operation was successful E_NOT_OK: Operation failed
Description:	Service to report the value of Diagnostic Readiness 2 (DM21) computed by the Dem.	

Table 8.99: Dem_J1939DcmReadDiagnosticReadiness2

⌋(*SRS_Diag_04113*)

8.3.6.3.5 Dem_J1939DcmReadDiagnosticReadiness3

[SWS_Dem_00770] ⌈

Service name:	Dem_J1939DcmReadDiagnosticReadiness3	
Syntax:	`Std_ReturnType Dem_J1939DcmReadDiagnosticReadiness3( Dem_J1939DcmDiagnosticReadiness3Type* DataValue, uint8 ClientId )`	
Service ID[hex]:	0x9d	
Sync/Async:	Synchronous	
Reentrancy:	Re-entrant for different ClientIDs, Non re-entrant for same ClientId.	
Parameters (in):	ClientId	ClientId to address the J1939 event memory
Parameters (inout):	None	
Parameters (out):	DataValue	Buffer of 8 bytes containing the contents of Diagnostic Readiness 3 (DM26) computed by the Dem.
Return value:	Std_ReturnType	E_OK: Operation was successful E_NOT_OK: Operation failed
Description:	Service to report the value of Diagnostic Readiness 3 (DM26) computed by the Dem.	

Table 8.100: Dem_J1939DcmReadDiagnosticReadiness3

⌋(*SRS_Diag_04113*)

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 (inout):	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_NO_SUCH_ELEMENT: The requested event data is not currently stored (but the request was valid). DEM_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. The OBD-II freeze frame is not returned by this function.	

Table 8.101: Dem_DltGetMostRecentFreezeFrameRecordData

⌋(*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 (inout):	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_NO_SUCH_ELEMENT: The requested event data is not currently stored (but the request was valid). DEM_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 8.102: 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 (inout):	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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.103: Dem_SetEventDisabled

⌋()

8.3.8.2 Dem_ReplUMPRFaultDetect

[SWS_Dem_00313] ⌈

Service name:	Dem_ReplUMPRFaultDetect	
Syntax:	`Std_ReturnType Dem_RepIUMPRFaultDetect( Dem_RatioIdType RatioID )`	
Service ID[hex]:	0x73	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant for different RatioIDs. Non reentrant for the same RatioID.	
Parameters (in):	RatioID	Ratio Identifier reporting that a respective monitor could have found a fault - only used when interface option "API" is selected
Parameters (inout):	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 API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.104: 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 ConditionStatus	Identification of a IUMPR denominator condition ID (General Denominator, Cold start, EVAP, 500mi). Status of the IUMPR denominator condition (Not-reached, reached, not reachable / inhibited)
Parameters (inout):	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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.105: Dem_SetIUMPRDenCondition

⌋(*SRS_Diag_04095*)

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 (inout):	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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.106: Dem_GetIUMPRDenCondition

⌋()

8.3.8.5 Dem_ReplUMPRDenLock

[SWS_Dem_00314] ⌈

Service name:	Dem_ReplUMPRDenLock	
Syntax:	`Std_ReturnType Dem_RepIUMPRDenLock( 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 (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK: report of IUMPR denominator status was successfully reported E_NOT_OK: 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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.107: Dem_ReplUMPRDenLock

⌋(*SRS_Diag_04141*)

8.3.8.6 Dem_ReplUMPRDenRelease

[SWS_Dem_00315] ⌈

Service name:	Dem_ReplUMPRDenRelease	
Syntax:	`Std_ReturnType Dem_RepIUMPRDenRelease( 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 (inout):	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 API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.108: Dem_ReplUMPRDenRelease

⌋(*SRS_Diag_04082*)

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 (inout):	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 API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.109: Dem_SetPtoStatus

⌋(*SRS_Diag_04082*)

8.3.8.8 Dem_ReadDataOfPID01

[SWS_Dem_01167] ⌈

Service name:	Dem_ReadDataOfPID01

Syntax:	`Std_ReturnType Dem_ReadDataOfPID01(`
	`uint8* PID01value`
	`)`	

Service ID[hex]:	0xb3	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	PID01value	Buffer containing the contents of PID $01 computed by the Dem. The buffer is provided by the application with the size of 4 bytes.
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 8.110: Dem_ReadDataOfPID01

⌋()

8.3.8.9 Dem_GetDataOfPID21

[SWS_Dem_01093] ⌈

Service name:	Dem_GetDataOfPID21	

Syntax:	`Std_ReturnType Dem_GetDataOfPID21(`
	`uint8* PID21value`
	`)`	

Service ID[hex]:	0xb1	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	PID21value	Content of PID $21 as raw hex value.
Return value:	Std_ReturnType	Always E_OK is returned, as E_NOT_OK will never appear.
Description:	Service to get the value of PID $21 from the Dem by a software component. API is needed in OBD-relevant ECUs only. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral/DemGeneralOBD.DemOBDCentralizedPID21Handling)} == true) && ({ecuc(Dem/DemGeneral.DemOBDSupport)} == DEM_OBD_MASTER_ECU)	

Table 8.111: Dem_GetDataOfPID21

⌋()

8.3.8.10 Dem_SetDataOfPID21

[SWS_Dem_00735] ⌈

Service name:	Dem_SetDataOfPID21	
Syntax:	`Std_ReturnType Dem_SetDataOfPID21( const 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 (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 $21 in the Dem by a software component. API is needed in OBD-relevant ECUs only. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.112: Dem_SetDataOfPID21

⌋(*SRS_Diag_04082*)

8.3.8.11 Dem_SetDataOfPID31

[SWS_Dem_00736] ⌈

Service name:	Dem_SetDataOfPID31	
Syntax:	`Std_ReturnType Dem_SetDataOfPID31( const 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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.113: Dem_SetDataOfPID31

⌋()

8.3.8.12 Dem_SetDataOfPID4D

[SWS_Dem_00737] ⌈

Service name:	Dem_SetDataOfPID4D	
Syntax:	`Std_ReturnType Dem_SetDataOfPID4D( const 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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.114: Dem_SetDataOfPID4D

⌋()

8.3.8.13 Dem_SetDataOfPID4E

[SWS_Dem_00738] ⌈

Service name:	Dem_SetDataOfPID4E	
Syntax:	`Std_ReturnType Dem_SetDataOfPID4E( const 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 (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 $4E in the Dem by a software component. API is needed in OBD-relevant ECUs only. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.115: Dem_SetDataOfPID4E

⌋(*SRS_Diag_04082*)

8.3.8.14 Dem_GetCycleQualified

[SWS_Dem_00740] ⌈

Service name:	Dem_GetCycleQualified	
Syntax:	`Std_ReturnType Dem_GetCycleQualified( uint8 OperationCycleId, boolean* isQualified )`	
Service ID[hex]:	0xab	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	OperationCycleId	Identification of a configured DemOperationCycle.
Parameters (inout):	None	
Parameters (out):	isQualified	TRUE: The dependent operation cylcle is qualified. FALSE: The qualification conditions of the dependent operation cylcle have not been met.
Return value:	Std_ReturnType	Always E_OK is returned, as E_NOT_OK will never appear.

Description:	Returns the qualification state of the dependent operation cycle. API Availability: This API will be available only if any of the ({ecuc(Dem/DemGeneral/DemOperationCycle.DemLeadingCycleRef)} != NULL)

Table 8.116: Dem_GetCycleQualified

⌋()

8.3.8.15 Dem_SetCycleQualified

[SWS_Dem_91001] ⌈

Service name:	Dem_SetCycleQualified	
Syntax:	`Std_ReturnType Dem_SetCycleQualified( uint8 OperationCycleId )`	
Service ID[hex]:	0x56	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	OperationCycleId	Identification of a configured DemOperationCycle
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	Always E_OK is returned.
Description:	Sets a dependent operation cycle as qualified, so it may be processed along with its leading cycle.	

Table 8.117: Dem_SetCycleQualified

⌋()

8.3.8.16 Dem_GetDTCSeverityAvailabilityMask

[SWS_Dem_01168] ⌈

Service name:	Dem_GetDTCSeverityAvailabilityMask	
Syntax:	`Std_ReturnType Dem_GetDTCSeverityAvailabilityMask( uint8 ClientId, Dem_DTCSeverityType* DTCSeverityMask )`	
Service ID[hex]:	0xb2	
Sync/Async:	Synchronous	
Reentrancy:	Re-entrant for different ClientIDs, Non re-entrant for same ClientId.	
Parameters (in):	ClientId	Unique client id, assigned to the instance of the calling module.
Parameters (inout):	None	

Parameters (out):	DTCSeverityMask	DTCSeverityMask The value DTCSeverityMask indicates the supported DTC severity 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 severity mask was successful E_NOT_OK: get of DTC severity mask failed
Description:	Gets the DTC Severity availability mask.	

Table 8.118: Dem_GetDTCSeverityAvailabilityMask

⌋(*SRS_Diag_04141*)

8.3.8.17 Dem_GetB1Counter

[SWS_Dem_01169] ⌈

Service name:	Dem_GetB1Counter	

Syntax:	`Std_ReturnType Dem_GetB1Counter(`
	`uint16* B1Counter`
	`)`	

Service ID[hex]:	0xb4	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	B1Counter	Buffer containing the B1 counter. The buffer is provided by the application with the size of 2 bytes.
Return value:	Std_ReturnType	Always E_OK is returned, as E_NOT_OK will never appear.
Description:	Service to report the value of the B1 counter computed by the Dem. API is needed in WWH-OBD relevant ECUs only	

Table 8.119: Dem_GetB1Counter

⌋(*SRS_Diag_04141*)

8.3.8.18 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 for different DTRIds. Non reentrant for the same DTRId.	
Parameters (in):	DTRId TestResult LowerLimit UpperLimit Ctrlval	Identification of a DTR element by assigned DTRId. Test result of DTR Lower limit of DTR Upper limit of DTR Control value of the DTR to support its interpretation Dem-internally.
Parameters (inout):	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. API Availability: This API will be available only if ({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT)	

Table 8.120: Dem_SetDTR

⌋(*SRS_Diag_04181*)

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***

Table 8.121: Dem Mandatory Interfaces

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 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 Dlt to get informed about Dem status changes.
FiM_DemInit	This service re-initializes the FIM.
FiM_DemTriggerOnComponent Status	Triggers on changes of the component failed status.
FiM_DemTriggerOnMonitorStatus	This service is provided to be called by the Dem in order to inform the Fim about monitor status changes.
J1939Dcm_DemTriggerOnDTC Status	Trigger for DM01 message that a UDS 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 a permanent RAM block or the status of the explicit synchronization of a NVRAM block.
NvM_WriteBlock	Service to copy the data of the RAM block to its corresponding NV block.

Table 8.122: Dem Optional Interfaces

⌋(*SRS_BSW_00171*)

Note: Based on implementation strategy FiM_DemInit can be used (refer also to chapter chapter 7.11.3).

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 7.11.5).

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.

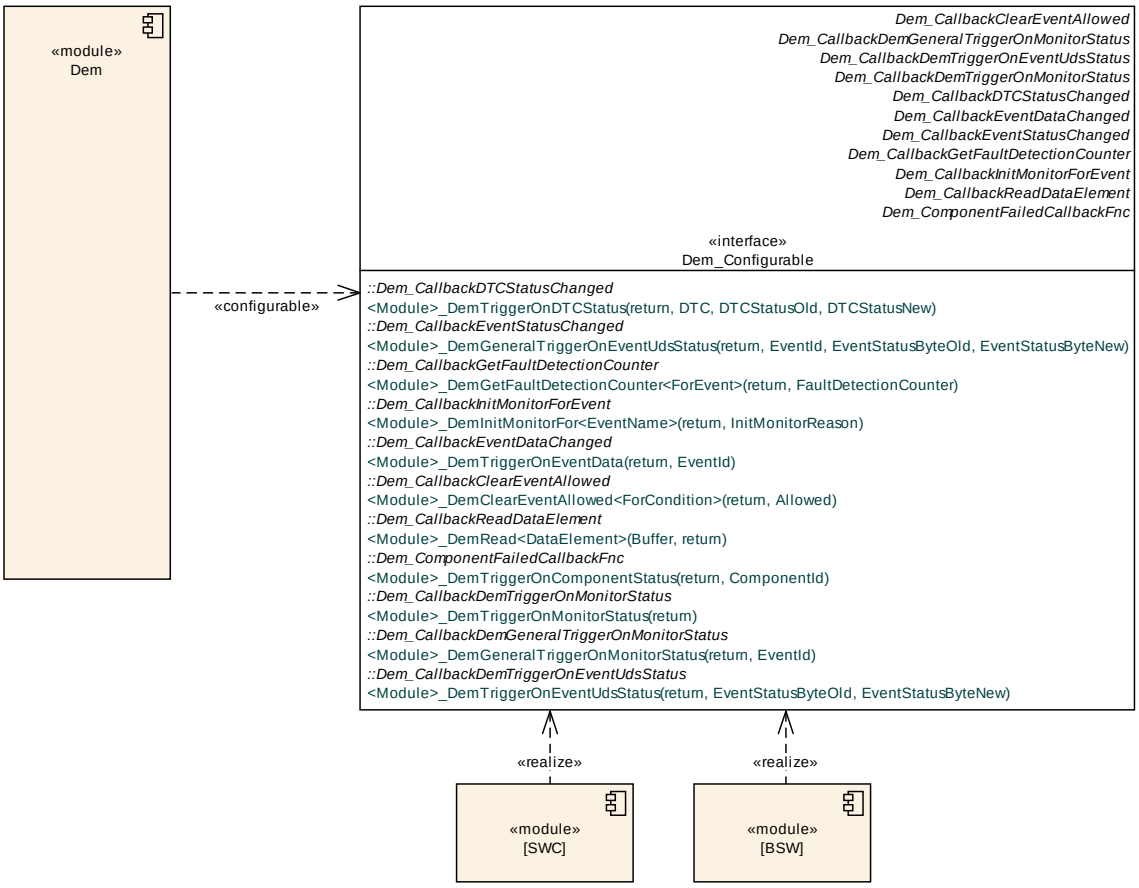


Figure 8.4: Configuration interfaces of the Dem module

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 )`	
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 8.123: InitMonitorForEvent

⌋(*SRS_BSW_00310*, *SRS_BSW_00101*)

8.4.3.1.2 DemTriggerOnComponentStatus

[SWS_Dem_01116] ⌈

Service name:	<Module>_DemTriggerOnComponentStatus	
Syntax:	`Std_ReturnType <Module>_DemTriggerOnComponentStatus( Dem_ComponentIdType ComponentId )`	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	ComponentId	Identification of a DemComponent.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	Return value unused - only for compatibility with according RTE operation.
Description:	Triggers on changes of the DemComponent failed status.	

Table 8.124: DemTriggerOnComponentStatus

⌋(*SRS_Diag_04142*)

8.4.3.2 ClearDtcNotification

[SWS_Dem_91002] ⌈

Service name:	<Module>_ClearDtcNotification

Syntax:	`Std_ReturnType <Module>_ClearDtcNotification(`
	`uint32 DTC,`
	`Dem_DTCFormatType DTCFormat,`
	`Dem_DTCOriginType DTCOrigin`
	`)`	

Sync/Async:	Asynchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	DTC DTCFormat DTCOrigin	DTC or group of DTC that is cleared. Format of the DTC value. Event memory which is selected by the current clear operation.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	E_OK is always returned
Description:	Called by the Dem when performing a clear DTC operation.	

Table 8.125: <Module>_ClearDtcNotification

⌋()

8.4.3.3 DemGeneralTriggerOnMonitorStatus

[SWS_Dem_91009] ⌈

Service name:	<Module>_DemGeneralTriggerOnMonitorStatus	

Syntax:	`Std_ReturnType <Module>_DemGeneralTriggerOnMonitor`
	`Status(`
	`Dem_EventIdType EventId`
	`)`	

Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	EventId	Identification of an event by assigned EventId.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	Return value unused - only for compatibility with according RTE operation.
Description:	Triggers on changes of the monitor status. Called synchronously in context of event status reporting.	

Table 8.126: <Module>_DemGeneralTriggerOnMonitorStatus

⌋()

8.4.3.4 DemGeneralTriggerOnEventUdsStatus

[SWS_Dem_00259] ⌈

Service name:	<Module>_DemGeneralTriggerOnEventUdsStatus	
Syntax:	`Std_ReturnType <Module>_DemGeneralTriggerOnEventUds Status( Dem_EventIdType EventId, Dem_UdsStatusByteType EventStatusByteOld, Dem_UdsStatusByteType EventStatusByteNew )`	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	EventId	Identification of an event by assigned EventId.
	EventStatusByteOld	UDS DTC status byte of event before change (refer to chapter "Status bit support").
	EventStatusByte New	UDS DTC status byte of event after change (refer to chapter "Status bit support").
Parameters (inout):	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 8.127: DemTriggerOnEventStatus

⌋(*SRS_Diag_04148*)

8.4.3.5 DemTriggerOnEventUdsStatus

[SWS_Dem_91006] ⌈

Service name:	<Module>_DemTriggerOnEventUdsStatus	
Syntax:	`Std_ReturnType <Module>_DemTriggerOnEventUdsStatus( Dem_UdsStatusByteType EventStatusByteOld, Dem_UdsStatusByteType EventStatusByteNew )`	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	EventStatusByteOld	UDS DTC status byte of event before change (refer to chapter "Status bit support").
	EventStatusByte New	UDS DTC status byte of event after change (refer to chapter "Status bit support").
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	ReturnType Return value unused - only for compatibility with according RTE operation.
Description:	Triggers on changes of the UDS DTC status byte.	

Table 8.128: <Module>_DemTriggerOnEventUdsStatus

⌋()

8.4.3.6 DemTriggerOnDTCStatus

[SWS_Dem_00260] ⌈

Service name:	<Module>_DemTriggerOnDTCStatus	
Syntax:	`Std_ReturnType <Module>_DemTriggerOnDTCStatus( uint32 DTC, Dem_UdsStatusByteType DTCStatusOld, Dem_UdsStatusByteType DTCStatusNew )`	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	DTC DTCStatusOld DTCStatusNew	Diagnostic Trouble Code in UDS format. UDS status before change UDS status after change
Parameters (inout):	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 status byte.	

Table 8.129: DemTriggerOnDTCStatus

⌋()

8.4.3.7 DemTriggerOnMonitorStatus

[SWS_Dem_91010] ⌈

Service name:	<Module>_DemTriggerOnMonitorStatus	
Syntax:	`Std_ReturnType <Module>_DemTriggerOnMonitorStatus( void )`	
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:	Triggers on changes of the monitor status. Called synchronously in context of event status reporting.	

Table 8.130: <Module>_DemTriggerOnMonitorStatus

⌋()

8.4.3.8 EventDataChanged

[SWS_Dem_00562] ⌈

Service name:	<Module>_DemTriggerOnEventData	
Syntax:	`Std_ReturnType <Module>_DemTriggerOnEventData( Dem_EventIdType EventId )`	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	EventId	Identification of an event by assigned EventId.
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Std_ReturnType	Return value unused - only for compatibility with according RTE operation.
Description:	Triggers on changes of the event related data in the event memory.	

Table 8.131: DemTriggerOnEventData

⌋*(SRS_Diag_04160)*

8.4.3.9 ClearEventAllowed

[SWS_Dem_00563] ⌈

Service name:	<Module>_DemClearEventAllowed<ForCondition>	
Syntax:	`Std_ReturnType <Module>_DemClearEventAllowed<ForCondition>( boolean* Allowed )`	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	None	
Parameters (inout):	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. Parameter "Allowed" will be unchanged in case E_NOT_OK is returned.	

Table 8.132: DemClearEventAllowed

⌋*()*

8.4.3.10 ReadDataElement

[SWS_Dem_00564] ⌈

Service name:	<Module>_DemRead<DataElement>	
Syntax:	`Std_ReturnType <Module>_DemRead<DataElement>( uint8* Buffer )`	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	None	
Parameters (inout):	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 8.133: DemRead

⌋(*SRS_Diag_04074*)

8.4.3.11 GetFaultDetectionCounter

[SWS_Dem_00263] ⌈

Service name:	<Module>_DemGetFaultDetectionCounter<ForEvent>	
Syntax:	`Std_ReturnType <Module>_DemGetFaultDetectionCounter<ForEvent>( sint8* FaultDetectionCounter )`	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	None	
Parameters (inout):	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 8.134: 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.

Table 8.135: Dem_MainFunction

⌋()

[SWS_Dem_00125] ⌈The function Dem_MainFunction shall process all not event based Dem module internal functions. ⌋(*SRS_BSW_00373*)

For details refer to the chapter 8.5 “Scheduled Functions” in SWS_BSWGeneral.

8.5.2 Runnable Entity MainFunction

```
1 RunnableEntity MainFunction
2        symbol "Dem_MainFunction"
3        canbeInvokedConcurrently = FALSE
4        SSCP = port CBEventUdsStatusChanged_*,
             CallbackEventUdsStatusChanged
5        SSCP = port GeneralCBStatusEvt,
             GeneralCallbackEventUdsStatusChanged
6        SSCP = port CBStatusDTC_*, DTCStatusChanged
7        SSCP = port CBDataEvt_*, EventDataChanged
8        SSCP = port GeneralCBDataEvt, EventDataChanged
9        SSCP = port CBReadData_*, ReadData
```

8.6 Service Interfaces

8.6.1 Implementation Data Types

8.6.1.1 DataArrayType

[SWS_Dem_01192] ⌈

Name	DataArrayType_{Data}		
Kind	Array	***Element type***	uint8
Size	{ecuc(Dem/DemGeneral/DemDataElementClass/DemExternalCS DataElementClass/DemDataElementDataSize)} Elements		
Description	--		
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)}		

Table 8.136: Implementation Data Type DataArrayType_{Data}

⌋()

8.6.1.2 Dem_DTCOriginType

[SWS_Dem_00934] ⌈

Name	Dem_DTCOriginType		
Kind	Type		
Derived from	uint16		
Description	This enum is used to define the location of the events. The definition and use of the different memory types is OEM-specific.		
Range	DEM_DTC_ORIGIN_ PRIMARY_MEMORY	0x0001	Event information located in the primary memory
	DEM_DTC_ORIGIN_ MIRROR_MEMORY	0x0002	Event information located in the mirror memory
	DEM_DTC_ORIGIN_ PERMANENT_MEMORY	0x0003	The Event information is located in the permanent memory
	DEM_DTC_ORIGIN_OBD_ RELEVANT_MEMORY	0x0004	Selects all memories which are storing OBD events (specified by configuration)
	DEM_DTC_ORIGIN_ USERDEFINED_MEMORY_ <Name>	0x01XX	Event information located in the user defined memory, where XX is the configured DemUserDefinedMemoryIden-tifier in hexadecimal and <Name> is the Short-Name of the DemUserDefinedMemory.

Variation	--

Table 8.137: Implementation Data Type Dem_DTCOriginType

⌋(*SRS_Diag_04066*)

8.6.1.3 Dem_DebouncingStateType

[SWS_Dem_01000] ⌈

Name	Dem_DebouncingStateType			
Kind	Bitfield			
Derived from	uint8			
Elements	***Kind***	***Name***	***Mask***	***Description***
	bit	DEM_TEMPORARILY_ DEFECTIVE	0x01	Bit 0: Temporarily Defective (corresponds to 0 < FDC < 127)
	bit	DEM_FINALLY_ DEFECTIVE	0x02	Bit 1: finally Defective (corresponds to FDC = 127)
	bit	DEM_TEMPORARILY_ HEALED	0x04	Bit 2: temporarily healed (corresponds to -128 < FDC < 0)
	bit	DEM_TEST_COMPLETE	0x08	Bit 3: Test complete (corresponds to FDC = -128 or FDC = 127)
	bit	DEM_DTR_UPDATE	0x10	Bit 4: DTR Update (= Test complete && Debouncing complete && enable conditions / storage conditions fulfilled)
Description	--			

Table 8.138: Implementation Data Type Dem_DebouncingStateType

⌋(*SRS_Diag_04105*)

8.6.1.4 Dem_DebounceResetStatusType

[SWS_Dem_00927] ⌈

Name	Dem_DebounceResetStatusType		
Kind	Type		
Derived from	uint8		
Description	This type contains all definitions to control an internal debounce counter/timer via the function Dem_ResetEventDebounceStatus().		
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
Variation	--		

Table 8.139: Implementation Data Type Dem_DebounceResetStatusType

⌋()

8.6.1.5 Dem_DTRControlType

[SWS_Dem_00941] ⌈

Name	Dem_DTRControlType		
Kind	Type		
Derived from	uint8		
Description	Control parameter for the interpretation of the reported test results.		
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.
Variation	--		

Table 8.140: Implementation Data Type Dem_DTRControlType

⌋()

8.6.1.6 Dem_EventIdType

[SWS_Dem_00925] ⌈

Name	Dem_EventIdType
Kind	Type

Derived from	uint16		
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.		
Range	1..65535		Internal identifier of a diagnostic event Remark: 0 is not a valid value
Variation	--		

Table 8.141: Implementation Data Type Dem_EventIdType

⌋()

8.6.1.7 Dem_EventStatusType

[SWS_Dem_00926] ⌈

Name	Dem_EventStatusType		
Kind	Type		
Derived from	uint8		
Description	This type contains all monitor test result values, which can be reported via Dem_SetEventStatus().		
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-internally).
	DEM_EVENT_STATUS_PREFAILED	0x03	Monitor reports non-qualified test result pre-failed (debounced Dem-internally).
	DEM_EVENT_STATUS_FDC_THRESHOLD_REACHED	0x04	Monitor triggers the storage of ExtendedDataRecords and FreezeFrames ON_FDC_THRESHOLD.
		0x05 - 0xFF	reserved
Variation	--		

Table 8.142: Implementation Data Type Dem_EventStatusType

⌋()

8.6.1.8 Dem_DTCFormatType

[SWS_Dem_00933] ⌈

Name	Dem_DTCFormatType		
Kind	Type		
Derived from	uint8		
Description	This type is used to select the format of the DTC value.		
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)
Variation	--		

Table 8.143: Implementation Data Type Dem_DTCFormatType

⌋(*SRS_Diag_04082*, *SRS_Diag_04000*, *SRS_Diag_04112*)

8.6.1.9 Dem_DTCKindType

[SWS_Dem_00932] ⌈

Name	Dem_DTCKindType		
Kind	Type		
Derived from	uint8		
Description	This type is used to filter DTCs for their kind.		
Range	DEM_DTC_KIND_ALL_DTCS	0x01	Select all DTCs
	DEM_DTC_KIND_ EMISSION_REL_DTCS	0x02	Select OBD-relevant DTCs
Variation	--		

Table 8.144: Implementation Data Type Dem_DTCKindType

⌋(*SRS_Diag_04129*)

8.6.1.10 Dem_InitMonitorReasonType

[SWS_Dem_00942] ⌈

Name	Dem_InitMonitorReasonType
Kind	Type
Derived from	uint8

Description	(Re-)Initialization reason returned by the callback <Module>_DemInitMonitorFor<EventName>().		
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.
	DEM_INIT_MONITOR_ STORAGE_REENABLED	0x04	Storage condition reenabled.
Variation	--		

Table 8.145: Implementation Data Type Dem_InitMonitorReasonType

⌋(*SRS_Diag_04063*)

8.6.1.11 Dem_IumprDenomCondIdType

[SWS_Dem_00943] ⌈

Name	Dem_IumprDenomCondIdType		
Kind	Type		
Derived from	uint8		
Description	This type contains all possible additional IUMPR denominator conditions to be broadcasted among OBD-relevant ECUs.		
Range	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"
	DEM_IUMPR_GENERAL_ INDIVIDUAL_DENOMINATOR	0x05	Individual denominators to support different conditions than the general denominator. It acts on individual denominators and allows a different condition to be set than for the general denominator. If the standard individual denominator conditions differ from the general denominator conditions, they typically differ by a "fueled engine" criterion.
	DEM_IUMPR_GENERAL_ OBDCOND	0x06	IUMPR denominator condition "General Denominator" for output with Infotype $08/$0B
Variation	--		

Table 8.146: Implementation Data Type Dem_IumprDenomCondIdType

⌋()

8.6.1.12 Dem_IumprDenomCondStatusType

[SWS_Dem_00944] ⌈

Name	Dem_IumprDenomCondStatusType		
Kind	Type		
Derived from	uint8		
Description	This type contains all possible states of an additional IUMPR denominator condition to be broadcasted among OBD-relevant ECUs.		
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
Variation	--		

Table 8.147: Implementation Data Type Dem_IumprDenomCondStatusType

⌋()

8.6.1.13 Dem_MaxDataValueType

[SWS_Dem_01072] ⌈

Name	Dem_MaxDataValueType		
Kind	Array	***Element type***	uint8
Size	size of largest Extended data class / Freeze frame record Elements		
Description	--		
Variation	--		

Table 8.148: Implementation Data Type Dem_MaxDataValueType

⌋()

8.6.1.14 Dem_MonitorStatusType

[SWS_Dem_91005] ⌈

Name	Dem_MonitorStatusType			
Kind	Bitfield			
Derived from	uint8			
Elements	***Kind***	***Name***	***Mask***	***Description***
	bit	DEM_MONITOR_STATUS_TF	0x01	Bit0: TestFailed
	bit	DEM_MONITOR_STATUS_TNCTOC	0x02	Bit1: TestNotCompletedThisOperationCycle
Description	This type contains possible monitor status values.			

Table 8.149: Implementation Data Type Dem_MonitorStatusType

⌋*()*

8.6.1.15 Dem_OperationCycleStateType

[SWS_Dem_00929] ⌈

Name	Dem_OperationCycleStateType		
Kind	Type		
Derived from	uint8		
Description	This type contains operation cycle state values, which can be reported via Dem_SetOperationCycleState()/ Dem_GetOperationCycleState().		
Range	DEM_CYCLE_STATE_START	0x00	Start/restart the operation cycle
	DEM_CYCLE_STATE_END	0x01	End the operation cycle
Variation	--		

Table 8.150: Implementation Data Type Dem_OperationCycleStateType

⌋*(SRS_Diag_04076)*

8.6.1.16 Dem_IndicatorStatusType

[SWS_Dem_00930] ⌈

Name	Dem_IndicatorStatusType		
Kind	Type		
Derived from	uint8		
Description	Indicator mode used by Dem_GetIndicatorStatus().		
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
	DEM_INDICATOR_ON_DEMAND	0x06	Indicator on-demand mode
	DEM_INDICATOR_SHORT	0x07	Indicator short mode
Variation	--		

Table 8.151: Implementation Data Type Dem_IndicatorStatusType

⌋()

8.6.1.17 Dem_PID21valueType

[SWS_Dem_01073] ⌈

Name	Dem_PID21valueType		
Kind	Array	***Element type***	uint8
Size	2 Elements		
Description	--		
Variation	--		

Table 8.152: Implementation Data Type Dem_PID21valueType

⌋(*SRS_Diag_04082*)

8.6.1.18 Dem_PID31valueType

[SWS_Dem_01074] ⌈

Name	Dem_PID31valueType		
Kind	Array	***Element type***	uint8
Size	2 Elements		
Description	--		
Variation	--		

Table 8.153: Implementation Data Type Dem_PID31valueType

⌋()

8.6.1.19 Dem_RatioIdType

[SWS_Dem_00940] ⌈

Name	Dem_RatioIdType		
Kind	Type		
Derived from	*Base Type*	*Variation*	
	uint16	Configurable, size depends on system complexity (refer to range of configuration parameter DemRatioId)	
	uint8	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.		
Range	0..255, 0..65535		Configurable, size depends on system complexity (refer to range of configuration parameter DemRatioId)
Variation	--		

Table 8.154: Implementation Data Type Dem_RatioIdType

⌋()

8.6.1.20 Dem_UdsStatusByteType

[SWS_Dem_00928] ⌈

Name	Dem_UdsStatusByteType			
Kind	Bitfield			
Derived from	uint8			
Lower limit	0x00 Elements			
Upper limit	0xFF Elements			
Elements	***Kind***	***Name***	***Mask***	***Description***
	bit	DEM_UDS_STATUS_TF	0x01	bit 0: TestFailed
	bit	DEM_UDS_STATUS_ TFTOC	0x02	bit 1: TestFailedThisOpera-tionCycle
	bit	DEM_UDS_STATUS_ PDTC	0x04	bit 2: PendingDTC
	bit	DEM_UDS_STATUS_ CDTC	0x08	bit 3: ConfirmedDTC
	bit	DEM_UDS_STATUS_ TNCSLC	0x10	bit 4: TestNotComplet-edSinceLastClear
	bit	DEM_UDS_STATUS_ TFSLC	0x20	bit 5: TestFailedSinceLastClear
	bit	DEM_UDS_STATUS_ TNCTOC	0x40	bit 6: TestNotComplet-edThisOperationCycle
	bit	DEM_UDS_STATUS_WIR	0x80	bit 7: WarningIndicatorRe-quested

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 8.155: Implementation Data Type Dem_UdsStatusByteType

⌋()

8.6.2 Sender-Receiver-Interfaces

8.6.2.1 DataServices_{Data}

[SWS_Dem_00850] ⌈

Name	DataServices_{Data}	
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

Table 8.156: Service Interface DataServices_{Data}

⌋(*SRS_Diag_04181*)

8.6.3 Client-Server-Interfaces

8.6.3.1 CallbackClearEventAllowed

[SWS_Dem_00620] ⌈

Name	CallbackClearEventAllowed	
Comment	If configured, it gets the permission to clear a specific event from the SW-C. For each event, there can be one port of this interface type.	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.157: Service Interface CallbackClearEventAllowed

Operations

ClearEventAllowed			
Comments	--		
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	--	

Table 8.158: Operation ClearEventAllowed

⌋(*SRS_Diag_04117*)

8.6.3.2 CallbackComponentStatusChanged

[SWS_Dem_01195] ⌈

Name	CallbackComponentStatusChanged	
Comment	--	
IsService	true	
Variation	--	
Possible Errors	--	--

Table 8.159: Service Interface CallbackComponentStatusChanged

Operations

ComponentStatusChanged			
Comments	--		
Variation	--		
Parameters	ComponentFailedStatus	Comment	--
		Type	boolean
		Variation	--
		Direction	IN

Table 8.160: Operation ComponentStatusChanged

⌋*(SRS_Diag_04142)*

8.6.3.3 CallbackDTCStatusChange

[SWS_Dem_00617] ⌈

Name	CallbackDTCStatusChange	
Comment	If configured it triggers SW-Cs on DTC status byte changes via DemCallbackDTCStatusChanged, DemCallbackOBDDTCStatusChanged, and / or DemCallbackJ1939DTCStatusChanged. There can be several ports of this interface type, provided globally by the Dem Service Component.	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.161: Service Interface CallbackDTCStatusChange

Operations

DTCStatusChanged	
Comments	--
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	--	

Table 8.162: Operation DTCStatusChanged

⌋(*SRS_Diag_04061*)

8.6.3.4 CallbackEventDataChanged

[SWS_Dem_00618] ⌈

Name	CallbackEventDataChanged	
Comment	If configured it triggers SW-Cs on event related data changes. For each event, there can be one port of this interface type.	
IsService	true	
Variation	--	
Possible Errors	--	--

Table 8.163: Service Interface CallbackEventDataChanged

Operations

EventDataChanged	
Comments	--
Variation	--

Table 8.164: Operation EventDataChanged

⌋(*SRS_Diag_04155*)

8.6.3.5 CallbackEventUdsStatusChanged

[SWS_Dem_00615] ⌈

Name	CallbackEventUdsStatusChanged	
Comment	If configured it triggers SW-Cs on event status byte changes. For each event, there can be several ports of this interface type.	
IsService	true	
Variation	--	
Possible Errors	--	--

Table 8.165: Service Interface CallbackEventUdsStatusChanged

Operations

CallbackEventUdsStatusChanged			
Comments	--		
Variation	--		
Parameters	EventStatusByteOld	Comment	--
		Type	Dem_UdsStatusByteType
		Variation	--
		Direction	IN
	EventStatusByteNew	Comment	--
		Type	Dem_UdsStatusByteType
		Variation	--
		Direction	IN

Table 8.166: Operation CallbackEventUdsStatusChanged

⌋()

8.6.3.6 CallbackGetFaultDetectCounter

[SWS_Dem_00622] ⌈

Name	CallbackGetFaultDetectCounter	
Comment	If configured it get the monitor-internal fault detection counter value of a specific event from the SW-C.	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.167: Service Interface CallbackGetFaultDetectCounter

Operations

GetFaultDetectionCounter			
Comments	--		
Variation	--		
Parameters	FaultDetectionCounter	Comment	Value of FaultDetectionCounter
		Type	sint8

		Variation	--
		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.168: Operation GetFaultDetectionCounter

⌋(*SRS_Diag_04025*)

8.6.3.7 CallbackInitMonitorForEvent

[SWS_Dem_00613] ⌈

Name	CallbackInitMonitorForEvent	
Comment	If configure it triggers an event-specific initialization of the monitor part of the SW-C). For each event, there can be one port of this interface type.	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.169: Service Interface CallbackInitMonitorForEvent

Operations

InitMonitorForEvent			
Comments	--		
Variation	--		
Parameters	InitMonitorReason	Comment	--
		Type	Dem_InitMonitorReasonType
		Variation	--
		Direction	IN
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.170: Operation InitMonitorForEvent

⌋(*SRS_BSW_00457*)

8.6.3.8 CallbackMonitorStatusChange

[SWS_Dem_91011] ⌈

Name	CallbackMonitorStatusChange

Comment	If configured it triggers SW-Cs on monitor status changes. For each event, there can be several ports of this interface type.	
IsService	true	
Variation	--	
Possible Errors	--	--

Table 8.171: Service Interface CallbackMonitorStatusChange

Operations

MonitorStatusChanged	
Comments	--
Variation	--

Table 8.172: Operation MonitorStatusChanged

⌋()

8.6.3.9 ClearDtcNotification

[SWS_Dem_91003] ⌈

Name	ClearDtcNotification	
Comment	--	
IsService	true	
Variation	(count({ecuc(Dem/DemGeneral/DemEventMemorySet/ DemClearDTCNotification/DemClearDtcNotificationFnc)} == NULL) > 0)	
Possible Errors	0	E_OK

Table 8.173: Service Interface ClearDtcNotification

Operations

ClearDtcNotification			
Comments	--		
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	

Table 8.174: Operation ClearDtcNotification

⌋()

8.6.3.10 ClearDTC

[SWS_Dem_00666] ⌈

Name	ClearDTC	
Comment	Provides the operations only related to complex device drivers. One port of this interface type is provided globally by the Dem Service Component. It has ClientId as a port-defined argument.	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK
	4	DEM_PENDING
	5	DEM_CLEAR_BUSY
	6	DEM_CLEAR_MEMORY_ERROR
	7	DEM_CLEAR_FAILED
	8	DEM_WRONG_DTC
	9	DEM_WRONG_DTCORIGIN
	22	DEM_BUSY

Table 8.175: Service Interface ClearDTC

Operations

ClearDTC		
Comments	--	
Variation	--	
Possible Errors	E_OK	Operation successful
	E_NOT_OK	No DTC selected
	DEM_PENDING	Clearing the DTCs is currently in progress. The caller shall call this function again at a later moment.
	DEM_CLEAR_BUSY	Another client is currently clearing DTCs. The requested operation will not be started and the caller shall try again at a later moment.
	DEM_CLEAR_ MEMORY_ERROR	An error occurred during erasing a memory location (e.g. if DemClearDTCBehavior is set to DEM_CLRRESP_NON-VOLATILE_FINISH and erasing of non-volatile-block failed).
	DEM_CLEAR_FAILED	DTC clearing failed
	DEM_WRONG_DTC	Selected DTC value in selected format does not exist or clearing is restricted by configuration to group of all DTCs only.

	DEM_WRONG_DTCORIGIN	Selected DTCOrigin does not exist.
	DEM_BUSY	A different Dem_SelectDTC dependent operation according to SWS_Dem_01253 of this client is currently in progress.

Table 8.176: Operation ClearDTC

SelectDTC			
Comments	--		
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	No DTC selected	

Table 8.177: Operation SelectDTC

⌋(*SRS_Diag_04122*)

8.6.3.11 CycleQualified

[SWS_Dem_00743] ⌈

Name	CycleQualified	
Comment	--	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.178: Service Interface CycleQualified

Operations

GetCycleQualified	
Comments	--
Variation	--

Parameters	isQualified	Comment	--
		Type	boolean
		Variation	--
		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.179: Operation GetCycleQualified

SetCycleQualified		
Comments	--	
Variation	--	
Possible Errors	E_OK	Operation successful
	E_NOT_OK	--

Table 8.180: Operation SetCycleQualified

⌋(*SRS_Diag_04082*)

8.6.3.12 DTCSuppression

[SWS_Dem_00608] ⌈

Name	DTCSuppression	
Comment	Provides the capability to control the suppression of DTCs. One port of this interface type is provided globally by the Dem Service Component.	
IsService	true	
Variation	({ecuc(Dem/DemGeneral/DemSuppressionSupport)} == DEM_DTC_SUPPRESSION)	
Possible Errors	0	E_OK
	1	E_NOT_OK
	4	DEM_PENDING
	8	DEM_WRONG_DTC
	9	DEM_WRONG_DTCORIGIN

Table 8.181: Service Interface DTCSuppression

Operations

GetDTCSuppression			
Comments	--		
Variation	--		
Parameters	ClientID	Comment	Unique client id, assigned to the instance of the calling module.
		Type	uint8
		Variation	--

		Direction	IN
	SuppressionStatus	Comment	Defines whether the respective DTC is suppressed (TRUE) or enabled (FALSE).
		Type	boolean
		Variation	--
		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	No DTC selected.	
	DEM_PENDING	The requested value is calculated asynchronously and currently not available. The caller can retry later.	
	DEM_WRONG_DTC	Selected DTC value in selected format does not exist.	
	DEM_WRONG_ DTCORIGIN	Selected DTCOrigin does not exist.	

Table 8.182: Operation GetDTCSuppression

SetDTCSuppression			
Comments	--		
Variation	--		
Parameters	ClientID	Comment	Unique client id, assigned to the instance of the calling module.
		Type	uint8
		Variation	--
		Direction	IN
	SuppressionStatus	Comment	--
		Type	boolean
		Variation	--
		Direction	IN
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	No DTC selected.	
	DEM_WRONG_DTC	Selected DTC value in selected format does not exist.	
	DEM_WRONG_ DTCORIGIN	Selected DTCOrigin does not exist.	

Table 8.183: Operation SetDTCSuppression

⌋(*SRS_Diag_04181*)

8.6.3.13 DataServices_{Data}

[SWS_Dem_00621] ⌈

Name	DataServices_{Data}

Comment	If configured it gets 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 "Dem and Dcm module requests PID data elements of SW-C via ReadData operation"). For each data element, one port of this interface type is provided by the SW-Cs.	
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

Table 8.184: Service Interface DataServices_{Data}

Operations

ReadData			
Comments	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_{Data}
		Variation	Data = {ecuc(Dem/DemGeneral/DemDataElementClass.SHORT-NAME)}
		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.185: Operation ReadData

⌋(*SRS_Diag_04061*)

8.6.3.14 DTRCentralReport

[SWS_Dem_00769] ⌈

Name	DTRCentralReport
Comment	Available if OBD support is configured.
IsService	true
Variation	--

Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.186: Service Interface DTRCentralReport

Operations

SetDTR			
Comments	--		
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	--	

Table 8.187: Operation SetDTR

⌋(*SRS_Diag_04082*)

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.

8.6.3.15 DiagnosticInfo

[SWS_Dem_00599] ⌈

Name	DiagnosticInfo	
Comment	Provides 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.	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK
	10	DEM_E_NO_DTC_AVAILABLE
	14	DEM_E_NO_FDC_AVAILABLE
	21	DEM_BUFFER_TOO_SMALL
	48	DEM_NO_SUCH_ELEMENT

Table 8.188: Service Interface DiagnosticInfo

Operations

GetDTCOfEvent			
Comments	--		
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	

Table 8.189: Operation GetDTCOfEvent

GetDebouncingOfEvent	
Comments	--
Variation	(({ecuc(Dem/DemConfigSet/DemEventParameter/DemDebounceAlgorithmClass)} instanceof {ecuc(Dem/DemConfigSet/DemEventParameter/DemDebounceAlgorithmClass/DemDebounceCounterBased)}) ||({ecuc(Dem/DemConfigSet/DemEventParameter/DemDebounceAlgorithmClass)} instanceof {ecuc(Dem/DemConfigSet/DemEventParameter/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	--	

Table 8.190: Operation GetDebouncingOfEvent

GetEventExtendedDataRecordEx			
Comments	--		
Variation	--		
Parameters	RecordNumber	Comment	--
		Type	uint8
		Variation	--
		Direction	IN
	DestBuffer	Comment	--
		Type	Dem_MaxDataValueType
		Variation	--
		Direction	OUT
	Bufsize	Comment	--
		Type	uint16
		Variation	--
		Direction	INOUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	
	DEM_BUFFER_TOO_SMALL	The provided buffer size is too small	
	DEM_NO_SUCH_ELEMENT	The requested event data is not currently stored (but the request was valid) OR The requested record number is not supported by the event OR The requested DID is not supported by the freeze frame (GetEventFreezeFrameDataEx)	

Table 8.191: Operation GetEventExtendedDataRecordEx

GetEventFreezeFrameDataEx	
Comments	--
Variation	--

Parameters	RecordNumber	Comment	--
		Type	uint8
		Variation	--
		Direction	IN
	DataId	Comment	--
		Type	uint16
		Variation	--
		Direction	IN
	DestBuffer	Comment	--
		Type	Dem_MaxDataValueType
		Variation	--
		Direction	OUT
	BufSize	Comment	--
		Type	uint16
		Variation	--
		Direction	INOUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	
	DEM_BUFFER_TOO_SMALL	The provided buffer size is too small	
	DEM_NO_SUCH_ELEMENT	The requested event data is not currently stored (but the request was valid) OR The requested record number is not supported by the event OR The requested DID is not supported by the freeze frame (GetEventFreezeFrameDataEx)	

Table 8.192: Operation GetEventFreezeFrameDataEx

GetEventStatus (obsolete)			
Comments	Gets the current UDS status byte assigned to the DTC for the event **Tags:** atp.Status=obsolete atp.StatusUseInstead=GetEventUdsStatus		
Variation	--		
Parameters	UDSStatusByte	Comment	--
		Type	Dem_UdsStatusByteType
		Variation	--
		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.193: Operation GetEventStatus (obsolete)

GetEventUdsStatus			
Comments	Gets the current UDS status byte assigned to the DTC for the event		
Variation	--		
Parameters	UDSStatusByte	Comment	--
		Type	Dem_UdsStatusByteType

		Variation	--
		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.194: Operation GetEventUdsStatus

GetFaultDetectionCounter			
Comments	--		
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	

Table 8.195: Operation GetFaultDetectionCounter

GetMonitorStatus			
Comments	--		
Variation	--		
Parameters	MonitorStatus	Comment	Monitor status byte of the requested event. If the return value of the function call is E_NOT_OK, this parameter does not contain valid data.
		Type	Dem_MonitorStatusType
		Variation	--
		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.196: Operation GetMonitorStatus

⌋(*SRS_Diag_04010*)

Note: These DiagnosticInfo ports are also available for each BSW event (so that also SW-Cs can access on).

8.6.3.16 DiagnosticMonitor

[SWS_Dem_00598] ⌈

Name	DiagnosticMonitor

Comment	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.	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.197: Service Interface DiagnosticMonitor

Operations

ClearPrestoredFreezeFrame		
Comments	--	
Variation	{ecuc(Dem/DemGeneral/DemMaxNumberPrestoredFF)} > 0	
Possible Errors	E_OK	Request to reset the event status was successful accepted.
	E_NOT_OK	Request to reset the event status failed or is not allowed, because the event is already tested in this operation cycle.

Table 8.198: Operation ClearPrestoredFreezeFrame

PrestoreFreezeFrame		
Comments	--	
Variation	{ecuc(Dem/DemGeneral/DemMaxNumberPrestoredFF)} > 0	
Possible Errors	E_OK	Request to reset the event status was successful accepted.
	E_NOT_OK	Request to reset the event status failed or is not allowed, because the event is already tested in this operation cycle.

Table 8.199: Operation PrestoreFreezeFrame

ResetEventDebounceStatus			
Comments	--		
Variation	({ecuc(Dem/DemGeneral/DemDebounceCounterBasedSupport)} == true) || ({ecuc(Dem/DemGeneral/ DemDebounceTimeBasedSupport)} == true)		
Parameters	DebounceResetStatus	Comment	--
		Type	Dem_DebounceResetStatusType
		Variation	--
		Direction	IN
Possible Errors	E_OK	Request to reset the event status was successful accepted.	
	E_NOT_OK	Request to reset the event status failed or is not allowed, because the event is already tested in this operation cycle.	

Table 8.200: Operation ResetEventDebounceStatus

ResetEventStatus		
Comments	--	
Variation	--	
Possible Errors	E_OK	Operation successful
	E_NOT_OK	Request to reset the event status failed or is not allowed, because the event is already tested in this operation cycle.

Table 8.201: Operation ResetEventStatus

SetEventDisabled		
Comments	--	
Variation	{ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT	
Possible Errors	E_OK	Request to reset the event status was successful accepted.
	E_NOT_OK	Request to reset the event status failed or is not allowed, because the event is already tested in this operation cycle.

Table 8.202: Operation SetEventDisabled

SetEventStatus			
Comments	--		
Variation	--		
Parameters	EventStatus	Comment	--
		Type	Dem_EventStatusType
		Variation	--
		Direction	IN
Possible Errors	E_OK	Request to reset the event status was successful accepted.	
	E_NOT_OK	Request to reset the event status failed or is not allowed, because the event is already tested in this operation cycle.	

Table 8.203: Operation SetEventStatus

⌋(*SRS_Diag_04061*)

Note: Each port of the DiagnosticMonitor interface is only connected to one monitor port.

Caveat: all operations within this client/server interface are mutual exclusive per server port (same eventId). Preempted invocations may be ignored and not processed by the Dem.

8.6.3.17 EnableCondition

[SWS_Dem_00604] ⌈

Name	EnableCondition	
Comment	If at least one enable condition is configured it provide the capability to set an enable condition. One port of this interface type is provided per enable condition by the Dem Service Component. It has EnableConditionId as a port-defined argument.	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.204: Service Interface EnableCondition

Operations

SetEnableCondition			
Comments	--		
Variation	--		
Parameters	ConditionFulfilled	Comment	--
		Type	boolean
		Variation	--
		Direction	IN
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.205: Operation SetEnableCondition

⌋()

8.6.3.18 EventAvailable

[SWS_Dem_01193] ⌈

Name	EventAvailable	
Comment	--	
IsService	true	
Variation	{ecuc(Dem/DemGeneral/DemAvailabilitySupport)} == DEM_EVENT_AVAILABILITY	
Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.206: Service Interface EventAvailable

Operations

SetEventAvailable

Comments	--		
Variation	--		
Parameters	AvailableStatus	Comment	--
		Type	boolean
		Variation	--
		Direction	IN
Possible Errors	E_OK	Request to set the availability status was successful.	
	E_NOT_OK	Request to set the availability status not accepted.	

Table 8.207: Operation SetEventAvailable

⌋()

8.6.3.19 EventFailureCycleCounterThreshold

[SWS_Dem_91018] ⌈

Name	EventFailureCycleCounterThreshold	
Comment	Provides the capability for dynamical adaptation of the failure cycle threshold. One port of this interface is provided per diagnostic event with EventId as a port-defined argument.	
IsService	true	
Variation	(count({ecuc(Dem/DemConfigSet/DemEventParameter/ DemEventFailureCycleCounterThresholdAdaptable)} == TRUE) > 0)	
Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.208: Service Interface EventFailureCycleCounterThreshold

Operations

SetEventFailureCycleCounterThreshold			
Comments	--		
Variation	--		
Parameters	FailureCycleCounterThreshold	Comment	Failure cycle counter threshold of event to be set.
		Type	uint8
		Variation	--
		Direction	IN
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.209: Operation SetEventFailureCycleCounterThreshold

⌋()

8.6.3.20 EvMemOverflowIndication

[SWS_Dem_00607] ⌈

Name	EvMemOverflowIndication	
Comment	If the respective event memory is configured it provides the status of the event memory). 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.	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.210: Service Interface EvMemOverflowIndication

Operations

GetEventMemoryOverflow			
Comments	--		
Variation	--		
Parameters	OverflowIndication	Comment	--
		Type	boolean
		Variation	--
		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.211: Operation GetEventMemoryOverflow

GetNumberOfEventMemoryEntries			
Comments	--		
Variation	--		
Parameters	GetNumberOfEventMemoryEntries	Comment	--
		Type	uint8
		Variation	--
		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.212: Operation GetNumberOfEventMemoryEntries

⌋(*SRS_Diag_04105*)

8.6.3.21 EventStatus

[SWS_Dem_00838] ⌈

Name	EventStatus	
Comment	Provides 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.	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.213: Service Interface EventStatus

Operations

SetWIRStatus			
Comments	--		
Variation	--		
Parameters	WIRStatus	Comment	--
		Type	boolean
		Variation	--
		Direction	IN
Possible Errors	E_OK	Request to set the WIR status was successful.	
	E_NOT_OK	Request to set the WIR status was not accepted (e.g. disabled controlDTCSetting) and should be repeated.	

Table 8.214: Operation SetWIRStatus

⌋(*SRS_Diag_04128*)

Note: Each port of the EventStatus interface is only connected to one port of the failsafe SW-C.

8.6.3.22 GeneralCallbackEventDataChanged

[SWS_Dem_00619] ⌈

Name	GeneralCallbackEventDataChanged	
Comment	--	
IsService	true	
Variation	--	
Possible Errors	--	--

Table 8.215: Service Interface GeneralCallbackEventDataChanged

Operations

EventDataChanged

Comments	--		
Variation	--		
Parameters	EventId	Comment	--
		Type	Dem_EventIdType
		Variation	--
		Direction	IN

Table 8.216: Operation EventDataChanged

⌋(*SRS_Diag_04155*)

8.6.3.23 GeneralCallbackEventUdsStatusChanged

[SWS_Dem_00616] ⌈

Name	GeneralCallbackEventUdsStatusChanged	
Comment	--	
IsService	true	
Variation	--	
Possible Errors	--	--

Table 8.217: Service Interface GeneralCallbackEventUdsStatusChanged

Operations

GeneralCallbackEventUdsStatusChanged			
Comments	--		
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

Table 8.218: Operation GeneralCallbackEventUdsStatusChanged

⌋(*SRS_Diag_04061*)

8.6.3.24 GeneralCallbackMonitorStatusChanged

[SWS_Dem_91013] ⌈

Name	GeneralCallbackMonitorStatusChanged	
Comment	--	
IsService	true	
Variation	--	
Possible Errors	--	--

Table 8.219: Service Interface GeneralCallbackMonitorStatusChanged

Operations

MonitorStatusChanged			
Comments	--		
Variation	--		
Parameters	EventId	Comment	--
		Type	Dem_EventIdType
		Variation	--
		Direction	IN

Table 8.220: Operation MonitorStatusChanged

⌋()

8.6.3.25 GeneralDiagnosticInfo

[SWS_Dem_00600] ⌈

Name	GeneralDiagnosticInfo	
Comment	--	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK
	10	DEM_E_NO_DTC_AVAILABLE
	14	DEM_E_NO_FDC_AVAILABLE
	21	DEM_BUFFER_TOO_SMALL
	48	DEM_NO_SUCH_ELEMENT

Table 8.221: Service Interface GeneralDiagnosticInfo

Operations

GetDTCOfEvent	
Comments	--
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	

Table 8.222: Operation GetDTCOfEvent

GetDebouncingOfEvent			
Comments	--		
Variation	(({ecuc(Dem/DemConfigSet/DemEventParameter/ DemDebounceAlgorithmClass)} instanceof {ecuc(Dem/ DemConfigSet/DemEventParameter/DemDebounceAlgorithmClass/ DemDebounceCounterBased)}) ||({ecuc(Dem/DemConfigSet/ DemEventParameter/DemDebounceAlgorithmClass)} instanceof {ecuc(Dem/DemConfigSet/DemEventParameter/ 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	--	

Table 8.223: Operation GetDebouncingOfEvent

GetEventExtendedDataRecordEx			
Comments	--		
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
	Bufsize	Comment	--
		Type	uint16
		Variation	--
		Direction	INOUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	
	DEM_BUFFER_TOO_SMALL	The provided buffer size is too small	
	DEM_NO_SUCH_ELEMENT	The requested event data is not currently stored (but the request was valid) OR The requested record number is not supported by the event OR The requested DID is not supported by the freeze frame (GetEventFreezeFrameDataEx)	

Table 8.224: Operation GetEventExtendedDataRecordEx

GetEventFreezeFrameDataEx			
Comments	--		
Variation	--		
Parameters	EventId	Comment	--
		Type	Dem_EventIdType
		Variation	--
		Direction	IN
	RecordNumber	Comment	--
		Type	uint8
		Variation	--
		Direction	IN
	DataId	Comment	--
		Type	uint16
		Variation	--
		Direction	IN
	DestBuffer	Comment	--
		Type	Dem_MaxDataValueType
		Variation	--
		Direction	OUT

	BufSize	Comment	--
		Type	uint16
		Variation	--
		Direction	INOUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	
	DEM_BUFFER_TOO_SMALL	The provided buffer size is too small	
	DEM_NO_SUCH_ELEMENT	The requested event data is not currently stored (but the request was valid) OR The requested record number is not supported by the event OR The requested DID is not supported by the freeze frame (GetEventFreezeFrameDataEx)	

Table 8.225: Operation GetEventFreezeFrameDataEx

GetEventStatus (obsolete)			
Comments	Gets the current UDS status byte assigned to the DTC for the event **Tags:** atp.Status=obsolete atp.StatusUseInstead=GetEventUdsStatus		
Variation	--		
Parameters	EventId	Comment	--
		Type	Dem_EventIdType
		Variation	--
		Direction	IN
	UDSStatusByte	Comment	--
		Type	Dem_UdsStatusByteType
		Variation	--
		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.226: Operation GetEventStatus (obsolete)

GetEventUdsStatus			
Comments	Gets the current UDS status byte assigned to the DTC for the event		
Variation	--		
Parameters	EventId	Comment	--
		Type	Dem_EventIdType
		Variation	--
		Direction	IN
	UDSStatusByte	Comment	--
		Type	Dem_UdsStatusByteType
		Variation	--
		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.227: Operation GetEventUdsStatus

GetFaultDetectionCounter			
Comments	--		
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	

Table 8.228: Operation GetFaultDetectionCounter

GetMonitorStatus			
Comments	--		
Variation	--		
Parameters	EventID	Comment	Identification of an event by assigned EventId.
		Type	Dem_EventIdType
		Variation	--
		Direction	IN
	MonitorStatus	Comment	Monitor status byte of the requested event. If the return value of the function call is E_NOT_OK, this parameter does not contain valid data.
		Type	Dem_MonitorStatusType
		Variation	--
		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.229: Operation GetMonitorStatus

⌋(*SRS_Diag_04181*)

8.6.3.26 GetDataOfPID21

[SWS_Dem_01092] ⌈

Name	GetDataOfPID21	
Comment	--	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.230: Service Interface GetDataOfPID21

Operations

GetDataOfPID21			
Comments	--		
Variation	--		
Parameters	PID21value	Comment	--
		Type	Dem_PID21valueType
		Variation	--
		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.231: Operation GetDataOfPID21

⌋(*SRS_Diag_04061*, *SRS_Diag_04001*)

8.6.3.27 IndicatorStatus

[SWS_Dem_00606] ⌈

Name	IndicatorStatus	
Comment	One port of this interface type is provided per indicator by the Dem Service Component. It has IndicatorId as a port-defined argument.	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.232: Service Interface IndicatorStatus

Operations

GetIndicatorStatus			
Comments	--		
Variation	--		
Parameters	IndicatorStatus	Comment	--
		Type	Dem_IndicatorStatusType
		Variation	--
		Direction	OUT

Possible Errors	E_OK	Operation successful
	E_NOT_OK	--

Table 8.233: Operation GetIndicatorStatus

⌋(*SRS_Diag_04128*)

8.6.3.28 IUMPRDenominator

[SWS_Dem_00611] ⌈

Name	IUMPRDenominator	
Comment	If OBD is configured it provides the capability to define the number of times the vehicle operation has been fulfilled. One port of this interface type is provided per ratio Id by the Dem Service Component. It has RatioID as a port-defined argument.	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.234: Service Interface IUMPRDenominator

Operations

RepIUMPRDenLock		
Comments	--	
Variation	--	
Possible Errors	E_OK	Operation successful
	E_NOT_OK	--

Table 8.235: Operation RepIUMPRDenLock

RepIUMPRDenRelease		
Comments	--	
Variation	--	
Possible Errors	E_OK	Operation successful
	E_NOT_OK	--

Table 8.236: Operation RepIUMPRDenRelease

⌋()

8.6.3.29 IUMPRDenominatorCondition

[SWS_Dem_00742] ⌈

Name	IUMPRDenominatorCondition	
Comment	If OBD is configured it broadcasts the status information of the General Denominator and additional denominator conditions among all OBD relevant ECUs. 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.	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.237: Service Interface IUMPRDenominatorCondition

Operations

GetIUMPRDenCondition			
Comments	--		
Variation	--		
Parameters	ConditionStatus	Comment	--
		Type	Dem_IumprDenomCondStatusType
		Variation	--
		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.238: Operation GetIUMPRDenCondition

SetIUMPRDenCondition			
Comments	--		
Variation	--		
Parameters	ConditionStatus	Comment	--
		Type	Dem_IumprDenomCondStatusType
		Variation	--
		Direction	IN
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.239: Operation SetIUMPRDenCondition

⌋(*SRS_Diag_04082*)

8.6.3.30 IUMPRNumerator

[SWS_Dem_00610] ⌈

Name	IUMPRNumerator	
Comment	If OBD is configured it provides the capability to define the number of times a fault could have been found. One port of this interface type is provided per ratio Id by the Dem Service Component. It has RatioID as a port-defined argument.	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.240: Service Interface IUMPRNumerator

Operations

RepIUMPRFaultDetect		
Comments	--	
Variation	--	
Possible Errors	E_OK	Operation successful
	E_NOT_OK	--

Table 8.241: Operation RepIUMPRFaultDetect

⌋(*SRS_Diag_04082*)

8.6.3.31 OperationCycle

[SWS_Dem_00601] ⌈

Name	OperationCycle	
Comment	Provides 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.	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.242: Service Interface OperationCycle

Operations

GetOperationCycleState			
Comments	--		
Variation	--		
Parameters	CycleState	Comment	--
		Type	Dem_OperationCycleStateType
		Variation	--

		Direction	OUT
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.243: Operation GetOperationCycleState

SetOperationCycleState			
Comments	--		
Variation	--		
Parameters	CycleState	Comment	--
		Type	Dem_OperationCycleStateType
		Variation	--
		Direction	IN
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.244: Operation SetOperationCycleState

⌋(*SRS_Diag_04076*)

8.6.3.32 PowerTakeOff

[SWS_Dem_00612] ⌈

Name	PowerTakeOff	
Comment	Available if OBD support is configured. One port of this interface type is provided by the Dem Service Component.	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.245: Service Interface PowerTakeOff

Operations

SetPtoStatus			
Comments	--		
Variation	--		
Parameters	PtoStatus	Comment	--
		Type	boolean
		Variation	--
		Direction	IN
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.246: Operation SetPtoStatus

⌋(*SRS_Diag_04082*)

8.6.3.33 SetDataOfPID21

[SWS_Dem_00745] ⌈

Name	SetDataOfPID21	
Comment	--	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.247: Service Interface SetDataOfPID21

Operations

SetDataOfPID21			
Comments	--		
Variation	--		
Parameters	PID21value	Comment	--
		Type	Dem_PID21valueType
		Variation	--
		Direction	IN
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.248: Operation SetDataOfPID21

⌋(*SRS_Diag_04061*, *SRS_Diag_04001*)

8.6.3.34 SetDataOfPID31

[SWS_Dem_00746] ⌈

Name	SetDataOfPID31	
Comment	--	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.249: Service Interface SetDataOfPID31

Operations

SetDataOfPID31

Comments	--		
Variation	--		
Parameters	PID31value	Comment	--
		Type	Dem_PID31valueType
		Variation	--
		Direction	IN
Possible Errors	E_OK	Operation successful	
	E_NOT_OK	--	

Table 8.250: Operation SetDataOfPID31

⌋(*SRS_Diag_04082*)

8.6.3.35 StorageCondition

[SWS_Dem_00605] ⌈

Name	StorageCondition	
Comment	Provides 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.	
IsService	true	
Variation	--	
Possible Errors	0	E_OK
	1	E_NOT_OK

Table 8.251: Service Interface StorageCondition

Operations

SetStorageCondition			
Comments	--		
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	--	

Table 8.252: Operation SetStorageCondition

⌋(*SRS_Diag_04095*)

8.6.4 Ports

8.6.4.1 CBClrEvt

[SWS_Dem_91024] ⌈

Name	CBClrEvt_{Name}		
Kind	RequiredPort	***Interface***	CallbackClearEventAllowed
Description	--		
Variation	(({ecuc(Dem/DemConfigSet/DemEventParameter/ DemCallbackClearEventAllowed)} != NULL) &&({ecuc(Dem/ DemConfigSet/DemEventParameter/ DemCallbackClearEventAllowed/ DemCallbackClearEventAllowedFnc)} == NULL)) Name = {ecuc(Dem/DemConfigSet/DemEventParameter/ DemCallbackClearEventAllowed.SHORT-NAME)}		

Table 8.253: Port CBClrEvt_{Name}

⌋()

8.6.4.2 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)}		

Table 8.254: Port CBDataEvt_{Name}

⌋(*SRS_Diag_04155*)

8.6.4.3 CBFaultDetectCtr

[SWS_Dem_01004] ⌈

Name	CBFaultDetectCtr_{Name}		
Kind	RequiredPort	***Interface***	CallbackGetFaultDetect Counter
Description	--		

Variation	({ecuc(Dem/DemConfigSet/DemEventParameter/ DemDebounceAlgorithmClass/DemDebounceMonitorInternal/ DemCallbackGetFDC/DemCallbackGetFDCFnc)} == NULL) Name = {ecuc(Dem/DemConfigSet/DemEventParameter/ DemDebounceAlgorithmClass/DemDebounceMonitorInternal/ DemCallbackGetFDC.SHORT-NAME)}

Table 8.255: Port CBFaultDetectCtr_{Name}

⌋(*SRS_Diag_04125*)

8.6.4.4 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)}		

Table 8.256: Port CBInitEvt_{Name}

⌋(*SRS_BSW_00457*)

8.6.4.5 CBStatusDTC

[SWS_Dem_01007] ⌈

Name	CBStatusDTC_{Name}		
Kind	RequiredPort	***Interface***	CallbackDTCStatusChange
Description	--		
Variation	Name = {ecuc(Dem/DemGeneral/DemClient/ DemCallbackDTCStatusChanged.SHORT-NAME)}		

Table 8.257: Port CBStatusDTC_{Name}

⌋(*SRS_Diag_04142*)

8.6.4.6 CBEventUdsStatusChanged

[SWS_Dem_01008] ⌈

Name	CBEventUdsStatusChanged_{EventName}_{CallbackName}		
Kind	RequiredPort	**Interface**	CallbackEventUdsStatus Changed
Description	--		
Variation	EventName = {ecuc(Dem/DemConfigSet/DemEventParameter. SHORT-NAME)} CallbackName = {ecuc(Dem/DemConfigSet/ DemEventParameter/DemCallbackEventUdsStatusChanged. SHORT-NAME)}		

Table 8.258: Port CBEventUdsStatusChanged_{EventName}_{CallbackName}

⌋()

8.6.4.7 CBMonitorStatusChanged

[SWS_Dem_91012] ⌈

Name	CBMonitorStatusChanged_{EventName}_{CallbackName}		
Kind	RequiredPort	**Interface**	CallbackMonitorStatusChange
Description	--		
Variation	({ecuc(Dem/DemConfigSet/DemEventParameter.DemEventKind)} == DEM_EVENT_KIND_SWC) EventName = {ecuc(Dem/ DemConfigSet/DemEventParameter.SHORT-NAME)} CallbackName = {ecuc(Dem/DemConfigSet/DemEventParameter/ DemCallbackMonitorStatusChanged.SHORT-NAME)}		

Table 8.259: Port CBMonitorStatusChanged_{EventName}_{CallbackName}

⌋()

8.6.4.8 CBStatusComp

[SWS_Dem_01197] ⌈

Name	CBStatusComp_{ComponentName}		
Kind	RequiredPort	**Interface**	CallbackComponentStatus Changed
Description	--		
Variation	ComponentName = {ecuc(Dem/DemConfigSet/DemComponent. SHORT-NAME)}		

Table 8.260: Port CBStatusComp_{ComponentName}

⌋(*SRS_Diag_04142*)

8.6.4.9 ClearDTC

[SWS_Dem_01009] ⌈

Name	ClearDTC_{Client}		
Kind	ProvidedPort	***Interface***	ClearDTC
Description	--		
Port Defined Argument Value(s)	Type	uint8	
	Value	{ecuc(Dem/DemGeneral/DemClient. Short-Name)}	
Variation	({ecuc(Dem/DemGeneral/DemClient/DemClientUsesRte)}==True) && ({ecuc(Dem/DemGeneral/DemClient/DemClientFunctionality)} == DEM_CLIENT_USES_FULL_FUNCTIONALITY) Client = {ecuc(Dem/DemGeneral/DemClient.SHORT-NAME)}		

Table 8.261: Port ClearDTC_{Client}

⌋()

8.6.4.10 ClearDtcNotification

[] ⌈

Name	ClearDtcNotification_{EventMemorySet}_{Notification}		
Kind	ProvidedPort	***Interface***	ClearDtcNotification
Description	called by the Dem when performing a clear DTC operation		
Variation	({ecuc(Dem/DemGeneral/DemEventMemorySet/ DemClearDTCNotification/DemClearDtcNotificationFnc)}== NULL) EventMemorySet = {ecuc(Dem/DemGeneral/DemEventMemorySet. SHORT-NAME)} Notification = {ecuc(Dem/DemGeneral/ DemEventMemorySet/DemClearDTCNotification.SHORT-NAME)}		

Table 8.262: Port ClearDtcNotification_{EventMemorySet}_{Notification}

⌋()

8.6.4.11 ControlDTCSuppression

[SWS_Dem_01010] ⌈

Name	ControlDTCSuppression		
Kind	ProvidedPort	***Interface***	DTCSuppression
Description	--		
Variation	({ecuc(Dem/DemGeneral/DemSuppressionSupport)} == DEM_DTC_SUPPRESSION)		

Table 8.263: Port ControlDTCSuppression

⌋()

8.6.4.12 ControlEventAvailable

[SWS_Dem_01011] ⌈

Name	ControlEventAvailable		
Kind	ProvidedPort	***Interface***	EventAvailable
Description	--		
Port Defined Argument Value(s)	Type	Dem_EventIdType	
	Value	{ecuc(Dem/DemConfigSet/ DemEventParameter/DemEventId. value)}	
Variation	{ecuc(Dem/DemGeneral/DemAvailabilitySupport)} == DEM_EVENT_AVAILABILITY		

Table 8.264: Port ControlEventAvailable

⌋()

8.6.4.13 ControlEventFailureCycleCounterThreshold

[SWS_Dem_91019] ⌈

Name	ControlEventFailureCycleCounterThreshold_{Name}		
Kind	ProvidedPort	***Interface***	EventFailureCycleCounter Threshold
Description	--		
Port Defined Argument Value(s)	Type	Dem_EventIdType	
	Value	{ecuc(Dem/DemConfigSet/ DemEventParameter/DemEventId. value)}	
Variation	{ecuc(Dem/DemConfigSet/DemEventParameter/ DemEventFailureCycleCounterThresholdAdaptable)} == TRUE Name = {ecuc(Dem/DemConfigSet/DemEventParameter. SHORT-NAME)}		

Table 8.265: Port ControlEventFailureCycleCounterThreshold_{Name}

⌋()

8.6.4.14 CycleQualified

[SWS_Dem_01021] ⌈

Name	CycleQualified		
Kind	ProvidedPort	***Interface***	CycleQualified
Description	--		
Port Defined Argument Value(s)	Type	uint8	
	Value	{ecuc(Dem/DemGeneral/ DemOperationCycle/ DemOperationCycleId.value)}	
Variation	({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT) &&({ecuc(Dem/DemGeneral/ DemEventMemorySet/DemMaxNumberEventEntryPermanent)} > 0)		

Table 8.266: Port CycleQualified

⌋(*SRS_Diag_04076*)

8.6.4.15 DataServices_{Data}

[SWS_Dem_01012] ⌈

Name	DataServices_{Data}		
Kind	RequiredPort	***Interface***	DataServices_{Data}
Description	--		
Variation	(({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)}) Data = {ecuc(Dem/DemGeneral/DemDataElementClass.SHORT-NAME)}		

Table 8.267: Port DataServices_{Data}

⌋()

8.6.4.16 DTR

[SWS_Dem_01039] ⌈

Name	DTR_{Name}		
Kind	ProvidedPort	***Interface***	DTRCentralReport
Description	--		

Port Defined Argument Value(s)	Type	uint16
	Value	{ecuc(Dem/DemConfigSet/DemDtrs/DemDtr/DemDtrId.value)}
Variation	({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT) Name = {ecuc(Dem/DemConfigSet/DemDtrs/DemDtr.SHORT-NAME)}	

Table 8.268: Port DTR_{Name}

⌋()

8.6.4.17 EnableCond

[SWS_Dem_01038] ⌈

Name	EnableCond_{Name}		
Kind	ProvidedPort	***Interface***	EnableCondition
Description	--		
Port Defined Argument Value(s)	Type	uint8	
	Value	{ecuc(Dem/DemGeneral/DemEnableCondition/DemEnableConditionId.value)}	
Variation	Name = {ecuc(Dem/DemGeneral/DemEnableCondition.SHORT-NAME)}		

Table 8.269: Port EnableCond_{Name}

⌋()

8.6.4.18 Event

[SWS_Dem_01037] ⌈

Name	Event_{Name}		
Kind	ProvidedPort	***Interface***	DiagnosticMonitor
Description	--		
Port Defined Argument Value(s)	Type	Dem_EventIdType	
	Value	{ecuc(Dem/DemConfigSet/DemEventParameter/DemEventId.value)}	
Variation	Name = {ecuc(Dem/DemConfigSet/DemEventParameter.SHORT-NAME)}		

Table 8.270: Port Event_{Name}

⌋(*SRS_Diag_04063*)

8.6.4.19 EventStatus

[SWS_Dem_01036] ⌈

Name	EventStatus_{Name}		
Kind	ProvidedPort	***Interface***	EventStatus
Description	--		
Port Defined Argument Value(s)	Type	Dem_EventIdType	
	Value	{ecuc(Dem/DemConfigSet/DemEventParameter/DemEventId.value)}	
Variation	Name = {ecuc(Dem/DemConfigSet/DemEventParameter.SHORT-NAME)}		

Table 8.271: Port EventStatus_{Name}

⌋()

8.6.4.20 EventInfo

[SWS_Dem_01034] ⌈

Name	EventInfo_{Name}		
Kind	ProvidedPort	***Interface***	DiagnosticInfo
Description	--		
Port Defined Argument Value(s)	Type	Dem_EventIdType	
	Value	{ecuc(Dem/DemConfigSet/DemEventParameter/DemEventId.value)}	
Variation	Name = {ecuc(Dem/DemConfigSet/DemEventParameter.SHORT-NAME)}		

Table 8.272: Port EventInfo_{Name}

⌋(*SRS_Diag_04063*)

8.6.4.21 GeneralCBDataEvt

[SWS_Dem_01041] ⌈

Name	GeneralCBDataEvt		
Kind	RequiredPort	***Interface***	GeneralCallbackEventDataChanged
Description	--		
Variation	({ecuc(Dem/DemGeneral/DemGeneralInterfaceSupport)} == True)		

Table 8.273: Port GeneralCBDataEvt

⌋()

8.6.4.22 GeneralCBMonitorStatusChanged

[SWS_Dem_91014] ⌈

Name	GeneralCBMonitorStatusChanged		
Kind	RequiredPort	***Interface***	GeneralCallbackMonitorStatusChanged
Description	--		
Variation	({ecuc(Dem/DemGeneral/DemGeneralInterfaceSupport)} == True)		

Table 8.274: Port GeneralCBMonitorStatusChanged

⌋()

8.6.4.23 GeneralCBStatusEvt

[SWS_Dem_01032] ⌈

Name	GeneralCBStatusEvt		
Kind	RequiredPort	***Interface***	GeneralCallbackEventUdsStatusChanged
Description	--		
Variation	({ecuc(Dem/DemGeneral/DemGeneralInterfaceSupport)} == True)		

Table 8.275: Port GeneralCBStatusEvt

⌋(*SRS_Diag_04142*)

8.6.4.24 GeneralEvtInfo

[SWS_Dem_01031] ⌈

Name	GeneralEvtInfo		
Kind	ProvidedPort	**Interface**	GeneralDiagnosticInfo
Description	--		
Variation	({ecuc(Dem/DemGeneral/DemGeneralInterfaceSupport)} == True)		

Table 8.276: Port GeneralEvtInfo

⌋()

8.6.4.25 IndStatus

[SWS_Dem_01030] ⌈

Name	IndStatus_{Name}		
Kind	ProvidedPort	**Interface**	IndicatorStatus
Description	--		
Port Defined Argument Value(s)	Type	uint8	
	Value	{ecuc(Dem/DemGeneral/ DemEventMemorySet/DemIndicator/ DemIndicatorID.value)}	
Variation	Name = {ecuc(Dem/DemGeneral/DemEventMemorySet/ DemIndicator.SHORT-NAME)}		

Table 8.277: Port IndStatus_{Name}

⌋()

8.6.4.26 IUMPRDenominator

[SWS_Dem_01029] ⌈

Name	IUMPRDenominator_{Name}		
Kind	ProvidedPort	**Interface**	IUMPRDenominator
Description	--		
Port Defined Argument Value(s)	Type	Dem_RatioIdType	
	Value	{ecuc(Dem/DemGeneral/DemRatio/ DemRatioId.value)}	
Variation	({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT) Name = {ecuc(Dem/ DemGeneral/DemRatio/DemRatioId.SHORT-NAME)}		

Table 8.278: Port IUMPRDenominator_{Name}

⌋()

8.6.4.27 IUMPRDenominatorCondition

[SWS_Dem_01028] ⌈

Name	IUMPRDenominatorCondition_{Name}		
Kind	ProvidedPort	***Interface***	IUMPRDenominatorCondition
Description	--		
Port Defined Argument Value(s)	Type	Dem_IumprDenomCondIdType	
	Value	TBD	
Variation	({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT) Name = {ecuc(Dem/DemGeneral/DemRatio/DemIUMPRDenGroup.SHORT-NAME)}		

Table 8.279: Port IUMPRDenominatorCondition_{Name}

⌋()

8.6.4.28 IUMPRNumerator

[SWS_Dem_01027] ⌈

Name	IUMPRNumerator_{Name}		
Kind	ProvidedPort	***Interface***	IUMPRNumerator
Description	--		
Port Defined Argument Value(s)	Type	Dem_RatioIdType	
	Value	{ecuc(Dem/DemGeneral/DemRatio/DemRatioId.value)}	
Variation	({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT) Name = {ecuc(Dem/DemGeneral/DemRatio/DemRatioId.SHORT-NAME)}		

Table 8.280: Port IUMPRNumerator_{Name}

⌋()

8.6.4.29 OpCycle

[SWS_Dem_01026] ⌈

Name	OpCycle_{Name}

Kind	ProvidedPort	***Interface***	OperationCycle
Description	--		
Port Defined Argument Value(s)	Type	uint8	
	Value	{ecuc(Dem/DemGeneral/ DemOperationCycle/ DemOperationCycleId.value)}	
Variation	Name = {ecuc(Dem/DemGeneral/DemOperationCycle. SHORT-NAME)}		

Table 8.281: Port OpCycle_{Name}

⌋(*SRS_Diag_04076*)

8.6.4.30 OverflowIndMirrorMemory

[SWS_Dem_01025] ⌈

Name	OverflowIndMirrorMemory_{Client}		
Kind	ProvidedPort	***Interface***	EvMemOverflowIndication
Description	--		
Port Defined Argument Value(s)	Type	uint8	
	Value	--	
	Type	Dem_DTCOriginType	
	Value	0x02	
Variation	({ecuc(Dem/DemGeneral/DemEventMemorySet/ DemMirrorMemory)} != NULL) &&({ecuc(Dem/DemGeneral/ DemClient/DemClientUsesRte)} ==TRUE)) Client= {ecuc(Dem/ DemGeneral/DemClient.SHORT-NAME)}		

Table 8.282: Port OverflowIndMirrorMemory_{Client}

⌋()

8.6.4.31 OverflowIndPermanentMemory

[SWS_Dem_01024] ⌈

Name	OverflowIndPermanentMemory		
Kind	ProvidedPort	***Interface***	EvMemOverflowIndication
Description	--		
Port Defined Argument Value(s)	Type	Dem_DTCOriginType	

	Value	0x03
Variation	({ecuc(Dem/DemGeneral/DemEventMemorySet/ DemMaxNumberEventEntryPermanent)} > 0)	

Table 8.283: Port OverflowIndPermanentMemory

⌋()

8.6.4.32 OverflowIndPrimaryMemory

[SWS_Dem_01023] ⌈

Name	OverflowIndPrimaryMemory_{Client}		
Kind	ProvidedPort	***Interface***	EvMemOverflowIndication
Description	--		
Port Defined Argument Value(s)	Type	uint8	
	Value	--	
	Type	Dem_DTCOriginType	
	Value	0x01	
Variation	(({ecuc(Dem/DemGeneral/DemClient/DemClientUsesRte)} ==TRUE)) Client= {ecuc(Dem/DemGeneral/DemClient. SHORT-NAME})		

Table 8.284: Port OverflowIndPrimaryMemory_{Client}

⌋(*SRS_Diag_04093*)

8.6.4.33 OverflowIndUserDefinedMemory

[SWS_Dem_01022] ⌈

Name	OverflowIndUserDefinedMemory_{Client}_{UserDefinedMemory}		
Kind	ProvidedPort	***Interface***	EvMemOverflowIndication
Description	--		
Port Defined Argument Value(s)	Type	uint8	
	Value	--	
	Type	Dem_DTCOriginType	

	Value	{ecuc(Dem/DemGeneral/ DemEventMemorySet/ DemUserDefinedMemory/ DemUserDefinedMemoryIdentifier. value)}
Variation	({ecuc(Dem/DemGeneral/DemClient/ DemEventMemorySetRef->DemEventMemorySet/ DemUserDefinedMemory != NULL)})(({ecuc(Dem/DemGeneral/ DemClient/DemClientUsesRte)} ==TRUE)) Client= {ecuc(Dem/ DemGeneral/DemEventMemorySet.SHORT-NAME}) UserDefinedMemory = {ecuc(Dem/DemGeneral/DemClient/ DemEventMemorySetRef->DemEventMemorySet/ DemUserDefinedMemory.SHORT-NAME)}	

Table 8.285: Port OverflowIndUserDefinedMemory_{Client}_{UserDefinedMemory}

⌋()

8.6.4.34 PowerTakeOffStatus

[SWS_Dem_01020] ⌈

Name	PowerTakeOffStatus		
Kind	ProvidedPort	***Interface***	PowerTakeOff
Description	--		
Variation	{ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT		

Table 8.286: Port PowerTakeOffStatus

⌋()

8.6.4.35 GetDataOfPID21

[SWS_Dem_01094] ⌈

Name	GetDataOfPID21		
Kind	ProvidedPort	***Interface***	GetDataOfPID21
Description	--		
Variation	({ecuc(Dem/DemGeneral/DemGeneralOBD. DemOBDCentralizedPID21Handling)} == true) && ({ecuc(Dem/ DemGeneral.DemOBDSupport)} == DEM_OBD_MASTER_ECU)		

Table 8.287: Port GetDataOfPID21

⌋()

8.6.4.36 SetDataOfPID21

[SWS_Dem_01017] ⌈

Name	SetDataOfPID21		
Kind	ProvidedPort	***Interface***	SetDataOfPID21
Description	--		
Variation	({ecuc(Dem/DemGeneral/DemGeneralOBD. DemOBDCentralizedPID21Handling)} == true) && ({ecuc(Dem/ DemGeneral.DemOBDSupport)} == DEM_OBD_PRIMARY_ECU)		

Table 8.288: Port SetDataOfPID21

⌋(*SRS_Diag_04142*)

8.6.4.37 SetDataOfPID31

[SWS_Dem_01016] ⌈

Name	SetDataOfPID31		
Kind	ProvidedPort	***Interface***	SetDataOfPID31
Description	--		
Variation	({ecuc(Dem/DemGeneral.DemOBDSupport)} != DEM_OBD_NO_OBD_SUPPORT) && ({ecuc(Dem/DemGeneral/ DemGeneralOBD.DemOBDCentralizedPID31Handling)} == true)		

Table 8.289: Port SetDataOfPID31

⌋()

8.6.4.38 StorageCond

[SWS_Dem_01015] ⌈

Name	StorageCond_{Name}		
Kind	ProvidedPort	***Interface***	StorageCondition
Description	--		
Port Defined Argument Value(s)	Type	uint8	
	Value	{ecuc(Dem/DemGeneral/ DemStorageCondition/ DemStorageConditionId.value)}	
Variation	Name = {ecuc(Dem/DemGeneral/DemStorageCondition. SHORT-NAME)}		

Table 8.290: Port StorageCond_{Name}

⌋()

9 Sequence Diagrams

9.1 ControlDTCSetting

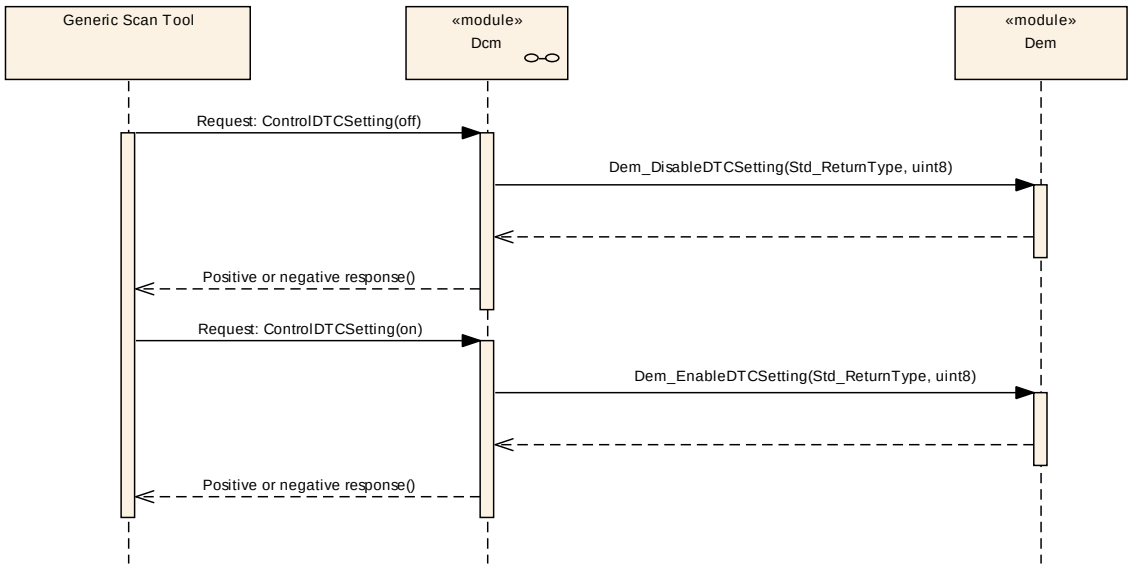


Figure 9.1: Sequence diagram of ControlDTCSetting

9.2 Dem_ClearDTC

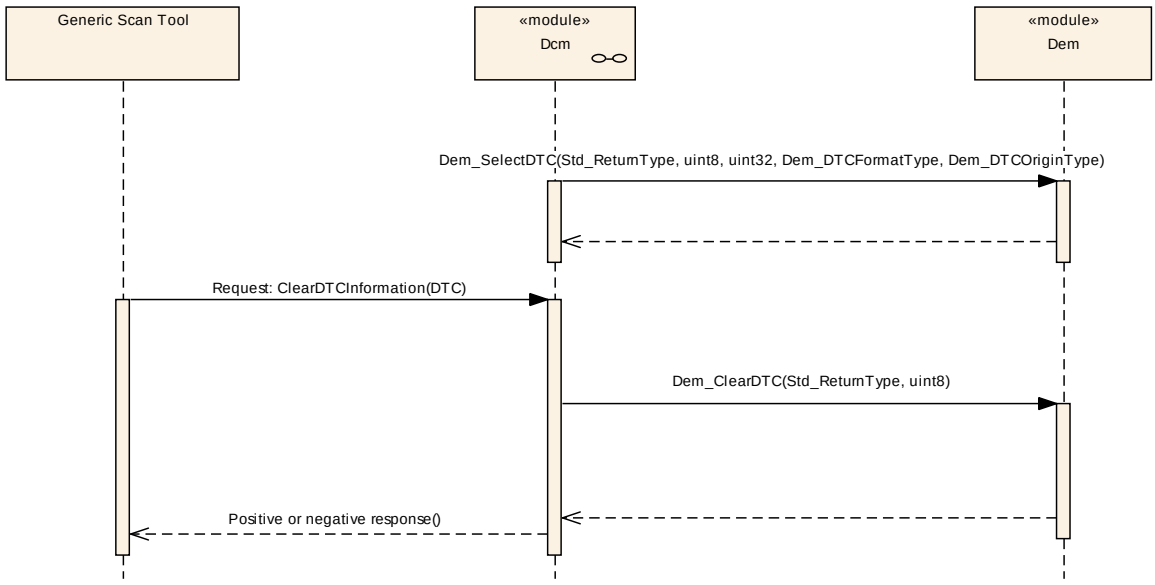


Figure 9.2: Sequence diagram of Dem_ClearDTC

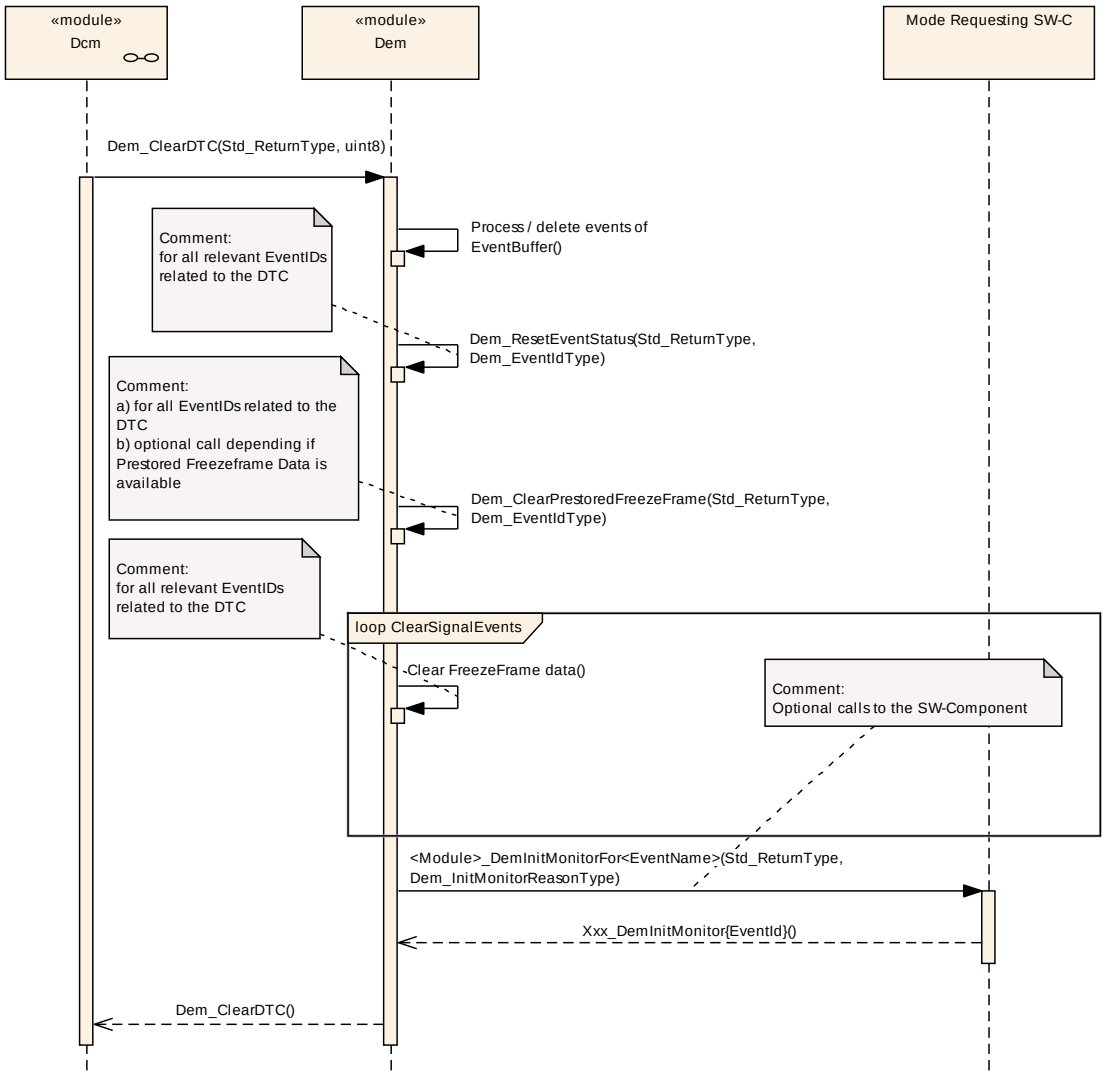


Figure 9.3: Sequence diagram of Dem_ClearDTC clearing a single DTC Dem-internally

9.3 Dem_GetDTCByOccurrenceTime

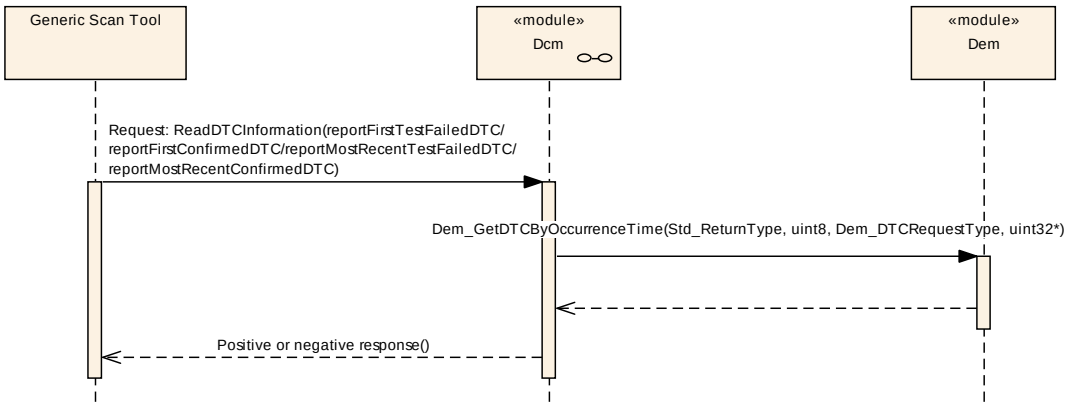


Figure 9.4: Sequence diagram of Dem_DcmGetDTCByOccurenceTime

9.4 Dem_GetNextExtendedDataRecord

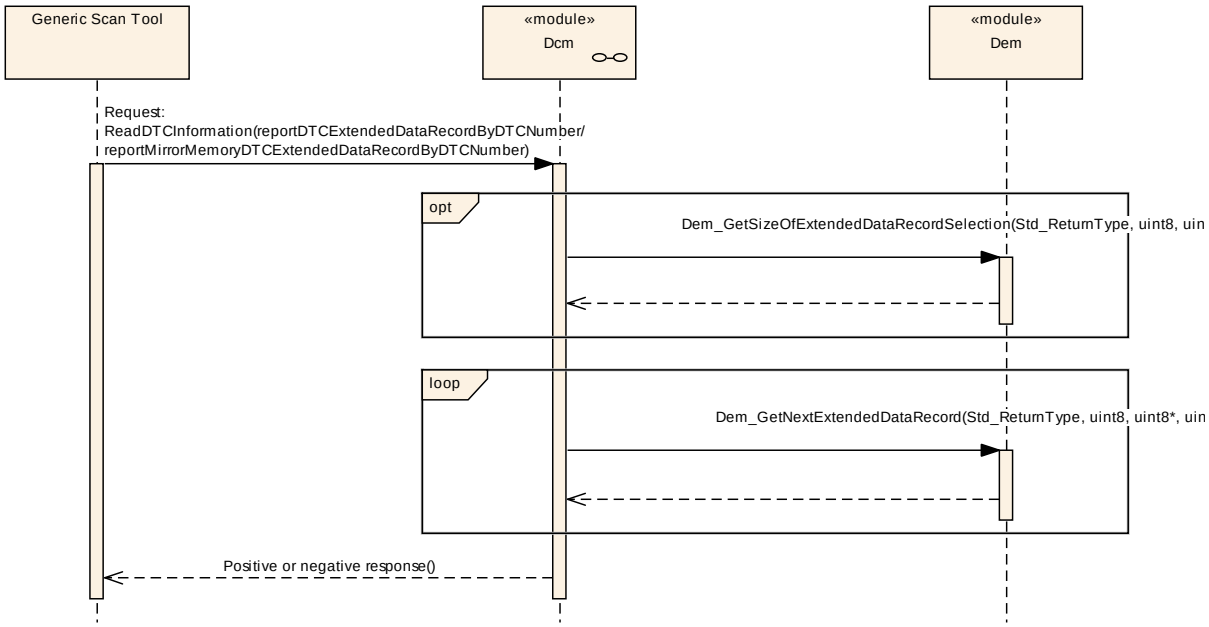


Figure 9.5: Sequence diagram of Dem_GetNextExtendedDataRecord

9.5 Dem_DcmGetStatusOfDTC

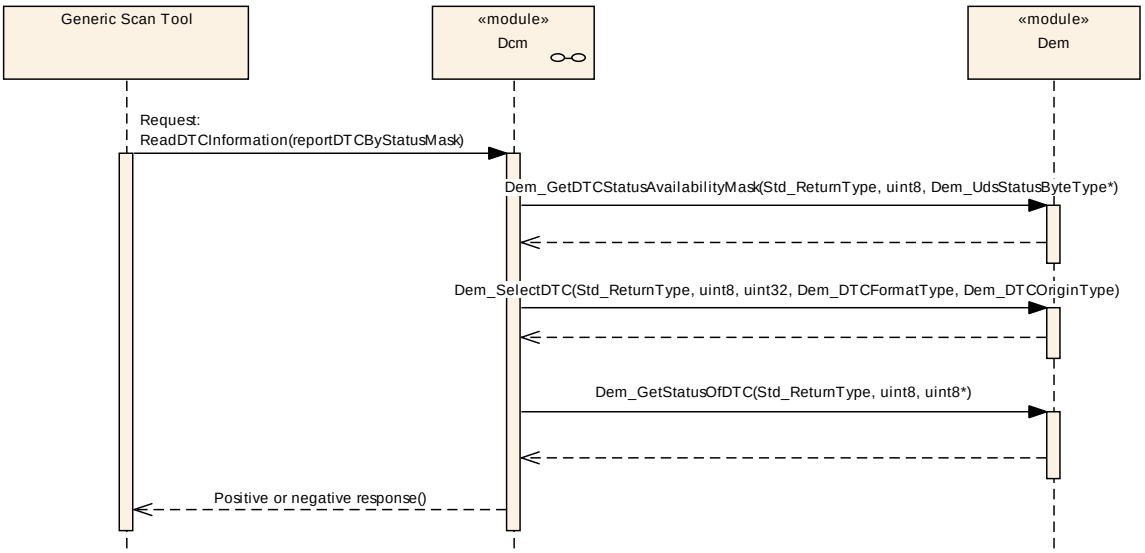


Figure 9.6: Sequence diagram of `Dem_GetStatusOfDTC`

9.6 Retrieving freeze frames

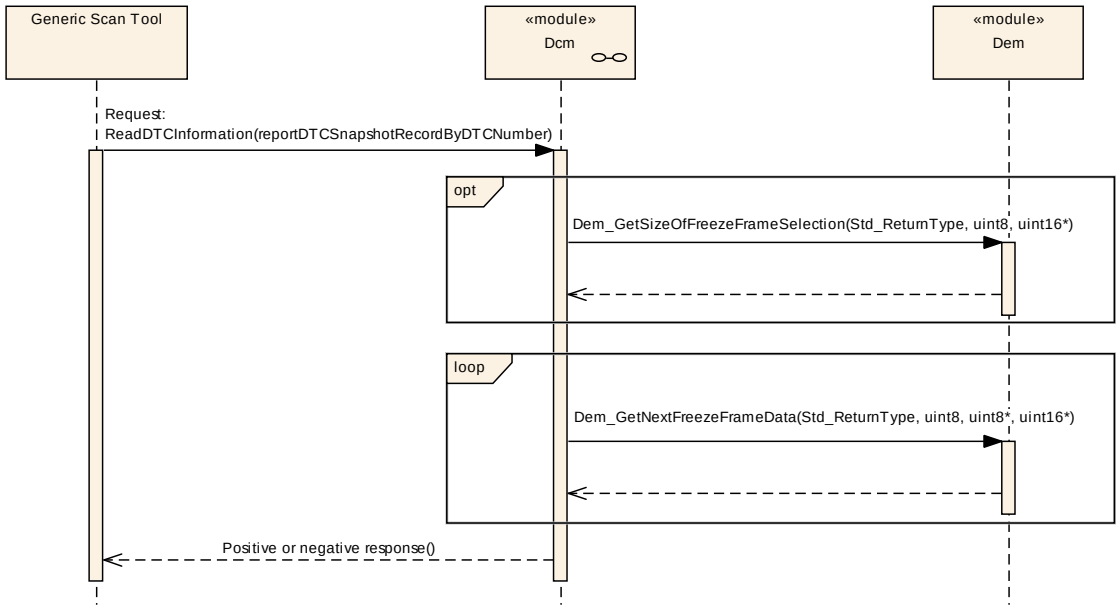


Figure 9.7: Sequence diagram to retrieve freeze frames

9.7 GetOBDFaultInformation

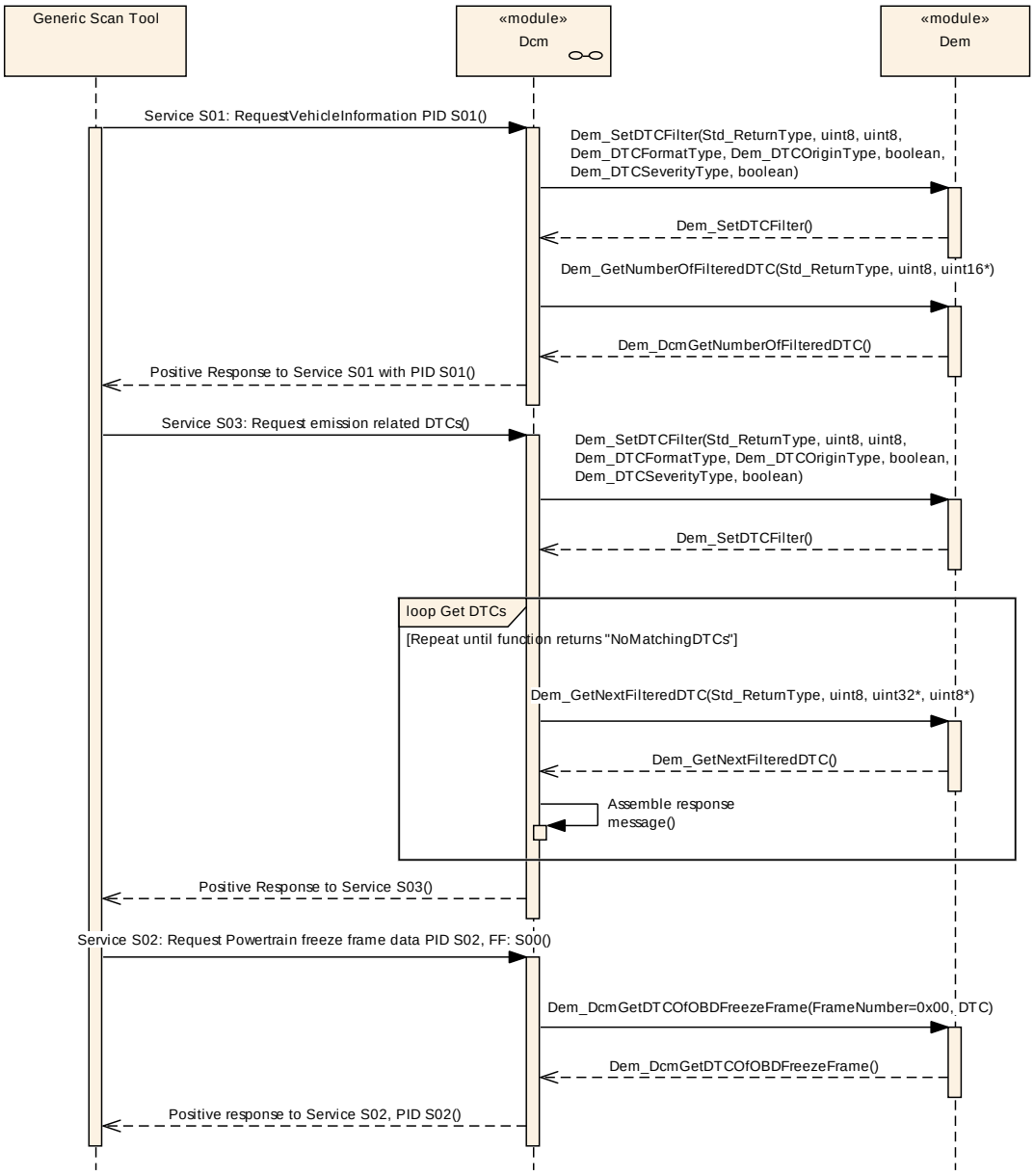


Figure 9.8: Sequence diagram of GetOBDFaultInformation

9.8 ReportDTCByStatusMask

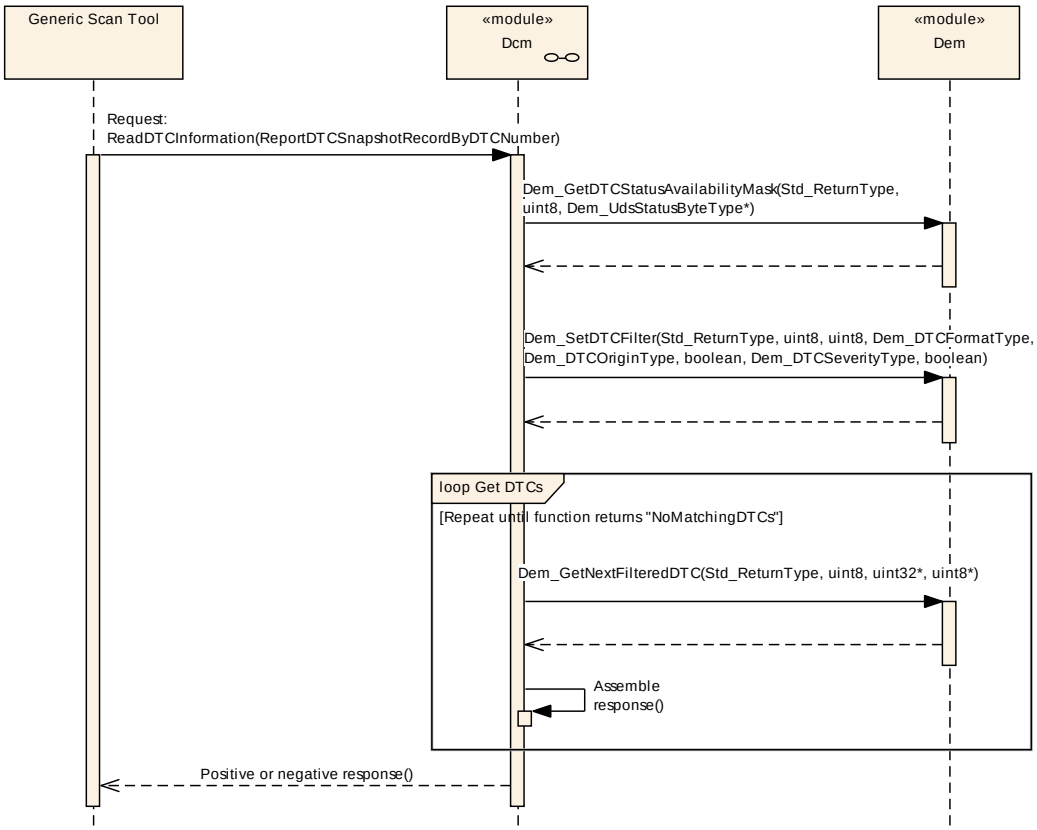


Figure 9.9: Sequence diagram of ReportDTCStatusMask

9.9 FiM_DemTriggerOnEventStatus

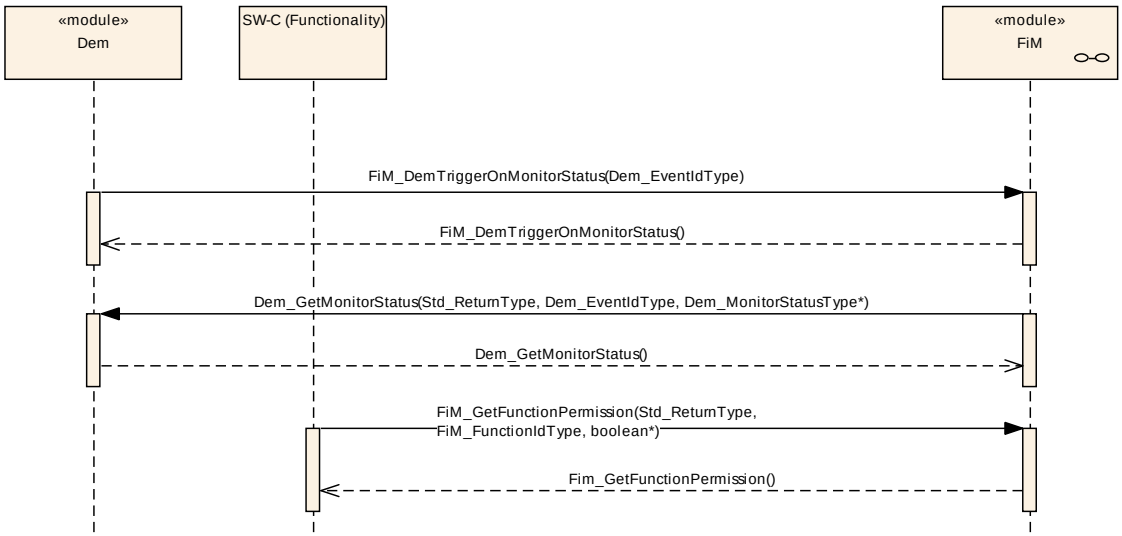


Figure 9.10: Sequence diagram of FiM_DemTriggerOnEventStatus

9.10 ProcessEvent (Example)

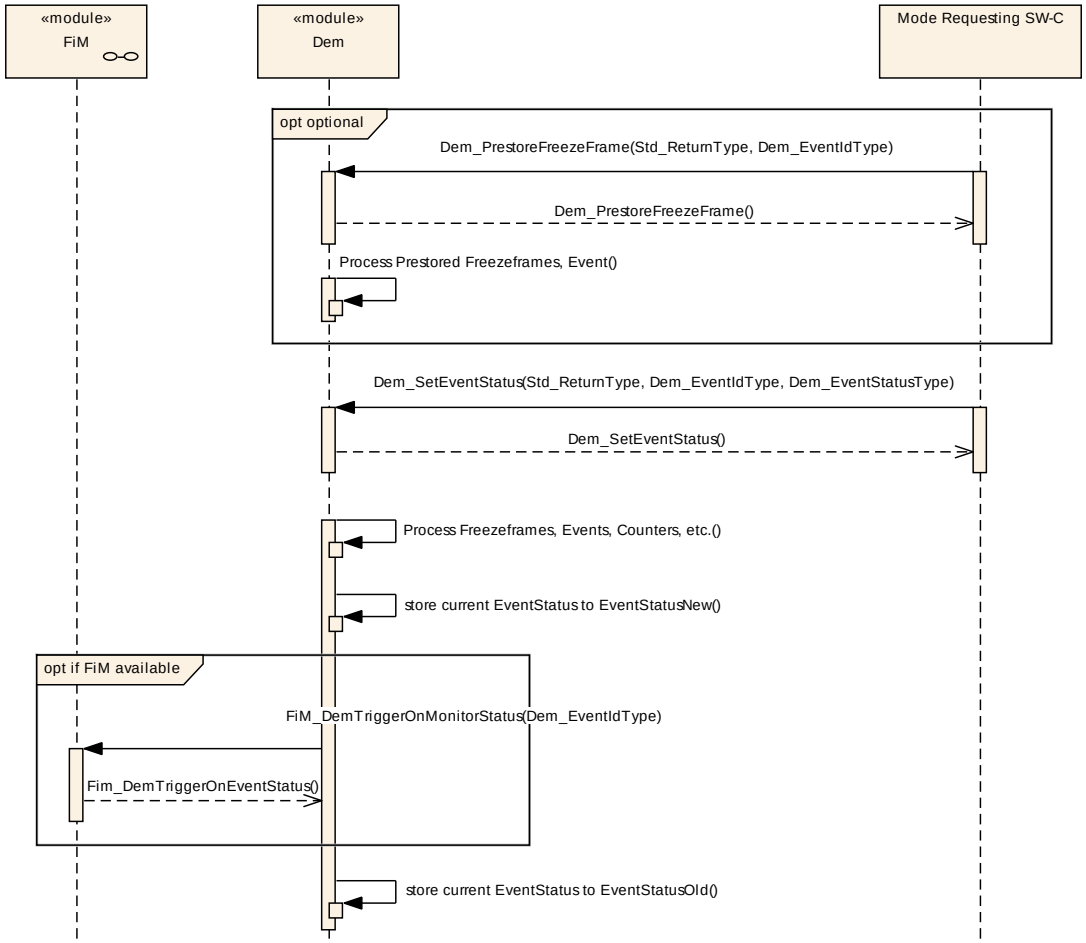


Figure 9.11: Sequence diagram for an example of ProcessEvent Dem-internally

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 Dem

Module SWS Item	ECUC_Dem_00928	
Module Name	Dem	
Module Description	Configuration of the Dem (Diagnostic Event Manager) module.	
Post-Build Variant Support	true	
Supported Config Variants	VARIANT-POST-BUILD, VARIANT-PRE-COMPILE	
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.
DemGeneral	1	This container contains the configuration (parameters) of the BSW Dem

10.2.2 General

10.2.2.1 DemGeneral

SWS Item	[ECUC_Dem_00677]

Container Name	DemGeneral
Description	This container contains the configuration (parameters) of the BSW Dem
Configuration Parameters	

Name	DemAgingRequiresNotFailedCycle [ECUC_Dem_00918]		
Parent Container	DemGeneral		
Description	Defines if the aging cycle counter is processed in operation cycles with test failed report or not. True: Aging cycle counter is processed only in operation cycles without test failed. False (Default): No effect on aging cycle counter processing.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemAgingRequiresTestedCycle [ECUC_Dem_00877]		
Parent Container	DemGeneral		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemAvailabilitySupport [ECUC_Dem_00878]		
Parent Container	DemGeneral		
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	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemBswErrorBufferSize [ECUC_Dem_00625]		
Parent Container	DemGeneral		
Description	Maximum number of elements in buffer for handling of BSW errors (ref. to SWS_Dem_00207).		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default Value			
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemClearDTCBehavior [ECUC_Dem_00766]	
Parent Container	DemGeneral	
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	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemClearDTCLimitation [ECUC_Dem_00790]		
Parent Container	DemGeneral		
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.	
	DEM_ONLY_CLEAR_ALL_DTCS	Dem_<...>ClearDTC accepts ClearAllDTCs only.	
Default Value	DEM_ALL_SUPPORTED_DTCS		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDataElementDefaultEndianness [ECUC_Dem_00858]		
Parent Container	DemGeneral		
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	
Post-Build Variant Value	false		

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemDebounceCounterBasedSupport [ECUC_Dem_00724]		
Parent Container	DemGeneral		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDebounceTimeBasedSupport [ECUC_Dem_00725]		
Parent Container	DemGeneral		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDevErrorDetect [ECUC_Dem_00648]		
Parent Container	DemGeneral		
Description	Switches the development error detection and notification on or off. • true: detection and notification is enabled. • false: detection and notification is disabled.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemEnvironmentDataCapture [ECUC_Dem_00895]		
Parent Container	DemGeneral		
Description	DemEnvironmentDataCapture defines the point in time, when the data actually is captured.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	DEM_CAPTURE_ASYNCHRONOUS_TO_REPORTING	The data capturing is postponed to the next cycle of the Dem_Mainfunction. (This means that there is a minimum delay between report of the failure and capturing the data).	
	DEM_CAPTURE_SYNCHRONOUS_TO_REPORTING	The data is captured immediately within the context of Dem_SetEventStatus.	
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemEventCombinationSupport [ECUC_Dem_00740]		
Parent Container	DemGeneral		
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	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemEventDisplacementStrategy [ECUC_Dem_00742]		
Parent Container	DemGeneral		
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).	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemEventMemoryEntryStorageTrigger [ECUC_Dem_00797]		
Parent Container	DemGeneral		
Description	Configures the primary trigger to allocate an event memory entry.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_TRIGGER_ON_CONFIRMED	Event Memory entries are triggered if the UDS status bit 3 (confirmedDTC) changes from 0 to 1.	
	DEM_TRIGGER_ON_FDC_THRESHOLD	Event Memory entries are triggered when the FDC threshold is reached.	
	DEM_TRIGGER_ON_TEST_FAILED	Event Memory entries are triggered if the UDS status bit 0 (testFailed) changes from 0 to 1.	
Default Value	DEM_TRIGGER_ON_TEST_FAILED		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemGeneralCallbackMonitorStatusChangedFnc [ECUC_Dem_00938]		
Parent Container	DemGeneral		
Description	Function name of prototype "<Module>_DemTriggerOnMonitorStatus". Function for the General Monitor Status Changed notification.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default Value			
Regular Expression			
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemGeneralInterfaceSupport [ECUC_Dem_00880]		
Parent Container	DemGeneral		
Description	The interfaces GeneralEvtInfo, GeneralCallbackEventDataChanged, GeneralCallbackMonitorStatusChanged and GeneralCallbackEventUdsStatusChange are provided if DemGeneralInterfaceSupport is equal to true.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemHeaderFileInclusion [ECUC_Dem_00722]		
Parent Container	DemGeneral		
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\._])*		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemMaxNumberEventEntryEventBuffer [ECUC_Dem_00899]	
Parent Container	DemGeneral	
Description	Specifies the size of the buffer for storing environmental data (freezeframes and extended data) until they are processed and stored to the event memory.	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	1 .. 250	
Default Value		
Post-Build Variant Multiplicity	false	

Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemMaxNumberPrestoredFF [ECUC_Dem_00692]		
Parent Container	DemGeneral		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOBDSupport [ECUC_Dem_00698]	
Parent Container	DemGeneral	
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
Post-Build Variant Value	false	

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOccurrenceCounterProcessing [ECUC_Dem_00767]		
Parent Container	DemGeneral		
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.	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOperationCycleStatusStorage [ECUC_Dem_00764]		
Parent Container	DemGeneral		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemPTOSupport [ECUC_Dem_00704]
Parent Container	DemGeneral
Description	This configuration switch defines, whether PTO support (and therefore PID $1E support) is enabled or not.
Multiplicity	1
Type	EcucBooleanParamDef
Default Value	
Post-Build Variant Value	false

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemResetConfirmedBitOnOverflow [ECUC_Dem_00799]		
Parent Container	DemGeneral		
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		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemStatusBitHandlingTestFailedSinceLastClear [ECUC_Dem_00784]		
Parent Container	DemGeneral		
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 Value	DEM_STATUS_BIT_NORMAL		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemStatusBitStorageTestFailed [ECUC_Dem_00714]		
Parent Container	DemGeneral		
Description	Activate/Deactivate the permanent storage of the "TestFailed" status bits. true: storage activated false: storage deactivated		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemSuppressionSupport [ECUC_Dem_00793]		
Parent Container	DemGeneral		
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_NO_SUPPRESSION	Suppression is not supported	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemTaskTime [ECUC_Dem_00715]		
Parent Container	DemGeneral		
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 .. INF[		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemTriggerDcmReports [ECUC_Dem_00754] (Obsolete)		
Parent Container	DemGeneral		
Description	Activate/Deactivate the notification to the Diagnostic Communication Manager for ROE processing. true: Dcm ROE notification activated false: Dcm ROE notification deactivated **Tags:** atp.Status=obsolete		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	

Scope / Dependency	scope: ECU

Name	DemTriggerDltReports [ECUC_Dem_00718]		
Parent Container	DemGeneral		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemTriggerFiMReports [ECUC_Dem_00719]		
Parent Container	DemGeneral		
Description	Activate/Deactivate the notification to the Function Inhibition Manager. true: FiM notification activated false: FiM notification deactivated		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemTriggerMonitorInitBeforeClearOk [ECUC_Dem_00765]		
Parent Container	DemGeneral		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	

Scope / Dependency	scope: ECU dependency: DemClearDTCBehavior

Name	DemTypeOfFreezeFrameRecordNumeration [ECUC_Dem_00778]		
Parent Container	DemGeneral		
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	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemVersionInfoApi [ECUC_Dem_00721]		
Parent Container	DemGeneral		
Description	Activate/Deactivate the version information API. true: version information activated false: version information deactivated		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemClearEventsWithoutDTCEventMemoryRef [ECUC_Dem_00941]
Parent Container	DemGeneral
Description	Indicating the event memory used as trigger to clear events without assigned DTCs.
Multiplicity	0..1
Type	Reference to DemPrimaryMemory
Post-Build Variant Multiplicity	false

Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemOBDEventMemorySetRef [ECUC_Dem_00940]		
Parent Container	DemGeneral		
Description	References the DemEventMemorySet used for OBD ECU.		
Multiplicity	0..1		
Type	Reference to DemEventMemorySet		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
DemClient	1..255	This container contains possible clients that are using the Dem APIs.
DemDataElementClass	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.
DemEventMemorySet	1..255	This container is a collection of referenced event memories and related information for a Dem client.
DemExtendedDataClass	0..*	This class contains the combinations of extended data records for an extended data class.

DemExtendedDataRecordClass	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 OBD2 and WWH-OBD relevant freeze frame class.
DemFreezeFrameRecNumClass	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
DemFreezeFrameRecordClass	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.
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	This container holds all parameters that are relevant to configure an operation cycle.
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.
DemStorageConditionGroup	0..255	This container contains the configuration (parameters) for storage condition groups.

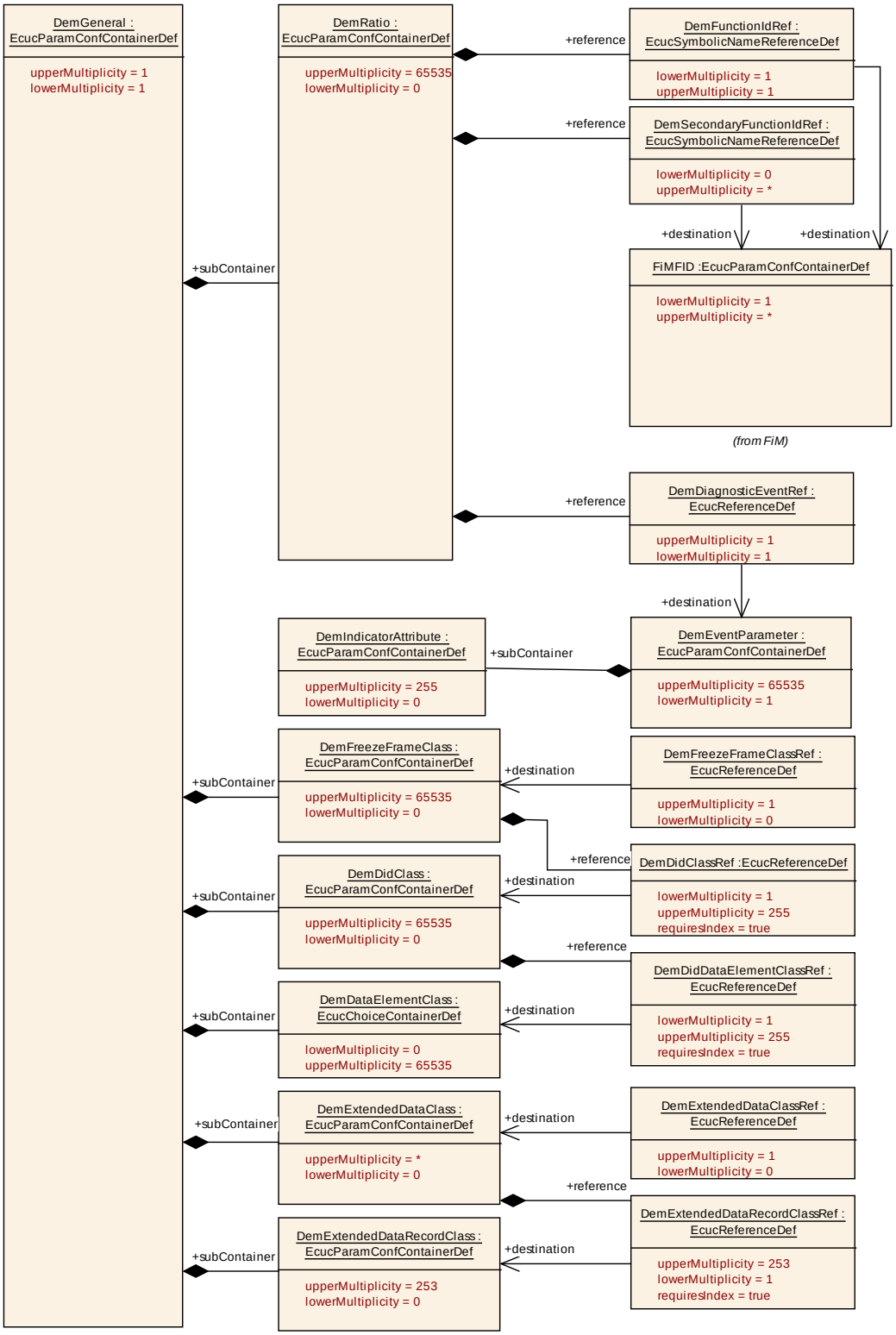


Figure 10.1: Configuration overview for DemGeneral (part I)

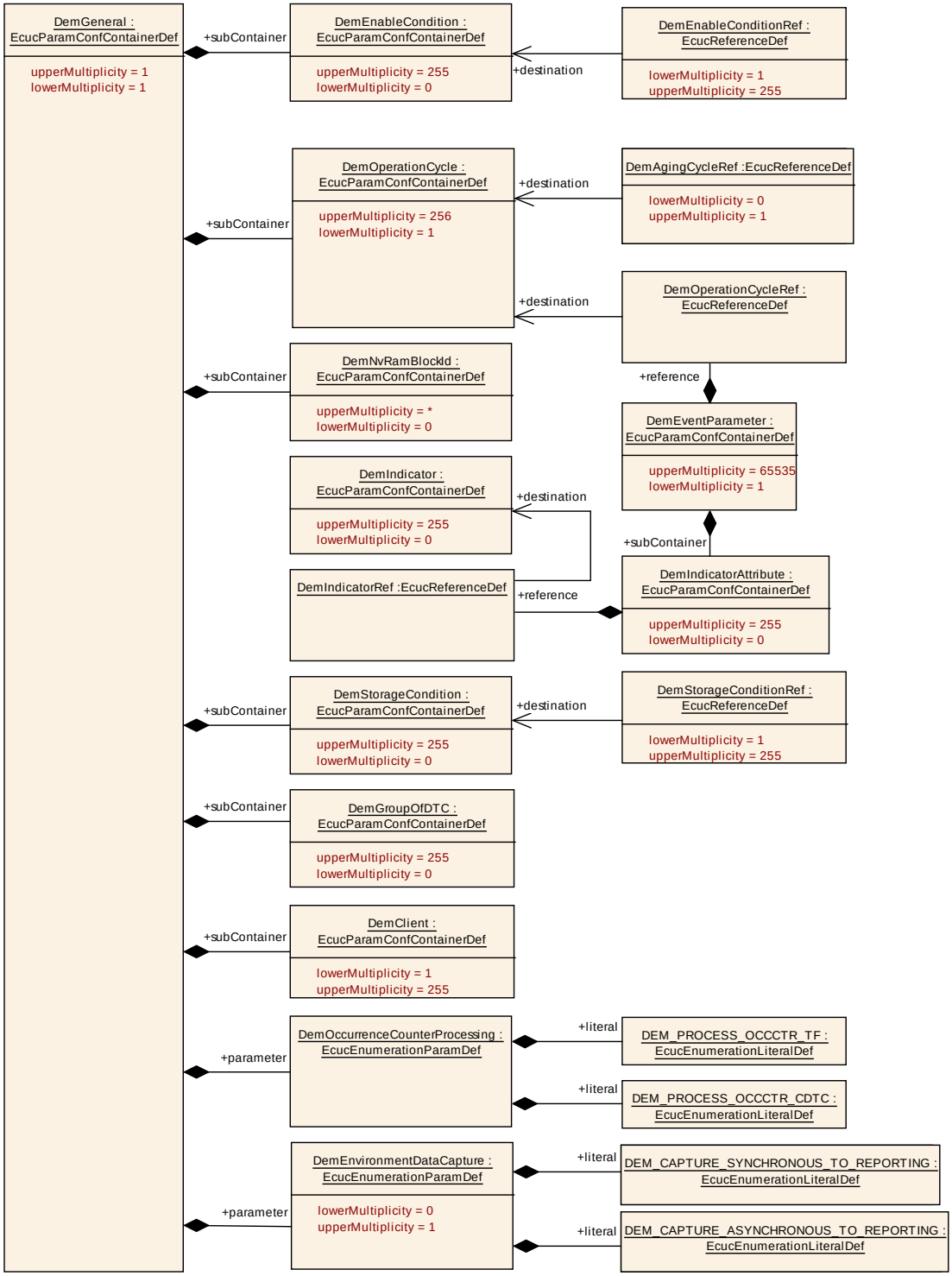


Figure 10.2: Configuration overview for DemGeneral (part II)

10.2.2.2 DemConfigSet

SWS Item	[ECUC_Dem_00634]
Container Name	DemConfigSet

Description	This container contains the configuration parameters and sub containers of the Dem module supporting multiple configuration sets.
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
DemComponent	0..*	This container configures the monitored components and system dependencies.
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 Time Based Class.
DemDtrs	0..1	This container holds the configuration of DTRs collection.
DemEventParameter	1..65535	This container contains the configuration (parameters) for events.
DemMultiEventTriggering	0..65535	Configures an event that will trigger other events whenever the event is reported.
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.

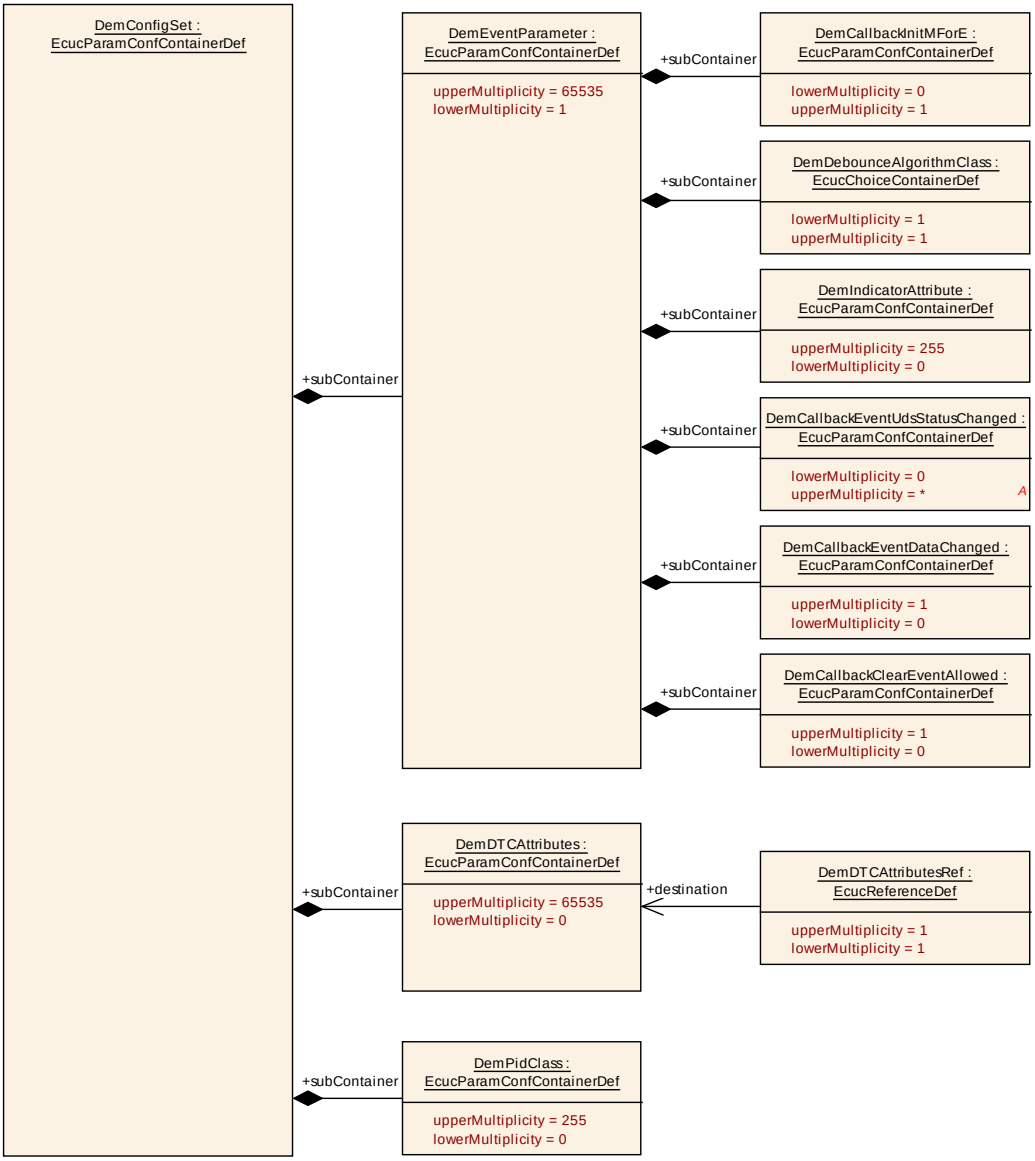


Figure 10.3: Configuration overview for DemConfigSet

10.2.2.3 DemClient

SWS Item	[ECUC_Dem_00931]		
Container Name	DemClient		
Description	This container contains possible clients that are using the Dem APIs.		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

Name	DemClientFunctionality [ECUC_Dem_00943]		
Parent Container	DemClient		
Description	Functionality provided by Dem for the DemClient.		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	DEM_CLIENT_ONLY_USES_EVENTOVERFLOW_INTERFACE	The Dem provides only the client related functionality EvMemOverflowIndication for this client Id. Other functionality like DTCFilter or Clear is not allowed to be used by this client.	
	DEM_CLIENT_USES_FULL_FUNCTIONALITY	The Dem provides all client related functionality for the selected client (e.g. clear APIs and overflow indications).	
Default Value	DEM_CLIENT_USES_FULL_FUNCTIONALITY		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name	DemClientId [ECUC_Dem_00932]		
Parent Container	DemClient		
Description	Defines a unique identifier for a Dem client. This number is used by this client in the ClientID parameter in all API with this parameter.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 255		
Default Value			
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemClientUsesRte [ECUC_Dem_00933]		
Parent Container	DemClient		
Description	If set to true, this client shall only use the DEM via RTE (Dem will provide the C/S interfaces: CddIf, EvMemOverflowIndication). The client must not do any C_API calls to the DEM.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemEventMemorySetRef [ECUC_Dem_00942]		
Parent Container	DemClient		
Description	References to the client assigned event memory container that contains client specific settings and event memories.		
Multiplicity	1		
Type	Reference to DemEventMemorySet		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
DemCallbackDTCStatusChanged	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.

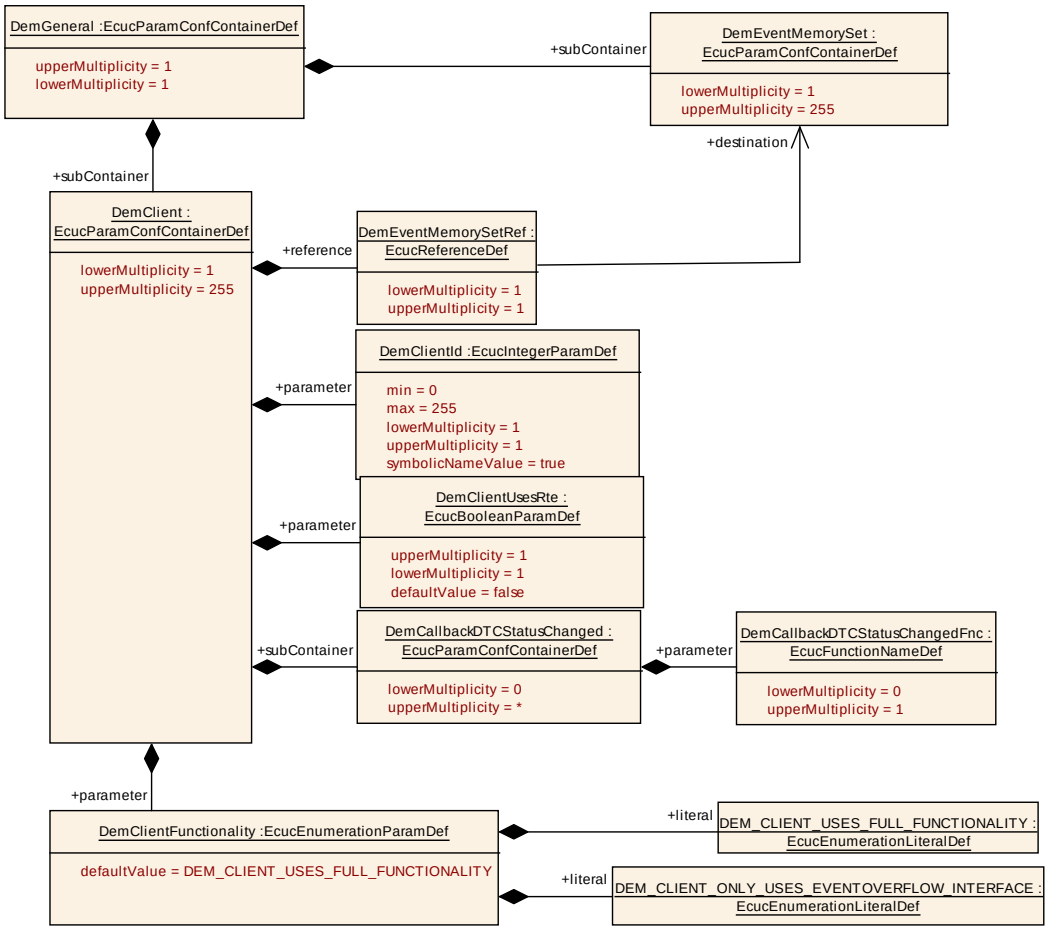


Figure 10.4: DemClient

10.2.2.4 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]		
Parent Container	DemDTCAttributes		
Description	Switch to allow aging/unlearning of the event or not. true: aging allowed false: aging not allowed		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemAgingCycleCounterThreshold [ECUC_Dem_00623]		
Parent Container	DemDTCAttributes		
Description	Number of aging cycles needed to unlearn/delete the event.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default Value			
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemAgingCycleCounterThresholdForTFSLC [ECUC_Dem_00897]		
Parent Container	DemDTCAttributes		
Description	Number of aging cycles needed to reset the testFailedSinceLastClear Bit.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default Value			
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDTCPriority [ECUC_Dem_00662]		
Parent Container	DemDTCAttributes		
Description	Priority of the event/dtc, in view of full event memory. A lower value means higher priority.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDTCSignificance [ECUC_Dem_00779]	
Parent Container	DemDTCAttributes	
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
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemMaxNumberFreezeFrameRecords [ECUC_Dem_00605]		
Parent Container	DemDTCAttributes		
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			
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemAgingCycleRef [ECUC_Dem_00624]
Parent Container	DemDTCAttributes
Description	Reference to the cycle which is triggering the aging of the event.
Multiplicity	0..1
Type	Reference to DemOperationCycle
Post-Build Variant Multiplicity	false

Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemExtendedDataClassRef [ECUC_Dem_00667]		
Parent Container	DemDTCAttributes		
Description	This reference defines the link to an extended data class sampler.		
Multiplicity	0..1		
Type	Reference to DemExtendedDataClass		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name	DemFreezeFrameClassRef [ECUC_Dem_00674]		
Parent Container	DemDTCAttributes		
Description	These references define the links to a freeze frame class sampler.		
Multiplicity	0..1		
Type	Reference to DemFreezeFrameClass		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name	DemFreezeFrameRecNumClassRef [ECUC_Dem_00776]		
Parent Container	DemDTCAttributes		
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		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemJ1939ExpandedFreezeFrameClassRef [ECUC_Dem_00834]		
Parent Container	DemDTCAttributes		
Description	These references define the links to a J1939 freeze frame class sampler.		
Multiplicity	0..1		
Type	Reference to DemJ1939FreezeFrameClass		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name	DemJ1939FreezeFrameClassRef [ECUC_Dem_00835]		
Parent Container	DemDTCAttributes		
Description	These references define the links to a J1939 freeze frame class sampler.		
Multiplicity	0..1		
Type	Reference to DemJ1939FreezeFrameClass		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name	DemMemoryDestinationRef [ECUC_Dem_00890]		
Parent Container	DemDTCAttributes		
Description	The memory destination assigns DTCs to one or two memory destinations. If more than one memory destination is assigned to a specific DTC, the DTC can be present in the corresponding event memories. In this case one of the references has to be DemMirrorMemory (SWS_Dem_CONSTR_6104).		
Multiplicity	1..2		
Type	Choice reference to [DemMirrorMemory,DemPrimaryMemory,DemUserDefinedMemory]		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemWWHOBDFreezeFrameClassRef [ECUC_Dem_00911]		
Parent Container	DemDTCAttributes		
Description	This reference defines the link to a WWH-OBD freeze frame class sampler.		
Multiplicity	0..1		
Type	Reference to DemFreezeFrameClass		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

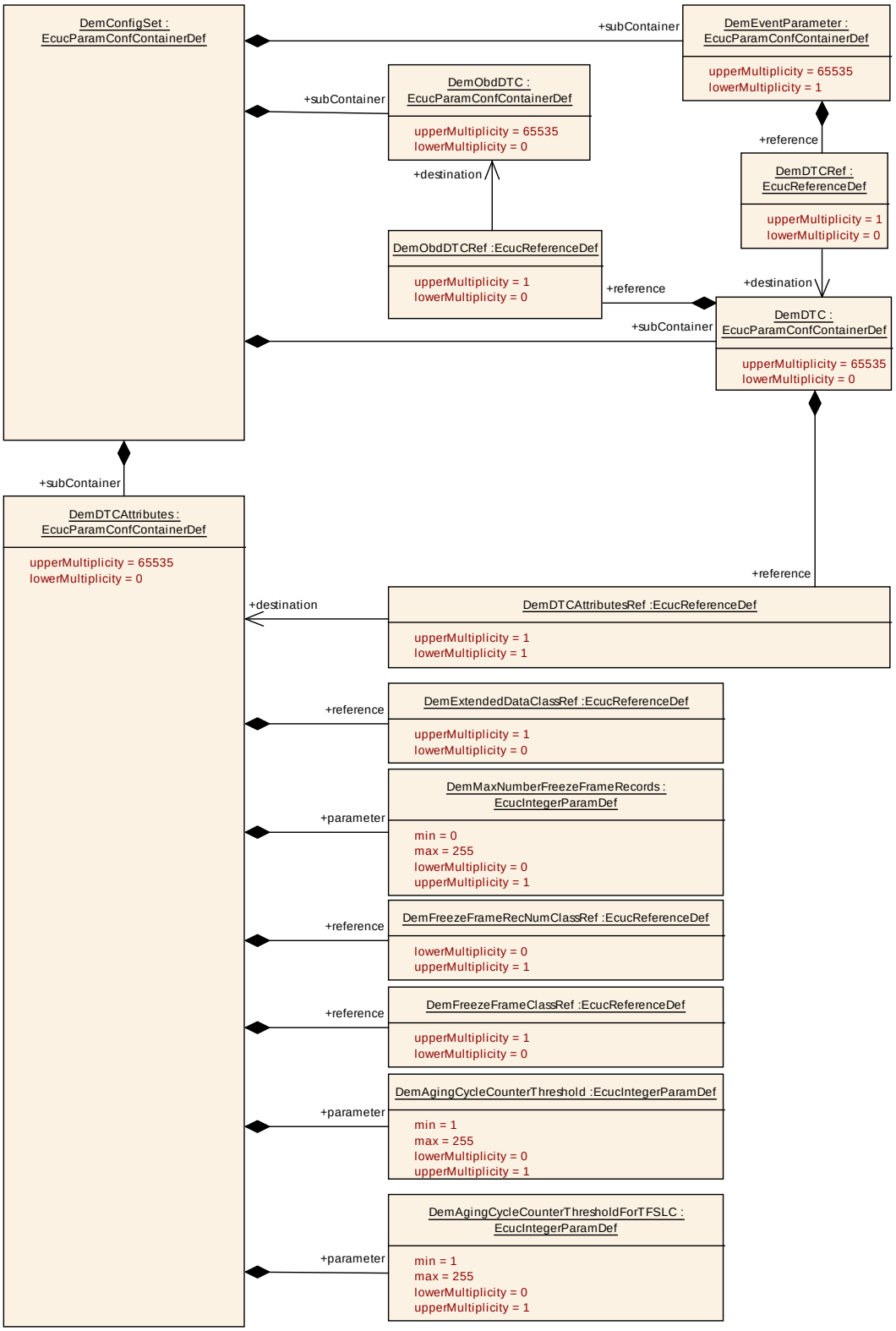


Figure 10.5: DTC attribute configuration (part I)

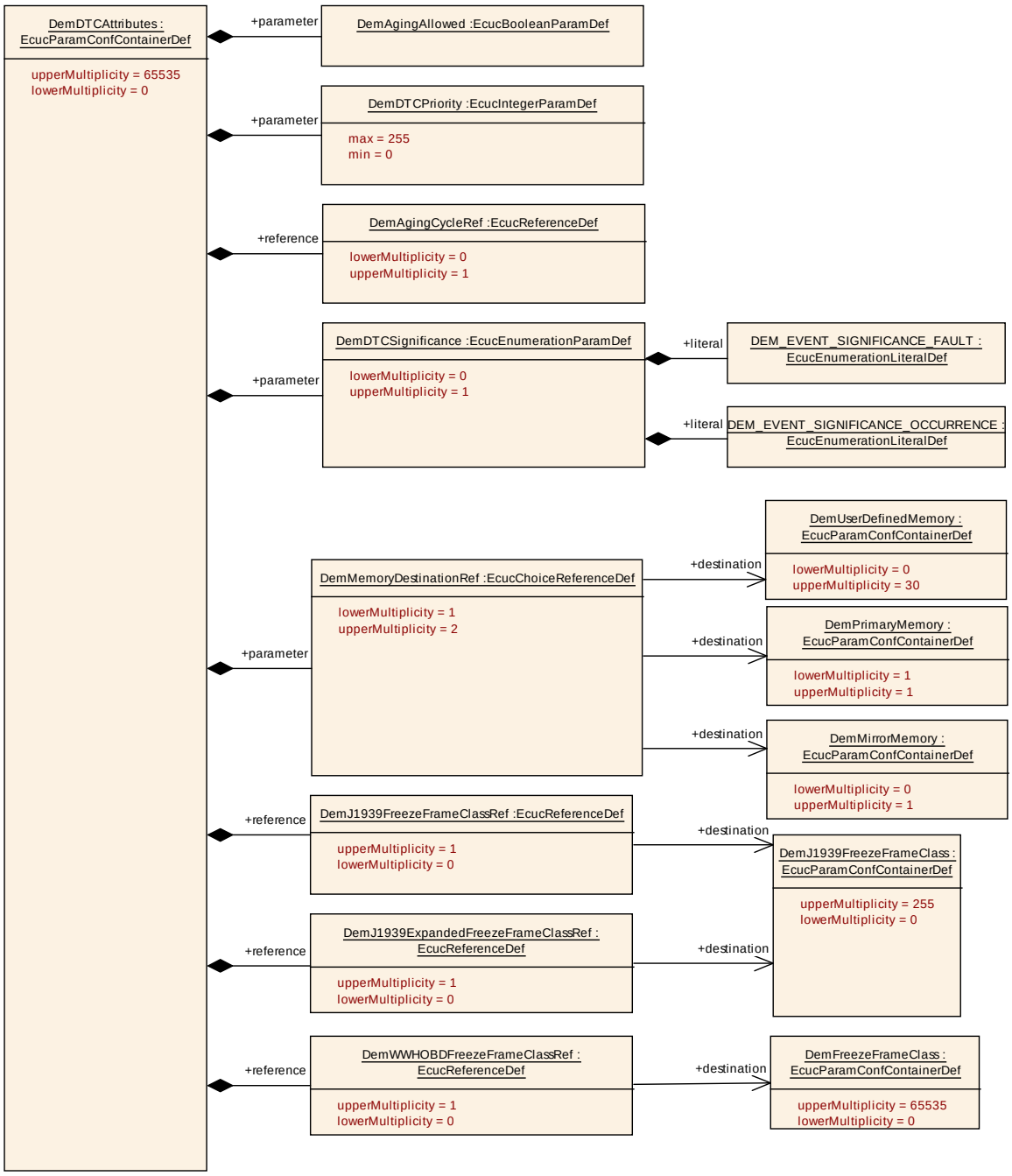


Figure 10.6: DTC attribute configuration (part II)

10.2.2.5 DemEventParameter

SWS Item	[ECUC_Dem_00661]
Container Name	DemEventParameter
Description	This container contains the configuration (parameters) for events.
Configuration Parameters	

Name	DemCausalityDelayTime [ECUC_Dem_00921]		
Parent Container	DemEventParameter		
Description	Time to wait until the event is considered as causal. The parameter is specified in seconds.		
Multiplicity	0..1		
Type	EcucFloatParamDef		
Range	[0 .. 2.5]		
Default Value	0		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemComponentPriority [ECUC_Dem_00909]		
Parent Container	DemEventParameter		
Description	Specifies the priority within the component. A lower value means higher priority.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default Value			
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemEventAvailable [ECUC_Dem_00792]		
Parent Container	DemEventParameter		
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			
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

Name	DemEventConfirmationThreshold [ECUC_Dem_00924]		
Parent Container	DemEventParameter		
Description	Defines the operation cycle threshold of the DTC confirmation status according "Confirmation Threshold" of ISO 14229-1.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default Value	1		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

Name	DemEventFailureCycleCounterThresholdAdaptable [ECUC_Dem_00929]		
Parent Container	DemEventParameter		
Description	Indicates whether the events confirmation Cycle threshold can be adapted by Dem_SetEventFailureCycleCounterThreshold.		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default Value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemEventId [ECUC_Dem_00659]		
Parent Container	DemEventParameter		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemEventKind [ECUC_Dem_00660]		
Parent Container	DemEventParameter		
Description	This parameter is used to distinguish between SW-C and BSW events.		
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	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name	DemEventRecoverableInSameOperationCycle [ECUC_Dem_00916]		
Parent Container	DemEventParameter		
Description	If parameter is configured to FALSE, reporting of PASSED will be ignored if the event is already "testfailed this operation cycle".		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default Value	true		
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemFFPrestorageSupported [ECUC_Dem_00671]		
Parent Container	DemEventParameter		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemReportBehavior [ECUC_Dem_00894]		
Parent Container	DemEventParameter		
Description	Indicates the reporting behavior of the BSW Module (DemEventKind == DEM_EVENT_KIND_BSW) in order to determine the size of the reporting queue. If the parameter is not defined it means REPORT_BEFORE_INIT.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	REPORT_AFTER_INIT	Indicates that the Event will not be reported before Dem_Init().	
	REPORT_BEFORE_INIT	Indicates that the Event may be reported before Dem_Init().	
Default Value	REPORT_BEFORE_INIT		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemComponentClassRef [ECUC_Dem_00908]		
Parent Container	DemEventParameter		
Description	Reference to the monitored component.		
Multiplicity	0..1		
Type	Reference to DemComponent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemDTCRef [ECUC_Dem_00888]		
Parent Container	DemEventParameter		
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		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name	DemEnableConditionGroupRef [ECUC_Dem_00746]		
Parent Container	DemEventParameter		
Description	References an enable condition group.		
Multiplicity	0..1		
Type	Reference to DemEnableConditionGroup		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOBDGroupingAssociativeEventsRef [ECUC_Dem_00839]		
Parent Container	DemEventParameter		
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		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOperationCycleRef [ECUC_Dem_00702]
Parent Container	DemEventParameter
Description	Kind of operation cycle for the event (e.g. power cycle, driving cycle, ...)
Multiplicity	1
Type	Reference to DemOperationCycle
Post-Build Variant Value	false

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemStorageConditionGroupRef [ECUC_Dem_00769]		
Parent Container	DemEventParameter		
Description	References a storage condition group.		
Multiplicity	0..1		
Type	Reference to DemStorageConditionGroup		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
DemCallbackClearEventAllowed	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.
DemCallbackEventDataChanged	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.

DemCallbackEventUdsStatusChanged	0..*	The presence of this container indicates, that the Dem has access to an "EventUdsStatusChanged" callback, which the Dem will call to notify other components about the change in the status of an event. In case there is a DemCallbackEventUdsStatusChangedFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackEventUdsStatusChangedFnc, the Dem will have an R-Port requiring the interface CallbackEventUdsStatusChanged, 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.
DemCallbackMonitorStatusChanged	0..*	The presence of this container indicates, that the Dem has access to an "MonitorStatusChanged" callback, which the Dem will call to notify other components about the change in the status of an event. In case there is a DemCallbackMonitorStatusChangedFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackMonitorStatusChangedFnc, the Dem will have an R-Port requiring the interface CallbackMonitorStatusChanged, whose name is generated by using the unique callback-prefix followed by the event name.
DemDebounceAlgorithmClass	1	Debounce algorithm class: counter based, time based, or monitor internal.
DemIndicatorAttribute	0..255	This container contains the event specific configuration of Indicators.

10.2.2.6 DemMultiEventTriggering

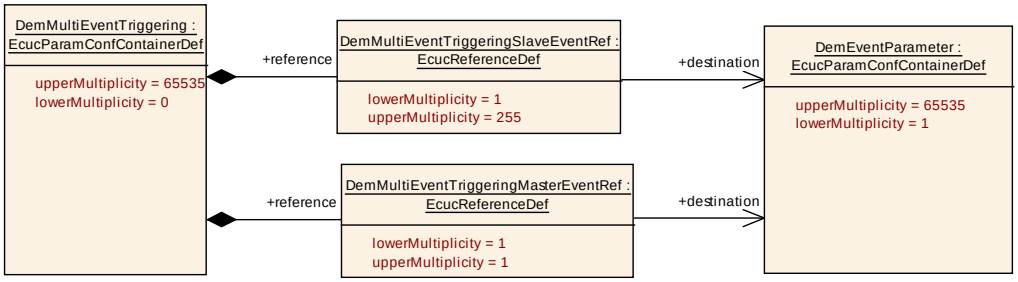


Figure 10.7: DemMultiEventTriggering

SWS Item	[ECUC_Dem_00944]
Container Name	DemMultiEventTriggering
Description	Configures an event that will trigger other events whenever the event is reported.
Configuration Parameters	

Name	DemMultiEventTriggeringMasterEventRef [ECUC_Dem_00945]		
Parent Container	DemMultiEventTriggering		
Description	Reference to the event that will trigger other events upon reception of this event.		
Multiplicity	1		
Type	Reference to DemEventParameter		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemMultiEventTriggeringSlaveEventRef [ECUC_Dem_00946]		
Parent Container	DemMultiEventTriggering		
Description	Reference to the event that is triggered upon triggering the master event.		
Multiplicity	1..255		
Type	Reference to DemEventParameter		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.2.7 DemComponent

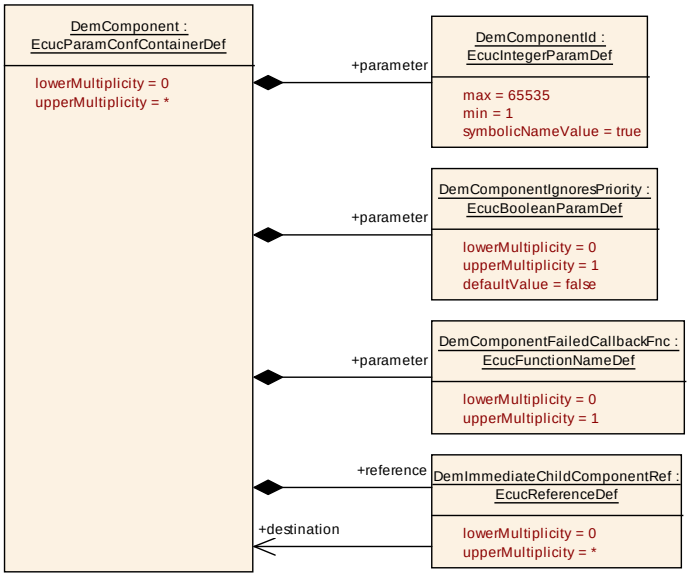


Figure 10.8: DemComponent

SWS Item	[ECUC_Dem_00904]
Container Name	DemComponent
Description	This container configures the monitored components and system dependencies.
Configuration Parameters	

Name	DemComponentFailedCallbackFnc [ECUC_Dem_00906]		
Parent Container	DemComponent		
Description	Specifies the function to be called on component failed status changes.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default Value			
Regular Expression			
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemComponentId [ECUC_Dem_00930]		
Parent Container	DemComponent		
Description	Unique identifier of a DemComponent. Component Id should be configured in sequential order beginning with 1 and no gaps in between. 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		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	1 .. 65535		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemComponentIgnoresPriority [ECUC_Dem_00905]		
Parent Container	DemComponent		
Description	This configuration switch defines, whether the priority of events at this component shall be ignored.		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default Value	false		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemImmediateChildComponentRef [ECUC_Dem_00907]		
Parent Container	DemComponent		
Description	Reference to all immediate children of the current component.		
Multiplicity	0..*		
Type	Reference to DemComponent		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

10.2.2.8 DemDTC

SWS Item	[ECUC_Dem_00886]
Container Name	DemDTC
Description	This container contains the configuration (parameters) for DemUdsDTC.
Configuration Parameters	

Name	DemDTCFunctionalUnit [ECUC_Dem_00643]	
Parent Container	DemDTC	
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		
Post-Build Variant Multiplicity	false	
Post-Build Variant Value	false	

Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDTCSeverity [ECUC_Dem_00645]		
Parent Container	DemDTC		
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	
Default Value	DEM_SEVERITY_NO_SEVERITY		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDtcValue [ECUC_Dem_00887]		
Parent Container	DemDTC		
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			
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

Name	DemNvStorageStrategy [ECUC_Dem_00127]		
Parent Container	DemDTC		
Description	This parameter defines when a specific event memory entry is allowed to be stored in the NVRAM.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	DURING_SHUTDOWN	write during shutdown	
	IMMEDIATE_AT_FIRST_OCCURRENCE	write at first fail occurrence and during shutdown	
Default Value	DURING_SHUTDOWN		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemWWHOBDDTCClass [ECUC_Dem_00912]		
Parent Container	DemDTC		
Description	DTC Class according to ISO 14229-1 [2013 version]. This parameter depends on the automotive manufacturer. If it is not configured, the value is marked as 'unclassified'. If this parameter is configured for no DTC, the Dem provides no DTC WWHOBD class information.		
Multiplicity	0..1		
Type	EcucEnumerationParamDef		
Range	DEM_DTC_WWHOBD_CLASS_A	Class A	
	DEM_DTC_WWHOBD_CLASS_B1	Class B1	
	DEM_DTC_WWHOBD_CLASS_B2	Class B2	
	DEM_DTC_WWHOBD_CLASS_C	Class C	
	DEM_DTC_WWHOBD_CLASS_NOCLASS	No Class information	
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDTCAttributesRef [ECUC_Dem_00642]		
Parent Container	DemDTC		
Description	This parameter defines the DTC Attributes associated with the DemDTC.		
Multiplicity	1		
Type	Reference to DemDTCAttributes		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name	DemObdDTCRef [ECUC_Dem_00889]		
Parent Container	DemDTC		
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		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.2.9 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]	
Parent Container	DemGroupOfDTC	
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 : 0x000000 to 0x0000ff and 0xffff00 to 0xffffff.	
Multiplicity	1	
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)	
Range	256 .. 16776959	
Default Value		
Post-Build Variant Value	false	

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.2.10 DemOperationCycle

SWS Item	[ECUC_Dem_00701]
Container Name	DemOperationCycle
Description	This container holds all parameters that are relevant to configure an operation cycle.
Configuration Parameters	

Name	DemOperationCycleAutostart [ECUC_Dem_00805]		
Parent Container	DemOperationCycle		
Description	The autostart property defines if the operation cycles is automatically (re-)started during Dem_PreInit.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOperationCycleId [ECUC_Dem_00898]	
Parent Container	DemOperationCycle	
Description	This parameter's value is used, together with the aggregating container, to define a symbolic name of the operation cycle.	
Multiplicity	1	
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)	
Range	0 .. 255	
Default Value		
Post-Build Variant Multiplicity	false	
Post-Build Variant Value	false	

Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemLeadingCycleRef [ECUC_Dem_00919]		
Parent Container	DemOperationCycle		
Description	Defines the operation cycle, which is relevant for processing this operation cycle.		
Multiplicity	0..1		
Type	Symbolic name reference to DemOperationCycle		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.2.11 DemIndicator

SWS Item	[ECUC_Dem_00680]
Container Name	DemIndicator
Description	This container contains the configuration (parameters) for Indicators.
Configuration Parameters	

Name	DemIndicatorID [ECUC_Dem_00683]	
Parent Container	DemIndicator	
Description	Unique identifier of an indicator.	
Multiplicity	1	
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)	
Range	0 .. 255	
Default Value		
Post-Build Variant Value	false	

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.2.12 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]		
Parent Container	DemIndicatorAttribute		
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'	
Post-Build Variant Value	true		
Value 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]		
Parent Container	DemIndicatorAttribute		
Description	Defines the number of failure cycles for the WarningIndicatorOnCriteria.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default Value			
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemIndicatorHealingCycleCounterThreshold [ECUC_Dem_00748]		
Parent Container	DemIndicatorAttribute		
Description	Defines the number of healing cycles for the WarningIndicatorOffCriteria.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemIndicatorRef [ECUC_Dem_00687]		
Parent Container	DemIndicatorAttribute		
Description	Reference to the used indicator.		
Multiplicity	1		
Type	Reference to DemIndicator		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	

Scope / Dependency	scope: ECU

No Included Containers

10.2.2.13 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]		
Parent Container	DemNvRamBlockId		
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		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.3 OBD

10.2.3.1 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]		
Parent Container	DemGeneralOBD		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOBDCentralizedPID31Handling [ECUC_Dem_00879]		
Parent Container	DemGeneralOBD		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOBDCompliancy [ECUC_Dem_00795]	
Parent Container	DemGeneralOBD	
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" DM05 message of J1939-73	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	0 .. 255	
Default Value		
Post-Build Variant Value	true	

Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

Name	DemOBDDelayedDCYConfirmedAndMIL [ECUC_Dem_00917]		
Parent Container	DemGeneralOBD		
Description	Controls the delayed calculation of the confirmed status for the OBD driving cycle.		
Multiplicity	0..1		
Type	EcucBooleanParamDef		
Default Value	false		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemOBDEngineType [ECUC_Dem_00900]	
Parent Container	DemGeneralOBD	
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
Post-Build Variant Multiplicity	false	
Post-Build Variant Value	false	

Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOBDEventDisplacement [ECUC_Dem_00796]		
Parent Container	DemGeneralOBD		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemOBDDrivingCycleRef [ECUC_Dem_00923]		
Parent Container	DemGeneralOBD		
Description	Defines the operationCycle which denotes the OBD driving cycle.		
Multiplicity	1		
Type	Reference to DemOperationCycle		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name	DemOBDInputAcceleratorPedalInformation [ECUC_Dem_00763]		
Parent Container	DemGeneralOBD		
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]		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOBDInputAmbientPressure [ECUC_Dem_00762]		
Parent Container	DemGeneralOBD		
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]		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOBDInputAmbientTemperature [ECUC_Dem_00761]		
Parent Container	DemGeneralOBD		
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]		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOBDInputDistanceInformation [ECUC_Dem_00759]		
Parent Container	DemGeneralOBD		
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]		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOBDInputEngineSpeed [ECUC_Dem_00757]		
Parent Container	DemGeneralOBD		
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]		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOBDInputEngineTemperature [ECUC_Dem_00772]		
Parent Container	DemGeneralOBD		
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]		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOBDInputProgrammingEvent [ECUC_Dem_00760]		
Parent Container	DemGeneralOBD		
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]		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOBDInputVehicleSpeed [ECUC_Dem_00758]		
Parent Container	DemGeneralOBD		
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]		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemOBDPFCCycleRef [ECUC_Dem_00920]		
Parent Container	DemGeneralOBD		
Description	Defines the operationCycle, which is relevant for processing the OBDPFCCycle.		
Multiplicity	1		
Type	Reference to DemOperationCycle		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name	DemOBDTimeSinceEngineStart [ECUC_Dem_00827]
Parent Container	DemGeneralOBD
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]
Post-Build Variant Multiplicity	false
Post-Build Variant Value	false

Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemOBDWarmUpCycleRef [ECUC_Dem_00922]		
Parent Container	DemGeneralOBD		
Description	Defines the operationCycle which denotes the OBD warm-up cycle.		
Multiplicity	1		
Type	Reference to DemOperationCycle		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
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.

10.2.3.2 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]		
Parent Container	DemObdDTC		
Description	This parameter is TRUE, when the event is affected by the Dem PTO handling.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDtcValue [ECUC_Dem_00885]		
Parent Container	DemObdDTC		
Description	Unique Diagnostic Trouble Code value for OBD		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 65535		
Default Value			
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

Name	DemEventOBDReadinessGroup [ECUC_Dem_00755]	
Parent Container	DemObdDTC	
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_EGR	EGR system - spark, compr.	
	DEM_OBD_RDY_EGSENS	Exhaust Gas Sensor - 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	
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemJ1939DTCValue [ECUC_Dem_00892]	
Parent Container	DemObdDTC	
Description	Unique Diagnostic Trouble Code value for J1939 (consisting of SPN and FMI)	
Multiplicity	0..1	
Type	EcucIntegerParamDef	
Range	1 .. 16777214	
Default Value		
Post-Build Variant Multiplicity	true	
Post-Build Variant Value	true	

Multiplicity Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
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

10.2.3.3 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	DemIUMPRDenGroup [ECUC_Dem_00838]	
Parent Container	DemRatio	
Description	This parameter specifies the assigned denominator type which is applied in addition to the DEM_IUMPR_GENERAL_INDIVIDUAL_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_INDIVIDUAL_DENOMINATOR only.
	DEM_IUMPR_DEN_PHYS_API	Additional physical condition (component activity) computed wihtin the SW-C and reported via Dem_RepIUMPRDenLock / Dem_RepIUMPRDenRelease.
Post-Build Variant Value	false	

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemIUMPRGroup [ECUC_Dem_00737]	
Parent Container	DemRatio	
Description	This parameter specifies the assigned IUMPR group of the ratio Id.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	DEM_IUMPR_AFRI1	Air Fuel Ratio Imbalance Monitor Bank 1
	DEM_IUMPR_AFRI2	Air Fuel Ratio Imbalance Monitor Bank 2
	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_PF1	Particulate Filter Monitor Bank 1
	DEM_IUMPR_PF2	Particulate Filter Monitor Bank 2
	DEM_IUMPR_PMFILTER	
	DEM_IUMPR_PRIVATE	
	DEM_IUMPR_SAIR	
	DEM_IUMPR_SECOXS1	
	DEM_IUMPR_SECOXS2	
Post-Build Variant Value	false	

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemRatioId [ECUC_Dem_00787]		
Parent Container	DemRatio		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemRatioKind [ECUC_Dem_00741]		
Parent Container	DemRatio		
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	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDiagnosticEventRef [ECUC_Dem_00735]		
Parent Container	DemRatio		
Description	This reference contains the link to a diagnostic event.		
Multiplicity	1		
Type	Reference to DemEventParameter		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemFunctionIdRef [ECUC_Dem_00736]		
Parent Container	DemRatio		
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		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemSecondaryFunctionIdRef [ECUC_Dem_00782]		
Parent Container	DemRatio		
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		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.3.4 DemDtrs

SWS Item	[ECUC_Dem_00913]
Container Name	DemDtrs
Description	This container holds the configuration of DTRs collection.
Configuration Parameters	

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
DemDtr	0..65535	This container holds the configuration of one individual DTR.

10.2.3.5 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]		
Parent Container	DemDtr		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDtrCompuNumerator0 [ECUC_Dem_00813]		
Parent Container	DemDtr		
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			
Post-Build Variant Value	false		

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDtrCompuNumerator1 [ECUC_Dem_00814]		
Parent Container	DemDtr		
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 xˆ 1, i.e., the factor.		
Multiplicity	1		
Type	EcucFloatParamDef		
Range	[-INF .. INF]		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDtrId [ECUC_Dem_00807]		
Parent Container	DemDtr		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDtrMid [ECUC_Dem_00809]		
Parent Container	DemDtr		
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			
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

Name	DemDtrTid [ECUC_Dem_00810]		
Parent Container	DemDtr		
Description	The OBDTID of the DTR.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default Value			
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

Name	DemDtrUasid [ECUC_Dem_00811]		
Parent Container	DemDtr		
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			
Post-Build Variant Multiplicity	true		
Post-Build Variant Value	true		
Multiplicity Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

Name	DemDtrUpdateKind [ECUC_Dem_00812]		
Parent Container	DemDtr		
Description	Update conditions applied by the Dem to reports of DTR values. Only supported if a related Event is configured		
Multiplicity	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.	
Default Value	DEM_DTR_UPDATE_ALWAYS		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemDtrEventRef [ECUC_Dem_00808]		
Parent Container	DemDtr		
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		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.3.6 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]		
Parent Container	DemPidClass		
Description	identifier of the PID		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 255		
Default Value			
Post-Build Variant Value	true		
Value Configuration Class	**Pre-compile time**	X	VARIANT-PRE-COMPILE
	Link time	–	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: ECU		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
DemPidDataElement	1..255	This container contains the different data elements contained in the specific PID.

10.2.3.7 DemPidDataElement

SWS Item	[ECUC_Dem_00896]		
Container Name	DemPidDataElement		
Description	This container contains the different data elements contained in the specific PID.		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Configuration Parameters			

Name	DemPidDataElementClassRef [ECUC_Dem_00733]		
Parent Container	DemPidDataElement		
Description	This reference contains the link to a data element class.		
Multiplicity	1		
Type	Reference to DemDataElementClass		
Post-Build Variant Value	true		
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

10.2.4 J1939

10.2.4.1 DemGeneralJ19139

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	DemJ1939ClearDtcSupport [ECUC_Dem_00872]		
Parent Container	DemGeneralJ1939		
Description	This configuration switch defines whether clearing J1939 DTCs (DM03 und DM11) is supported or not. This switches on and off the API Dem_J1939DcmClearDTC.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemJ1939Dm31Support [ECUC_Dem_00868]		
Parent Container	DemGeneralJ1939		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemJ1939ExpandedFreezeFrameSupport [ECUC_Dem_00873]		
Parent Container	DemGeneralJ1939		
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			
Post-Build Variant Value	false		

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemJ1939FreezeFrameSupport [ECUC_Dem_00866]		
Parent Container	DemGeneralJ1939		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemJ1939RatioSupport [ECUC_Dem_00867]		
Parent Container	DemGeneralJ1939		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemJ1939Readiness1Support [ECUC_Dem_00869]		
Parent Container	DemGeneralJ1939		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemJ1939Readiness2Support [ECUC_Dem_00870]		
Parent Container	DemGeneralJ1939		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemJ1939Readiness3Support [ECUC_Dem_00871]		
Parent Container	DemGeneralJ1939		
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			
Post-Build Variant Value	false		

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemJ1939ReadingDtcSupport [ECUC_Dem_00865]		
Parent Container	DemGeneralJ1939		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
DemCallbackJ1939DTCStatusChanged	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.
DemJ1939FreezeFrameClass	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.

10.2.4.2 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]		
Parent Container	DemJ1939FreezeFrameClass		
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		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.4.3 DemSPNClass

SWS Item	[ECUC_Dem_00830]
Container Name	DemSPNClass
Description	This container contains the configuration (parameters) for a SPN.
Configuration Parameters	

Name	DemSPNId [ECUC_Dem_00831]		
Parent Container	DemSPNClass		
Description	Suspect parameter number		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 524287		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemSPNDataElementClassRef [ECUC_Dem_00832]		
Parent Container	DemSPNClass		
Description	This reference contains the link to a data element class.		
Multiplicity	1		
Type	Reference to DemDataElementClass		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.5 Conditions

10.2.5.1 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]		
Parent Container	DemEnableCondition		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemEnableConditionStatus [ECUC_Dem_00656]		
Parent Container	DemEnableCondition		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.5.2 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]		
Parent Container	DemEnableConditionGroup		
Description	References an enable condition.		
Multiplicity	1..255		
Type	Reference to DemEnableCondition		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.5.3 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]		
Parent Container	DemStorageCondition		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemStorageConditionStatus [ECUC_Dem_00731]		
Parent Container	DemStorageCondition		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemStorageConditionReplacementEventRef [ECUC_Dem_00893]		
Parent Container	DemStorageCondition		
Description	Specifies the reference to an event which is stored to event memory and supports failure analysis.		
Multiplicity	0..1		
Type	Reference to DemEventParameter		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.5.4 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]		
Parent Container	DemStorageConditionGroup		
Description	References an enable condition.		
Multiplicity	1..255		
Type	Reference to DemStorageCondition		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.6 Event Memory

10.2.6.1 DemEventMemorySet

SWS Item	[ECUC_Dem_00939]
Container Name	DemEventMemorySet
Description	This container is a collection of referenced event memories and related information for a Dem client.
Configuration Parameters	

Name	DemDtcStatusAvailabilityMask [ECUC_Dem_00652]		
Parent Container	DemEventMemorySet		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemMaxNumberEventEntryPermanent [ECUC_Dem_00689]		
Parent Container	DemEventMemorySet		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemTypeOfDTCSupported [ECUC_Dem_00720]		
Parent Container	DemEventMemorySet		
Description	This parameter defines the format returned by Dem_GetTranslationType 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 = SAE_J2012-DA_DTCFormat_00 (2 byte format)	
	DEM_DTC_TRANSLATION_SAEJ1939_73	SAEJ1939-73 DTC format	
	DEM_DTC_TRANSLATION_SAE_J2012_DA_DTCFORMAT_04	SAE_J2012-DA_DTCFormat_00 (3 byte format)	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemAmberWarningLampIndicatorRef [ECUC_Dem_00821]		
Parent Container	DemEventMemorySet		
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		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemMILIndicatorRef [ECUC_Dem_00723]		
Parent Container	DemEventMemorySet		
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		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
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	DemProtectLampIndicatorRef [ECUC_Dem_00822]		
Parent Container	DemEventMemorySet		
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		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemRedStopLampIndicatorRef [ECUC_Dem_00820]		
Parent Container	DemEventMemorySet		
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		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
DemClearDTC Notification	0..255	Contains callback function definition which are called during clear DTC operations.
DemIndicator	0..255	This container contains the configuration (parameters) for Indicators.
DemMirrorMemory	0..1	This container contains the mirror event memory specific parameters of the Dem module.
DemPrimaryMemory	1	This container contains the primary event memory specific parameters of the Dem module.

DemUserDefinedMemory	0..30	This container contains the user defined event memory specific parameters of the Dem module.

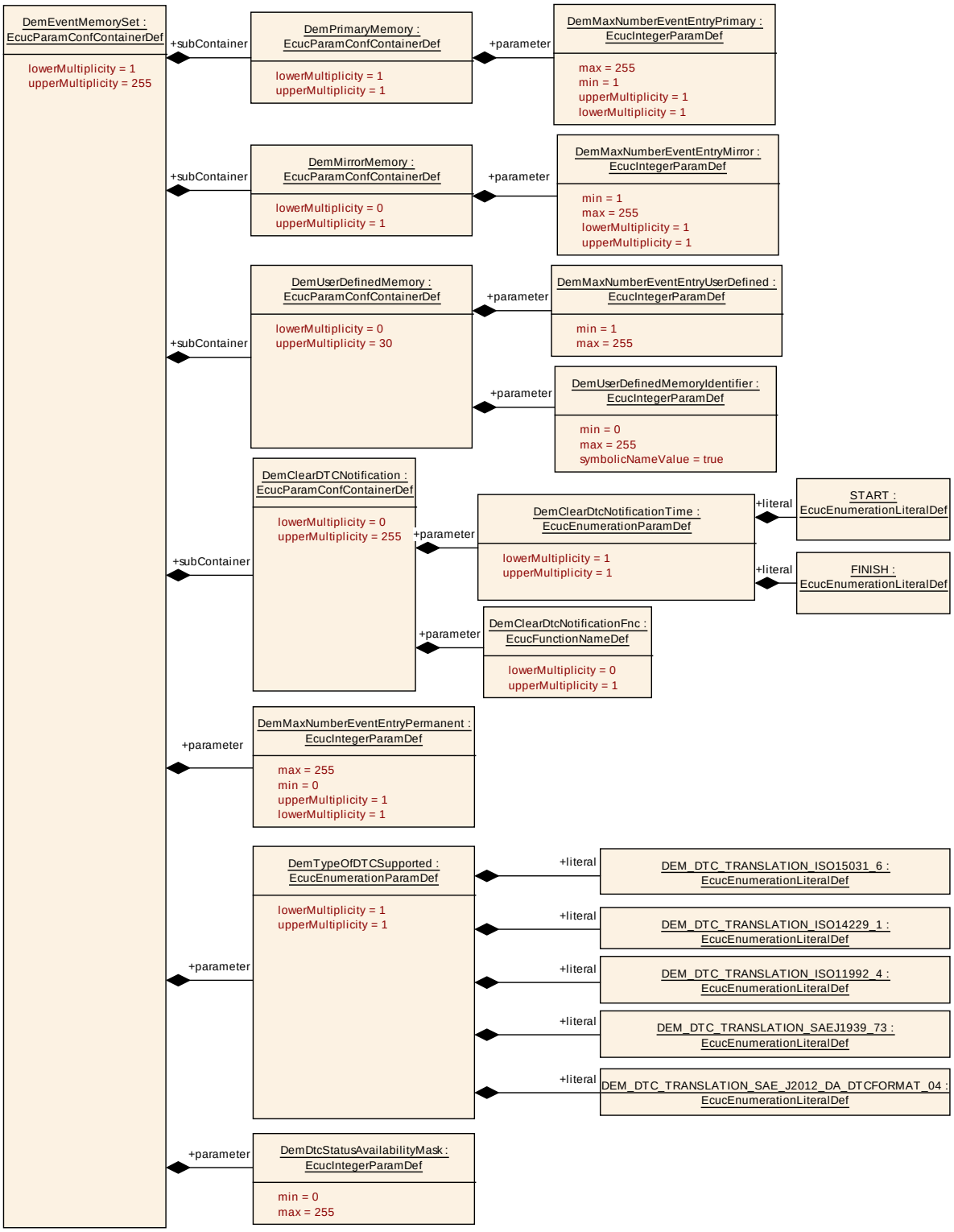


Figure 10.9: DemEventMemorySet configuration

10.2.6.2 DemPrimaryMemory

SWS Item	[ECUC_Dem_00901]
Container Name	DemPrimaryMemory
Description	This container contains the primary event memory specific parameters of the Dem module.
Configuration Parameters	

Name	DemMaxNumberEventEntryPrimary [ECUC_Dem_00690]		
Parent Container	DemPrimaryMemory		
Description	Maximum number of events which can be stored in the primary memory		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.6.3 DemMirrorMemory

SWS Item	[ECUC_Dem_00902]
Container Name	DemMirrorMemory
Description	This container contains the mirror event memory specific parameters of the Dem module.
Configuration Parameters	

Name	DemMaxNumberEventEntryMirror [ECUC_Dem_00688]	
Parent Container	DemMirrorMemory	
Description	Maximum number of events which can be stored in the mirror memory	
Multiplicity	1	
Type	EcucIntegerParamDef	
Range	1 .. 255	
Default Value		
Post-Build Variant Value	false	

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.6.4 DemUserDefinedMemory

SWS Item	[ECUC_Dem_00910]
Container Name	DemUserDefinedMemory
Description	This container contains the user defined event memory specific parameters of the Dem module.
Configuration Parameters	

Name	DemMaxNumberEventEntryUserDefined [ECUC_Dem_00691]		
Parent Container	DemUserDefinedMemory		
Description	Maximum number of events which can be stored in the user defined memory.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemUserDefinedMemoryIdentifier [ECUC_Dem_00903]		
Parent Container	DemUserDefinedMemory		
Description	Identifier used by external tester to identify the User defined event memory.		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 255		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	

Scope / Dependency	scope: ECU

No Included Containers

10.2.7 Debouncing

10.2.7.1 DemDebounceAlgorithmClass

SWS Item	[ECUC_Dem_00604]
Container Name	DemDebounceAlgorithmClass
Description	Debounce algorithm class: counter based, time based, or monitor internal.
Configuration Parameters	

Container Choices		
Container Name	**Multiplicity**	**Scope / Dependency**
DemDebounceCounterBased	0..1	This container contains the configuration (parameters) for counter based debouncing.
DemDebounceMonitorInternal	0..1	This container contains the configuration (parameters) for monitor internal debouncing.
DemDebounceTimeBase	0..1	This container contains the configuration (parameters) for time based debouncing.

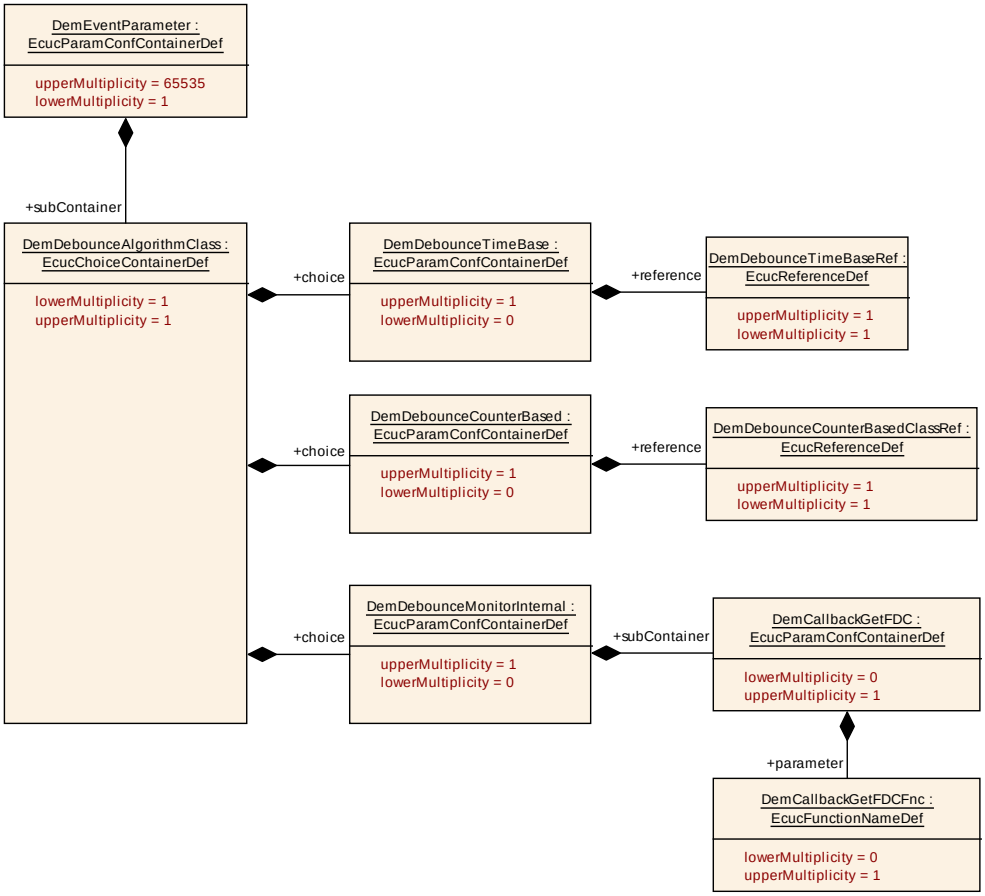


Figure 10.10: Debounce Algorithm Class

10.2.7.2 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]
Parent Container	DemDebounceCounterBased
Description	This reference selects the DemDebounceCounterBasedClass applied for the debouncing of the DemEventParameter.
Multiplicity	1
Type	Reference to DemDebounceCounterBasedClass
Post-Build Variant Value	false

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.7.3 DemDebounceCounterBasedClass

SWS Item	[ECUC_Dem_00881]
Container Name	DemDebounceCounterBasedClass
Description	This container contains the configuration of Debounce Counter Based Class
Configuration Parameters	

Name	DemCounterBasedFdcThresholdStorageValue [ECUC_Dem_00914]		
Parent Container	DemDebounceCounterBasedClass		
Description	Threshold to allocate an event memory entry and to capture the Freeze Frame.		
Multiplicity	0..1		
Type	EcucIntegerParamDef		
Range	1 .. 32767		
Default Value			
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemDebounceBehavior [ECUC_Dem_00786]		
Parent Container	DemDebounceCounterBasedClass		
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.	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemDebounceCounterDecrementStepSize [ECUC_Dem_00635]		
Parent Container	DemDebounceCounterBasedClass		
Description	Defines the step size for decrementation of the internal debounce counter (PREPASSED).		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 32768		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDebounceCounterFailedThreshold [ECUC_Dem_00618]		
Parent Container	DemDebounceCounterBasedClass		
Description	Defines the value of the internal debounce counter, which indicates the failed status.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 32767		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDebounceCounterIncrementStepSize [ECUC_Dem_00637]		
Parent Container	DemDebounceCounterBasedClass		
Description	Defines the step size for incrementation of the internal debounce counter (PREFAILED).		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 32767		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDebounceCounterJumpDown [ECUC_Dem_00685]		
Parent Container	DemDebounceCounterBasedClass		
Description	Switch for the activation of Jump-Down. true: Jump-Down activated false: Jump-Down deactivated		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDebounceCounterJumpDownValue [ECUC_Dem_00638]		
Parent Container	DemDebounceCounterBasedClass		
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		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDebounceCounterJumpUp [ECUC_Dem_00686]		
Parent Container	DemDebounceCounterBasedClass		
Description	Switch for the activation of Jump-Up. true: Jump-Up activated false: Jump-Up deactivated		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDebounceCounterJumpUpValue [ECUC_Dem_00639]		
Parent Container	DemDebounceCounterBasedClass		
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		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDebounceCounterPassedThreshold [ECUC_Dem_00636]		
Parent Container	DemDebounceCounterBasedClass		
Description	Defines the value of the internal debounce counter, which indicates the passed status.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	-32768 .. -1		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDebounceCounterStorage [ECUC_Dem_00791]		
Parent Container	DemDebounceCounterBasedClass		
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		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

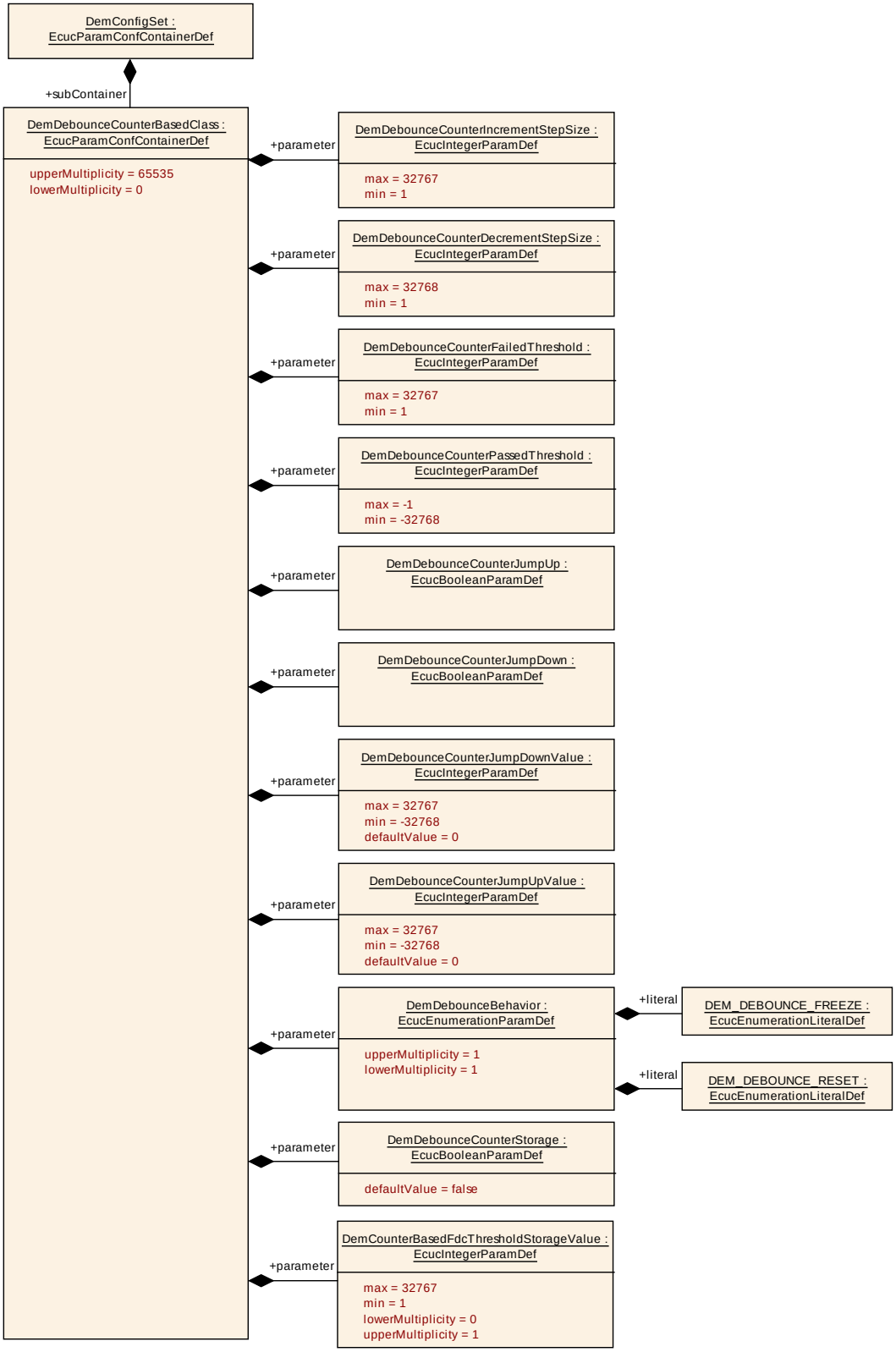


Figure 10.11: Debounce Counter Based Class

10.2.7.4 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]		
Parent Container	DemDebounceTimeBase		
Description	This reference selects the DemDebounceTimeBaseClass applied for the debouncing of the DemEventParameter.		
Multiplicity	1		
Type	Reference to DemDebounceTimeBaseClass		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.7.5 DemDebounceTimeBaseClass

SWS Item	[ECUC_Dem_00882]
Container Name	DemDebounceTimeBaseClass
Description	This container contains the configuration of Debounce Time Based Class.
Configuration Parameters	

Name	DemDebounceBehavior [ECUC_Dem_00789]		
Parent Container	DemDebounceTimeBaseClass		
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.	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemDebounceTimeFailedThreshold [ECUC_Dem_00716]		
Parent Container	DemDebounceTimeBaseClass		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	

Scope / Dependency	scope: ECU

Name	DemDebounceTimePassedThreshold [ECUC_Dem_00717]		
Parent Container	DemDebounceTimeBaseClass		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemTimeBasedFdcThresholdStorageValue [ECUC_Dem_00915]		
Parent Container	DemDebounceTimeBaseClass		
Description	Threshold to allocate an event memory entry and to capture the Freeze Frame.		
Multiplicity	0..1		
Type	EcucFloatParamDef		
Range	[0.001 .. 3600]		
Default Value			
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

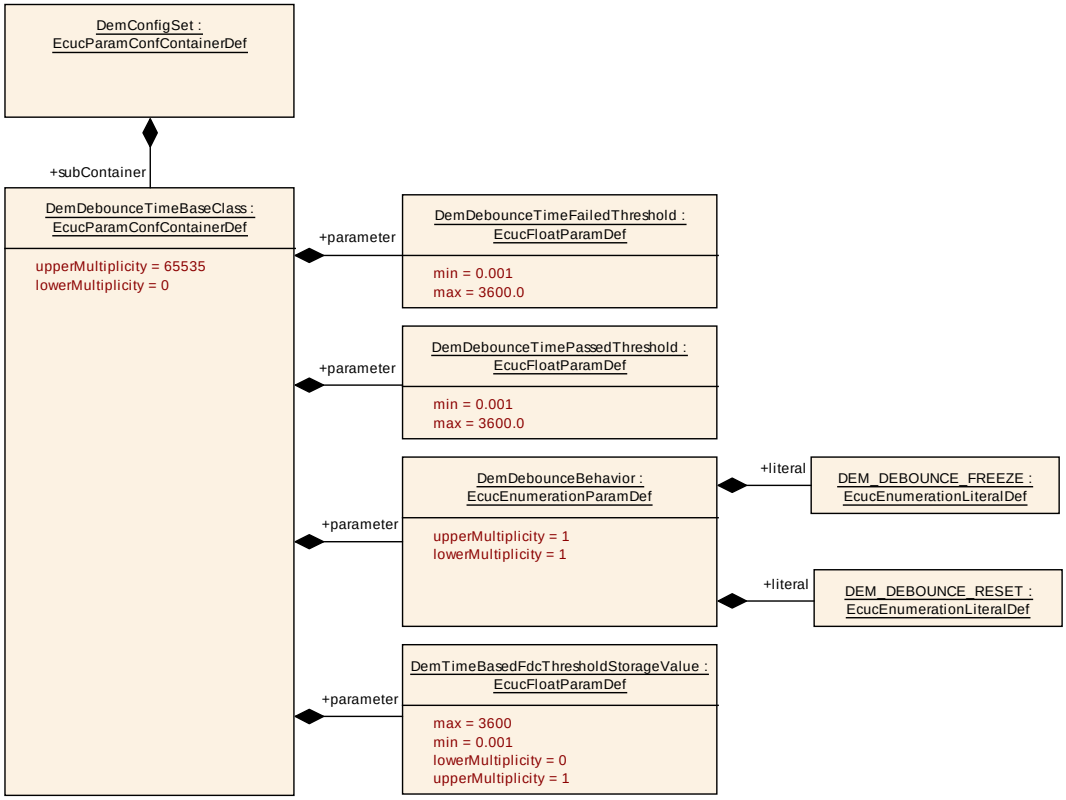


Figure 10.12: Debounce Time Based Class

10.2.7.6 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 / Dependency**
DemCallbackGetFDC	0..1	DemCallbackGetFDC specifies the callback (parameter DemCallbackGetFDCFnc is present) or R-Port (no parameter DemCallbackGetFDCFnc is present) to retrieve the fault detection counter value. In case no container is configured, no fault detection counter will be available.

10.2.8 Callbacks

10.2.8.1 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]		
Parent Container	DemCallbackClearEventAllowed		
Description	Function name of prototype "ClearEventAllowed".		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default Value			
Regular Expression			
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemClearEventAllowedBehavior [ECUC_Dem_00788]	
Parent Container	DemCallbackClearEventAllowed	
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 UDS status byte keeps unchanged.
	DEM_ONLY_THIS_CYCLE_AND_READINESS	The <...>ThisOperationCycle and readiness bits of the UDS status byte are reset.
Default Value	DEM_NO_STATUS_BYTE_CHANGE	
Post-Build Variant Value	false	

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

10.2.8.2 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]		
Parent Container	DemCallbackDTCStatusChanged		
Description	Function name of prototype "DTCStatusChanged".		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default Value			
Regular Expression			
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.8.3 DemCallbackGetFDC

SWS Item	[ECUC_Dem_00630]		
Container Name	DemCallbackGetFDC		
Description	DemCallbackGetFDC specifies the callback (parameter DemCallbackGetFDCFnc is present) or R-Port (no parameter DemCallbackGetFDCFnc is present) to retrieve the fault detection counter value. In case no container is configured, no fault detection counter will be available.		
Configuration Parameters			

Name	DemCallbackGetFDCFnc [ECUC_Dem_00631]		
Parent Container	DemCallbackGetFDC		
Description	This parameter defines the name of the function that the Dem will call to retrieve the fault-detection counter value from a complex driver.		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default Value			
Regular Expression			
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.8.4 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]		
Parent Container	DemCallbackEventDataChanged		
Description	Function name of prototype "EventDataChanged"		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default Value			
Regular Expression			
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.8.5 DemCallbackEventUdsStatusChanged

SWS Item	[ECUC_Dem_00628]
Container Name	DemCallbackEventUdsStatusChanged

Description	The presence of this container indicates, that the Dem has access to an "EventUdsStatusChanged" callback, which the Dem will call to notify other components about the change in the status of an event. In case there is a DemCallbackEventUdsStatusChangedFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackEventUdsStatusChangedFnc, the Dem will have an R-Port requiring the interface CallbackEventUdsStatusChanged, whose name is generated by using the unique callback-prefix followed by the event name.
Configuration Parameters	

Name	DemCallbackEventUdsStatusChangedFnc [ECUC_Dem_00629]		
Parent Container	DemCallbackEventUdsStatusChanged		
Description	Function name of prototype "Dem_CBEventUdsStatusChanged"		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default Value			
Regular Expression			
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.8.6 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]		
Parent Container	DemCallbackInitMForE		
Description	Function name of prototype "InitMonitorForEvent".		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default Value			
Regular Expression			
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.8.7 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]		
Parent Container	DemCallbackJ1939DTCStatusChanged		
Description	Function name of prototype "DTCStatusChanged".		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default Value			
Regular Expression			
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.8.8 DemCallbackMonitorStatusChanged

SWS Item	[ECUC_Dem_00936]
Container Name	DemCallbackMonitorStatusChanged

Description	The presence of this container indicates, that the Dem has access to an "MonitorStatusChanged" callback, which the Dem will call to notify other components about the change in the status of an event. In case there is a DemCallbackMonitorStatusChangedFnc, this parameter defines the name of the function that the Dem will call. In case there is no DemCallbackMonitorStatusChangedFnc, the Dem will have an R-Port requiring the interface CallbackMonitorStatusChanged, whose name is generated by using the unique callback-prefix followed by the event name.
Configuration Parameters	

Name	DemCallbackMonitorStatusChangedFnc [ECUC_Dem_00937]		
Parent Container	DemCallbackMonitorStatusChanged		
Description	Function name of prototype "<Module>_DemTriggerOnMonitorStatus"		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default Value			
Regular Expression			
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.8.9 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]		
Parent Container	DemCallbackOBDDTCStatusChanged		
Description	Function name of prototype "DTCStatusChanged".		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default Value			
Regular Expression			
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.8.10 DemClearDTCNotification

SWS Item	[ECUC_Dem_00925]
Container Name	DemClearDTCNotification
Description	Contains callback function definition which are called during clear DTC operations.
Post-Build Variant Multiplicity	false
Configuration Parameters	

Name	DemClearDtcNotificationFnc [ECUC_Dem_00926]		
Parent Container	DemClearDTCNotification		
Description	Notification callback function name which is called on a clear DTC operation (refer to <Module>_ClearDtcNotification).		
Multiplicity	0..1		
Type	EcucFunctionNameDef		
Default Value			
Regular Expression			
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemClearDtcNotificationTime [ECUC_Dem_00927]		
Parent Container	DemClearDTCNotification		
Description	Configure, whether the callback shall be called on start of a clear or after finishing a clear DTC operation (refer to <Module>_ClearDtcNotification)		
Multiplicity	1		
Type	EcucEnumerationParamDef		
Range	FINISH		
	START		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

10.2.9 Event related data

10.2.9.1 DemFreezeFrameClass

SWS Item	[ECUC_Dem_00673]
Container Name	DemFreezeFrameClass

Description	This container contains the combinations of DIDs for a non OBD2 and WWH-OBD relevant freeze frame class.
Configuration Parameters	

Name	DemDidClassRef [ECUC_Dem_00707]		
Parent Container	DemFreezeFrameClass		
Description	Reference to the DID elements which shall be contained in the freeze frame. **Attributes:** requiresIndex=true		
Multiplicity	1..255		
Type	Reference to DemDidClass		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

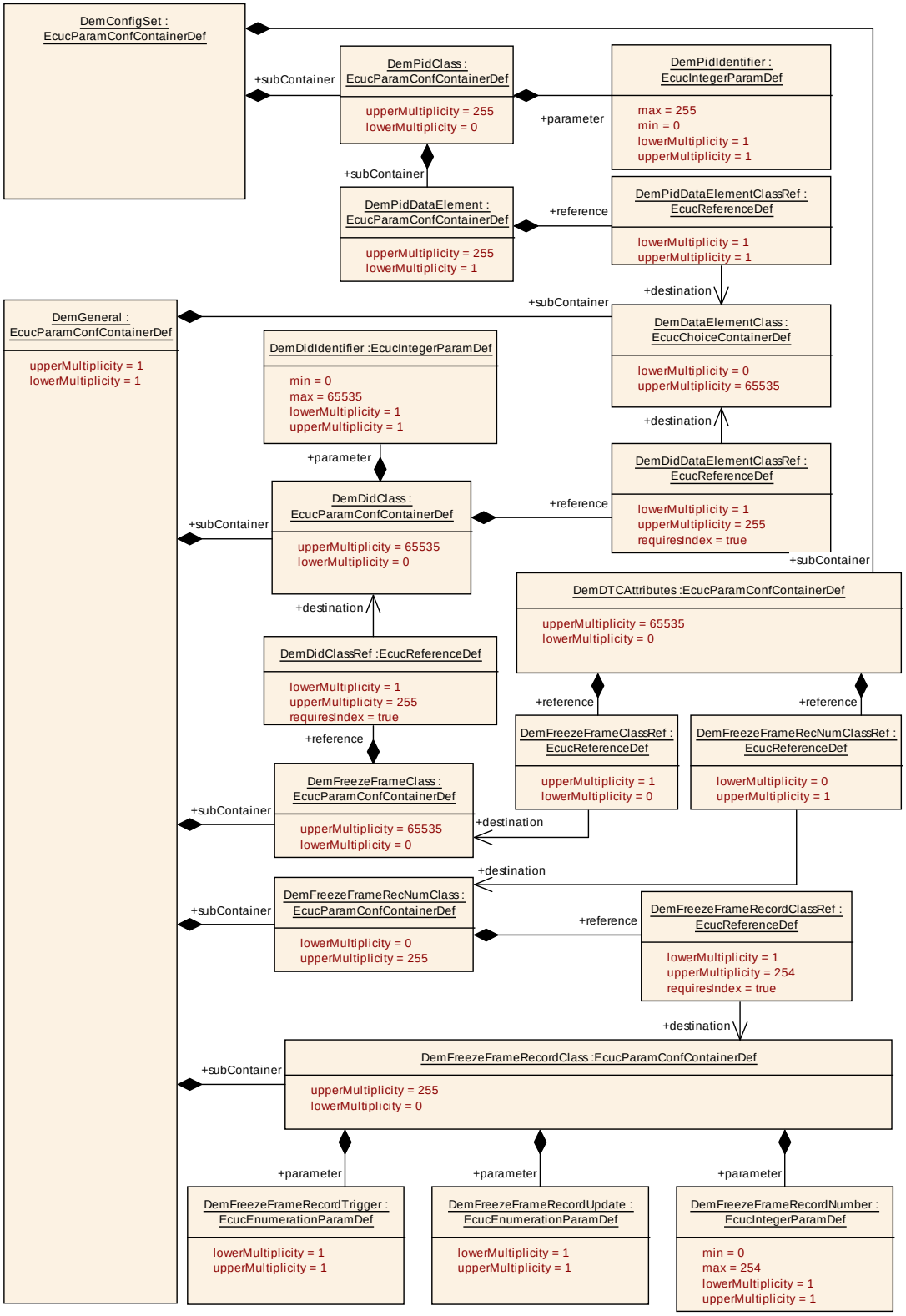


Figure 10.13: DemFreezeFrameClass

10.2.9.2 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]		
Parent Container	DemFreezeFrameRecordClass		
Description	This parameter defines a record number for a freeze frame record. This record number is unique per freeze frame record number class.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 254		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemFreezeFrameRecordTrigger [ECUC_Dem_00803]	
Parent Container	DemFreezeFrameRecordClass	
Description	Defines the trigger to store the FreezeFrameRecord.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	DEM_TRIGGER_ON_CONFIRMED	Event Memory entries are triggered if the UDS status bit 3 (confirmedDTC) changes from 0 to 1.
	DEM_TRIGGER_ON_FDC_THRESHOLD	Event Memory entries are triggered when the FDC threshold is reached.
	DEM_TRIGGER_ON_PENDING	Event Memory entries are triggered if the UDS status bit 2 (pendingDTC) changes from 0 to 1.
	DEM_TRIGGER_ON_TEST_FAILED	Event Memory entries are triggered if the UDS status bit 0 (testFailed) changes from 0 to 1.
Post-Build Variant Value	false	

Value Configuration Class			
	Pre-compile time	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemFreezeFrameRecordUpdate [ECUC_Dem_00802]		
Parent Container	DemFreezeFrameRecordClass		
Description	This parameter defines the case, when the freeze frame record is stored/updated.		
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.	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.9.3 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]
Parent Container	DemFreezeFrameRecNumClass
Description	This parameter references record number(s) for a freeze frame record. **Attributes:** requiresIndex=true
Multiplicity	1..254
Type	Reference to DemFreezeFrameRecordClass
Post-Build Variant Multiplicity	false
Post-Build Variant Value	false

Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.9.4 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]		
Parent Container	DemExtendedDataClass		
Description	This reference contains the link to an extended data class record. **Attributes:** requiresIndex=true		
Multiplicity	1..253		
Type	Reference to DemExtendedDataRecordClass		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.9.5 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	DemExtendedDataRecordNumber [ECUC_Dem_00666]		
Parent Container	DemExtendedDataRecordClass		
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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemExtendedDataRecordTrigger [ECUC_Dem_00804]	
Parent Container	DemExtendedDataRecordClass	
Description	Defines the trigger to store the ExtendedDataRecord.	
Multiplicity	1	
Type	EcucEnumerationParamDef	
Range	DEM_TRIGGER_ON_CONFIRMED	ExtendedDataRecord will be stored when the UDS status confirmed bit changes from 0 to 1.
	DEM_TRIGGER_ON_FDC_THRESHOLD	ExtendedDataRecord will be stored when the FDC reaches its threshold.
	DEM_TRIGGER_ON_MIRROR	ExtendedDataRecord will be stored when the event is transferred to the mirror memory.
	DEM_TRIGGER_ON_PASSED	ExtendedDataRecord will be stored when the event is reported as passed (testFailed bit transition 1 --> 0).

	DEM_TRIGGER_ON_PENDING	ExtendedDataRecord will be stored when the UDS status pending bit changes from 0 to 1.	
	DEM_TRIGGER_ON_TEST_FAILED	ExtendedDataRecord will be stored when the UDS status test failed bit changes from 0 to 1.	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Name	DemExtendedDataRecordUpdate [ECUC_Dem_00621]		
Parent Container	DemExtendedDataRecordClass		
Description	This extended data record is captured if the configured trigger condition in "DemExtendedDataRecordTrigger" is fulfilled.		
Multiplicity	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.	
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDataElementClassRef [ECUC_Dem_00771]		
Parent Container	DemExtendedDataRecordClass		
Description	This reference contains the link to a data element class. **Attributes:** requiresIndex=true		
Multiplicity	1..255		
Type	Reference to DemDataElementClass		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.10 Data elements

10.2.10.1 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	

Container Choices		
Container Name	**Multiplicity**	**Scope / Dependency**
DemExternalCSData ElementClass	0..1	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.
DemExternalSRData ElementClass	0..1	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.
DemInternalDataElement Class	0..1	This container contains the configuration (parameters) for an internal data element class.

10.2.10.2 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]		
Parent Container	DemDataElementInstance		
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		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.10.3 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	DemDidIdentifier [ECUC_Dem_00650]		
Parent Container	DemDidClass		
Description	Identifier of the Data ID.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 65535		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name	DemDidDataElementClassRef [ECUC_Dem_00617]		
Parent Container	DemDidClass		
Description	This reference contains the link to a data element class. **Attributes:** requiresIndex=true		
Multiplicity	1..255		
Type	Reference to DemDataElementClass		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.10.4 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]		
Parent Container	DemInternalDataElementClass		
Description	Defines the size of the data element in bytes.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemInternalDataElement [ECUC_Dem_00616]	
Parent Container	DemInternalDataElementClass	
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_AGINGCTR_UPCNT_FIRST_ACTIVE	Map up-counting Dem-internal aging counter, max. range: 1 byte, starting at 1 when aging starts.
	DEM_CLR_DIST	Distance since diagnostic trouble codes cleared (PID31)
	DEM_CLR_TIME	Time since diagnostic trouble codes cleared (PID4E)
	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_MIL_DIST	Distance Travelled While MIL is Activated (PID21)
	DEM_MIL_TIME	Time run by the engine while MIL is activated (PID4D)
	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 DemDTCSignificance), max. range: 1 byte (0 = OCCURRENCE, 1 = FAULT)
	DEM_WARM_UPS	Number of warm-ups since diagnostic trouble codes cleared (PID30)
Post-Build Variant Value	false	

Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.10.5 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]		
Parent Container	DemExternalCSDataElementClass		
Description	Defines the size of the data element in bytes.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDataElementReadFnc [ECUC_Dem_00619]
Parent Container	DemExternalCSDataElementClass
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	
Post-Build Variant Multiplicity	false

Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDataElementUsePort [ECUC_Dem_00647]		
Parent Container	DemExternalCSDataElementClass		
Description	If the parameter is set to True, a R-Port is generated, to obtain the data element (interface DataServices_{Data}). 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			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

No Included Containers

10.2.10.6 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]		
Parent Container	DemExternalSRDataElementClass		
Description	Defines the size of the data element in bytes.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	1 .. 255		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDataElementDataType [ECUC_Dem_00840]		
Parent Container	DemExternalSRDataElementClass		
Description	Provide the implementation data type of data belonging to a external data.		
Multiplicity	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.	
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemDataElementEndianness [ECUC_Dem_00841]		
Parent Container	DemExternalSRDataElementClass		
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	
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
DemDiagnosisScaling	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	0..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.

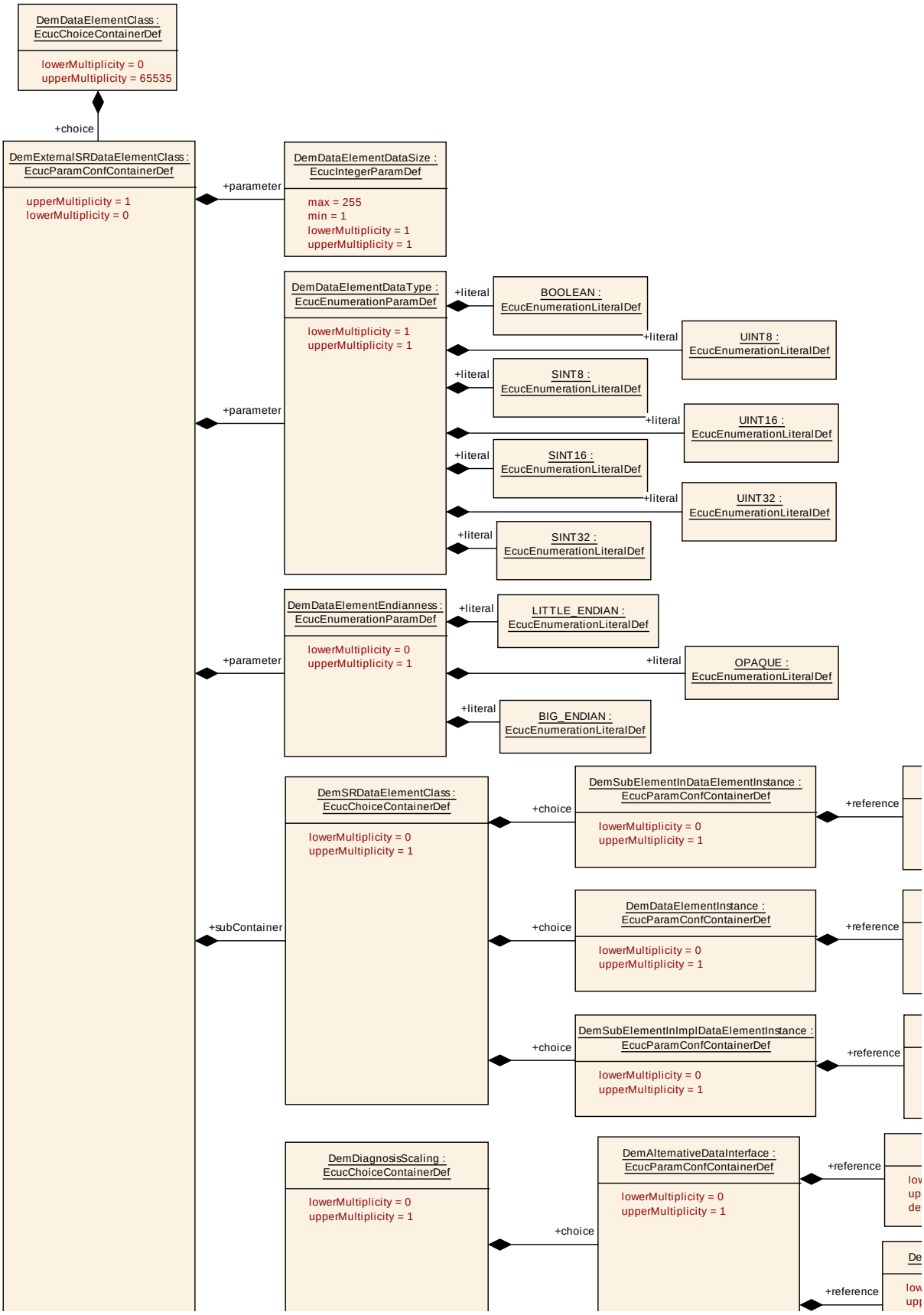


Figure 10.14: DemExternalSRDataElementClass

10.2.10.7 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	

Container Choices		
Container Name	**Multiplicity**	**Scope / Dependency**
DemDataElement Instance	0..1	Instance Reference to the primitive data in a port where the data element is typed with an ApplicationPrimitveDataType or an ImplementationDataType.
DemSubElementInData ElementInstance	0..1	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.
DemSubElementInImpl DataElementInstance	0..1	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.

10.2.10.8 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]		
Parent Container	DemSubElementInDataElementInstance		
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 APPLICATION-COMPOSITE-ELEMENT-DATA-PROTOTYPE context: ROOT-SW-COMPOSITION-PROTOTYPE SW-COMPONENT-PROTOTYPE PORT-PROTOTYPE AUTOSAR-DATA-PROTOTYPE APPLICATION-COMPOSITE-ELEMENT-DATA-PROTOTYPE*		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.10.9 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]		
Parent Container	DemSubElementInImplDataElementInstance		
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*		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.10.10 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	

Container Choices		
Container Name	**Multiplicity**	**Scope / Dependency**
DemAlternativeData Interface	0..1	This container contains the configuration (parameters) of an alternative Diagnosis Representation by the means of a VariableDataPrototoype in a DataInterface. Additionally a reference to PortInterfaceMapping can be defined which provide already the mapping rules between the VariableDataPrototoype in a DataInterface used by the software component (DemSRDataElementClass) and the intended Diagnosis Representation defined by DemExternalSRDataElementClass.

DemAlternativeDataType	0..1	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 that refers to a CompuMethod of category TEXTTABLE and SCALE_LINEAR_AND_TEXTTABLE.
DemAlternative DiagnosticDataElement	0..1	This container contains the configuration (parameters) of an alternative Diagnosis Representation by the means of Diagnostic Extract.

10.2.10.11 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]		
Parent Container	DemAlternativeDataInterface		
Description	Alternative Diagnosis Representation for the data defined by the means of a VariableDataPrototype in a DataInterface. The CompuMethod of the data type of the referenced VariableDataPrototype will be applied to the data type of the VariableDataPrototype in the interface used by the Dem.		
Multiplicity	1		
Type	Foreign reference to VARIABLE-DATA-PROTOTYPE		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Name	DemPortInterfaceMapping [ECUC_Dem_00846]		
Parent Container	DemAlternativeDataInterface		
Description	Optional reference to PortInterfaceMapping which defines the mapping rules. The PortInterfaceMapping is used to get the DataPrototypeMapping that describes a conversion between the data prototype referenced by DemDataElement and the data prototype referenced from DcmDspExternalSRDataElementClass.		
Multiplicity	0..1		
Type	Foreign reference to PORT-INTERFACE-MAPPING		
Post-Build Variant Multiplicity	false		
Post-Build Variant Value	false		
Multiplicity Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

No Included Containers

10.2.10.12 DemAlternativeDiagnosticDataElement

SWS Item	[ECUC_Dem_00934]
Container Name	DemAlternativeDiagnosticDataElement
Description	This container contains the configuration (parameters) of an alternative Diagnosis Representation by the means of Diagnostic Extract.
Configuration Parameters	

Name	DemDiagnosticDataElementRef [ECUC_Dem_00935]		
Parent Container	DemAlternativeDiagnosticDataElement		
Description	Alternative Diagnosis Representation for the data defined by the means of a DiagnosticDataElement in the Diagnostic Extract. This EcucForeignReference enables the access to all SwDataDefProps, in particular BaseType, CompuMethod and DataConstr The CompuMethod and DataConstr that applies to the referenced DiagnosticDataElement will be applied to the data type of the VariableDataPrototype in the interface used by the Dem. The mapped ImplementationDataType needs to match the given BaseType.		
Multiplicity	1		
Type	Foreign reference to DIAGNOSTIC-DATA-ELEMENT		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: local		

No Included Containers

10.2.10.13 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 that refers to a CompuMethod of category TEXTTABLE and SCALE_LINEAR_AND_TEXTTABLE.
Configuration Parameters	

Name	DemApplicationDataType [ECUC_Dem_00848]		
Parent Container	DemAlternativeDataType		
Description	Alternative Diagnosis Representation for the data defined by the means of a ApplicationDataType of category VALUE, BOOLEAN or ARRAY. The CompuMethod that applies to the referenced ApplicationDataType in case of category VALUE or BOOLEAN will be applied to the data type of the VariableDataPrototype in the interface used by the Dem.		
Multiplicity	1		
Type	Foreign reference to APPLICATION-DATA-TYPE		
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency			

Included Containers		
Container Name	**Multiplicity**	**Scope / Dependency**
DemTextTableMapping	0..*	The purpose of the DemDspTextTableMapping is to associate a texttable value defined in the context of the Dem to a texttable value defined in the context of a CompuMethod referenced by a DataType that shall be taken to create a dataElement in a SenderReceiverInterface. By this means it is possible to create a primitive version of a TexttableMapping (which can only be applied if a dataElement already exists). In other words, the DemDspTextTableMapping provides a similar mechanism to the TexttableMapping in a situation where the TexttableMapping cannot be applied since the SenderReceiverInterface for the PortPrototype on the Dem ServiceComponent does not yet exist.

10.2.10.14 DemTextTableMapping

SWS Item	[ECUC_Dem_00849]
Container Name	DemTextTableMapping

Description	The purpose of the DemDspTextTableMapping is to associate a texttable value defined in the context of the Dem to a texttable value defined in the context of a CompuMethod referenced by a DataType that shall be taken to create a dataElement in a SenderReceiverInterface. By this means it is possible to create a primitive version of a TexttableMapping (which can only be applied if a dataElement already exists). In other words, the DemDspTextTableMapping provides a similar mechanism to the TexttableMapping in a situation where the TexttableMapping cannot be applied since the SenderReceiverInterface for the PortPrototype on the Dem ServiceComponent does not yet exist.
Configuration Parameters	

Name	DemDiagnosisRepresentationDataValue [ECUC_Dem_00851]		
Parent Container	DemTextTableMapping		
Description	The data value in the diagnosis representation.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 18446744073709551615		
Default Value			
Post-Build Variant Value	false		
Value Configuration Class	**Pre-compile time**	X	All Variants
	Link time	–	
	Post-build time	–	
Scope / Dependency	scope: ECU		

Name	DemInternalDataValue [ECUC_Dem_00850]		
Parent Container	DemTextTableMapping		
Description	The ECU internal data value.		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 18446744073709551615		
Default Value			
Post-Build Variant Value	false		
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.

11 Not Applicable Requirements

[SWS_Dem_00999] ⌈ These requirements are not applicable to this specification. ⌋(*SRS_BSW_00005*, *SRS_BSW_00161*, *SRS_BSW_00162*, *SRS_BSW_00164*, *SRS_BSW_00168*, *SRS_BSW_00170*, *SRS_BSW_00327*, *SRS_BSW_00331*, *SRS_BSW_00341*, *SRS_BSW_00347*, *SRS_BSW_00348*, *SRS_BSW_00350*, *SRS_BSW_00353*, *SRS_BSW_00357*, *SRS_BSW_00359*, *SRS_BSW_00360*, *SRS_BSW_00361*, *SRS_BSW_00374*, *SRS_BSW_00375*, *SRS_BSW_00379*, *SRS_BSW_00433*, *SRS_Diag_04005*, *SRS_Diag_04006*, *SRS_Diag_04007*, *SRS_Diag_04015*, *SRS_Diag_04016*, *SRS_Diag_04019*, *SRS_Diag_04020*, *SRS_Diag_04021*, *SRS_Diag_04032*, *SRS_Diag_04033*, *SRS_Diag_04059*, *SRS_Diag_04064*, *SRS_Diag_04086*, *SRS_Diag_04087*, *SRS_Diag_04089*, *SRS_Diag_04090*, *SRS_Diag_04091*, *SRS_Diag_04097*, *SRS_Diag_04098*, *SRS_Diag_04100*, *SRS_Diag_04101*, *SRS_Diag_04119*, *SRS_Diag_04120*, *SRS_Diag_04121*, *SRS_Diag_04135*, *SRS_Diag_04136*, *SRS_Diag_04139*, *SRS_Diag_04143*, *SRS_Diag_04144*, *SRS_Diag_04145*, *SRS_Diag_04146*)