MCAL DIO Module Software Design Document

Document Version : 56
Document Owner : Texas Instruments
Document Status : Published
Last Approval Date : Mar 11, 2022

- Template Revision History

1 Revision History

Version	Date	Author	Document Status	Comments
0.1	25 Jul 2018	Vibha Pant	Draft	First Version
0.2	09 Oct 2018	Sujith S	In Review	Format change and re-order
0.3	09 Oct 2018	Vibha Pant	In Review	Review Comments Addressed
0.4	28 Jan 2019	Vibha Pant	In Review	Section for AUTOSAR 2.3.1 Version
0.5	19 Jan 2020	Sunil M S	In Review	Updates w.r.o porting AUTOSAR 4.3.1 Version
0.6	21 Apr 2021	Nishit Dhas	In Review	Format change as per ASPICE

Version	Date	Author	Document Status	Comments
0.7	12 Aug 2021	Nikki Shah	In Review	Adding Design IDs
0.8	19 Aug 2021	Nishit Dhas	In Approval	Fixing Review Comments
1.0	07 Sep 2021	Nikki S	Published	Upload to Galileo
1.1	24 Jan 2022	Nikki S	In Review	JAINCTOREQ-1870
v56	04 Mar 2022	Nikki S	Published	Review Comments Addressed.

2 Terms and Abbreviations

Abbreviation /Term	Meaning / Explanation
DIO	Digital Input Output
AUTOSAR	AUTomotive Open System ARchitecture
BSW	Basic Software
RTE	Runtime Environment
MCAL	MicroController Abstraction Layer
SBL	Serial Bootloader
API	Application Programming Interface
ECU	Electronic Control Unit

Abbreviation /Term	Meaning / Explanation
DIO channel	Represents a single general-purpose digital input/output pin
DIO port	Represents several DIO channels that are grouped by hardware
DIO channel group	Represents several adjoining DIO channels represented by a logical group. A DIO channel group shall belong to one DIO port.
ID	Identifier
DET	Default Error Tracer
DEM	Diagnostic Event Manager – module to handle diagnostic relevant events.
DAR	Decision Analysis and Resolution
SoC	System on a Chip

3 Introduction

This document describes the design of the AUTOSAR BSW module DIO.

- Supported AUTOSAR Release **: 4.3.1**
- Supported Configuration Variants **: Pre-Compile & Link Time**
- Vendor ID **: DIO_VENDOR_ID (44)**
- Module ID **: DIO_MODULE_ID (120)**

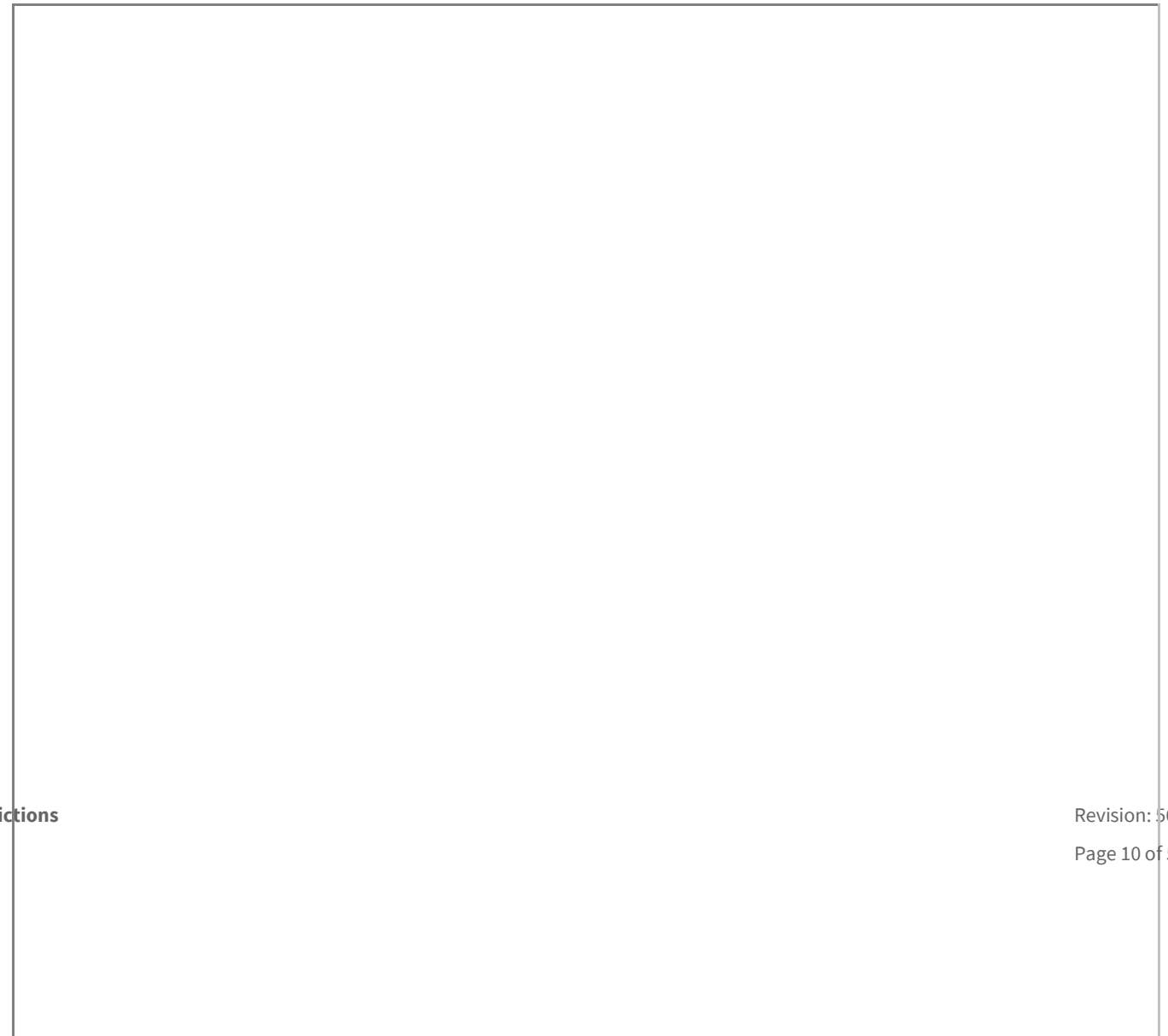
3.1 Overview

The figure below depicts the AUTOSAR layered architecture as 3 distinct layers,

- Application
- Runtime Environment (RTE) and
- Basic Software (BSW).

The BSW is further divided into 4 layers:

- Services
- Electronic Control Unit Abstraction
- MicroController Abstraction (MCAL) and
- Complex Drivers.



AUTOSAR Architecture

The DIO driver is a part of the microcontroller (peripheral) Driver module which is a part of the Basic Software. The figure below shows the position of the DIO driver in the AUTOSAR Architecture.

AUTOSAR Architecture – DIO MCAL

3.2 Purpose and Scope

The Detailed Design document provides the design details of DIO driver and aims to provide a guide to a design that could be implemented by a software developer.

The scope of this document is to describe the software design procedure of DIO module.

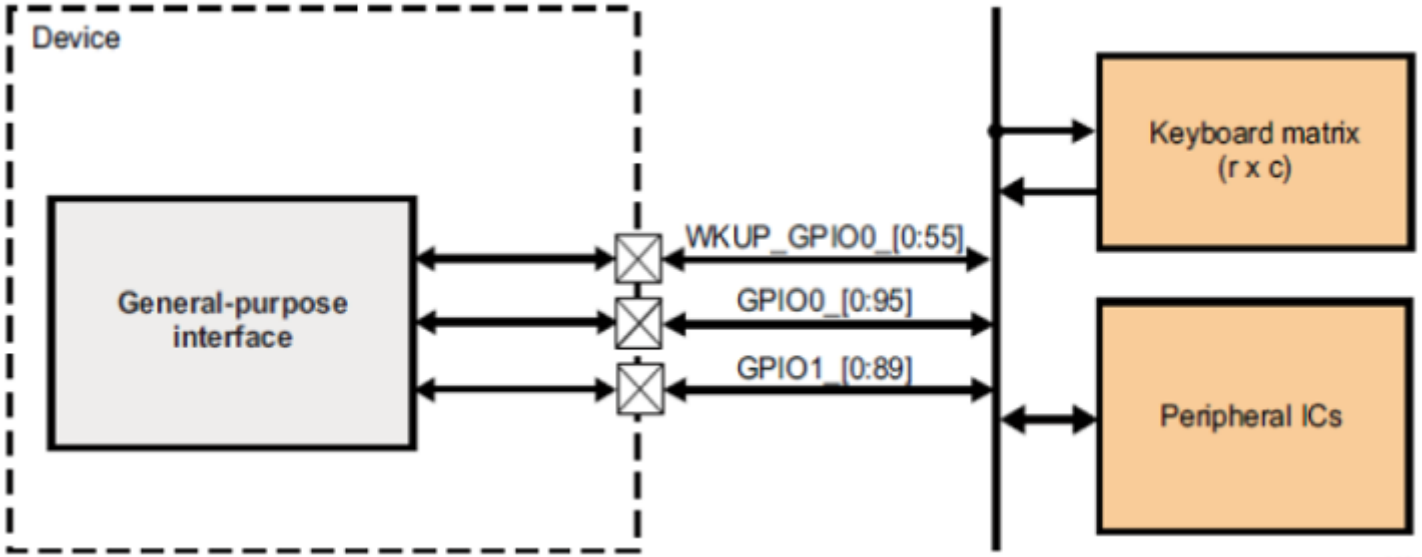
3.3 Module Overview

The DIO driver provides port and channel based read and write access to the internal general purpose I/O ports, using hardware IP "gpio_144_10_rel.1.5.x". The read and write behavior is unbuffered. The basic behavior of this driver is synchronous.

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

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

The DIO driver provides an interface to external peripherals by abstracting the input and output pins on the microcontroller device. The DIO pins are general purpose in nature. These instances are generally associated with specific domain, e.g. wakeup, main domains. Diagrams below are from device TRMs. Please refer the device specific data Manual for operating voltages and current sourcing capabilities (these can be found in the SoC User Manual).



DIO Typical Application

3.4 Requirements

The DIO Driver abstracts the access to the microcontroller's hardware pins. The DIO Driver implements a standardized interface as specified in Reference 1 - AUTOSAR 4.3.1.

3.4.1 Features Supported

The DIO SWS driver defines functions allowing read and write access to the internal general purpose I/O channels, ports and channel groups.

3.4.2 Features Not Supported / NON Compliance

- **[NON Compliance]** As the microcontroller currently doesnt support direct read back, requirement pertaining to direct read back is not supported.
- Supports additional configuration parameters, refer section (Dio_RegisterReadback) & generates global (Global Variables)

3.5 Assumptions

Below listed are assumed to valid for this design/implementation, exceptions and other deviations are listed for each explicitly. Care should be taken to ensure these assumptions are addressed by an entity outside DIO driver.

1. This module works on pins and ports which must be configured external to this device. Overall configuration and initialization of the port structure which is used in the DIO module.
2. The DIO functions are valid only after the Port Driver has been initialized. If it is not initialized, then DIO behavior is undefined. In cases where MCAL Port module is not present, the SBL/GEL files will initialize the pin functionality.
3. The functional clock to the DIO module is expected to be ON before calling any DIO module API.
4. Please Note that an entity outside DIO module will take care to configured required voltage level for DIO.

Note that assumption 1 & 2 are specified by AUTOSAR DIO specification. Assumptions 3 & 4 are device specific.

3.6 Constraints

Some of the PINs are reserved and cannot be used by DIO module, please refer device specific manual for details.

3.7 Hardware and SW platforms

Hardware Platforms

- Refer to specified SoC User Manual to check if ADC module is supported.

Software Platforms

- Bare-Metal

3.8 Dependencies

The DIO module does not provide APIs for overall configuration and initialization of the port structure which is used in the DIO module. The initialization and configuration will be done by other entities.

TI MCAL does not provide a PORT module. User must configure required pins in start-up application code using direct calls to registers. TI Pinmux utility can be used to generate reference code.

Design Identifier	Description
MCAL-5513	[SWS_Dio_00001] : Dio Initialization
MCAL-5468	[SWS_Dio_00002] : Dio Reconfiguration
MCAL-5428	[SWS_Dio_00063] : Dio Scalability

3.9 Stakeholders

- Developers
- Test Engineers
- Customer Integrator

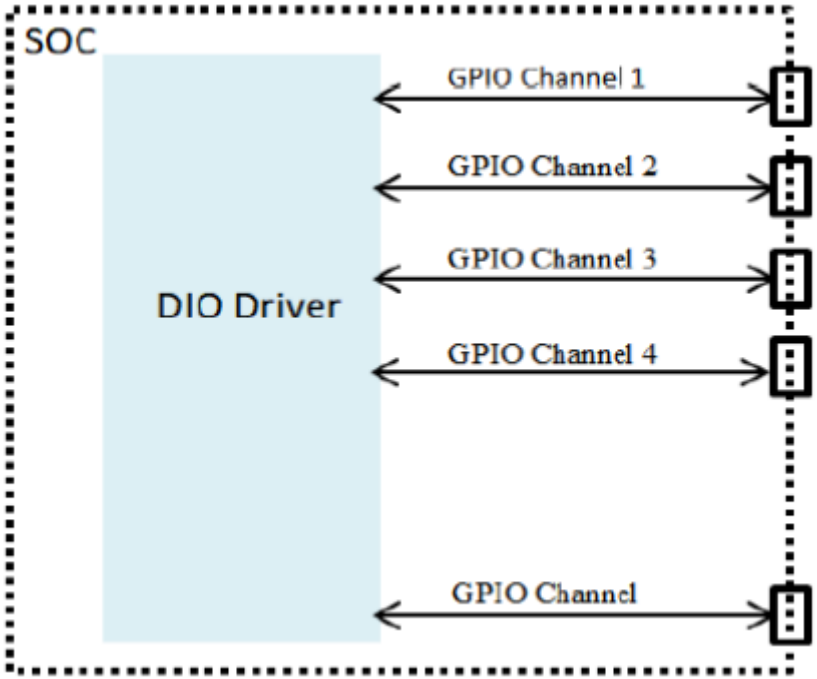
3.10 References

	Specification	Comment/Link
1	AUTOSAR 4.3.1	AUTOSAR Specification for DIO Driver.

	Specification	Comment/Link
2	BSW General Requirements / Coding guidelines	Autosar and Coding guidelines for the Mcal drivers.
3	Software Product Specification (SPS)	Product Functional requirements.
4	Software Architecture	Mcal Software Architecture.

4 Design Description

The DIO driver provides an interface to the external connections. The top level diagram of DIO module is as show below. (n= varies for each SOC)



DIO Block Diagram

4.1 DIO Channel, Port And Channel Group

A DIO channel represents a single general-purpose digital input/output pin. A DIO Port is a grouping of several DIO channels by hardware (typically controlled by one hardware register). A DIO Channel Group consists of several adjoining DIO channels represented by a logical group. A DIO channel group belongs to one DIO port as illustrated below

DIO Channel Group

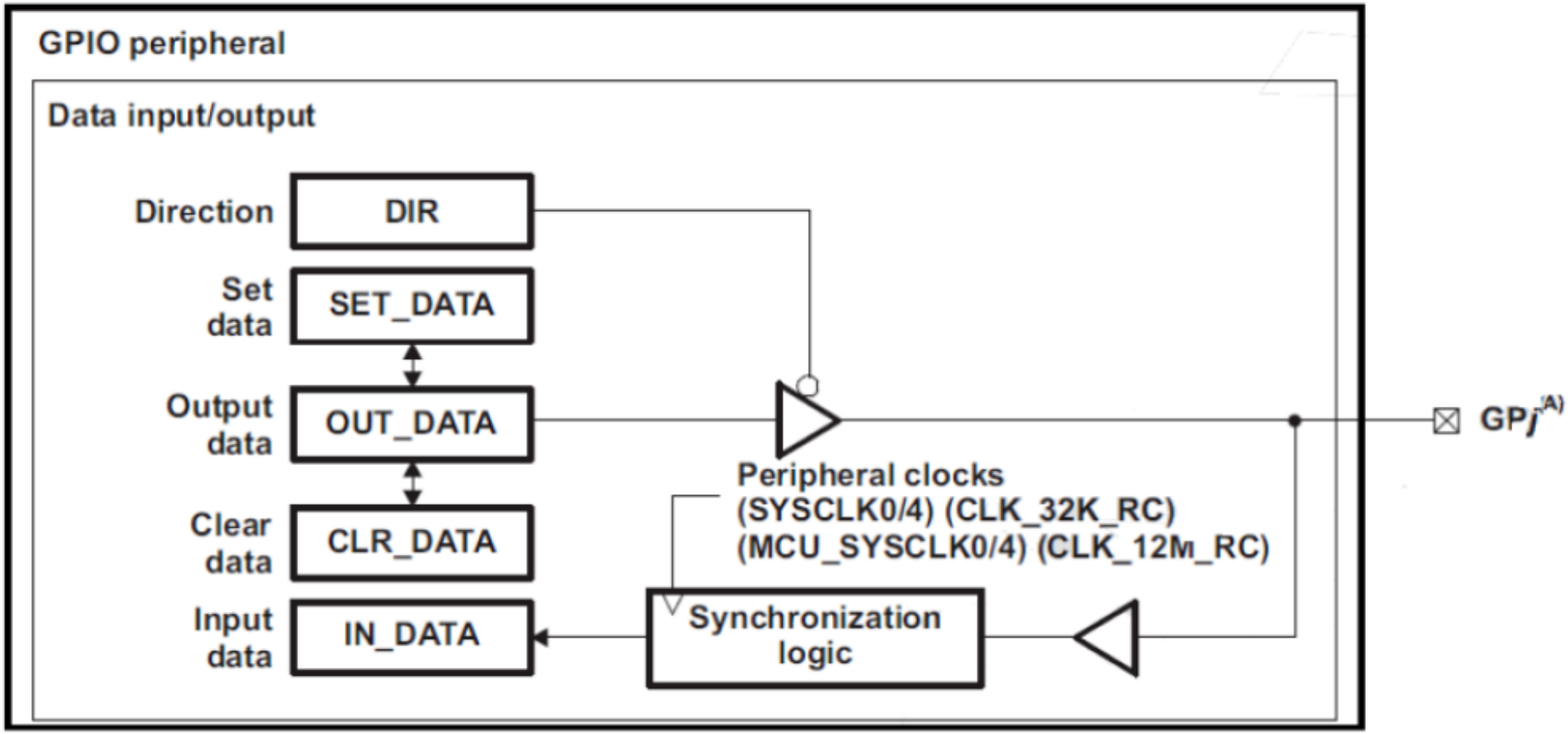
The allocation of DIO instances is dependent on the variant of the device being used. Please refer Device Specific TRM for details. The table below depicts the DIO allocation for DRA80X.

DIO Modules in Different Domain

In general, channels/pins can be configured as input or output. Each DIO instance supports 9 banks of 16 DIO pins each. There are in total 3 instances, one in wakeup domain and two in the main domain. Refer to the SoC User Manual for SoC specific details.

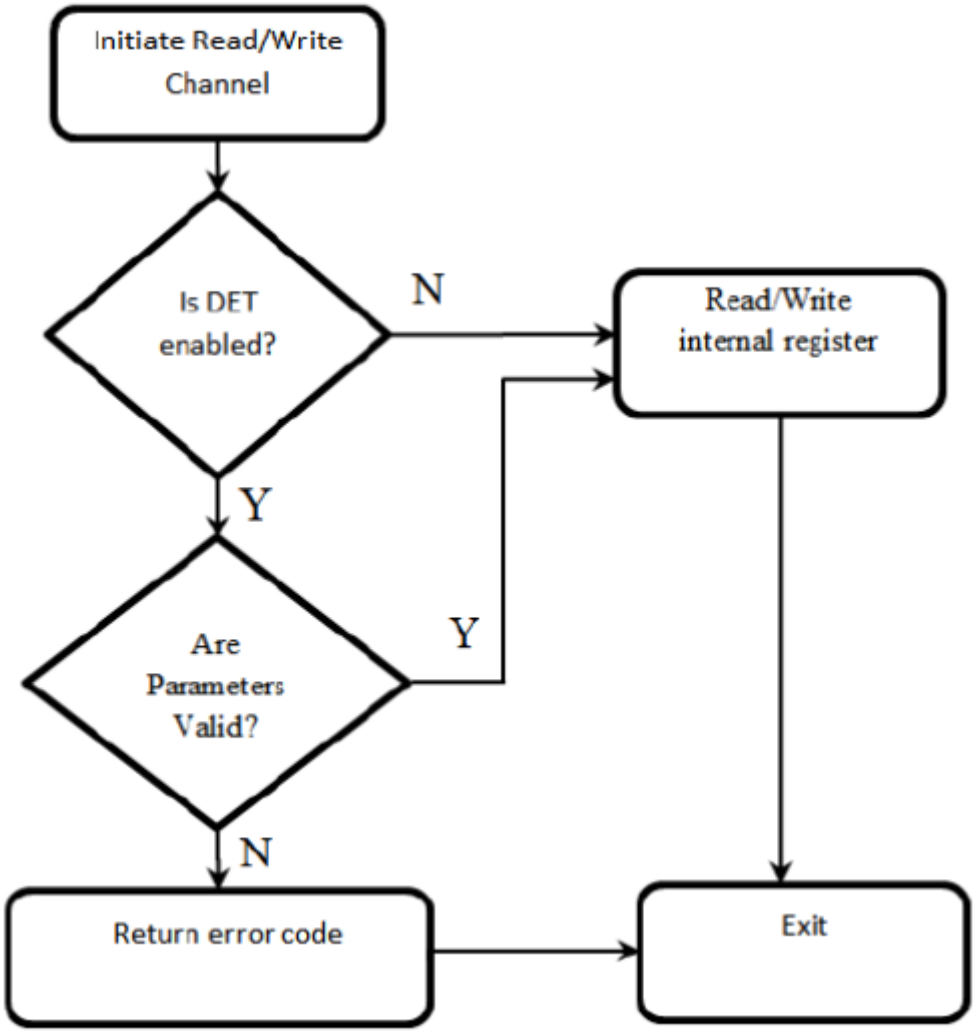
4.2 Input/Output Functionality

The DIO peripheral provides the main functionality of input and output. Each pin can be configured independently as input or output with the help of the GPIO direction registers.


GPIO peripheral
Data input/output
Direction
DIR
Set data
SET_DATA
Output data
OUT_DATA
Clear data
CLR_DATA
Input data
IN_DATA
Peripheral clocks
(SYSCLK0/4) (CLK_32K_RC)
(MCU_SYSCLK0/4) (CLK_12M_RC)
Synchronization logic
GPj(A)

DIO Functional Block Diagram

The main services implemented for the input/output pins are the read and write services for channels, ports and channel groups. The following sequence diagrams elaborate the sequence followed for a typical read and write service.


Initiate Read/Write Channel
Is DET enabled?
N
Read/Write internal register
Y
Are Parameters Valid?
Y
N
Return error code
Exit

DIO Read/Write Service Flow Chart

4.3 Supporting DIO for AUTOSAR versioned 4.3.1

The following have been removed in v4.3.1 specification, but it is included in the code in order to set the event status when it fails.

- SWS_Dio_00131 Imported Types: Dem Module and specifically Imported types: Dem_EventIdType

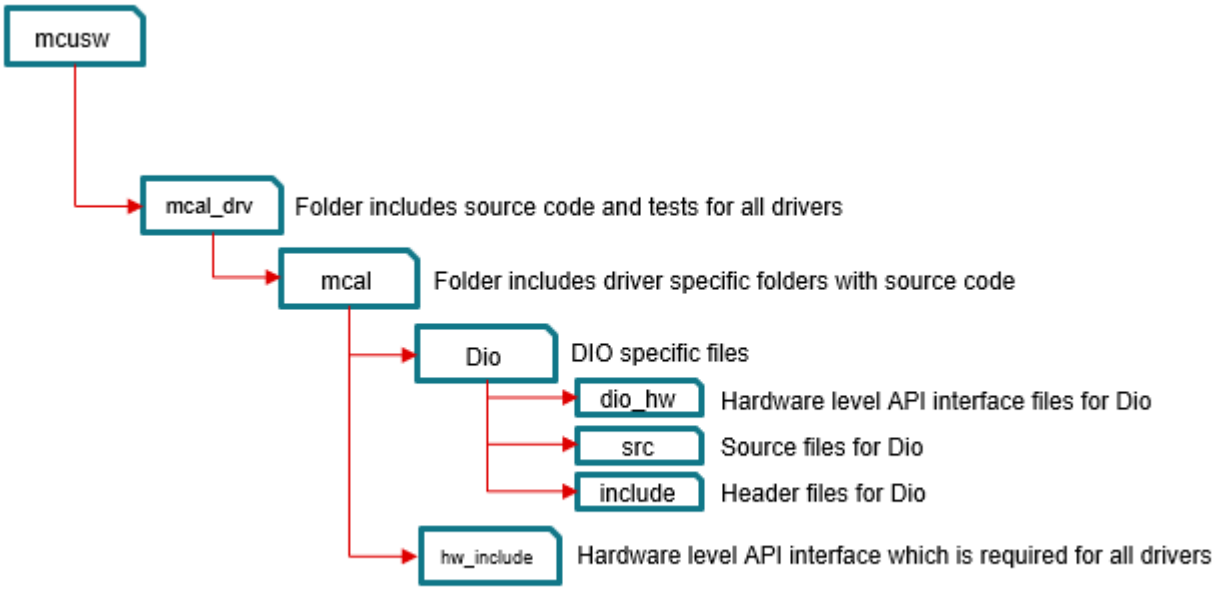
4.4 Directory Structure

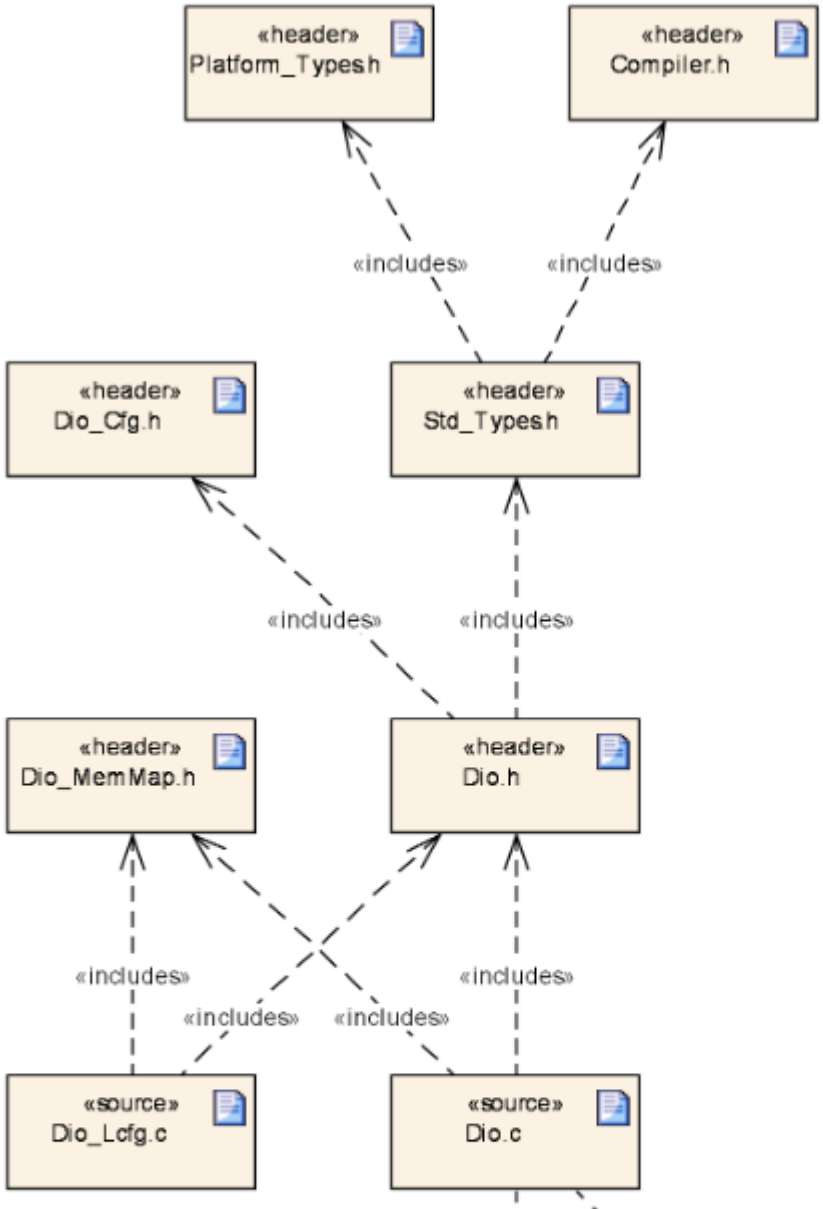
Design Identifier	Description
MCAL-5509	[SWS_Dio_00170] : Dio File structure
MCAL-5500	[SWS_Dio_00194] : Dio File structure

The directory structure is as depicted in figures below, the source files can be categorized under “Driver Implementation” and “Example Application”.

Driver Implemented by

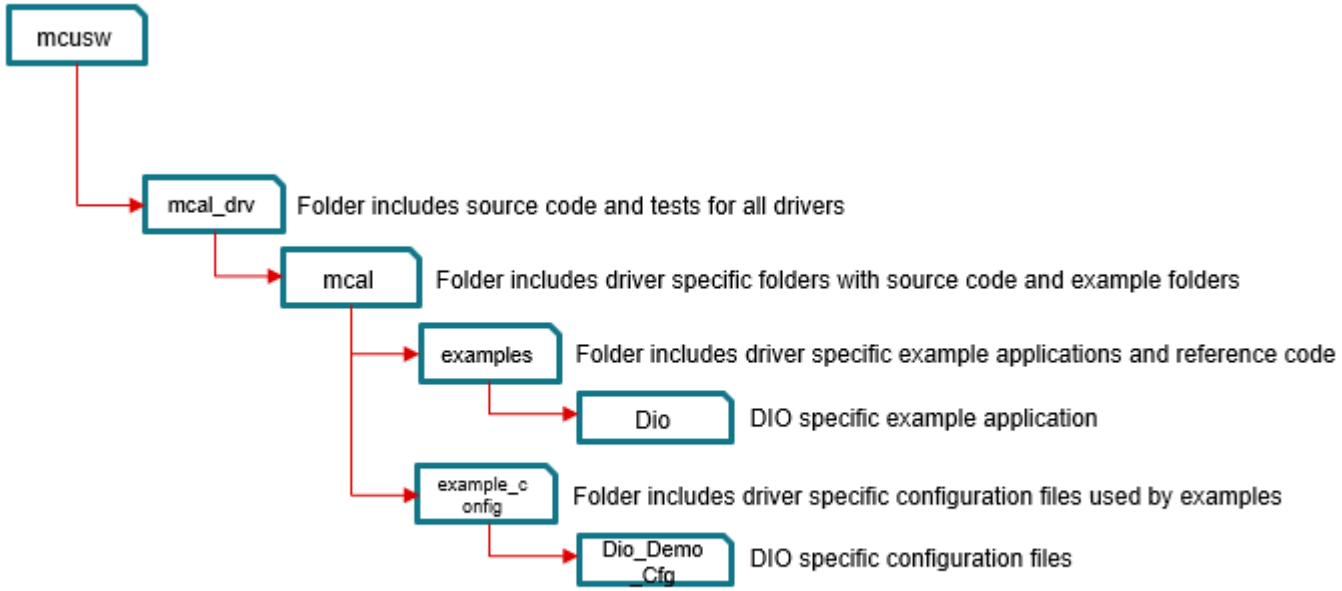
- Dio.h : Shall implement the interface provided by the driver.
- Dio.c, Dio_Priv.h : Shall implement the driver functionality.
- lld_gpio.h, lldr_gpio.h : Shall include the SOC specific register definitions.


mcusw
mcal_drv
Folder includes source code and tests for all drivers
mcal
Folder includes driver specific folders with source code
Dio
DIO specific files
dio_hw
Hardware level API interface files for Dio
src
Source files for Dio
include
Header files for Dio
hw_include
Hardware level API interface which is required for all drivers


«header»
Platform_Types.h
«header»
Compiler.h
«includes»
«includes»
«header»
Dio_Cfg.h
«header»
Std_Types.h
«includes»
«includes»
«header»
Dio_MemMap.h
«header»
Dio.h
«includes»
«includes»
«includes»
«includes»
«source»
Dio_Lcfg.c
«source»
Dio.c

Example Application

- Dio_Cfg.h and Dio_Cfg.c: Shall implement the generated configuration for pre-compile variant.
- Dio_Cfg.h and Dio_Lcfg.c: Shall implement the generated configuration for link-time variant.
- DioApp.c: Shall implement the example application that demonstrates the use of the driver.


mcusw
mcal_drv
Folder includes source code and tests for all drivers
mcal
Folder includes driver specific folders with source code and example folders
examples
Folder includes driver specific example applications and reference code
Dio
DIO specific example application
example_config
Folder includes driver specific configuration files used by examples
Dio_Demo_Cfg
DIO specific configuration files

4.5 Configurator

Design Identifier	Description
MCAL-5532	ECUC_Dio_00150 : DioPortMask
MCAL-5528	ECUC : DioDemEventParameterRef
MCAL-5527	ECUC_Dio_00141: DioGeneral
MCAL-5525	ECUC_Dio_00145 : DioPortId
MCAL-5510	ECUC_Dio_00151 : DioPortOffset
MCAL-5501	ECUC_Dio_00142: DioDevErrorDetect
MCAL-5492	ECUC_Dio_00152 : DioConfig
MCAL-5482	ECUC_Dio_00146 : DioChannel
MCAL-5481	DioRegisterReadbackApi
MCAL-5480	ECUC_Dio_00149 : DioChannelGroupIdentification

Design Identifier	Description
MCAL-5473	ECUC_Dio_00143 : DioVersionInfoApi
MCAL-5470	ECUC :DIO_WRITE_PORT_EVENT_ID
MCAL-5461	ECUC_Dio_00154 : Dio Configuration Container
MCAL-5460	ECUC_Dio_00153 : DioFlipChannelApi
MCAL-5457	ECUC_Dio_00148 : DioChannelGroup
MCAL-5455	ECUC_Dio_00144 : DioPort
MCAL-5452	ECUC_Dio_00147 : DioChannelId
MCAL-5448	ECUC : DioDeviceVariant

The AUTOSAR DIO Driver Specification details mandatory parameters that shall be configurable via the configurator. Please refer section 10 of Reference 1 - AUTOSAR 4.3.1.

4.5.1 NON Standard configurable parameters

The design's specific configurable parameters are as follows:

Parameter	Usage comment
DioRegisterReadbackApi	This shall allow integrators to specify if the read back of critical registers using the API is required or not.
DioDeviceVariant	This shall allow integrators to select the device variant for which integration is being performed.This parameter shall be used by driver to impose device specific constraints. The user guide shall detail the device specific constraints.

4.5.2 Variant Support

The driver shall support both VARIANT-LINK-TIME & VARIANT-PRE-COMPILE

4.6 Error Classification

Design Identifier	Description
MCAL-5434	[SWS_Dio_00075] : Dio Error Detection
MCAL-5495	[SWS_Dio_00074] : Dio Error Detection

Design Identifier	Description
MCAL-5489	[SWS_Dio_00114] : Dio Error Detection
MCAL-5443	[SWS_Dio_00175] : DIO_E_PARAM_INVALID_CHANNEL_ID
MCAL-5441	[SWS_Dio_00177] : DIO_E_PARAM_INVALID_PORT_ID
MCAL-5493	[SWS_Dio_00178] : DIO_E_PARAM_INVALID_GROUP
MCAL-5425	[SWS_Dio_00188] : DIO_E_PARAM_POINTER
MCAL-5497	[SWS_Dio_00065] : Dio Error and Exceptions
MCAL-5496	[SWS_Dio_00189] : Dio Error Handling
~~MCAL-5501~~	ECUC_Dio_00142: DioDevErrorDetect
MCAL-5514	[SWS_Dio_00118] : Dio Read Service Development errors
MCAL-5504	[SWS_Dio_00020] : Dio Port Type Parameter Specification
MCAL-5503	[SWS_Dio_00015] : Dio Channel Type Parameter Specification

Errors are classified in two categories, development error and runtime / production error.

4.6.1 Error Detection

The detection of development errors is configurable (ON / OFF) at pre-compile time. The switch DioDevErrorDetect will activate or deactivate the detection ofall development errors.

4.6.2 Development Errors

Type of Error	Related Error Code	Value (Hex)
Invalid channel name requested	DIO_E_PARAM_INVALID_CHANNEL_ID	0x0A
Parameter is NULL Pointer	DIO_E_PARAM_CONFIG	0x10
Invalid Port Nam3	DIO_E_PARAM_INVALID_PORT_ID	0x14
API parameter checking: invalid channel	DIO_E_PARAM_INVALID_GROUP	0x1F

Type of Error	Related Error Code	Value (Hex)
NULL Pointer	DIO_E_PARAM_POINTER	0x20

4.6.3 Error notification (DET)

All detected development errors are reported to Det_ReportError service of the Development Error Tracer (DET).

5 Implementation Details

5.1 Data structures and resources

5.1.1 MACROS, Data Types & Structures

The sections below lists some of key data structures that shall be implemented and used in driver implementation.

Design Identifier	Description
MCAL-5469	[SWS_Dio_00131] : Dio Imported Types
MCAL-5533	[SWS_Dio_00183] : Dio Port Type Specification
MCAL-5526	[SWS_Dio_00021] : Dio Channel Group Type Specification
MCAL-5519	[SWS_Dio_00185] : Dio Level Type Specification
MCAL-5517	[SWS_Dio_00023] : Dio Level Type Specification
MCAL-5486	[SWS_Dio_00024] : Dio Port Level Type Specification
MCAL-5484	[SWS_Dio_00186] : Dio Port Level Type Specification

Design Identifier	Description
MCAL-5467	[SWS_Dio_00184] : Dio Channel Group Type Specification
MCAL-5436	[SWS_Dio_00182] : Dio Channel Type Specification
MCAL-5518	[SWS_Dio_00022] : Dio Channel Group Type Parameter Specification
MCAL-5516	[SWS_Dio_00181] : Dio Port Type Parameter Specification
MCAL-5435	[SWS_Dio_00089] : Dio Software Channel Level
MCAL-5447	[SWS_Dio_00103] : Port Width Specification
MCAL-5446	[SWS_Dio_00017] : Dio Channel Type Parameter Specification
MCAL-5449	[SWS_Dio_00180] : Dio Channel Type Parameter Specification
MCAL-5521	[SWS_Dio_00018] : Dio Port Type Parameter Specification

Maximum number of DIO Instances

Type	Identifier	Comments
uint8	DIO_HW_UNIT_CNT	Defines the maximum number of Instaces of DIO driver that are configured.

Dio_ChannelType

Type definition used to specify the numeric id of the channel, please refer section 8.2.1 of Reference 1 - AUTOSAR 4.3.1

Dio_PortType

Type definition used to specify the numeric id of the port, please refer section 8.2.2 of Reference 1 - AUTOSAR 4.3.1

Dio_ChannelGroupType

Type definition used to specify the channel group, please refer section 8.2.3 of Reference 1 - AUTOSAR 4.3.1

Dio_LevelType

Used to specify the possible levels of a channel, please refer section 8.2.4 of Reference 1 - AUTOSAR 4.3.1

Dio_PortLevelType

Used to specify the possible levels of a port, please refer section 8.2.5 of Reference 1 - AUTOSAR 4.3.1

Dio_DirectionType

Used to specify the direction of a channel.

5.1.2 Global Variables

This design expects that implementation will require to use following global variables.

Variable	Type	Description	Default Value
Dio_ConfigValidChannelMask	uint32 array	Auto generated array of enabled ports	NA

5.2 Dynamic Behavior - Control Flow Diagram

Not Applicable

5.3 Dynamic Behavior - Data Flow Diagram

Not Applicable

5.4 Application Parameters

Dio_ReadChannel

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
ChannelId	ID of DIO Channel (Input Parameter)	0 to 432	-	-	N.A

Dio_WriteChannel

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
ChannelId	ID of DIO channel (Input Parameter)	0 to 432	-	-	N.A
Level	Value to be written (Input Parameter)	0 to 1	-	-	N.A

Dio_ReadPort

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
PortId	ID of DIO Port (Input Parameter)	0 to 144	-	-	N.A

Dio_WritePort

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
PortId	ID of DIO Port (Input Parameter)	0 to 144	-	-	N.A
Level	Value to be written (Input Parameter)	0 to 0xFFFFFFFF	-	-	N.A

Dio_ReadChannelGroup

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
ChannelGroupIdPtr	Pointer to ChannelGroup (Input Parameter)	0 to 0xFFFFFFFF	-	-	N.A

Dio_WriteChannelGroup

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
ChannelGroupIdPtr	Pointer to ChannelGroup (Input Parameter)	0 to 0xFFFFFFFF	-	-	N.A
Level	Value to be written (Input Parameter)	0 to 0xFFFFFFFF	-	-	

Dio_FlipChannel

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
ChannelId	ID of DIO channel (Input Parameter)	0 - 432	-	-	N.A

Dio_GetVersionInfo

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
versioninfo	Pointer to where to store the version information of this module. (Output Parameter)	-	-	-	N.A

Dio_RegisterReadback

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant

ChannelId	ID of DIO channel (Input Parameter)	0 - 432	-	-	N.A

5.5 Safety Diagnostic Features

GPIO1 - Software test of function using I/O loopback:

DIO module does not support a distinct I/O loopback mode. However, it is possible to support I/O checking using normal functionality. To do this software generates output and reads back and checks for the same value in the input registers. The DIO MCAL driver provides the API - **Dio_WritePort** and **Dio_ReadPort** to implement this diagnostic feature.

GPIO3 - Periodic Software Readback of Static Configuration Registers / GPIO4 - Software Readback of Written Configuration:

Software Readback of Written Configuration ensures that the configuration register are written with the expected value. Periodic readback of configuration registers can provide a diagnostic for inadvertent writes to these registers.

The DIO MCAL driver provides the API - **Dio_RegisterReadback** to readback static and written configuration registers to implement this diagnostic feature.

6 Low Level Definitions

6.1 DIO Read and Write Services

Design Identifier	Description
MCAL-5458	[SWS_Dio_00084] : Dio Direct Read-Back Service
MCAL-5472	[SWS_Dio_00005] : Dio Data Consistency
MCAL-5433	[SWS_Dio_00109] : Dio Write Service
MCAL-5432	[SWS_Dio_00119] : Dio Write Service Development errors
MCAL-5483	[SWS_Dio_00064] : Dio Write Service
MCAL-5499	[SWS_Dio_00070] : Dio Write Service
MCAL-5524	[SWS_Dio_00051] : Dio Buffering
MCAL-5514	[SWS_Dio_00118] : Dio Read Service Development errors

6.2 Driver APIs

For the standard APIs please refer 8.3 section of the DIO AutoSar Specification as listed in Reference 1. Sections below highlight other design considerations for the implementation.

6.2.1 Dio_ReadChannel

Refer section 8.3.1 of the DIO AutoSar Specification as listed in Reference 1 - AUTOSAR 4.3.1.

Design Identifier	Description
MCAL-5522	[SWS_Dio_00093] : Dio Read Channel Group Service
MCAL-5505	[SWS_Dio_00011] : Dio Read Channel Service
MCAL-5495	[SWS_Dio_00074] : Dio Error Detection
MCAL-5489	[SWS_Dio_00114] : Dio Error Detection
MCAL-5474	[SWS_Dio_00014] : Dio Read Port Service
MCAL-5471	[SWS_Dio_00037] : Dio Read Channel Group Service
MCAL-5444	[SWS_Dio_00092] : Dio Read Channel Group Service

Design Identifier	Description
MCAL-5424	[SWS_Dio_00027] : Dio Read Channel Service
MCAL-5465	[SWS_Dio_00012] : Dio Read Channel Service

6.2.2 Dio_WriteChannel

Refer section 8.3.2 of the DIO AutoSar Specification as listed in Reference 1 - AUTOSAR 4.3.1.

Design Identifier	Description
MCAL-5528	ECUC : DioDemEventParameterRef
MCAL-5507	[SWS_Dio_00091] : Dio Write Channel Group Service
MCAL-5495	[SWS_Dio_00074] : Dio Error Detection
MCAL-5489	[SWS_Dio_00114] : Dio Error Detection
MCAL-5487	[SWS_Dio_00008] : Dio Write Channel Group Service
MCAL-5478	[SWS_Dio_00028] : Dio Write Channel Service

Design Identifier	Description
MCAL-5476	[SWS_Dio_00039] : Dio Write Channel Group Service
MCAL-5464	[SWS_Dio_00029] : Dio Write Channel Service
MCAL-5459	[SWS_Dio_00040] : Dio Write Channel Group Service
MCAL-5454	[SWS_Dio_00079] : Dio Write Channel Service
MCAL-5427	[SWS_Dio_00006] : Dio Write Channel Service
MCAL-5426	[SWS_Dio_00090] : Dio Write Channel Group Service

6.2.3 Dio_ReadPort

Refer 8.3.3 of the DIO AutoSar Specification as listed in Reference 1 - AUTOSAR 4.3.1.

Design Identifier	Description
MCAL-5511	[SWS_Dio_00104] : Dio Read Port Service
MCAL-5475	[SWS_Dio_00013] : Dio Read Port Service

Design Identifier	Description
MCAL-5440	[SWS_Dio_00031] : Dio Read Port Service
MCAL-5434	[SWS_Dio_00075] : Dio Error Detection

6.2.4 Dio_WritePort

Refer section 8.3.4 of the DIO AutoSar Specification as listed in Reference 1 - AUTOSAR 4.3.1.

Design Identifier	Description
MCAL-5515	[SWS_Dio_00108] : Dio Write Port Service
MCAL-5512	[SWS_Dio_00105] : Dio Write Port Service
MCAL-5490	[SWS_Dio_00035] : Dio Write Port Service
MCAL-5470	ECUC :DIO_WRITE_PORT_EVENT_ID
MCAL-5456	[SWS_Dio_00004] : Dio Write Port Service
MCAL-5434	[SWS_Dio_00075] : Dio Error Detection

Design Identifier	Description
MCAL-5430	[SWS_Dio_00007] : Dio Write Port Service
MCAL-5429	[SWS_Dio_00034] : Dio Write Port Service

6.2.5 Dio_ReadChannelGroup

Refer section 8.3.5 of the DIO AutoSar Specification as listed in Reference 1 - AUTOSAR 4.3.1.

Design Identifier	Description
MCAL-5522	[SWS_Dio_00093] : Dio Read Channel Group Service
MCAL-5489	[SWS_Dio_00114] : Dio Error Detection
MCAL-5474	[SWS_Dio_00014] : Dio Read Port Service
MCAL-5471	[SWS_Dio_00037] : Dio Read Channel Group Service
MCAL-5444	[SWS_Dio_00092] : Dio Read Channel Group Service

6.2.6 Dio_WriteChannelGroup

Refer section 8.3.6 of the DIO AutoSar Specification as listed in Reference 1 - AUTOSAR 4.3.1.

Design Identifier	Description
MCAL-5507	[SWS_Dio_00091] : Dio Write Channel Group Service
MCAL-5489	[SWS_Dio_00114] : Dio Error Detection
MCAL-5487	[SWS_Dio_00008] : Dio Write Channel Group Service
MCAL-5476	[SWS_Dio_00039] : Dio Write Channel Group Service
MCAL-5459	[SWS_Dio_00040] : Dio Write Channel Group Service
MCAL-5426	[SWS_Dio_00090] : Dio Write Channel Group Service

6.2.7 Dio_FlipChannel

Refer section 8.3.8 of the DIO AutoSar Specification as listed in Reference 1 - AUTOSAR 4.3.1.

Design Identifier	Description
MCAL-5531	[SWS_Dio_00192] : Dio Flip Channel Service
MCAL-5529	[SWS_Dio_00191] : Dio Flip Channel Service
MCAL-5460	ECUC_Dio_00153 : DioFlipChannelApi
MCAL-5438	[SWS_Dio_00193] : Dio Flip Channel Service

6.2.8 **Dio_GetVersionInfo**

Refer section 8.3.7 of the DIO AutoSar Specification as listed in Reference 1 - AUTOSAR 4.3.1.

Design Identifier	Description
MCAL-5496	[SWS_Dio_00189] : Dio Error Handling

6.2.9 Dio_RegisterReadback

As noted from previous implementation, some of the critical configuration registers could potentially be corrupted by other entities (s/w or h/w). One of there commended detection methods would be to periodically read-back the configuration and confirm configuration is consistent. The service API defined below shall be implemented to enable this detection.

	Description	*Comments*
Service Name	Dio_RegisterReadback	Can potentially be turned OFF (see NON Standard configurable parameters)
Syntax	Std_ReturnType Dio_RegisterReadback(Dio_ChannelType ChannelId, Dio_RegisterReadbackType *DioRegRbPtr)	Returns critical DIO registers of associated DIO module
Service ID	NA	
Sync / Async	Sync	
Reentrancy	Non Reentrant	
Parameter in	ChannelId	Channel Identifier
Parameters out	DioRegRbPtr	A pointer of type Dio_RegisterReadbackType, which holds the read back register values
Return Value	Standard return type	E_OK or E_NOT_OK in case of NULL buffer pointer.

Design Identifier	**Description**
MCAL-5481	DioRegisterReadbackApi

Design Identifier	Description
MCAL-4945	GPIO: Safety Diagnostics: GPIO4 - Software readback of written configuration
MCAL-4944	GPIO: Safety Diagnostics: GPIO3 - Periodic software readback of static configuration registers
MCAL-4942	GPIO: Safety Diagnostics: GPIO1 - SoftwareTest of Function Using I/O Loopback

7 Performance Objectives

7.1 Resource Consumption Objectives

ROM - Program(KB)	ROM - Data(KB)	RAM - Program(KB)	RAM - Data(KB)	Stack Size (KB)	EEPROM (KB)	% CPU Utilization
5	NA	NA	1	2	NA	NA

7.2 Critical timing and Performance

Not Applicable

8 Decision Analysis & Resolution (DAR)

Sections below list some of the important design decisions and rational behind those decision.

8.1 Supporting Different SoCs

There are different numbers of pins in different SOCs as well as different instances available in domains such as wakeup and main domain. The Driver can also be hosted on different cores. Multiple approaches are available to support to support this and following are top 2 options.

No.	Decision Criteria	Alternatives	Selected alternative	Rationale	Trade-offs
1	Should be scalable (minimal changes to change reserved pins map) and easy to maintain.	**Separate Project per SoC :** For each SoC, a different configurator project is created and reserved pins are hard coded / checked in configuration generation. **Advantages:** • Simple to implement. No overhead in configuration. **Disadvantages:** • Not scalable as core could potentially change (that hosts AUTOSAR). Would require different project for a combination of SoC/core • Low ease of use, as customers will have to use right version of the configurator project. Potentially, customer can generate wrong configuration.	One Project and conditional generation of configuration	It is scalable and minimizes generation of wrong configuration .	Configuration development effort is high.

No.	Decision Criteria	Alternatives	Selected alternative	Rationale	Trade-offs
		One Project and conditional generation of configuration : Multiple SoCs (or core) supported in one project. Add conditional checks while generating/ validating pins **Advantages:** • Scalability and ease of use. **Disadvantages:** • Configuration development effort is high.			

8.2 Width of Port

AUTOSAR specification doesn't explicitly specify the width of the PORT, its dependent on the underlying hardware. As per the hardware specification each port is 16 bits wide. However, 2 ports are represented by a single 32 bit register in the hardware. This would require special handling, especially when the ASYNC APIs have to supported.

No.	Decision Criteria	Alternatives	Selected alternative	Rationale	Trade-offs

1	Low complexity implementation and scalable	**32 bits per port**
		• **Advantages:**
		  • No Exclusive area required to implement SYNC calls
		  • Usability (Similar to previous generation of SoC, customers will find it simpler to use)
		• **Disadvantages:**
		  • Doesn't reflect internal organization of port (HW). Customer will have to read this design document for complete picture of usage
			32 bits per port	It minimizes generation use of exclusive areas	Doesn't reflect internal organization of port (HW).

		16 bits per port
		• **Advantages:**
		  • Maps directly to hardware organization
		• **Disadvantages:**
		  • Would require additional exclusive area to implement SYNC APIs. As the 2 ports map to a single 32 bit register.			

9 Testing Guidelines

The sections below identify some of the aspects of design that would require emphasis during testing of this design implementation

- **Channel Read/ Write**
  - Test cases shall check for valid pin configurations
- **Channel Group Read/ Write**
  - Test cases shall exercise grouping of pins and test for validity
- **Negative Tests**
  - Set the direction of channel/port/channel group as INPUT and check for write
  - Check if read/write is possible for invalid configurations.

10 Template Revision History

Author Name	Description	Version	Date
Yaniv Machani	Initial version	0.1	03 Oct 2018
Yaniv Machani	Updated to include EP views	0.4	02 Nov 2018
Yaniv Weizman	Restructuring and editing to further meet the A-SPICE and EP requirements	0.5	27 Dec 2018
Yaniv Weizman	Adding link to Architecture review template	0.6	22 Oct 2019
Yaniv Weizman	Adding requirement type column for requirements table (Functional/Non-Functional). Adding DAR table	0.65	13 Nov 2019

Author Name	Description	Version	Date
Yaniv Weizman	Adding tables for Testing guidelines	0.7	18 Nov 2019
Krishna	Updated based on ASPICE requirements	0.8	20 Aug 2020
Krishna	Updated based on the feedback from Jon N	0.9	09 Oct 2020
Krishna	Updated the traceability scheme	1.0	17 Dec 2020