Document Title	Specification of DIO Driver
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	020
Document Classification	Standard

Document Version	2.5.0
Document Status	Final
Part of Release	4.0
Revision	3

Document Change History

Date	Version	Changed by	Change Description
30.09.2011	2.5.0	AUTOSAR Administration	• Removed Dem.h from DIO171 and added new requirement DIO194
15.10.2010	2.4.0	AUTOSAR Administration	• Added a new API "Dio_LevelType Dio_FlipChannel(Dio_ChannelType ChannelId)" to flip (change from 1 to 0 or from 0 to 1) the level of a channel and return the level of the channel after flip. • Removed requirement DIO174 and rephrased DIO106. • Added requirements DIO188 and DIO189, to report DET error DIO_E_PARAM_POINTER from Dio_GetVersionInfo().
07.12.2009	2.3.0	AUTOSAR Administration	• Clarification of DIO014 • DioVersionInfoApi added to DIO071 • Clean up of configuration parameters and header file inclusion structure • Legal disclaimer revised
23.06.2008	2.2.1	AUTOSAR Administration	• Legal disclaimer revised
19.12.2007	2.2.0	AUTOSAR Administration	• Harmonized initialization with MCAL modules • Optional inclusion of the DEM header file • Added explanation on dependency between DIO_PORT_MASK and DIO_PORT_OFFSET • Document meta information extended • Small layout adaptations made

Document Change History			
Date	**Version**	**Changed by**	**Change Description**
24.01.2007	2.1.0	AUTOSAR Administration	• File structure update • Removed BSW00324 • In the configuration where pre-compile and link time is possible the variant for pre-compile is now always "PC" and not "All variants". • Added Chapter 8.6 • Changes in referencing symbolic naming • Updated traceability matrix regarding BSW00435 and BSW00436 • Legal disclaimer revised • “Advice for users” revised • “Revision Information” added
26.01.2006	2.0.0	AUTOSAR Administration	• Major changes in chapter 10, Configuration specification • Structure of document changed partly • Readback support moved to PORT Driver
30.06.2005	1.0.0	AUTOSAR Administration	Initial Release

Disclaimer

Advice for users

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

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

.Table of Contents

1 Introduction and functional overview

This specification specifies the functionality, API and the configuration of the AUTOSAR Basic Software module DIO Driver.
This specification is applicable to drivers only for on chip DIO pins and ports.

The DIO Driver provides services for reading and writing to/from

- DIO Channels (Pins)
- DIO Ports
- DIO Channel Groups

The behaviour of those services is synchronous.

This module works on pins and ports which are configured by the PORT driver for this purpose. For this reason, there is no configuration and initialization of this port structure in the DIO Driver.

The diagram below identifies the DIO Driver functions, and the structure of the PORT Driver and DIO Driver within the MCAL software layer.

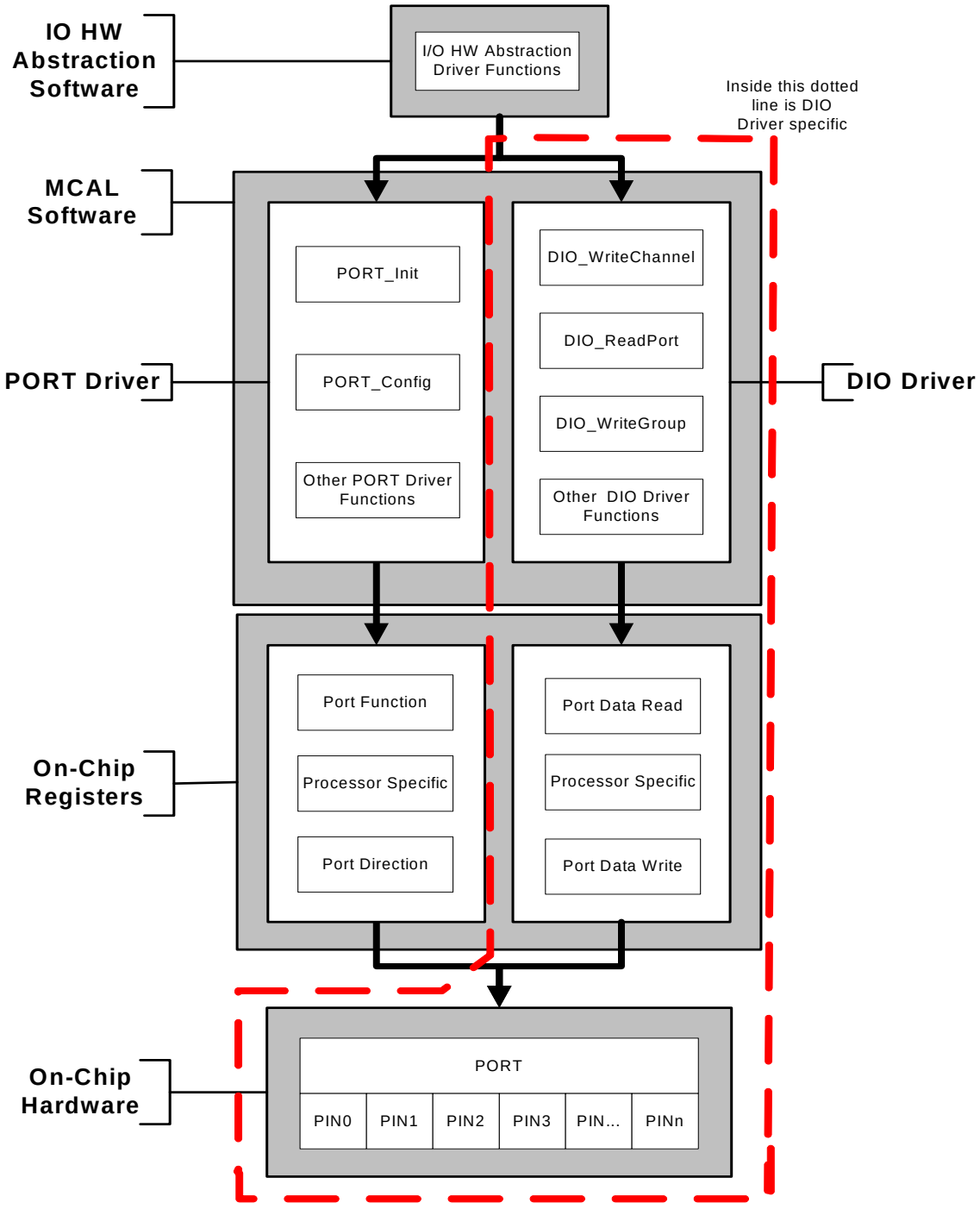

Figure 1: DIO Driver Structure and Integration

2 Acronyms and abbreviations

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

Abbreviation / Acronym:	***Description:***
DIO channel:	Represents a single general-purpose digital input/output pin
DIO port:	Represents several DIO channels that are grouped by hardware (typically controlled by one hardware register). Example: Port A (8 bit) of Freescale HC08
DIO channel group:	Represents several adjoining DIO channels represented by a logical group. A DIO channel group shall belong to one DIO port. Example: Port pins 2..6 of an 8 bit port addressing a multiplexer
Physical Level (Input):	Two states possible: LOW/HIGH. A bit value '0' represents a LOW, a bit value '1' represents a HIGH.
Physical Level (Output):	Two states possible: LOW/HIGH. A bit value '0' represents a LOW, a bit value '1' represents a HIGH.
LSB	Least Significant Bit
MSB	Most Significant Bit
DIO	Digital Input Output
ID	Identifier
ADC	Analog to Digital Converter
SPI	Serial Peripheral Interface
PWM	Pulse Width Modulation
ICU	Input Capture Unit
DET	Development Error Tracer
DEM	Diagnostic Event Manager

3 Related documentation

3.1 Deliverables of AUTOSAR

[1] Layered Software Architecture
AUTOSAR_EXP_LayeredSoftwareArchitecture.pdf

[2] List of Basic Software Modules
AUTOSAR_TR_BSWModuleList.pdf

[3] General Requirements on SPAL
AUTOSAR_SRS_SPALGeneral.pdf

[4] General Requirements on Basic Software Modules
AUTOSAR_SRS_BSWGeneral.pdf

[5] Specification of ECU Configuration
AUTOSAR_TPS_ECUConfiguration.pdf

[5] Specification of PORT Driver,
AUTOSAR_SWS_PortDriver.pdf

[6] Specification of Standard Types,
AUTOSAR_SWS_StandardTypes.pdf

[6] AUTOSAR Basic Software Module Description Template,
AUTOSAR_TPS_BSWModuleDescriptionTemplate.pdf

3.2 Related standards and norms

[7] Specification I/O Drivers,
http://www.automotive-his.de/download/
API_IODriver_2_1_3.pdf

4 Constraints and assumptions

4.1 Limitations

No limitations

4.2 Applicability to car domains

No restrictions.

5 Dependencies to other modules

Port Driver Module

Many ports and port pins are assigned by the PORT Driver Module to various functionalities as for example:

- General purpose I/O
- ADC
- SPI
- PWM

[DIO061] ⌈The Dio module shall not provide APIs for overall configuration and initialization of the port structure which is used in the Dio module. These actions are done by the PORT Driver Module.⌋ ()

[DIO063] ⌈The Dio module shall adapt its configuration and usage to the microcontroller and ECU.⌋ ()

[DIO102] ⌈The Dio module's user shall only use the Dio functions after the Port Driver has been initialized. Otherwise the Dio module will exhibit undefined behavior.⌋ ()

5.1 File structure

[DIO117] ⌈The Dio module shall comply with the following file structure

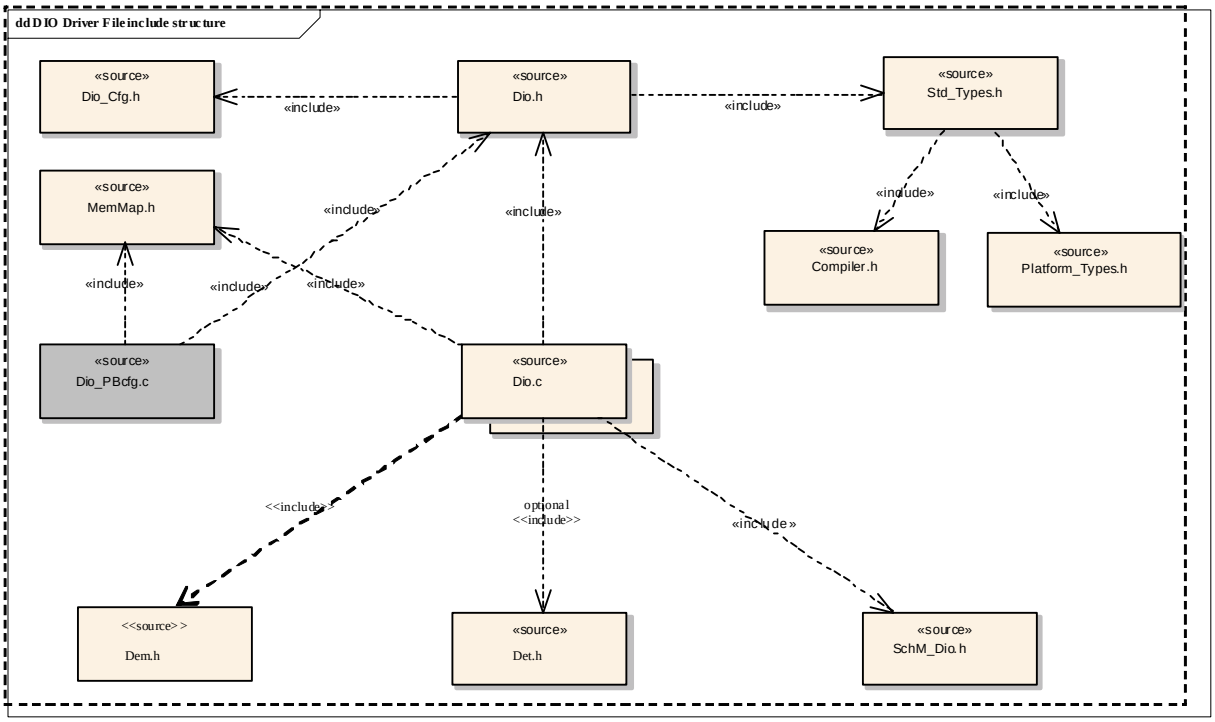

Figure 2: Include File Structure

」(BSW158, BSW00301, BSW00302, BSW00346, BSW00380, BSW00381, BSW00409, BSW00412, BSW00419, BSW00435, BSW00436)

[DIO168] 「`Dio.h` shall include `Dio _Cfg.h` for the API pre-compiler switches」()

[DIO169] 「`Dio.c` has access to the `Dio_Cfg.h` via the implicitly include through the `Dio.h` file.」()

[DIO170] 「Dio.h shall include Std_Types.h.」()

[DIO171] 「Dio.c shall include MemMap.h and SchM_Dio.h.」()

[DIO194] 「Dio.c shall include Det.h if detection of development error (DET) is enabled.」()

[DIO172] ⌈The module shall optionally include the Dem.h file if any production error will be issued by the implementation. By this inclusion the APIs to report errors as well as the required Event Id symbols are included.⌋ ()

Note: This specification defines the name of the Event Id symbols, which are provided by XML to the DEM configuration tool.

[DIO173] ⌈The DEM configuration tool assigns ECU dependent values to the Event Id symbols and publishes the symbols in Dem_IntErrId.h.⌋ ()

6 Requirements traceability

This chapter refers to input requirements specified in the SRS documents (Software Requirements Specifications) that are applicable for this software module.

The table below lists links to specification items of the DIO driver SWS document, that satisfy the input requirements. Only functional requirements are referenced.

Requirement	Satisfied by
-	DIO109
-	DIO183
-	DIO190
-	DIO135
-	DIO136
-	DIO140
-	DIO194
-	DIO163
-	DIO131
-	DIO177
-	DIO192
-	DIO167
-	DIO173
-	DIO024
-	DIO023
-	DIO161
-	DIO175
-	DIO102
-	DIO185
-	DIO134
-	DIO106
-	DIO133
-	DIO105
-	DIO179
-	DIO060
-	DIO170
-	DIO103
-	DIO137
-	DIO021
-	DIO176
-	DIO172
-	DIO182
-	DIO181

-	DIO193
-	DIO169
-	DIO108
-	DIO180
-	DIO171
-	DIO189
-	DIO061
-	DIO015
-	DIO178
-	DIO184
-	DIO063
-	DIO160
-	DIO104
-	DIO053
-	DIO168
-	DIO018
-	DIO164
-	DIO162
-	DIO186
-	DIO191
-	DIO188
-	DIO187
-	DIO138
-	DIO126
BSW00301	DIO117
BSW00302	DIO117
BSW00304	DIO195
BSW00306	DIO195
BSW00307	DIO195
BSW00308	DIO195
BSW00309	DIO195
BSW00314	DIO195
BSW00321	DIO195
BSW00323	DIO114, DIO065, DIO075, DIO074
BSW00325	DIO195
BSW00326	DIO195
BSW00327	DIO067, DIO065
BSW00328	DIO195
BSW00329	DIO195
BSW00330	DIO195
BSW00331	DIO195
BSW00333	DIO195

BSW00334	DIO195
BSW00335	DIO195
BSW00336	DIO195
BSW00337	DIO065
BSW00338	DIO066, DIO067, DIO073
BSW00339	DIO195
BSW00341	DIO195
BSW00342	DIO195
BSW00343	DIO195
BSW00344	DIO001, DIO002
BSW00346	DIO117
BSW00347	DIO195
BSW00350	DIO066
BSW00355	DIO195
BSW00357	DIO195
BSW00358	DIO165
BSW00359	DIO195
BSW00360	DIO195
BSW00369	DIO195
BSW00370	DIO195
BSW00371	DIO195
BSW00373	DIO195
BSW00375	DIO195
BSW00376	DIO195
BSW00377	DIO195
BSW00378	DIO195
BSW00380	DIO117
BSW00381	DIO117
BSW00382	DIO195
BSW00384	DIO195
BSW00387	DIO195
BSW00399	DIO195
BSW00400	DIO195
BSW00404	DIO195
BSW00405	DIO195
BSW00406	DIO195
BSW00407	DIO123
BSW00409	DIO117
BSW00410	DIO124
BSW00411	DIO139
BSW00412	DIO117
BSW00413	DIO195

BSW00414	DIO165
BSW00416	DIO195
BSW00417	DIO195
BSW00419	DIO117
BSW00420	DIO195
BSW00421	DIO066
BSW00422	DIO195
BSW00423	DIO195
BSW00424	DIO195
BSW00425	DIO195
BSW00426	DIO195
BSW00427	DIO195
BSW00428	DIO195
BSW00429	DIO195
BSW00431	DIO195
BSW00432	DIO195
BSW00433	DIO195
BSW00434	DIO195
BSW00435	DIO117
BSW00436	DIO117
BSW005	DIO195
BSW006	DIO195
BSW007	DIO195
BSW009	DIO195
BSW010	DIO195
BSW101	DIO165, DIO001, DIO002
BSW12003	DIO051, DIO089, DIO007, DIO004, DIO034, DIO035
BSW12004	DIO040, DIO051, DIO056, DIO089, DIO091, DIO090, DIO008, DIO039
BSW12005	DIO051, DIO079, DIO089, DIO006, DIO128, DIO127, DIO029, DIO028
BSW12006	DIO051, DIO089, DIO013, DIO031
BSW12007	DIO051, DIO056, DIO089, DIO092, DIO014, DIO093, DIO037
BSW12008	DIO051, DIO089, DIO011, DIO128, DIO127, DIO027
BSW12057	DIO166, DIO001, DIO002
BSW12063	DIO195
BSW12064	DIO001, DIO002
BSW12067	DIO195
BSW12068	DIO195
BSW12069	DIO195
BSW12075	DIO195
BSW12077	DIO195
BSW12078	DIO195
BSW12092	DIO195

BSW12125	DIO001, DIO002
BSW12129	DIO195
BSW12163	DIO001, DIO002
BSW12169	DIO195
BSW12263	DIO017, DIO020, DIO022
BSW12265	DIO195
BSW12267	DIO195
BSW12352	DIO064, DIO070, DIO084, DIO083, DIO012
BSW12355	DIO113, DIO017, DIO020, DIO022, DIO026
BSW12424	DIO005
BSW12448	DIO114, DIO118, DIO119, DIO075, DIO074
BSW12461	DIO001, DIO002
BSW12462	DIO001, DIO002
BSW12463	DIO001, DIO002
BSW157	DIO195
BSW158	DIO117
BSW160	DIO195
BSW161	DIO195
BSW162	DIO195
BSW164	DIO195
BSW167	DIO195
BSW168	DIO195
BSW170	DIO195
BSW171	DIO124
BSW172	DIO195

7 Functional specification

7.1 General Behaviour

7.1.1 Background & Rationale

The DIO Driver abstracts the access to the microcontroller's hardware pins. Furthermore, it allows the grouping of those pins.

7.1.2 Requirements

The Dio SWS shall define functions allowing

- Port-
- Channel-
- Channel-group -

-based read and write access to the internal general purpose I/O ports.

[DIO051] ⌈The Dio module shall not buffer data when providing read and write services.

The Dio SWS shall define synchronous read/write services.⌋ (BSW12003, BSW12004, BSW12005, BSW12006, BSW12007, BSW12008)

[DIO005] ⌈The Dio module's read and write services shall ensure for all services, that the data is consistent (Interruptible read-modify-write sequences are not allowed).⌋ (BSW12424)

[DIO089] ⌈Values used by the DIO Driver for the software level of Channels are either `STD_HIGH` or `STD_LOW`.⌋ (BSW12003, BSW12004, BSW12005, BSW12006, BSW12007, BSW12008)

[DIO128] ⌈A general-purpose digital IO pin represents a **DIO channel**.⌋ (BSW12005, BSW12008)

[DIO127] ⌈The Port module shall configure a DIO channel as input or output [DIO001 and DIO002].⌋ (BSW12005, BSW12008)

[DIO053] ⌈In the DIO Driver, it shall be possible to group several DIO channels by hardware (typically controlled by one hardware register) to represent a **DIO port**.⌋ ()

Note: The single DIO channel levels inside a DIO port represent a bit in the DIO port value, depending on their position inside the port.

[DIO056] ⌈A channel group is a formal logical combination of several adjoining DIO channels within a DIO port.

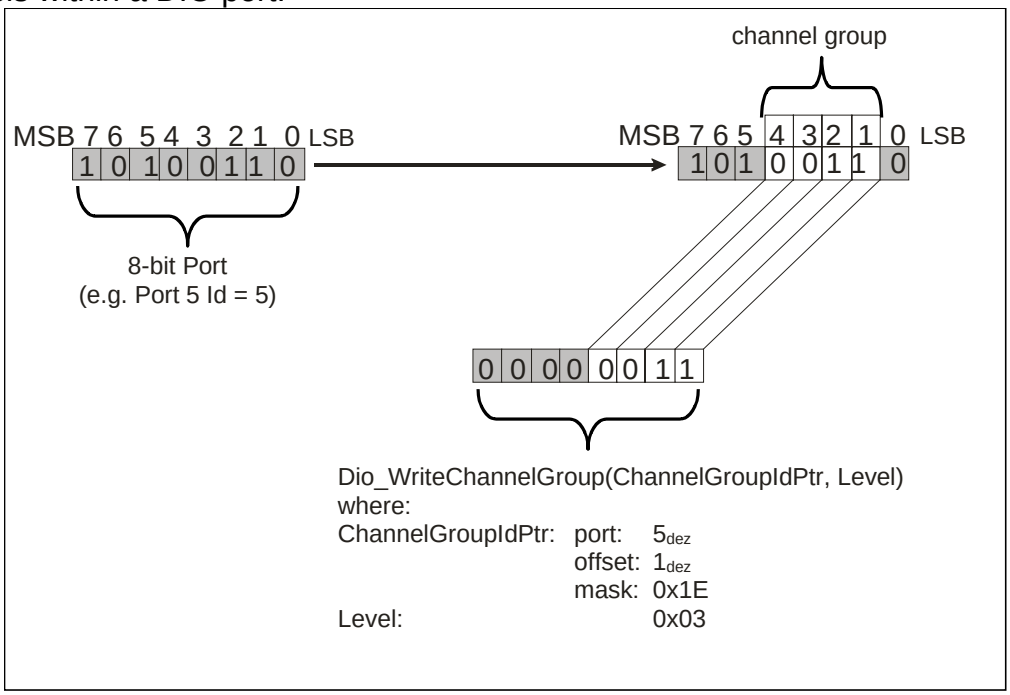

Figure 3: Schematic description of a ChannelGroup

The DIO Driver provides the following services:

- The Dio SWS shall define functions to modify the levels of output channels individually, for a port or for a channel group.
- The Dio SWS shall define functions to read the level of input and output (see DIO083) channels individually, for a port or for a channel group.

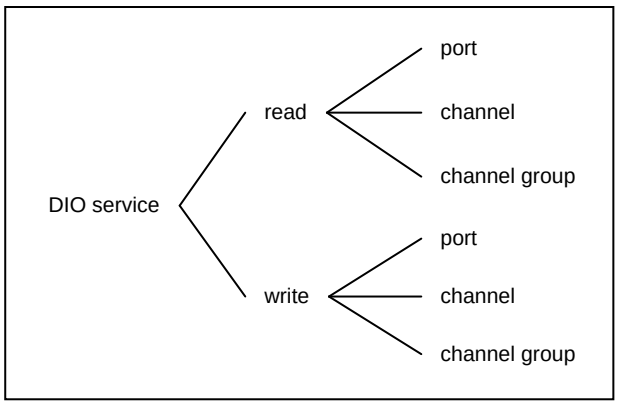

Figure 4: DIO Services

⌋(BSW12004, BSW12007)

[DIO060] ⌈All read and write functions of the Dio module shall be re-entrant.

Reason: The DIO Driver may be accessed by different upper layer handlers or drivers. These upper layer modules may access the driver concurrently. ⌋()

[DIO026] ⌈The configuration process for Dio module shall provide symbolic names for each configured DIO channel, port and group.⌋(BSW12355)

[DIO113] ⌈The Dio module shall publish the symbolic names which have been created during the configuration process in the file “Dio_Cfg.h”. ⌋(BSW12355)

7.1.3 Version check

7.1.3.1 Background & Rationale

The integration of incompatible files must be avoided. Minimum implementation is the version check of the header file inside the C file (version numbers of C and H file shall be identical)

7.1.3.2 Requirements

[DIO106] ⌈The DIO module shall perform Inter Module Checks to avoid integration of incompatible files.
The imported included files shall be checked by preprocessing directives.

The following version numbers shall be verified:

- <MODULENAME>_AR_RELEASE_MAJOR_VERSION
- <MODULENAME>_AR_RELEASE_MINOR_VERSION

Where <MODULENAME> is the module abbreviation of the other (external) modules which provide header files included by DIO module.

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

7.2 Initialization

7.2.1 Background & Rationale

Initialization of the hardware is done by the PORT Driver.

7.2.2 Requirements

[DIO001] ⌈The Dio module shall not provide an interface for initialization of the hardware. The Port Driver performs this. ⌋ (BSW101, BSW00344, BSW12057, BSW12064, BSW12125, BSW12163, BSW12461, BSW12462, BSW12463)

7.3 Runtime reconfiguration

7.3.1 Background & Rationale

Runtime reconfiguration is provided by the PORT Driver.

7.3.2 Requirements

[DIO002] ⌈The PORT driver shall provide the reconfiguration of the port pin direction during runtime. ⌋ (BSW101, BSW00344, BSW12057, BSW12064, BSW12125, BSW12163, BSW12461, BSW12462, BSW12463)

7.4 DIO write service

7.4.1 Background & Rationale

The DIO Driver provides services to transfer data to the microcontroller's pins

7.4.2 Requirements

[DIO064] ⌈The Dio module's write functions shall work on input and output channels. ⌋ (BSW12352)

[DIO070] ⌈If a Dio write function is used on an input channel, it shall have no effect on the physical output level. ⌋ (BSW12352)

[DIO109] ⌈If supported by hardware, the Dio module shall set/clear the output data latch of an input channel so that the required level is output from the pin when the port driver configures the pin as a DIO output pin. ⌋ ()

[DIO119] ⌈If development errors are enabled and an error ocurred, the Dio module's write functions shall NOT process the write command. ⌋ (BSW12448)

7.4.2.1 DIO channel write service

[DIO006] ⌈The `Dio_WriteChannel` function shall set the level of a single DIO channel to `STD_HIGH` or `STD_LOW`.⌋(BSW12005)

7.4.2.2 DIO port write service

[DIO007] ⌈The `Dio_WritePort` function shall simultaneously set the levels of all output channels. A bit value '0' sets the corresponding channel to physical `STD_LOW`, a bit value '1' sets the corresponding channel to physical `STD_HIGH`.⌋(BSW12003)

[DIO004] ⌈The `Dio_WritePort` function shall ensure that the functionality of the input channels of that port is not affected.⌋(BSW12003)

7.4.2.3 DIO channel group write service

[DIO008] ⌈The `Dio_WriteChannelGroup` function shall simultaneously set an adjoining subset of DIO channels (channel group). A bit value '0' sets the corresponding channel to physical `STD_LOW`, a bit value '1' sets the corresponding channel to physical `STD_HIGH`.⌋(BSW12004)

7.5 DIO Read Service

7.5.1 Background & Rationale

The DIO Driver provides services to transfer data from the microcontroller's pins.

7.5.2 Requirements

[DIO012] ⌈The Dio module’s read functions shall work on input and output channels.⌋(BSW12352)

[DIO118] ⌈If development errors are enabled and an error ocurred the Dio module’s read functions shall return with the value '0'.⌋(BSW12448)

7.5.2.1 DIO channel read Service

[DIO011] ⌈The `Dio_ReadChannel` function shall read the level of a single DIO channel.⌋(BSW12008)

7.5.2.2 DIO port read service

[DIO013] ⌈The `Dio_ReadPort` function shall read the levels of all channels of one port. A bit value '0' indicates that the corresponding channel is physical `STD_LOW`, a bit value '1' indicates that the corresponding channel is physical `STD_HIGH`. ⌋ (BSW12006)

7.5.2.3 DIO channel group read service

[DIO014] ⌈The `Dio_ReadChannelGroup` function shall read the levels of a DIO channel group. A bit value '0' indicates that the corresponding channel is physical `STD_LOW`, a bit value '1' indicates that the corresponding channel is physical `STD_HIGH`. ⌋ (BSW12007)

7.5.2.4 DIO readback of output pins

[DIO083] ⌈If the microcontroller supports the direct read-back of a pin value, the Dio module's read functions shall provide the real pin level, when they are used on a channel which is configured as an output channel.⌋ (BSW12352)

[DIO084] ⌈If the microcontroller does not support the direct read-back of a pin value, the Dio module's read functions shall provide the value of the output register, when they are used on a channel which is configured as an output channel.⌋ (BSW12352)

7.6 Error classification

[DIO067] ⌈The Dio module shall report production errors to the Diagnostic Event Manager. ⌋ (BSW00327, BSW00338)

[DIO065] ⌈The Dio module shall detect the following errors and exceptions depending on its build version (development/production mode). ⌋ (BSW00323, BSW00327, BSW00337)

Type of error	***Relevance***	***Related error code***	***Value [hex]***
[DIO175] ⌈Invalid channel name requested⌋ ()	Development	`DIO_E_PARAM_INVALID_CHANNEL_ID`	`0x0A`
[DIO176] ⌈API service called with "NULL	Development	`DIO_E_PARAM_CONFIG`	`0x10`

pointer" parameter」()			
[DIO177]「Invalid port name requested」( )	Development	DIO_E_PARAM_INVALID_PORT_ID	0x14
[DIO178] 「Invalid ChannelGroup passed」()	Development	DIO_E_PARAM_INVALID_GROUP	0x1F
[DIO188]「API service called with a NULL pointer. In case of this error, the API service shall return immediately without any further action, beside reporting this development error.」()	Development	DIO_E_PARAM_POINTER	0x20
--	Production	No error code specified	

7.7 Error detection

7.7.1 API Parameter checking

[DIO074] 「If development error detection is enabled, the services Dio_ReadChannel and Dio_WriteChannel shall check the "Channels" parameter to be valid within the current configuration. If the "Channels" parameter is invalid, the functions shall report the error code DIO_E_PARAM_INVALID_CHANNEL_ID to the DET. 」(BSW00323, BSW12448)

[DIO075]「If development error detection is enabled, the functions Dio_ReadPort and Dio_WritePort shall check the "Ports" parameter to be valid within the current configuration. If the "Ports" parameter is invalid, the functions shall report the error code DIO_E_PARAM_INVALID_PORT_ID to the DET. 」(BSW00323, BSW12448)

[DIO114] 「If development error detection is enabled, the functions Dio_ReadChannelGroup and Dio_WriteChannelGroup shall check the "ChannelGroupIdPtr" parameter to be valid within the current configuration. If the "ChannelGroupIdPtr" parameter is invalid, the functions shall report the error code DIO_E_PARAM_INVALID_GROUP to the DET. 」(BSW00323, BSW12448)

7.8 Error notification

[DIO066] ⌈The detection of all development errors shall be configurable (on/off) with the preprocessor switch `DioDevErrorDetect.`⌋ (BSW00338, BSW00350, BSW00421)

[DIO179] ⌈The Dio module shall report detected development errors to the error hook of the Development Error Tracer (DET) if the preprocessor switch `DioDevErrorDetect` is set (see chapter 10).⌋ ()

[DIO073] ⌈Additional errors that are detected because of specific implementation and/or specific hardware properties shall be added in the DIO device specific implementation specification. The classification and enumeration shall be compatible to the errors listed above DIO065.⌋ (BSW00338)

7.9 Debugging Support

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

[DIO160] ⌈Each variable that shall be accessible by AUTOSAR Debugging, shall be defined as global variable. ⌋ ()

[DIO161] ⌈All type definitions of variables which shall be debugged, shall be accessible by the header file Dio.h. ⌋ ()

[DIO162] ⌈The declaration of variables in the header file shall allow to calculate the size of the variables by C-"sizeof".⌋ ()

[DIO163] ⌈Variables available for debugging shall be described in the respective Basic Software Module Description. ⌋ ()

8 API specification

8.1 Imported types

In this chapter all types included from the following files are listed:

[DIO131] ⌈

Module	*Imported Type*
Dem	Dem_EventIdType
	Dem_EventStatusType
Std_Types	Std_ReturnType
	Std_VersionInfoType

⌋ ()

8.2 Type definitions

[DIO103] ⌈The port width within the types defined for the DIO Driver shall be the size of the largest port on the MCU which may be accessed by the DIO Driver. ⌋ ()

8.2.1 Dio_ChannelType

[DIO182] ⌈

Name:	`Dio_ChannelType`		
Type:	`uint`		
Range:	`This is implementation specific but not all values may be valid within the type.`	`--`	Shall cover all available DIO channels
Description:	Numeric ID of a DIO channel.		

⌋ ()

[DIO015] ⌈Parameters of type `Dio_ChannelType` contain the numeric ID of a DIO channel. ⌋ ()

[DIO180] ⌈The mapping of the ID is implementation specific but not configurable. ⌋ ()

[DIO017] ⌈For parameter values of type `Dio_ChannelType`, the Dio's user shall use the symbolic names provided by the configuration description.

Furthermore, DIO103 applies to the type `Dio_ChannelType`. ⌋ (BSW12263, BSW12355)

8.2.2 Dio_PortType

[DIO183] ⌈

Name:	`Dio_PortType`		
Type:	`uint`		
Range:	`0..<number of ports>`	`--`	Shall cover all available DIO Ports.
Description:	Numeric ID of a DIO port.		

⌋ ()

[DIO018] ⌈Parameters of type `Dio_PortType` contain the numeric ID of a DIO port. ⌋ ()

[DIO181] ⌈The mapping of ID is implementation specific but not configurable. ⌋ ()

[DIO020] ⌈For parameter values of type `Dio_PortType`, the user shall use the symbolic names provided by the configuration description.

Furthermore, DIO103 applies to the type `Dio_PortType`. ⌋ (BSW12263, BSW12355)

8.2.3 Dio_ChannelGroupType

[DIO184] ⌈

Name:	`Dio_ChannelGroupType`		
Type:	`Structure`		
Element:	`uint8/16/32`	`mask`	This element mask which defines the positions of the channel group.
	`uint8`	`offset`	This element shall be the position of the Channel Group on the port, counted from the LSB.
	`Dio_PortType`	`port`	This shall be the port on which the Channel group is defined.
Description:	Type for the definition of a channel group, which consists of several adjoining channels within a port.		

⌋ ()

[DIO021] ⌈`Dio_ChannelGroupType` is the type for the definition of a channel group, which consists of several adjoining channels within a port. ⌋ ()

[DIO022] ⌈For parameter values of type `Dio_ChannelGroupType`, the user shall use the symbolic names provided by the configuration description.

Furthermore, DIO056 applies to the type `Dio_ChannelGroupType`. ⌋(BSW12263, BSW12355)

8.2.4 Dio_LevelType

[DIO185] ⌈

Name:	`Dio_LevelType`		
Type:	`uint8`		
Range:	`STD_LOW`	`0x00`	Physical state 0V
	`STD_HIGH`	`0x01`	Physical state 5V or 3.3V
Description:	These are the possible levels a DIO channel can have (input or output)		

⌋()

[DIO023] ⌈`Dio_LevelType` is the type for the possible levels that a DIO channel can have (input or output). ⌋()

8.2.5 Dio_PortLevelType

[DIO186] ⌈

Name:	`Dio_PortLevelType`		
Type:	`uint`		
Range:	`0...xxx`	- -	If the µC owns ports of different port widths (e.g. 4, 8,16...Bit) Dio_PortLevelType inherits the size of the largest port
Description:	If the µC owns ports of different port widths (e.g. 4, 8,16...Bit) Dio_PortLevelType inherits the size of the largest port.		

⌋()

[DIO024] ⌈`Dio_PortLevelType` is the type for the value of a DIO port.

Furthermore, DIO103 applies to the type `Dio_PortLevelType`. ⌋()

8.2.6 Dio_ConfigType

[DIO187] ⌈

Name:	`Dio_ConfigType`
Type:	`Structure`
Range:	Implementation specific.
Description:	This structure contains all post-build configurable parameters of the DIO driver. A pointer to this structure is passed to the DIO driver initialization function for configuration.

⌋()

[DIO164] ⌈Dio_ConfigType is the type for all post-build configurable parameters of the DIO driver.⌋ ()

8.3 Function definitions

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

8.3.1 Dio_ReadChannel

[DIO133] ⌈

Service name:	Dio_ReadChannel	
Syntax:	`Dio_LevelType Dio_ReadChannel( Dio_ChannelType ChannelId )`	
Service ID[hex]:	0x00	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	ChannelId	ID of DIO channel
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Dio_LevelType	STD_HIGH The physical level of the corresponding Pin is STD_HIGH STD_LOW The physical level of the corresponding Pin is STD_LOW
Description:	Returns the value of the specified DIO channel.	

⌋ ()

[DIO027] ⌈The `Dio_ReadChannel` function shall return the value of the specified DIO channel.

Regarding the return value of the `Dio_ReadChannel` function, the requirements [DIO083] and [DIO084] are applicable.

Furthermore, the requirements DIO005, DIO118 and DIO026 are applicable to the `Dio_ReadChannel` function.⌋ (BSW12008)

8.3.2 Dio_WriteChannel

[DIO134] ⌈

Service name:	Dio_WriteChannel
Syntax:	`void Dio_WriteChannel( Dio_ChannelType ChannelId, Dio_LevelType Level )`

Service ID[hex]:	0x01	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	ChannelId	ID of DIO channel
	Level	Value to be written
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Service to set a level of a channel.	

⌋ ()

[DIO028] ⌈If the specified channel is configured as an output channel, the `Dio_WriteChannel` function shall set the specified Level for the specified channel. ⌋ (BSW12005)

[DIO029] ⌈If the specified channel is configured as an input channel, the `Dio_WriteChannel` function shall have no influence on the physical output. ⌋ (BSW12005)

[DIO079] ⌈If the specified channel is configured as an input channel, the `Dio_WriteChannel` function shall have no influence on the result of the next Read-Service.

Furthermore, the requirements DIO005, DIO119 and DIO026 are applicable to the `Dio_WriteChannel` function. ⌋ (BSW12005)

8.3.3 Dio_ReadPort

[DIO135] ⌈

Service name:	Dio_ReadPort	
Syntax:	`Dio_PortLevelType Dio_ReadPort( Dio_PortType PortId )`	
Service ID[hex]:	0x02	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	PortId	ID of DIO Port
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Dio_PortLevelType	Level of all channels of that port
Description:	Returns the level of all channels of that port.	

⌋ ()

[DIO031] ⌈The `Dio_ReadPort` function shall return the level of all channels of that port. ⌋ (BSW12006)

[DIO104] ⌈When reading a port which is smaller than the `Dio_PortType` using the `Dio_ReadPort` function (see [DIO103]), the function shall set the bits corresponding to undefined port pins to 0.

Furthermore, the requirements DIO005, DIO118 and DIO026 are applicable to the `Dio_ReadPort` function.⌋ ()

8.3.4 Dio_WritePort

[DIO136] ⌈

Service name:	Dio_WritePort	

Syntax:	`void Dio_WritePort(`
	`  Dio_PortType PortId,`
	`  Dio_PortLevelType Level`
	`)`	

Service ID[hex]:	0x03	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	PortId	ID of DIO Port
	Level	Value to be written
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Service to set a value of the port.	

⌋ ()

[DIO034] ⌈The `Dio_WritePort` function shall set the specified value for the specified port.⌋ (BSW12003)

[DIO035] ⌈When the `Dio_WritePort` function is called, DIO Channels that are configured as input shall remain unchanged.⌋ (BSW12003)

[DIO105] ⌈When writing a port which is smaller than the `Dio_PortType` using the `Dio_WritePort` function (see [DIO103]), the function shall ignore the MSB.⌋ ()

[DIO108] ⌈The `Dio_WritePort` function shall have no effect on channels within this port which are configured as input channels.

Furthermore, the requirements DIO005, DIO119 and DIO026 are applicable to the `Dio_WritePort` function.⌋ ()

8.3.5 Dio_ReadChannelGroup

[DIO137] ⌈

Service name:	Dio_ReadChannelGroup	
Syntax:	`Dio_PortLevelType Dio_ReadChannelGroup( const Dio_ChannelGroupType* ChannelGroupIdPtr )`	
Service ID[hex]:	0x04	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	ChannelGroupIdPtr	Pointer to ChannelGroup
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Dio_PortLevelType	Level of a subset of the adjoining bits of a port
Description:	This Service reads a subset of the adjoining bits of a port.	

⌋()

[DIO037] ⌈The `Dio_ReadChannelGroup` function shall read a subset of the adjoining bits of a port (channel group). ⌋(BSW12007)

[DIO092] ⌈The `Dio_ReadChannelGroup` function shall do the masking of the channel group. ⌋(BSW12007)

[DIO093] ⌈The `Dio_ReadChannelGroup` function shall do the shifting so that the values read by the function are aligned to the LSB.

Furthermore, the requirements DIO005, DIO056, DIO083, DIO084, DIO118 and DIO026 are applicable to the `Dio_ReadChannelGroup` function. ⌋(BSW12007)

8.3.6 Dio_WriteChannelGroup

[DIO138] ⌈

Service name:	Dio_WriteChannelGroup	
Syntax:	`void Dio_WriteChannelGroup( const Dio_ChannelGroupType* ChannelGroupIdPtr, Dio_PortLevelType Level )`	
Service ID[hex]:	0x05	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	ChannelGroupIdPtr	Pointer to ChannelGroup
	Level	Value to be written
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Service to set a subset of the adjoining bits of a port to a specified level.	

⌋()

[DIO039] ⌈The `Dio_WriteChannelGroup` function shall set a subset of the adjoining bits of a port (channel group) to a specified level. ⌋(BSW12004)

[DIO040] ⌈The `Dio_WriteChannelGroup` shall not change the remaining channels of the port and channels which are configured as input. ⌋(BSW12004)

[DIO090] ⌈The `Dio_WriteChannelGroup` function shall do the masking of the channel group.⌋(BSW12004)

[DIO091] ⌈The function `Dio_WriteChannelGroup` shall do the shifting so that the values written by the function are aligned to the LSB.

Furthermore, the requirements DIO005, DIO056, DIO119 and DIO026 are applicable for the `Dio_WriteChannelGroup` function.⌋(BSW12004)

8.3.7 Dio_GetVersionInfo

[DIO139] ⌈

Service name:	Dio_GetVersionInfo	
Syntax:	`void Dio_GetVersionInfo( Std_VersionInfoType* VersionInfo )`	
Service ID[hex]:	0x12	
Sync/Async:	Synchronous	
Reentrancy:	Reentrant	
Parameters (in):	None	
Parameters (inout):	None	
Parameters (out):	VersionInfo	Pointer to where to store the version information of this module.
Return value:	None	
Description:	Service to get the version information of this module.	

⌋(BSW00411)

[DIO123] ⌈The `Dio_GetVersionInfo` function shall return the version information of this module. The version information includes:

- Module Id (See Literature [2])
- Vendor Id
- Vendor specific version numbers (BSW00407).⌋(BSW00407)

[DIO126] ⌈If source code for caller and callee is available, the module Dio should realize the function `Dio_GetVersionInfo` as a macro defined in the module's header file. ⌋()

[DIO124] ⌈The `Dio_GetVersionInfo` function shall be pre-compile time configurable (`On/Off`) by the configuration parameter `DioVersionInfoApi`.⌋ (BSW171, BSW00410)

[DIO189] ⌈If DET is enabled for the DIO Driver module, the function Dio_GetVersionInfo shall raise DIO_E_PARAM_POINTER, if the argument is NULL pointer and return without any action.

See also Chapter 10.⌋()

8.3.8 Dio_Init

[DIO165] ⌈

Service name:	Dio_Init	

Syntax:	`void Dio_Init(`
	`  const Dio_ConfigType* ConfigPtr`
	`)`	

Service ID[hex]:	0x10	
Sync/Async:	Synchronous	
Reentrancy:	Non Reentrant	
Parameters (in):	ConfigPtr	Pointer to post-build configuration data
Parameters (inout):	None	
Parameters (out):	None	
Return value:	None	
Description:	Initializes the module.	

⌋(BSW101, BSW00358, BSW00414)

[DIO166] ⌈The Dio_Init function shall initialize all global variables of the DIO module.⌋(BSW12057)

[DIO167] ⌈When development error detection is enabled for the DIO module: The function Dio_Init shall check that the parameter ConfigPtr is not NULL. If this error is detected, the function Dio_Init shall not execute the initialization but raise the development error DIO_E_PARAM_CONFIG.⌋( )

8.3.9 Dio_FlipChannel

[DIO190] ⌈

Service name:	Dio_FlipChannel

Syntax:	`Dio_LevelType Dio_FlipChannel(`
	`  Dio_ChannelType ChannelId`
	`)`

Service ID[hex]:	0x11
Sync/Async:	Synchronous
Reentrancy:	Reentrant

Parameters (in):	ChannelId	ID of DIO channel
Parameters (inout):	None	
Parameters (out):	None	
Return value:	Dio_LevelType	STD_HIGH: The physical level of the corresponding Pin is STD_HIGH. STD_LOW: The physical level of the corresponding Pin is STD_LOW.
Description:	Service to flip (change from 1 to 0 or from 0 to 1) the level of a channel and return the level of the channel after flip.	

⌋ ()

[DIO191] ⌈If the specified channel is configured as an output channel, the `Dio_FlipChannel` function shall read level of the channel (requirements [DIO083] & [DIO084] are applicable) and invert it, then write the inverted level to the channel. The return value shall be the inverted level of the specified channel. ⌋()

[DIO192] ⌈If the specified channel is configured as an input channel, the `Dio_FlipChannel` function shall have no influence on the physical output. The return value shall be the level of the specified channel. ⌋()

[DIO193] ⌈If the specified channel is configured as an input channel, the `Dio_FlipChannel` function shall have no influence on the result of the next Read-Service.

Furthermore, the requirements DIO005, DIO119 and DIO026 are applicable to the `Dio_FlipChannel` function.

See also Chapter 10. ⌋()

8.4 Call-back notifications

This chaper lists all functions provided by the Dio module to lower layers.

The Dio module does not provide any callback notifications. Callbacks related to the functionality of the Dio module are implemented in another module (ICU Driver and/or complex drivers).

8.5 Scheduled functions

This chaper lists all functions called directly by the Basic Software Module Scheduler.

The Dio module has no scheduled functions.

8.6 Expected Interfaces

This chapter lists all functions the Dio module requires from other modules.

8.6.1 Mandatory Interfaces

None

8.6.2 Optional Interfaces

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

[DIO140] ⌈

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

⌋ ()

9 Sequence diagrams

The diagrams below show the sequences when calling the `Dio_ReadChannel()` and `Dio_WriteChannel()` service. They show normal operation mode and development mode with error condition. For development mode with no error the diagrams for normal operation mode are valid. Since all other services which are defined in chapter 8.3 have exactly the same synchronous behavior concerning, there are intentionally no further sequence diagrams in this document.

9.1 Read a value from a digital I/O - 1

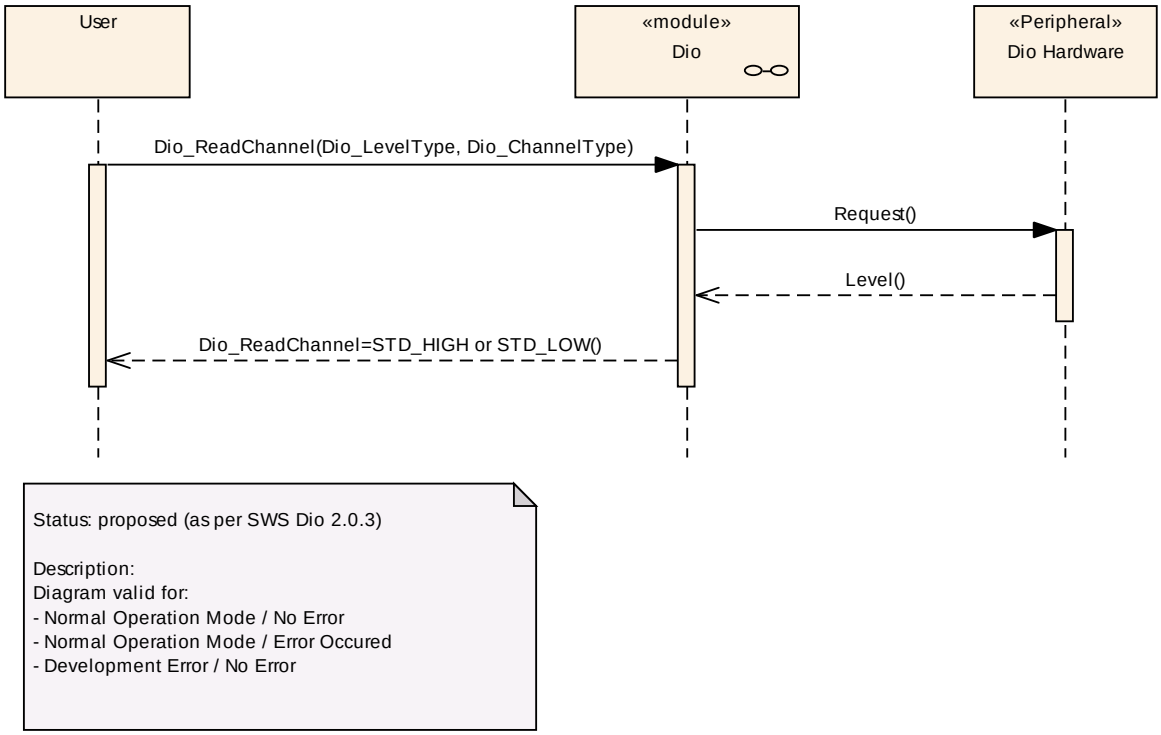

Figure 5: Read Service Sequence Chart (normal operation mode)

9.2 Read a value from a digital I/O - 2

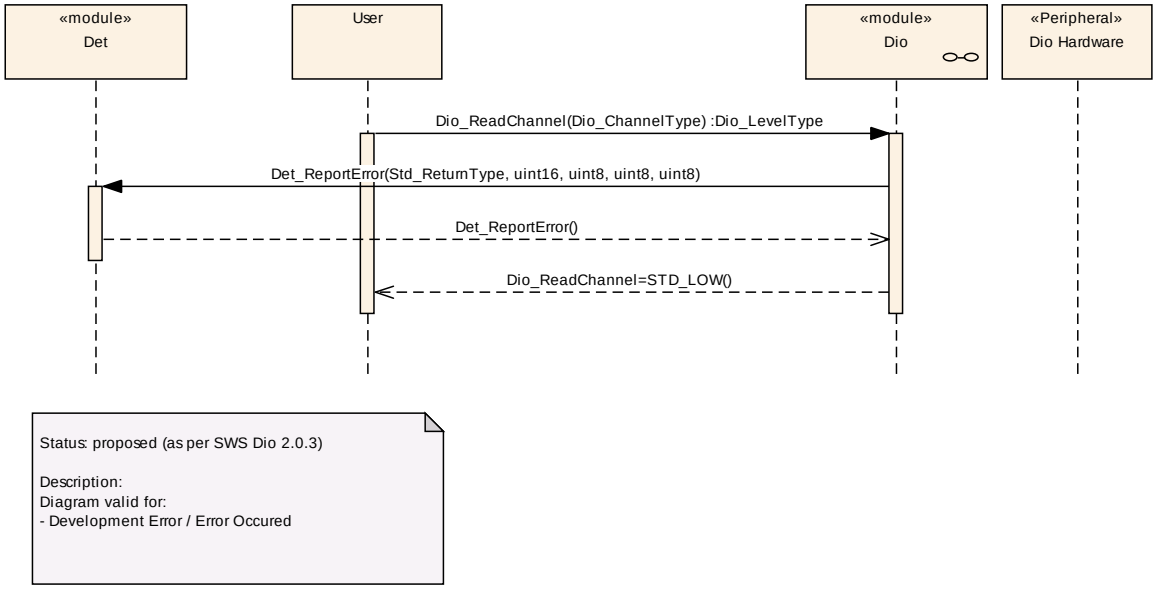

Figure 6: Read Service Sequence Chart (development error mode)

9.3 Write a value to a digital I/O - 1

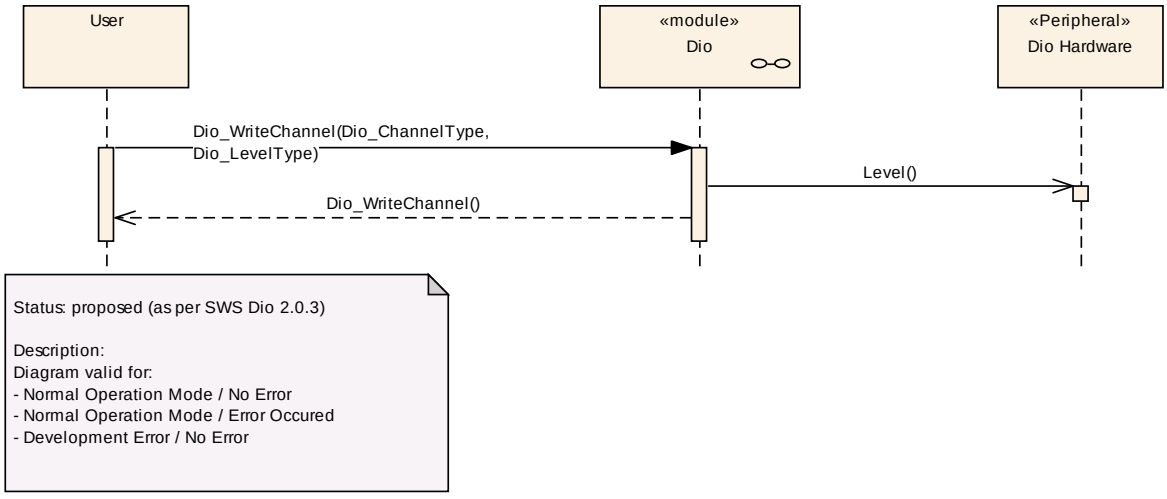

Figure 7: Write Service Sequence Chart (normal operation mode)

9.4 Write a value to a digital I/O - 2

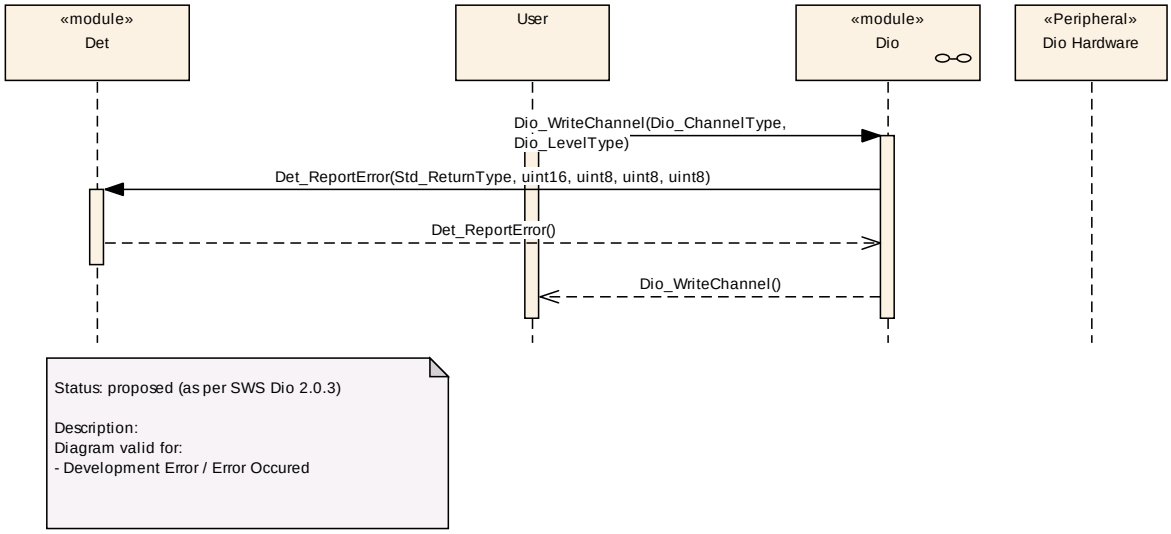

Figure 8: Write Service Sequence Chart (development error mode)

10 Configuration specification

This chapter defines configuration parameters and their clustering into containers.

10.1 Containers and configuration parameters

The following chapters summarize all configuration parameters. The detailed meanings of the parameters describe Chapters 7 and Chapter 8.

10.1.1 Variants

Configuration variants describe sets of configuration parameters:

- VARIANT-PRE-COMPILE (PC)
  Only parameters with "Pre-compile time" configuration are allowed in this variant.

- VARIANT-LINK-TIME (LT)
  Only parameters with "Pre-compile time" and "Link time" are allowed in this variant.

- VARIANT-POST-BUILD (PB)
  Parameters with "Pre-compile time", "Link time" and "Post-build time" are allowed in this variant.

[DIO129] ⌈At least one of the following variants has to be supported by implementation:
- VARIANT-PRE-COMPILE
- VARIANT-POST-BUILD⌋ ()

10.1.2 Dio

Module Name	*Dio*
Module Description	Configuration of the Dio (Digital IO) module.

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
DioConfig	1	This container contains the configuration parameters and sub containers of the AUTOSAR DIO module. This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set.
DioGeneral	1	General DIO module configuration parameters.

10.1.3 DioGeneral

SWS Item	**DIO141_Conf :**
Container Name	DioGeneral
Description	General DIO module configuration parameters.
Configuration Parameters	

SWS Item	**DIO142_Conf :**		
Name	DioDevErrorDetect {DIO_DEV_ERROR_DETECT}		
Description	Switches the Development Error Detection and Notification ON or OFF		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: Module		

SWS Item	**DIO153_Conf :**		
Name	DioFlipChannelApi {DIO_FLIP_CHANNEL_API}		
Description	Adds / removes the service Dio_FlipChannel() from the code.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: Module		

SWS Item	**DIO143_Conf :**		
Name	DioVersionInfoApi {DIO_VERSION_INFO_API}		
Description	Adds / removes the service Dio_ GetVersionInfo() from the code.		
Multiplicity	1		
Type	EcucBooleanParamDef		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: Module		

No Included Containers

10.1.4 DioPort

SWS Item	**DIO144_Conf :**
Container Name	DioPort
Description	Configuration of individual DIO ports, consisting of channels and possible channel groups. Note that this container definition does not explicitly define a symbolic name parameter. Instead, the container's short name

	will be used in the Ecu Configuration Description to specify the symbolic name of the port.
Configuration Parameters	

SWS Item	**DIO145_Conf :**		
Name	DioPortId {DIO_PORT_ID}		
Description	Numeric identifier of the DIO port. Not all MCU ports may be used for DIO, thus there may be "gaps" in the list of all IDs. This value will be assigned to the DIO port symbolic name (i.e. the SHORT-NAME of the DioPort container).		
Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 4294967295		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	--	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: Module		

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
DioChannel	0..*	Configuration of an individual DIO channel. Besides a HW specific channel name which is typically fixed for a specific micro controller, additional symbolic names can be defined per channel. Note hat this container definition does not explicitly define a symbolic name parameter. Instead, the container's short name will be used in the Ecu Configuration Description to specify the symbolic name of the channel.
DioChannelGroup	0..*	Definition and configuration of DIO channel groups. A channel group represents several adjoining DIO channels represented by a logical group. Note hat this container definition does not explicitly define a symbolic name parameter. Instead, the container's short name will be used in the Ecu Configuration Description to specify the symbolic name of the channel group.

10.1.5 DioChannel

SWS Item	**DIO146_Conf :**
Container Name	DioChannel
Description	Configuration of an individual DIO channel. Besides a HW specific channel name which is typically fixed for a specific micro controller, additional symbolic names can be defined per channel. Note hat this container definition does not explicitly define a symbolic name parameter. Instead, the container's short name will be used in the Ecu Configuration Description to specify the symbolic name of the channel.
Configuration Parameters	

SWS Item	**DIO147_Conf :**
Name	DioChannelId {DIO_CHANNEL_ID}
Description	Channel Id of the DIO channel. This value will be assigned to the symbolic names.

Multiplicity	1		
Type	EcucIntegerParamDef (Symbolic Name generated for this parameter)		
Range	0 .. 4294967295		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	--	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency	scope: Module		

No Included Containers

10.1.6 DioChannelGroup

SWS Item	**DIO148_Conf :**
Container Name	DioChannelGroup
Description	Definition and configuration of DIO channel groups. A channel group represents several adjoining DIO channels represented by a logical group. Note hat this container definition does not explicitly define a symbolic name parameter. Instead, the container's short name will be used in the Ecu Configuration Description to specify the symbolic name of the channel group.
Configuration Parameters	

SWS Item	**DIO149_Conf :**		
Name	DioChannelGroupIdentification {DIO_CHANNEL_GROUP_IDENTIFICATION}		
Description	The DIO channel group is identified in DIO API by a pointer to a data structure (of type Dio_ChannelGroupType). That data structure contains the channel group information. This parameter contains the code fragment that has to be inserted in the API call of the calling module to get the address of the variable in memory which holds the channel group information. Example values are "&MyDioGroup1" or "&MyDioGroupArray[0]"		
Multiplicity	1		
Type	EcucStringParamDef (Symbolic Name generated for this parameter)		
Default value	--		
maxLength	--		
minLength	--		
regularExpression	--		
ConfigurationClass	***Pre-compile time***	X	All Variants
	Link time	--	
	Post-build time	--	
Scope / Dependency	scope: Module		

SWS Item	**DIO150_Conf :**
Name	DioPortMask {DIO_PORT_MASK}
Description	This shall be the mask which defines the positions of the channel group. The channels shall consist of adjoining bits in the same port. The data type depends on the port width.
Multiplicity	1

Type	EcucIntegerParamDef		
Range	0 .. 4294967295		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	--	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency			

SWS Item	**DIO151_Conf :**		
Name	DioPortOffset {DIO_PORT_OFFSET}		
Description	The position of the Channel Group on the port, counted from the LSB. This value can be derived from DioPortMask. calculationFormula = Position of the first bit of DioPortMask which is set to '1' counted from LSB		
Multiplicity	1		
Type	EcucIntegerParamDef		
Range	0 .. 31		
Default value	--		
ConfigurationClass	***Pre-compile time***	X	VARIANT-PRE-COMPILE
	Link time	--	
	Post-build time	X	VARIANT-POST-BUILD
Scope / Dependency			

No Included Containers

10.1.7 DioConfig

SWS Item	**DIO152_Conf :**
Container Name	DioConfig [Multi Config Container]
Description	This container contains the configuration parameters and sub containers of the AUTOSAR DIO module. This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set.
Configuration Parameters	

Included Containers		
Container Name	***Multiplicity***	***Scope / Dependency***
DioPort	1..*	Configuration of individual DIO ports, consisting of channels and possible channel groups. Note that this container definition does not explicitly define a symbolic name parameter. Instead, the container's short name will be used in the Ecu Configuration Description to specify the symbolic name of the port.

10.2 Published Information

[DIO195] ⌈The standardized common published parameters as required by BSW00402 in the General Requirements on Basic Software Modules [4] shall be published within the header file of this module and need to be provided in the BSW Module Description. The according module abbreviation can be found in the List of Basic Software Modules [2].⌋ ( )

Additional module-specific published parameters are listed below if applicable.

10.3 Configuration Example

This chapter shall provide a better understanding of how and where configuration parameters are defined and used.

Use Cases:

1. Configuration of a DIO channel
2. Configuration of a DIO port
3. Configuration of a DIO channel group

10.3.1 Generation of DIO configuration data

10.3.1.1 Configuration of a DIO channel

Each channel with index of type `Dio_ChannelType` shall be referenced via symbolic names through the file Dio_Cfg.h.
Example:

```
#define MOTOR_START_STOP (DIO_CHANNEL_A_5)
#define MOTOR_DIRECTION (DIO_CHANNEL_A_6)
```

Where `DIO_CHANNEL_A_5` and `DIO_CHANNEL_A_6` may be defined in a derivative or board specific header file.
The mapping shall be done implementation specific.

10.3.1.2 Configuration of a DIO port

Each port with index of type `Dio_PortType` shall be referenced via symbolic names through the file Dio_Cfg.h.

Example:

```
#define MOTOR_CTL_PORT (DIO_PORT_A)
#define MUX_SEL_PORT (DIO_PORT_B)
```

Where DIO_PORT_A and DIO_PORT_B may be defined in a derivative or board specific header file.
The mapping shall be done implementation specific.

10.3.1.3 Configuration of a DIO channel group

Each channel group which is of type Dio_ChannelGroupType shall be referenced via symbolic names through the file Dio_Cfg.h.

Example:

```
#define MOTOR_CTL_GRP_PTR (&DioConfigData[0])
#define MUX_SEL_GRP_PTR (&DioConfigData[1])
```

For description of DioConfigData see section 10.3.2.

10.3.2 Instantiation of DIO configuration data

The file that contains the instantiation (=definition) of the DIO configuration structure includes Dio_Cfg.h and uses the defined values for initialization of structure elements. The filename should be Dio_Lcfg.c (BSW00346).

Example:

```
const Dio_ChannelGroupType DioConfigData[2] =
{
   {
      port    = MOTOR_CTL_PORT,
      offset  = 5,
      mask    = 0x60,
   },
   {
      port    = MUX_SEL_PORT,
      offset  = 1,
      mask    = 0x1E,
   }
};
```

11 Not applicable requirements

[DIO195] ⌈These requirements are not applicable to this specification.⌋ (BSW005, BSW006, BSW007, BSW009, BSW010, BSW160, BSW161, BSW162, BSW164, BSW167, BSW168, BSW170, BSW172, BSW00304, BSW00306, BSW00307, BSW00308, BSW00309, BSW00314, BSW00321, BSW00325, BSW00326, BSW00328, BSW00329, BSW00330, BSW00331, BSW00333, BSW00334, BSW00335, BSW00336, BSW00339, BSW00341, BSW00342, BSW00343, BSW00347, BSW00355, BSW00357, BSW00359, BSW00360, BSW00369, BSW00370, BSW00371, BSW00373, BSW00375, BSW00376, BSW00377, BSW00378, BSW00382, BSW00384, BSW00387, BSW00399, BSW00400, BSW00404, BSW00405, BSW00406, BSW00413, BSW00416, BSW00417, BSW00420, BSW00422, BSW00423, BSW00424, BSW00425, BSW00426, BSW00427, BSW00428, BSW00429, BSW00431, BSW00432, BSW00433, BSW00434, BSW157, BSW12063, BSW12067, BSW12068, BSW12069, BSW12075, BSW12077, BSW12078, BSW12092, BSW12129, BSW12169, BSW12265, BSW12267)