MCAL User Manual for Dio

32-bit TriCore^{TM} AURIX^{TM} TC3xx microcontroller

About this document

Scope and purpose

This User Manual is intended to enable users to integrate the Microcontroller Abstraction Layer (MCAL) software for the TriCore^{TM} AURIX^{TM} family of 32-bit microcontrollers.

This document describes responsibilities of integrator in-charge of integrating MCAL software with the basic software (BSW) stack. This document also provides detailed information on safety, configuration and functions along with examples of usage of significant features.

Note: Detailed information about package installation, safety and other generic information that are common across all modules are provided in MCAL User Manual General.

Intended audience

This document is intended for anyone using the Dio module of the TC3xx MCAL software.

Document conventions

Table 1 Conventions

Convention	Explanation
Bold	Emphasizes heading levels, column headings, table and figure captions, screen names, windows, dialog boxes, menus, sub-menus
Italics	Denotes variable(s) and reference(s)
`Courier New`	Denotes APIs, functions, interrupt handlers, events, data types, error handlers, file/folder names, directories, command line inputs, code snippets
>	Indicates that a cascading sub-menu opens when you select a menu item
[cover parentID=<alpha numeric value>]	Used for traceability completeness. Reader should ignore these.

Reference documents

This User Manual should be read in conjunction with the following documents:

- AURIX^{TM} TC3xx MCAL User Manual General
- Specification of DIO Driver, AUTOSAR_SWS_DIO_Driver, AUTOSAR Release 4.2.2
- Specification of DIO Driver, AUTOSAR_SWS_DIO_Driver, AUTOSAR Release 4.4.0

Table of contents

Table of contents

Table of contents

1 DIO driver

1.1 User information

1.1.1 Description

The DIO driver uses the port peripheral. The usage responsibility of the port peripheral is split by AUTOSAR into two modules. The PORT driver configures and sets the properties of port pin. The DIO driver reads or writes to the port pin .The DIO driver provides, port, channel and channel group based read and write access to the internal general purpose IO ports. All read and write services in the DIO driver are not buffered. Channel refers to individual general purpose IO pin, port refers to DIO channels that are grouped by the hardware, and channel group refers to the formal logical combination of several adjoining dio channels represented by a logical group. Note that a DIO channel group should belong to one DIO port.

1.1.2 Hardware-software mapping

This section describes the system view of the DIO driver and the peripherals administered by it.

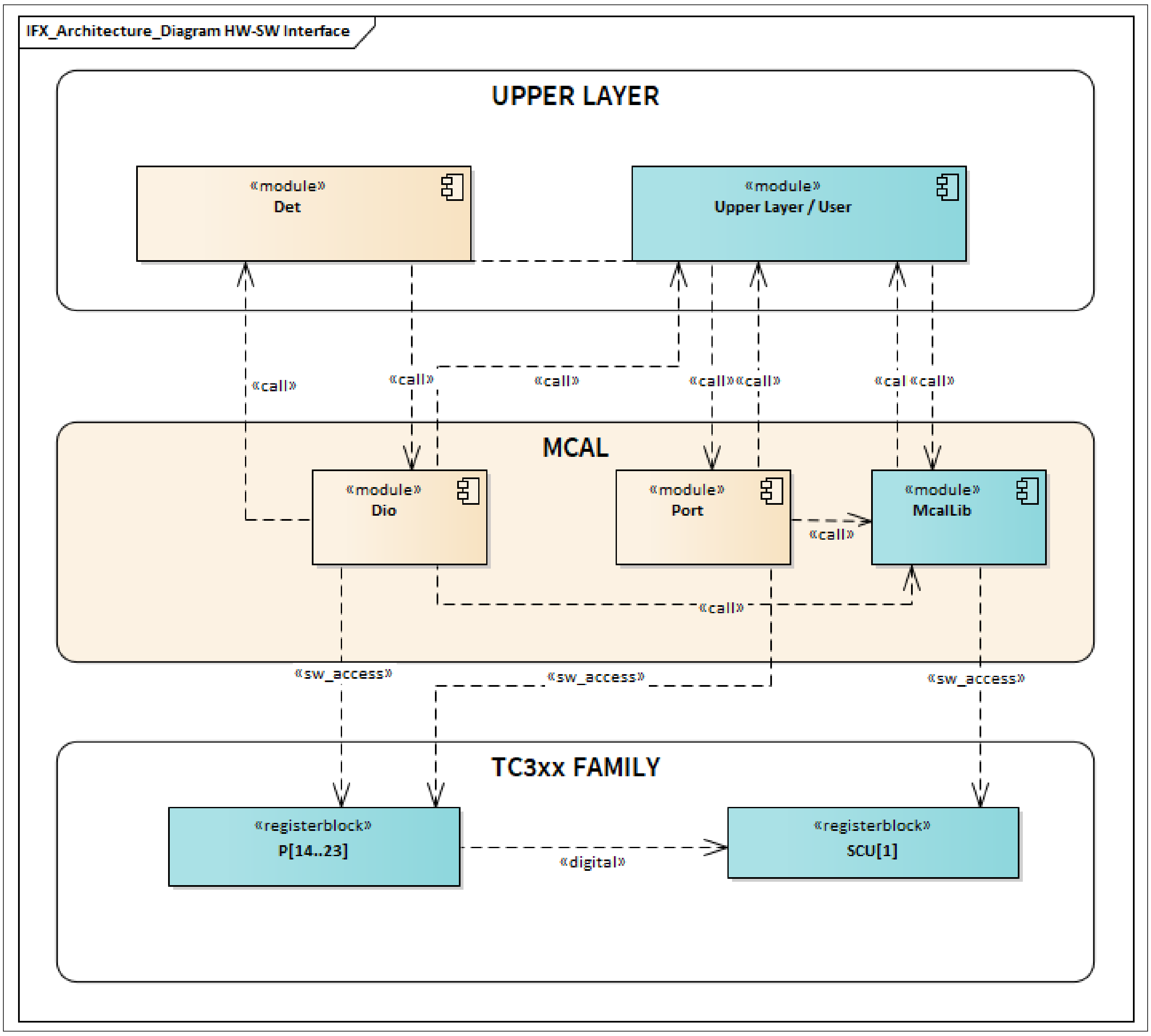

Figure 1 **Mapping of hardware-software interfaces**

1.1.2.1 Port: primary hardware peripheral.

Hardware functional features

The DIO driver is used for read and write access to the internal general purpose IO ports.

The key hardware functional features used by the driver are:

- Set, clear and toggle a portpin through the Pn_OUT and Pn_OMR register.

The unsupported features of the DIO (since these are configured by the PORT driver) are:

- LVDS pad control
- Emergency stop
- Function decision control
- Controller selection

- Access enable
- Drive mode

Users of the hardware

The PORT driver performs the configuration for port pins .The DIO driver performs input and output operation on the configured ports, therefore there is no conflict with the PORT driver. The user shall ensure that the port pins used by the other MCAL drivers are not conflicting with the DIO driver.

Hardware diagnostic features

Not applicable.

Hardware events

Not applicable.

1.1.2.2 SCU: Dependent Hardware peripheral

Hardware functional features

The DIO driver depends on the SCU IP for the clock, ENDINIT and reset functionalities.

The driver requires the SPB clock signals for functioning.

Users of the Hardware

The SCU IP supplies clock for all the peripherals and the MCU driver is responsible for configuring the clocktree. To avoid conflict due to simultaneous writes, update to all the ENDINIT protected registers are performed using the MCALLIB APIs.

Hardware diagnostic features

The SMU alarms configured for the SCU IP are not monitored by the DIO driver.

Hardware events

Hardware events from the SCU are not used by the DIO driver.

1.1.3 File structure

1.1.3.1 C file structure

This section provides details of the C files of the DIO driver.

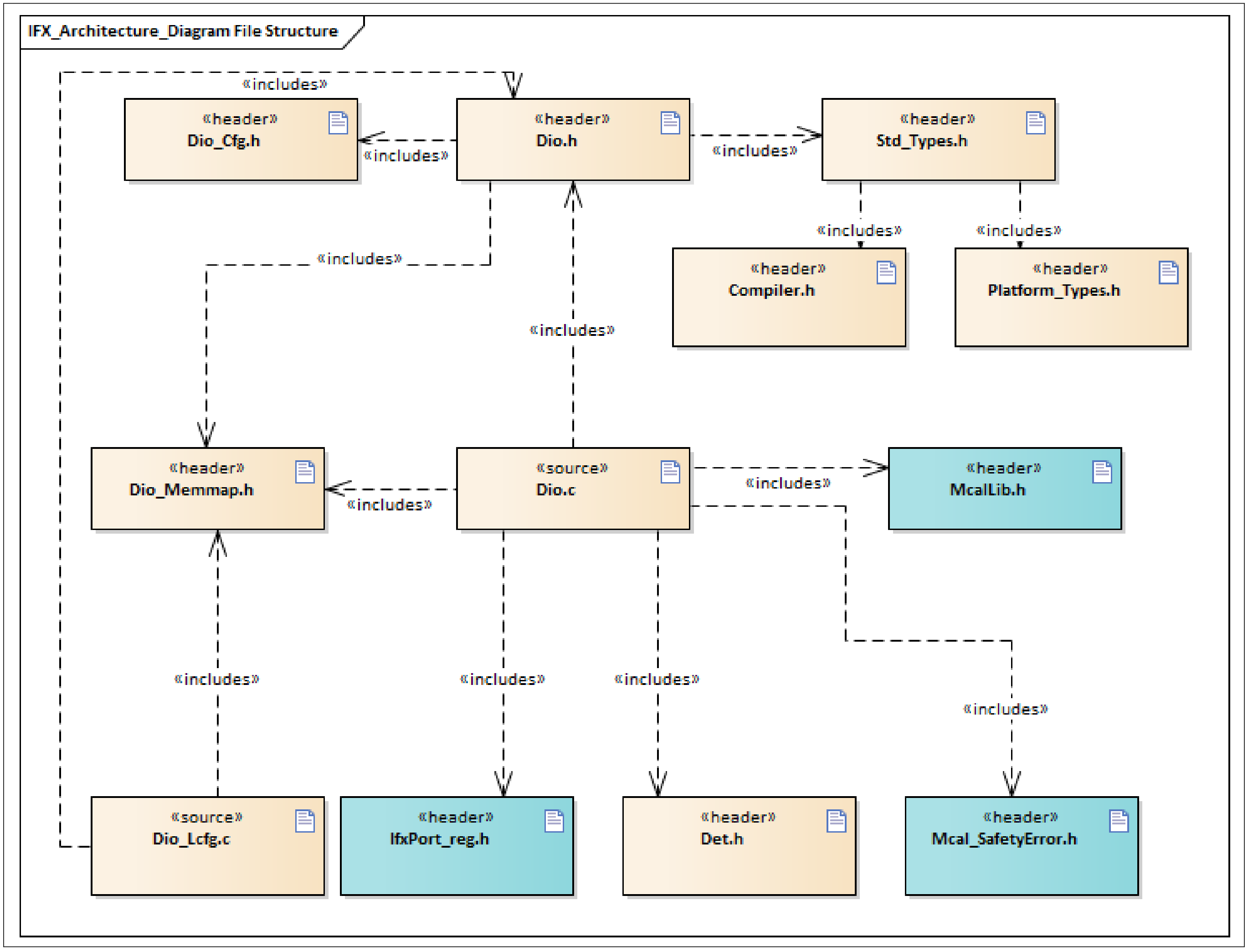

Figure 2 Dio_File_Structure-1.png

Table 2 C file structure

File name	Description
`Compiler.h`	Provides abstraction from compiler-specific keywords
`Det.h`	Provides the exported interfaces of Development Error Tracer
`Dio.c`	File (Static) containing implementation of APIs
`Dio.h`	Header file (Static) defining prototypes of data structures and APIs
`Dio_Cfg.h`	Header file (Generated) containing constants, symbolic names and pre-processor macros.
`Dio_Lcfg.c`	File (Generated) containing objects to data structures
`Dio_Memmap.h`	File (Static) containing the memory section definitions used by the DIO driver
`IfxPort_reg.h`	SFR header file for Port
`McalLib.h`	Static header file defining prototypes of data structure and APIs exported by the MCALLIB.
`Mcal_SafetyError.h`	Header file containing the prototype of the API for reporting safety-related errors

Table 2 C file structure (continued)

File name	Description
`Platform_Types.h`	Platform-specific type declaration file as defined by AUTOSAR
`Std_Types.h`	Standard type declaration file as defined by AUTOSAR. It is independent of compiler or platform.

1.1.3.2 Code generator plugin files

This section provides details of the code generator plugin files of the DIO driver.

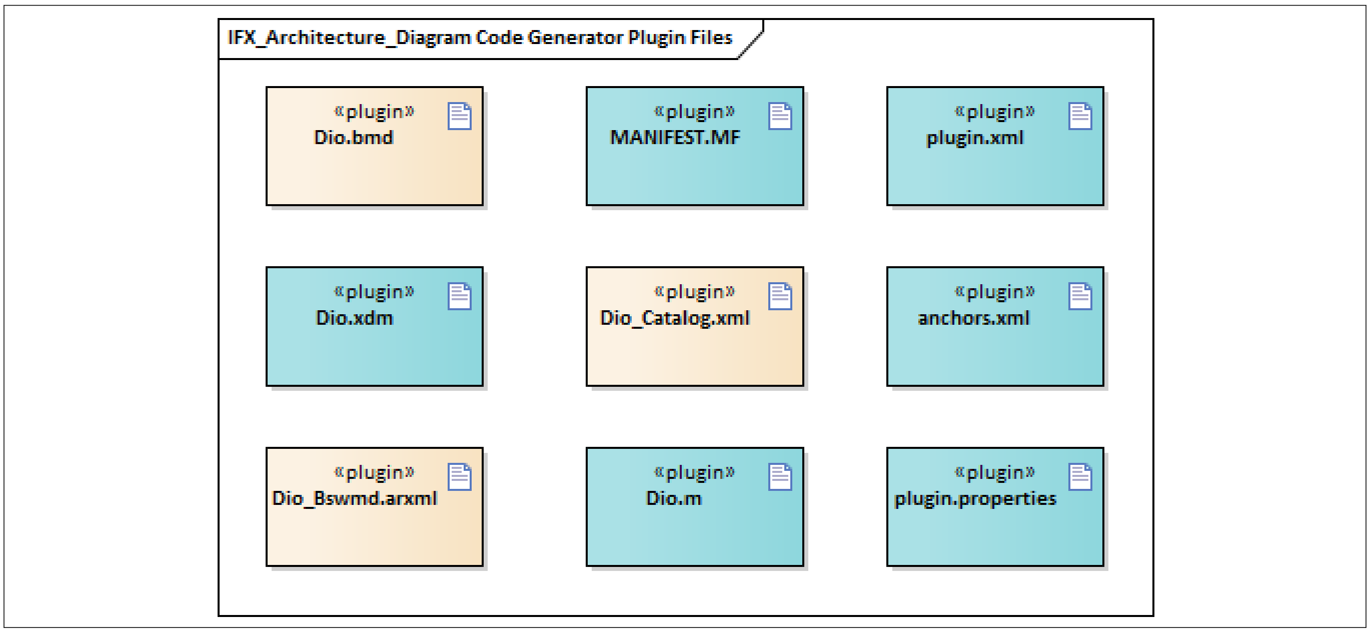

Figure 3 Dio_Code_Generator_Plugin_Files-1.png

Table 3 Code generator plugin files

File name	Description
`Dio.bmd`	AUTOSAR format XML data model schema file(for each device)
`Dio.m`	Code template macro file for DIO driver
`Dio.xdm`	Tresos format XML data model schema file
`Dio_Bswmd.arxml`	AUTOSAR format module description file
`Dio_Catalog.xml`	AUTOSAR format catalog file as per catalog_V3_0_0.ml.xsd
`MANIFEST.MF`	Tresos plugin support file containing the metadata for DIO driver
`anchors.xml`	Tresos anchors support file for the DIO driver
`plugin.properties`	Tresos plugin support file for the DIO driver
`plugin.xml`	Tresos plugin support file for the DIO driver

1.1.4 Integration hints

This section lists the key points that an integrator or user of the DIO driver must consider.

1.1.4.1 Intergration with AUTOSAR stack

- **EcuM**
  The EcuM module is not required for the integrating the DIO driver.
- **Memory mapping**
  Memory mapping is a concept from AUTOSAR that allows relocation of text, variables, constants and configuration data to user specific memory regions. To achieve this, all the relocatable elements of the driver are en-capsulated in different memory section macros. These macros are defined in the `Dio_MemMap.h` file.
  The `Dio_MemMap.h`file is provided in the MCAL package as a stub code. The integrator must place appropriate compiler pragmas within the memory-section macros. The pragmas ensure that the elements are re-located to the correct memory region. A sample implementation listing the memory-section macros is shown as follows:

```
/**** CONST DATA -- ****/
#if defined DIO_START_SEC_CONST_ASIL_B_GLOBAL_16
/***** User pragmas here *****/
#undef DIO_START_SEC_CONST_ASIL_B_GLOBAL_16
#undef MEMMAP_ERROR

#elif defined DIO_STOP_SEC_CONST_ASIL_B_GLOBAL_16
/***** User pragmas here *****/
#undef DIO_STOP_SEC_CONST_ASIL_B_GLOBAL_16
#undef MEMMAP_ERROR


/****** CONFIG DATA *****/
#elif defined DIO_START_SEC_CONFIG_DATA_ASIL_B_GLOBAL_UNSPECIFIED
/***** User pragmas here *****/
#undef DIO_START_SEC_CONFIG_DATA_ASIL_B_GLOBAL_UNSPECIFIED
#undef MEMMAP_ERROR
#elif defined DIO_STOP_SEC_CONFIG_DATA_ASIL_B_GLOBAL_UNSPECIFIED
/***** User pragmas here *****/
#undef DIO_STOP_SEC_CONFIG_DATA_ASIL_B_GLOBAL_UNSPECIFIED
#undef MEMMAP_ERROR

/***** CODE DATA ****/
#elif defined DIO_START_SEC_CODE_ASIL_B_GLOBAL
/***** User pragmas here *****/
#undef DIO_START_SEC_CODE_ASIL_B_GLOBAL
#undef MEMMAP_ERROR
#elif defined DIO_STOP_SEC_CODE_ASIL_B_GLOBAL
/***** User pragmas here *****/
#undef DIO_STOP_SEC_CODE_ASIL_B_GLOBAL
#undef MEMMAP_ERROR

#endif

#if defined MEMMAP_ERROR
#error "Dio_MemMap.h, wrong pragma command"
#endif
```

- **DET**

  The DET module is a part of the AUTOSAR stack that handles all the development and runtime errors reported by the BSW modules. The DIO driver reports all the development errors to the DET module through the `Det_ReportError()` API. The user of the DIO driver must process all the errors reported to the DET module through the `Det_ReportError()` API.

  The `Det.h` and `Det.c` files are provided in the MCAL package as a stub code and needs to be replaced with a complete DET module during the integration phase.
- **DEM**

  The DEM module is not required for the integration of the DIO driver.
- **SchM**

  The SchM is not required for the integration of the DIO driver.
- **Safety Error**

  The DIO driver will report all the detected safety errors through the API `Mcal_ReportSafetyError()`.

  The driver performs only detection and reporting of the safety errors. The handling of the reported errors shall be done by the user. The `Mcal_ReportSafetyError()` API is provided in the `Mcal_SafetyError.c` and `Mcal_SafetyError.h` files as a stub code, and must be updated by the integrator to handle the reported errors.

  Note: All DET errors are also reported as safety errors (error code used is same as DET).
- **Notifications and callbacks:**

  The DIO driver does not provide any call-backs or notifications.

1.1.4.2 Multicore and Resource Manager

The DIO driver supports the multicore functionality. The DIO driver service can be accessed from any core.

1.1.4.3 MCU support

The DIO driver does not use any services provided by the MCU driver.

1.1.4.4 Port support

The PORT driver configures the port pins of the entire microcontroller. The user must configure port pins used by the DIO driver through the PORT configuration and initialize the port pins prior to invoking the DIO APIs.

1.1.4.5 DMA support

The DIO driver does not use any services provided by the DMA driver.

1.1.4.6 Interrupt connections

The DIO driver does not use any interrupt source.

1.1.4.7 Example usage

DIO driver published symbolic names

The DIO channel and DIO port symbolic names are defined in the `Dio_Cfg.h` (derivative or board specific header file).

Configuration of DIO Channel

The symbolic names for DIO channels is generated as follows. These symbolic names are of type `Dio_ChannelType`.

Example for DIO channel configuration

```
/* User Defined Symbolic Names for the DIO CHANNELS */
 #define DioConf_DioChannel_MOTOR_START_STOP (DIO_CHANNEL_0_5)
 #define DioConf_DioChannel_MOTOR_DIRECTION (DIO_CHANNEL_0_8)
 #define DioConf_DioChannel_CAN_TRCV_ENT0 (DIO_CHANNEL_1_1)
 #define DioConf_DioChannel_CAN_TRCV_NSTB0 (DIO_CHANNEL_1_2)
```

Configuration of DIO Port

The symbolic names for DIO port is generated as follows. These symbolic names are of type `Dio_PortType`.

Example for DIO port configuration

```
/* User Defined Symbolic Names for the DIO PORTS */
 #define DioConf_DioPort_MOTOR_CTL_PORT (DIO_PORT_0)
 #define DioConf_DioPort_CAN_TRCV_PORT (DIO_PORT_1)
```

Configuration of DIO Channel Group

The symbolic names for DIO channel group is generated as follows. These symbolic names are of type`Dio_ChannelGroupType`.

Example for DIO channel group configuration

```
/* User Defined Symbolic Names for the DIO CHANNEL GROUPS */
 #define DioConf_DioChannelGroup_MOTOR_CTL_GRP
(&Dio_Config.Dio_ChannelGroupConfigPtr[0])
 #define DioConf_DioChannelGroup_CAN_TRCV_GRP
(&Dio_Config.Dio_ChannelGroupConfigPtr[1])
```

Using the APIs

The following code listing shows example calls to different services provided by the DIO driver. This code listing uses symbols as described earlier.

Using of DIO driver services

```
Dio_levelType ChannelVal;
Dio_PortLevelType PortVal;
Dio_PortLevelType ChannelGrpVal;

 /* Set level STD_HIGH for port 0 channel 5 */
 Dio_WriteChannel(DioConf_DioChannel_MOTOR_START_STOP, STD_HIGH);

 /* Read level of port 0 channel 8 */
 ChannelVal = Dio_ReadChannel(DioConf_DioChannel_MOTOR_DIRECTION);

 /* Write port 1 with all pins set to HIGH */
 Dio_WritePort(DioConf_DioPort_CAN_TRCV_PORT, (Dio_PortLevelType)0x7FFF);

 /* Read the level of all the pins of port 0 */
 PortVal = Dio_ReadPort(DioConf_DioPort_MOTOR_CTL_PORT);

 /* Write to channel group 0 */
 Dio_WriteChannelGroup(DioConf_DioChannelGroup_MOTOR_CTL_GRP,
(Dio_PortLevelType)0xA);

 /* Read from channel group 1 */
 ChannelGrpVal = Dio_ReadChannelGroup(DioConf_DioChannelGroup_CAN_TRCV_GRP);
```

1.1.5 Key architectural considerations

1.1.5.1 Implementation Type

The DIO driver is implemented as Variant Link Time.

1.1.5.2 User mode support

The DIO driver operates both in User-1 and Supervisory mode without the need of any configuration parameter to configure the behaviour.

1.2 Assumptions of Use (AoU)

The AoU for the Dio driver are as follows.

- **Configuration Check**

The user should ensure that the generated configuration is correct against the GUI configuration.
[cover parentID DIO={A4C58AA6-0186-47d1-810A-13AE19E45737}]

- **Dio Flip Channel**

Due to the configured pin drive strength and load capacitance connected to the pin, there is a delayed response on the pin to flip. After the call to Dio_FlipChannel() API, the user shall read the pin level using Dio_ReadChannel() API after the necessary delay and confirm the flipped level of the pin. For the delay information refer the datasheet. (Rise / Fall time)

Affected APIs: Dio_FlipChannel

[cover parentID DIO={50AE62EA-7A3B-421a-A9F5-1595EAFE62DD}]

- **Dio Readonly Usage**

The user should ensure that the DIO driver is not used on the analog pins.
[cover parentID DIO={EEBBE858-7E80-40bb-92B2-DA4D61CA9257}]

- **Dio Write Verification**

The user should perform read operation after each write operation to ensure realization of desired operations.
[cover parentID DIO={A62F0251-C5CC-4b25-B83A-AD9F504F62F6}]

- **Port Init Check**

The DIO driver needs PORT driver to be initialized prior to use of the DIO driver API's, therefore the Port_InitCheck (AoU) shall be performed by the integrator to check initialization of PORT driver as DIO driver works on pins and ports which are configured by the PORT driver.
[cover parentID DIO={A2AE117E-4BCF-46c2-9F85-3E871ABDF72F}]

1.3 Reference information

1.3.1 Configuration interfaces

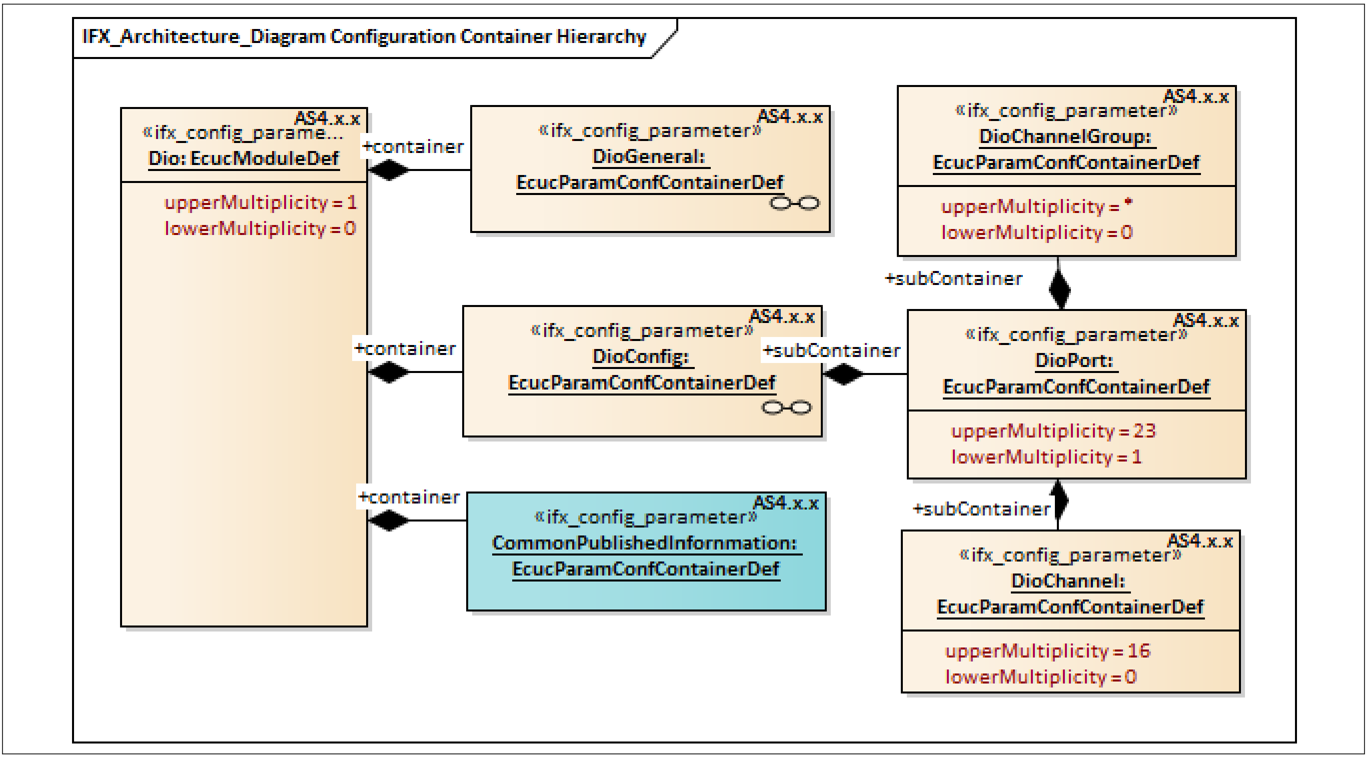

Figure 4 Container hierarchy along with their configuration parameters

1.3.1.1 Container: CommonPublishedInfornmation

This section describes the information about the module published by the Dio Driver.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.1.1 ArMajorVersion

Table 4 Specification for ArMajorVersion

Name	`ArMajorVersion`		
Description	This parameter provides the major version of the AUTOSAR Specification.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	4		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-

Table 4 Specification for ArMajorVersion (continued)

Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.2 ArMinorVersion

Table 5 Specification for ArMinorVersion

Name	`ArMinorVersion`		
Description	This parameter provides the minor version of the AUTOSAR Specification.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	As per AUTOSAR version		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.3 ArPatchVersion

Table 6 Specification for ArPatchVersion

Name	`ArPatchVersion`		
Description	This parameter provides the patch version of the AUTOSAR Specification.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	As per AUTOSAR version		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.4 ModuleId

Table 7 Specification for ModuleId

Name	`ModuleId`		
Description	Module ID of DIO		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 65535		
Default value	120		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.5 Release

Table 8 Specification for Release

Name	`Release`		
Description	Aurix derivative used for the implementation.		
Multiplicity	1..1	**Type**	EcucStringParamDef
Range	String		
Default value	As per Hardware derivative		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.6 SwMajorVersion

Table 9 Specification for SwMajorVersion

Name	`SwMajorVersion`
Description	This parameter provides the major version of the Software.

Table 9 Specification for SwMajorVersion (continued)

Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	As per driver version		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.7 SwMinorVersion

Table 10 Specification for SwMinorVersion

Name	`SwMinorVersion`		
Description	This parameter provides the minor version of the Software.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	As per driver version		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.8 SwPatchVersion

Table 11 Specification for SwPatchVersion

Name	`SwPatchVersion`		
Description	This parameter provides the patch version of the Software.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 255		
Default value	As per driver version		

Table 11 **Specification for SwPatchVersion (continued)**

Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.1.9 VendorID

Table 12 **Specification for VendorID**

Name	`VendorID`		
Description	This parameter provides the Vendor Id		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 65535		
Default value	17		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Published-Information	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.2 Container: Dio

Configuration of the Dio (Digital IO) module.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.2.1 Config Variant

Table 13 **Specification for Config Variant**

Name	`Config Variant`		
Description	None		
Multiplicity	1..1	**Type**	EcucEnumerationParamDef

Table 13 Specification for Config Variant (continued)

Range	Variant LinkTime: Only parameters with "Pre-compile time" and "Link time" are allowed in this variant.		
Default value	Variant LinkTime		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.3 Container: DioChannel

Configuration of an individual DIO channel.

Post-Build Variant Multiplicity: FALSE

Multiplicity Configuration Class: Pre-Compile

1.3.1.3.1 DioChannelEcucPartitionRef

Table 14 Specification for DioChannelEcucPartitionRef

Name	`DioChannelEcucPartitionRef`		
Description	Maps a DIO channel to zero or multiple ECUC partitions. The ECUC partitions referenced are a subset of the ECUC partitions where the related DIO port is mapped to.		
Multiplicity	1..1	**Type**	EcucReferenceDef
Range	Reference to Node: EcucPartition		
Default value	NULL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	TRUE
Value configuration class	Pre-Compile	**Multiplicity configuration class**	Pre-Compile
Origin	AUTOSAR_ECUC	**Scope**	ECU
Dependency	-		
Autosar Version	Applicable for Autosar version 4.4.0.		

1.3.1.3.2 DioChannelId

Table 15 Specification for DioChannelId

Name	`DioChannelId`		
Description	Channel Id of the DIO channel. This value will be assigned to the symbolic names and consecutive value is calculated for each new channel Id.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 15		
Default value	0		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	ECU
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4 Container: DioChannelGroup

Definition and configuration of DIO channel groups. A channel group represents several adjoining DIO channels represented by a logical group. 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 channel group.

Post-Build Variant Multiplicity: FALSE

Multiplicity Configuration Class: Link-Time

1.3.1.4.1 DioChannelGroupEcucPartitionRef

Table 16 Specification for DioChannelGroupEcucPartitionRef

Name	`DioChannelGroupEcucPartitionRef`		
Description	Parameter support is added only for AUTOSAR schema compliance, this parameter is not used in code generation logic, hence this parameter is made editable false.		
Multiplicity	1..1	**Type**	EcucReferenceDef
Range	Reference to Node: EcucPartition		
Default value	NULL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	TRUE
Value configuration class	Pre-Compile	**Multiplicity configuration class**	Pre-Compile
Origin	AUTOSAR_ECUC	**Scope**	ECU

Table 16 Specification for DioChannelGroupEcucPartitionRef (continued)

Dependency	-		
Autosar Version	Applicable for Autosar version 4.4.0.		

1.3.1.4.2 DioChannelGroupIdentification

Table 17 Specification for DioChannelGroupIdentification

Name	`DioChannelGroupIdentification`		
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 to the calling module to get the address of the variable in memory which holds the channel group information.		
Multiplicity	1..1	**Type**	EcucStringParamDef
Range	String		
Default value	0		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	ECU
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.3 DioPortMask

Table 18 Specification for DioPortMask

Name	`DioPortMask`		
Description	This should be the mask which defines the positions of the channel group. The channels should consist of adjoining bits in the same port. The data type depends on the port width.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 65535		
Default value	0		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Link-Time	**Multiplicity configuration class**	-

Table 18 **Specification for DioPortMask (continued)**

Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.4.4 DioPortOffset

Table 19 **Specification for DioPortOffset**

Name	`DioPortOffset`		
Description	The position of the Channel Group on the port counted from the LSB. This value can be derived from DioPortMask. Calculation Formula = Position of the first bit of DioPortMask which is set to '1' counted from LSB.		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 15		
Default value	0		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Link-Time	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.5 Container: DioConfig

This container contains the configuration parameters and sub containers of the AUTOSAR DIO module.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.6 Container: DioGeneral

General DIO module configuration parameters.

Post-Build Variant Multiplicity: -

Multiplicity Configuration Class: -

1.3.1.6.1 DioDevErrorDetect

Table 20 **Specification for DioDevErrorDetect**

Name	`DioDevErrorDetect`

Table 20 Specification for DioDevErrorDetect (continued)

Description	Switches the Default Error Tracer detection and notification ON or OFF. True: ON. False: OFF.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.2 DioEcucPartitionRef

Table 21 Specification for DioEcucPartitionRef

Name	`DioEcucPartitionRef`		
Description	Parameter support is added only for AUTOSAR schema compliance, this parameter is not used in code generation logic, hence this parameter is made editable false.		
Multiplicity	1..1	**Type**	EcucReferenceDef
Range	Reference to Node: EcucPartition		
Default value	NULL		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	ECU
Dependency	-		
Autosar Version	Applicable for Autosar version 4.4.0.		

1.3.1.6.3 DioFlipChannelApi

Table 22 Specification for DioFlipChannelApi

Name	`DioFlipChannelApi`		
Description	Switch to Adds / Removes the service of Dio_FlipChannel() from the code.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.4 DioMaskedWritePortApi

Table 23 Specification for DioMaskedWritePortApi

Name	`DioMaskedWritePortApi`		
Description	Switch to Adds / Removes the service of Dio_MaskedWritePort Api from the code.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX FOR AS4.2.2 VARIANT AND AUTOSAR_ECUC FOR AS4.4.0 VARIANT	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.5 DioSafetyEnable

Table 24 Specification for DioSafetyEnable

Name	`DioSafetyEnable`		
Description	Switch to enable reporting of safety Errors (Range and plausibility check).		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	TRUE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	IFX	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.6.6 DioVersionInfoApi

Table 25 Specification for DioVersionInfoApi

Name	`DioVersionInfoApi`		
Description	Switch for enabling the API Dio_GetVersionInfo() which returns the version information of the module.		
Multiplicity	1..1	**Type**	EcucBooleanParamDef
Range	TRUE FALSE		
Default value	FALSE		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	LOCAL
Dependency	-		
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.		

1.3.1.7 Container: DioPort

The 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.

Post-Build Variant Multiplicity: FALSE

Multiplicity Configuration Class: Link-Time

1.3.1.7.1 DioPortEcucPartitionRef

Table 26 Specification for DioPortEcucPartitionRef

Name	`DioPortEcucPartitionRef`		
Description	Parameter support is added only for AUTOSAR schema compliance, this parameter is not used in code generation logic, hence this parameter is made editable false.		
Multiplicity	1..1	**Type**	EcucReferenceDef
Range	Reference to Node: EcucPartition		
Default value	NULL		
Post-build variant value	TRUE	**Post-build variant multiplicity**	TRUE
Value configuration class	Pre-Compile	**Multiplicity configuration class**	Pre-Compile
Origin	AUTOSAR_ECUC	**Scope**	ECU
Dependency	-		
Autosar Version	Applicable for Autosar version 4.4.0.		

1.3.1.7.2 DioPortId

Table 27 Specification for DioPortId

Name	`DioPortId`		
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 PORTIDs. This value will be assigned to the DIO port symbolic name (i.e. the SHORT-NAME of the DioPort container).		
Multiplicity	1..1	**Type**	EcucIntegerParamDef
Range	0 - 41		
Default value	0		
Post-build variant value	FALSE	**Post-build variant multiplicity**	-
Value configuration class	Pre-Compile	**Multiplicity configuration class**	-
Origin	AUTOSAR_ECUC	**Scope**	ECU

Table 27 Specification for DioPortId (continued)

Dependency	-
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.2 Functions - Type definitions

This section lists all the data type of the DIO driver.

1.3.2.1 Dio_ChannelGroupType

Table 28 Specification for Dio_ChannelGroupType

Syntax	`Dio_ChannelGroupType`	
Type	Structure	
File	`Dio.h`	
Range	uint16 Mask	This element mask which defines the positions of the channel group. Range: 0x0 - 0xFFFF
	uint8 Offset	This element must be the position of the Channel Group on the port, counted from the LSB. Range: 0 - 15
	Dio_PortType Port	This should be the port on which the Channel group is defined. Range: Refer Data Type
Description	Type for the definition of a channel group, which consists of several adjoining channels within a port.	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.2 Dio_ChannelType

Table 29 Specification for Dio_ChannelType

Syntax	`Dio_ChannelType`	
Type	uint16	
File	`Dio.h`	
Range	0 to Number of channels available	Number of Channels in a port
Description	Numeric ID of a DIO channel	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.3 Dio_ConfigType

Table 30 Specification for Dio_ConfigType

Syntax	`Dio_ConfigType`
Type	Structure
File	`Dio.h`
Description	Defines the type for data structure containing the set of configuration parameters required for initializing the DIO driver.
Source	AUTOSAR
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.2.4 Dio_LevelType

Table 31 Specification for Dio_LevelType

Syntax	`Dio_LevelType`	
Type	uint8	
File	`Dio.h`	
Range	0x00	STD_LOW Physical state 0V
	0x01	STD_HIGH Physical state 5V or 3.3V
Description	These are the possible levels a DIO channel can have (input or output)	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.5 Dio_PortType

Table 32 Specification for Dio_PortType

Syntax	`Dio_PortType`	
Type	uint8	
File	`Dio.h`	
Range	0 to 41	Number of Dio Ports
Description	Numeric ID of a DIO Port	
Source	AUTOSAR	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.2.6 Dio_PortLevelType

Table 33 Specification for Dio_PortLevelType

Syntax	`Dio_PortLevelType`

Table 33 **Specification for Dio_PortLevelType (continued)**

Type	uint16	
File	`Dio.h`	
Range	0x0 – 0xFFFF	It is a type of the value of Dio Port. It inherits the size of the largest port.
Description	It is a type of the value of Dio Port. It inherits the size of the largest port.	
Source	IFX	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3 Functions - APIs

This section lists all the APIs of DIO driver.

1.3.3.1 Dio_FlipChannel

Table 34 **Specification for `Dio_FlipChannel` API**

Syntax	`Dio_LevelType  Dio_FlipChannel ( const Dio_ChannelType ChannelId )`	
Service ID	0x11	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	ChannelId	ID of DIO channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Dio_LevelType	The physical level of the corresponding Pin
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 the flip. The function will ignore to configure the level values for pin/s which is/are configured as INPUT. The user of Dio module SHALL use the symbolic names provided by the configuration of the Dio module as values for any parameters of type Dio_ChannelType. *Note:The Dio module's environment/user SHALL ensure that Dio APIs are called only after initialization of port driver.*	
Source	AUTOSAR	

Table 34 **Specification for `Dio_FlipChannel` API (continued)**

Error handling	DIO_E_PARAM_INVALID_CHANNEL_ID
Configuration dependencies	DioFlipChannelApi
User hints	-
SFR accessed	P_IN(r), P_OMR(w) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.2 Dio_GetVersionInfo

Table 35 **Specification for `Dio_GetVersionInfo` API**

Syntax	`void  Dio_GetVersionInfo`
	`(`
	`    Std_VersionInfoType * const VersionInfo`
	`)`	

Service ID	0x12	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	-	-
Parameters (out)	VersionInfo	Pointer to where to store the version information of this module.
Parameters (in - out)	-	-
Return	void	-
Description	Service to get the version information of this module.	
Source	AUTOSAR	
Error handling	DIO_E_PARAM_POINTER	
Configuration dependencies	DioVersionInfoApi	
User hints	-	
SFR accessed	-	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.3 Dio_MaskedWritePort

Table 36 Specification for `Dio_MaskedWritePort` API

Syntax	`void  Dio_MaskedWritePort (     const Dio_PortType PortId,     const Dio_PortLevelType Level,     const Dio_PortLevelType Mask )`	
Service ID	0x13	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	PortId Level Mask	ID of DIO Port Pin (Bit-wise) representation of STD_HIGH/STD_LOW in that port Channels to be masked in the port
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	Service to set the value of a given port with required mask. The level value in the bit positions which are not set in mask will be ignored. The function will ignore to configure the level values for pin/s which is/are configured as INPUT. The user of Dio module SHALL use the symbolic names provided by the configuration of the Dio module as values for any parameters of type Dio_PortType. *Note:The Dio module's environment/user SHALL ensure that Dio APIs are called only after initialization of port driver.*	
Source	IFX for AS4.2.2 variant and AUTOSAR for AS4.4.0 variant	
Error handling	DIO_E_PARAM_INVALID_PORT_ID	
Configuration dependencies	DioMaskedWritePortApi	
User hints	-	
SFR accessed	P_OMR(rw) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.4 Dio_ReadChannel

Table 37 Specification for `Dio_ReadChannel` API

Syntax	`Dio_LevelType  Dio_ReadChannel ( const Dio_ChannelType ChannelId )`	
Service ID	0x00	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	ChannelId	ID of DIO channel
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Dio_LevelType	The physical level of the corresponding Pin
Description	Returns the value of the specified DIO channel. The function will ignore to configure the level values for pin/s which is/are configured as INPUT. The user of Dio module SHALL use the symbolic names provided by the configuration of the Dio module as values for any parameters of type Dio_ChannelType. *Note:The Dio module's environment/user SHALL ensure that Dio APIs are called only after initialization of port driver.*	
Source	AUTOSAR	
Error handling	DIO_E_PARAM_INVALID_CHANNEL_ID	
Configuration dependencies	-	
User hints	-	
SFR accessed	P_IN(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.5 Dio_ReadChannelGroup

Table 38 Specification for `Dio_ReadChannelGroup` API

Syntax	`Dio_PortLevelType  Dio_ReadChannelGroup ( const Dio_ChannelGroupType * const ChannelGroupIdPtr )`	
Service ID	0x04	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	ChannelGroupIdPtr	Pointer to ChannelGroup
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	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. The user of Dio module SHALL use the symbolic names provided by the configuration of the Dio module as values for any parameters of type Dio_ChannelGroupType. *Note:The Dio module's environment/user SHALL ensure that Dio APIs are called only after initialization of port driver.*	
Source	AUTOSAR	
Error handling	DIO_E_PARAM_INVALID_GROUP , DIO_E_PARAM_POINTER	
Configuration dependencies	-	
User hints	-	
SFR accessed	P_IN(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.6 Dio_ReadPort

Table 39 Specification for `Dio_ReadPort` API

Syntax	`Dio_PortLevelType  Dio_ReadPort (`

Table 39 **Specification for `Dio_ReadPort` API (continued)**

	`    const Dio_PortType PortId`
	`)`	

Service ID	0x02	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	PortId	ID of DIO Port
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	Dio_PortLevelType	Level of all channels of that port
Description	Returns the level of all channels of that port. The user of Dio module SHALL use the symbolic names provided by the configuration of the Dio module as values for any parameters of type Dio_PortType. *Note:The Dio module's environment/user SHALL ensure that Dio APIs are called only after initialization of port driver.*	
Source	AUTOSAR	
Error handling	DIO_E_PARAM_INVALID_PORT_ID	
Configuration dependencies	-	
User hints	-	
SFR accessed	P_IN(r) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.7 Dio_WriteChannel

Table 40 **Specification for `Dio_WriteChannel` API**

Syntax	`void  Dio_WriteChannel`
	`(`
	`    const Dio_ChannelType ChannelId,`
	`    const Dio_LevelType Level`
	`)`

Service ID	0x01

Table 40 **Specification for `Dio_WriteChannel` API (continued)**

Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	ChannelId Level	ID of Dio Channel Value to be written (STD_HIGH / STD_LOW)
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	Service to set specified level for a channel. The function will ignore to configure the level values for pin/s which is/are configured as INPUT. The user of Dio module SHALL use the symbolic names provided by the configuration of the Dio module as values for any parameters of type Dio_ChannelType. *Note:The Dio module's environment/user SHALL ensure that Dio APIs are called only after initialization of port driver.*	
Source	AUTOSAR	
Error handling	DIO_E_PARAM_INVALID_CHANNEL_ID, DIO_E_PARAM_INVALID_LEVEL	
Configuration dependencies	-	
User hints	-	
SFR accessed	P_OMR(rw) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.8 Dio_WriteChannelGroup

Table 41 **Specification for `Dio_WriteChannelGroup` API**

Syntax	`void  Dio_WriteChannelGroup`
	`(`
	`    const Dio_ChannelGroupType * const ChannelGroupIdPtr,`
	`    const Dio_PortLevelType Level`
	`)`

Service ID	0x05
Sync/Async	Synchronous

Table 41 **Specification for `Dio_WriteChannelGroup` API (continued)**

ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	ChannelGroupIdPtr Level	Pointer to ChannelGroup Value to be written
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	Service to set a subset of the adjoining bits of a port to a specified level. The function will ignore to configure the level values for pin/s which is/are configured as INPUT. For group or multiple pins, level value in the bit positions which are not set in channel group will be ignored. The user of Dio module SHALL use the symbolic names provided by the configuration of the Dio module as values for any parameters of type Dio_ChannelGroupType. *Note:The Dio module's environment/user SHALL ensure that Dio APIs are called only after initialization of port driver.*	
Source	AUTOSAR	
Error handling	DIO_E_PARAM_INVALID_GROUP , DIO_E_PARAM_POINTER	
Configuration dependencies	-	
User hints	-	
SFR accessed	P_OMR(w) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.9 Dio_WritePort

Table 42 **Specification for `Dio_WritePort` API**

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

Service ID	0x03

Table 42 Specification for `Dio_WritePort` API (continued)

Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	PortId Level	ID of DIO Port Value to be written
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	void	-
Description	Service to set specified level for Dio Port. The function will ignore to configure the level values for pin/s which is/are configured as INPUT. The user of Dio module SHALL use the symbolic names provided by the configuration of the Dio module as values for any parameters of type Dio_PortType. *Note:The Dio module's environment/user SHALL ensure that Dio APIs are called only after initialization of port driver.*	
Source	AUTOSAR	
Error handling	DIO_E_PARAM_INVALID_PORT_ID	
Configuration dependencies	-	
User hints	-	
SFR accessed	P_OUT(rw) *Note : The list includes all the SFRs accessed in the context of the API. It lists the SFRs accessed by the driver and called interfaces from other drivers. During runtime, the SFRs accessed from this list may vary based on configuration and execution context.*	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.4 Notifications and Callbacks

The DIO driver does not provide any notification and callbacks.

1.3.5 Scheduled functions

The DIO driver does not provide any scheduled functions.

1.3.6 Interrupt service routines

The DIO driver does not provide any interrupt handlers.

1.3.7 Callout

The driver does not support any callout functions.

1.3.8 Errors Handling

This section describes the various errors reported by the DIO driver.

Error Name: Description	Source	Error ID (AS422)	Type (AS422)	Error ID (AS440)	Type (AS440)
DIO_E_PARAM_INVALID_PORT_ID : Invalid port name requested.	AUTOSAR	0x14	DET_SAFETY	0x14	DET_SAFETY
DIO_E_PARAM_INVALID_CHANNEL_ID: Invalid channel name requested.	AUTOSAR	0x0A	DET_SAFETY	0x0A	DET_SAFETY
DIO_E_PARAM_INVALID_GROUP : Invalid ChannelGroup requested.	AUTOSAR	0x1F	DET_SAFETY	0x1F	DET_SAFETY
DIO_E_PARAM_INVALID_LEVEL : This safety error code is reported if wrong level is passed to the API.	IFX	0x32	SAFETY	0x32	SAFETY
DIO_E_PARAM_POINTER: API service called with a NULL pointer.	AUTOSAR	0x20	DET_SAFETY	0x20	DET_SAFETY

1.3.9 Deviations and limitations

The section describes the deviations and limitations from software specification.

1.3.9.1 Deviations

This section describes the deviation of the DIO driver.

1.3.9.1.1 Software specification deviations

This section describes the deviations from software specification.

Table 43 Known Deviations

Reference	Deviation
AUTOSAR_SWS_DIODriver.pdf, AUTOSAR Release 4.2.2: Section 10.1.2 DIO	The DIO driver is implemented as post-build variant support false, instead of true. Issue is raised via Bugzilla(77125) and confirmed for update in future ASR release.
AUTOSAR_SWS_DIODriver.pdf, AUTOSAR Release 4.2.2:ECUC_DIO_00150: DioPortMask	The parameter DioPortMask is implemented as pre-compile instead of link time. The parameter DioPortMask is used for generating derived macros.

Table 43 **Known Deviations (continued)**

	Therefore, this parameter is implicitly converted to pre-compile.

1.3.9.1.2 AMDC Violations

The DIO driver does not have any AMDC violations.

1.3.9.1.3 VSMD Violations

The DIO driver does not have any VSMD violations.

1.3.9.2 Limitations

The DIO driver does not have any limitations.

Revision history

Table 44 **Revision History**

Date	Version	Description
2020-11-18	2.0	Document Released
2020-11-17	1.1	• SFR access information updated
2020-08-13	1.0	Document Released
2020-08-06	0.1	• Initial Version • Dio driver chapter moved from MC-ISAR_TC3xx_UM_Basic to this document • Dio_MaskedWritePortApi added for AS440

Edition 2020-11-18

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about any aspect of this document?

Email: erratum@infineon.com

Document reference
IFX-ocr1484806431059