User Manual

for S32K14X PORT Driver

Document Number: UM2PORTASR4.3 Rev0001R1.0.1
Rev. 1.0

Contents

Section number **Title** **Page**

Chapter 1
Revision History

Table 1-1. Revision History

Revision	Date	Author	Description
1.0	21/06/2019	NXP MCAL Team	Updated version for ASR 4.3.1S32K14XR1.0.1

Chapter 2 Introduction

This User Manual describes NXP Semiconductors AUTOSAR Port (Port) for S32K14X .

AUTOSAR Port driver configuration parameters and deviations from the specification are described in Port Driver chapter of this document. AUTOSAR Port driver requirements and APIs are described in the AUTOSAR Port driver software specification document.

2.1 Supported Derivatives

The software described in this document is intented to be used with the following microcontroller devices of NXP Semiconductors .

Table 2-1. S32K14X Derivatives

NXP Semiconductors	s32k148_lqfp144, s32k148_lqfp176, s32k148_mapbga100, s32k146_lqfp144, s32k146_lqfp100, s32k146_lqfp64, s32k146_mapbga100, s32k144_lqfp100, s32k144_lqfp64, s32k144_mapbga100, s32k142_lqfp100, s32k142_lqfp64, s32k118_lqfp48, s32k118_lqfp64, s32k142_lqfp48, s32k144_lqfp48, s32k148_lqfp100

All of the above microcontroller devices are collectively named as S32K14X .

2.2 Overview

AUTOSAR (AUTomotive Open System ARchitecture) is an industry partnership working to establish standards for software interfaces and software modules for automobile electronic control systems.

AUTOSAR

- paves the way for innovative electronic systems that further improve performance, safety and environmental friendliness.
- is a strong global partnership that creates one common standard: "Cooperate on standards, compete on implementation".
- is a key enabling technology to manage the growing electrics/electronics complexity. It aims to be prepared for the upcoming technologies and to improve cost-efficiency without making any compromise with respect to quality.
- facilitates the exchange and update of software and hardware over the service life of the vehicle.

2.3 About this Manual

This Technical Reference employs the following typographical conventions:

Boldface type: Bold is used for important terms, notes and warnings.

Italic font: Italic typeface is used for code snippets in the text. Note that C language modifiers such "const" or "volatile" are sometimes omitted to improve readability of the presented code.

Notes and warnings are shown as below:

Note

This is a note.

2.4 Acronyms and Definitions

Table 2-2. Acronyms and Definitions

Term	Definition
API	Application Programming Interface
AUTOSAR	Automotive Open System Architecture
ASM	Assembler
BSMI	Basic Software Make file Interface
CAN	Controller Area Network
DEM	Diagnostic Event Manager
DET	Development Error Tracer
C/CPP	C and C++ Source Code
VLE	Variable Length Encoding

Table continues on the next page...

Table 2-2. Acronyms and Definitions (continued)

Term	Definition
N/A	Not Applicable
MCU	Micro Controller Unit
DIO	Digital Input Output

2.5 Reference List

Table 2-3. Reference List

#	Title	Version
1	Specification of Port Driver	AUTOSAR Release 4.3.1
2	S32K14X Reference Manual	Reference Manual, Rev. 9, 9/2018
3	S32K142 Mask Set Errata for Mask 0N33V (0N33V)	30/11/2017
4	S32K144 Mask Set Errata for Mask 0N57U (0N57U)	30/11/2017
5	S32K146 Mask Set Errata for Mask 0N73V (0N73V)	30/11/2017
6	S32K148 Mask Set Errata for Mask 0N20V (0N20V)	25/10/2018
7	S32K118 Mask Set Errata for Mask 0N97V (0N97V)	07/01/2019

Chapter 3
Driver

3.1 Requirements

Requirements for this driver are detailed in the AUTOSAR 4.3 Rev0001Port Driver Software Specification document (See Table Reference List).

3.2 Driver Design Summary

This module provides the service for initializing the whole PORT structure of the microcontroller. Many ports and port pins can be assigned to various functionalities, e.g.

- General purpose I/O
- ADC
- SPI
- SCI
- PWM
- CAN
- LIN
- etc

For this reason, there is an overall configuration and initialization of this port structure. The configuration and mode of these port pins is microcontroller and ECU dependent.

Port initialisation data are written to each port as efficiently as possible. This PORT driver module completes the overall configuration and initialisation of the port structure which is used in the DIO driver module. Therefore, the DIO driver works on pins and ports which are configured by the PORT driver.

The PORT driver is initialised prior to use of the DIO functions. Otherwise DIO functions will exhibit undefined behaviour.

3.3 Hardware Resources

The hardware configured by the Port driver is PORT (Port Control and Interrupts).

Every PortPin configured in a PortContainer of the Port plugin can be mapped to one and only one microcontroller pin. The following steps must be followed in order to correctly map a Port plugin pin over a specific microcontroller pin:

- 1. Open the S32K1xx_IO_Signal_Description_Input_Multiplexing.xlsx Excel file attached to the Reference Manual
- 2. Go to 'IO Signal Table' sheet
- 3. Identify the microcontroller pin you want to use (eg. PTC7]), searching after the values in columns 'Module' and 'Function'.
- 3. Compute the number of the PCR (Pin Control Register) associated to the identified pin, using the following information: S32K14x platforms have 5 consecutive ports, listed as A to E and numbered from 0 to 4, like below:
    - 0 - PORTA
    - 1 - PORTB
    - 2 - PORTC
    - 3 - PORTD
    - 4 - PORTE

  Each of the 5 ports have a number of 32 pins, such that the pins are allocated to ports like below:
    - 0-31 -> PORTA
    - 32-63 -> PORTB
    - 64-95 -> PORTC
    - 96-127 -> PORTD
    - 128-159 -> PORTE

  The PCR number for a given pin (eg. PTC7) is computed like this:
    - Take the port information from the pin name (eg. C for PTC7) and multiply it's corresponding numeric identifier with 32
    - Take the pin information from the pin name (eg. 7 for PTC7)
    - Add the 2 values obtained above and note down the result (eg. 71 for PTC7)
- 4. Go to port container inside the Port plugin where you want to add the pin
- 5. Add a new PortPin in the port container list then double click the newly added PortPin to open it's properties
- 6. Go to the 'PortPinPcr' attribute and type the number noted down at above
- 7. Go to the 'PortPin Mode' attribute and choose the functionality you want to use for the selected pin

3.4 Deviation from Requirements

The driver deviates from the AUTOSAR Port Driver software specification in some places. Table identifies the AUTOSAR requirements that are not fully implemented, implemented differently, or out of scope for the Port driver. Table Table 3-1 provides Status column description.

Table 3-1. Deviations Status Column Description

Term	Definition
N/A	Not available
N/T	Not testable
N/S	Out of scope
N/I	Not implemented
N/F	Not fully implemented

Below table identifies the AUTOSAR requirements that are not fully implemented, implemented differently, or out of scope for the driver.

Table 3-2. Driver Deviations Table

Requirement	Status	Description	Notes
SWS_Port_00205	N/I	Port_Lcfg.c shall include Port_MemMap.h and Port.h.	Currently no support for link-time configuration is provided.
ECUC_Port_00128	N/A	Name : PortPinInitialMode {PORT_PIN_INITIAL_MODE} Description : Port pin mode from mode list for use with Port_Init() function. Range: PORT_PIN_MODE_ADC PORT_PIN_MODE_CAN PORT_PIN_MODE_DIO PORT_PIN_MODE_DIO_GPT PORT_PIN_MODE_DIO_WDG PORT_PIN_MODE_FLEXRAY PORT_PIN_MODE_ICUPort PORT_PIN_MODE_LINPort PORT_PIN_MODE_MEMPort PORT_PIN_MODE_PWMPort PORT_PIN_MODE_SPIPort	Currently implemented in a different mode in MCAL 4.0.
ECUC_Port_00130	N/A	Name : PortPinInitialMode {PORT_PIN_INITIAL_MODE} Description : Port pin mode from mode list. Note that more than one mode is allowed by default. That way it is e.g. possible to combine DIO with another mode such as ICU. Range: PORT_PIN_MODE_ADC PORT_PIN_MODE_CAN PORT_PIN_MODE_DIO	Replaced by requirement CPR-MCAL-781.port

Table continues on the next page...

Table 3-2. Driver Deviations Table (continued)

Requirement	Status	Description	Notes
		PORT_PIN_MODE_DIO_GPT PORT_PIN_MODE_DIO_WDG PORT_PIN_MODE_FLEXRAY PORT_PIN_MODE_ICUPort PORT_PIN_MODE_LINPort PORT_PIN_MODE_MEMPort PORT_PIN_MODE_PWMPort PORT_PIN_MODE_SPIPort	
SWS_Port_00227	N/S	These requirements are not applicable to this specification. (SRS_BSW_00005, SRS_BSW_00006, SRS_BSW_00007, SRS_BSW_00010, SRS_BSW_00160, SRS_BSW_00161, SRS_BSW_00162, SRS_BSW_00164, SRS_BSW_00167, SRS_BSW_00168, SRS_BSW_00170, SRS_BSW_00172, SRS_BSW_00307, SRS_BSW_00308, SRS_BSW_00309, SRS_BSW_00321, SRS_BSW_00325, SRS_BSW_00326, SRS_BSW_00328, SRS_BSW_00329, SRS_BSW_00330, SRS_BSW_00331, SRS_BSW_00333, SRS_BSW_00334, SRS_BSW_00335, SRS_BSW_00336, SRS_BSW_00341, SRS_BSW_00342, SRS_BSW_00343, SRS_BSW_00344, SRS_BSW_00347, SRS_BSW_00355, SRS_BSW_00357, SRS_BSW_00359, SRS_BSW_00360, SRS_SPAL_12463, SRS_SPAL_12462, SRS_SPAL_12265, SRS_SPAL_12092, SRS_SPAL_12078, SRS_SPAL_12077, SRS_SPAL_12067, SRS_SPAL_12064, SRS_SPAL_12129, SRS_SPAL_12075, SRS_SPAL_12063, SRS_SPAL_12169, SRS_SPAL_00157, SRS_SPAL_12069, SRS_SPAL_12068, SRS_SPAL_12267, SRS_SPAL_12056, SRS_BSW_00440, SRS_BSW_00439, SRS_BSW_00437, BSW00434, SRS_BSW_00433, SRS_BSW_00432, BSW00431, SRS_BSW_00429, SRS_BSW_00428, SRS_BSW_00427, SRS_BSW_00426, SRS_BSW_00425, SRS_BSW_00424, SRS_BSW_00423, BSW00421, BSW00420, SRS_BSW_00419, SRS_BSW_00417, SRS_BSW_00416, SRS_BSW_00413, SRS_BSW_00398, SRS_BSW_00395, SRS_BSW_00387, SRS_BSW_00378, SRS_BSW_00377, SRS_BSW_00376, SRS_BSW_00375, SRS_BSW_00373, SRS_BSW_00371, SRS_BSW_00370)	This is not a requirement

As a deviation from standard:

Port_PBcfg_< VariantNo >.c files will contain the definition for all parameters that are variant aware, independent of the configuration class that will be selected (PC, LT, PB).

Port_Cfg.c file will contain the definition for all parameters that are not variant aware.

3.5 PORT Driver limitations

S32K supports to set input pin to high-Z. However, it is not available in S32K11X (follow newest RM chapter 13). When in GPIO mode (PORT_PCRn[MUX] programmed to 0x1), the register PIDR cannot be written to 0x1. This programming helps to disable the input mode and this is not possible to achieve in S32K11x. The implication of this is that when the GPIO is configured in output mode (PDDR[n] programmed to 1), the data written on PDOR register would be reflected on PDIR after some delay if the output does not toggle to often. If the pad needs to be tristated, the PORT_PCRn[MUX] needs to be 00.

3.6 Driver usage and configuration tips

The Port driver is responsible with configuring the functionality that should be active on a platform hardware pin. The information about the functionalities available on each of the hardware pins of the platform can be found in the S32K1xx_IO_Signal_Description_Input_Multiplexing.xlsx Excel file attached to the Reference Manual.

The Port plugin allows the user to configure each pin’s functionality using 2 distinct mechanisms:

- A. Define the functionality of a specific pin. This can be done by adding a new entry in the PortContainer/PortPin list and setting the attributes of the pin. The following steps should be followed:
  - 1. Open the S32K1xx_IO_Signal_Description_Input_Multiplexing.xlsx Excel file attached to the Reference Manual
  - 2. Go to 'IO Signal Table' sheet
  - 3. Identify the microcontroller pin you want to use (eg. PTC7]), searching after the values in columns 'Module' and 'Function'.
  - 4. Compute the number of the PCR (Pin Control Register) associated to the identified pin, using the following information: S32K14x platforms have 5 consecutive ports, listed as A to E and numbered from 0 to 4, like below:
    - 0 - PORTA
    - 1 - PORTB

        - 2 - PORTC
        - 3 - PORTD
        - 4 - PORTE

      Each of the 5 ports have a number of 32 pins, such that the pins are allocated to ports like below:

        - 0-31 -> PORTA
        - 32-63 -> PORTB
        - 64-95 -> PORTC
        - 96-127 -> PORTD
        - 128-159 -> PORTE

      The PCR number for a given pin (eg. PTC7) is computed like this:

        - Take the port information from the pin name (eg. C for PTC7) and multiply it's corresponding numeric identifier with 32
        - Take the pin information from the pin name (eg. 7 for PTC7)
        - Add the 2 values obtained above and note down the result (eg. 71 for PTC7)

    - 5. Go to port container inside the Port plugin where you want to add the pin
    - 6. Add a new PortPin in the port container list then double click the newly added PortPin to open it's properties
    - 7. Go to the 'PortPinPcr' attribute and type the number noted down at step A.4
    - 8. Go to the 'PortPin Mode' attribute and choose the functionality you want to use for the selected pin
    - 9. Look at the other attributes of the PortPin and set them to the desired values
- B. Define pins that should not be touched by any Port driver functionality, including Port_Init() function. This option allows the user to configure a list of pins for which the driver will not touch their PCRs, leaving them containing the reset values. This list is named UnTouchedPortPin and is available in the PortConfigSet container and adding new entries in this list should follow the next steps:
    - 1. Open the S32K1xx_IO_Signal_Description_Input_Multiplexing.xlsx Excel file attached to the Reference Manual.
    - 2. Go to 'IO Signal Table' sheet.
    - 3. Identify the microcontroller pin you want the Port driver to not touch (eg. PTC7), searching after the values in columns 'Module' and 'Function'
    - 4. Go to UnTouchedPortPin list inside the PortConfigSet container
    - 5. Add a new entry in the list and double click it to open it's properties
    - 6. Go to the 'PortPin PCR' attribute and type the number noted down at step A.4
- C. Define the settings for all platform hardware pins that were not configured using mechanism described at point A or point B. This option allows the user to configure all platform pins that are not explicitely configured by the user as GPIOs, with some specific settings. These settings are available in the container NotUsedPortPin where the user can define the pin direction (in or out), pin level (high or low), pull up/down.

Every single platform hardware pin is configured by the Port driver, either by mechanism A, B or C. There are no hardware pins that are left untouched by the Port driver Port_Init() API.

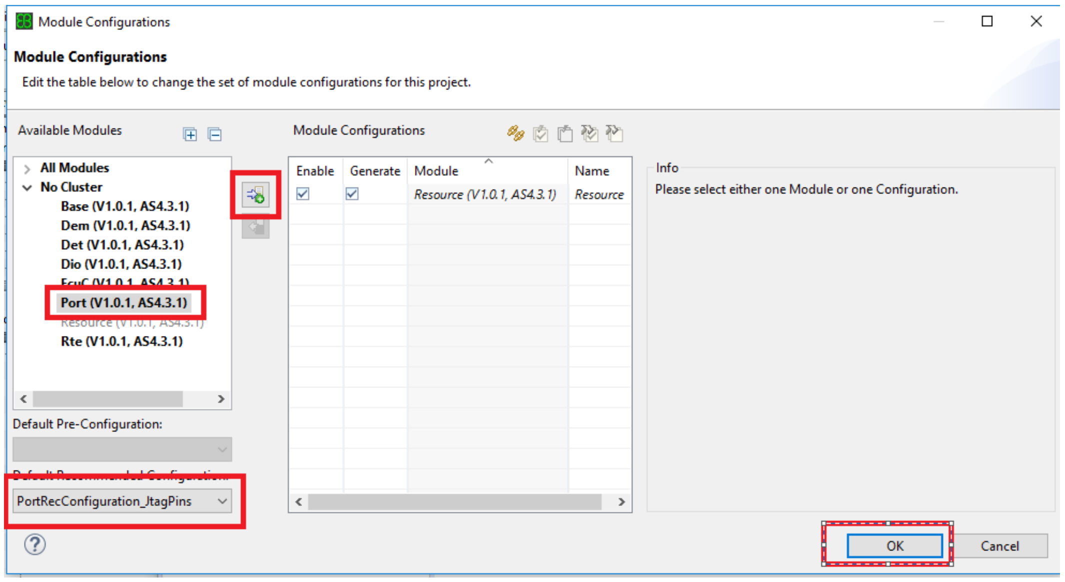
For this reason, if the platform contains hardware pins that need to have certain non GPIO functionalities, these pins must be explicitly added in the Port configuration using mechanism A. Otherwise, they will be configured by Port_Init() API as GPIOs.

Important note

In order to be able to use the debug capabilities, the JTAG pins need to be configured in the Port driver using mechanism A. This means that the following pins/functionalities need to be added in the PortContainer/PortPin list:

- PortPin_JTAG_TDI having PortPinPcr set to 69 and PortPinMode set to JTAG_TDI
- PortPin_JTAG_TDO having PortPinPcr set to 10 and PortPinMode set to JTAG_TDO
- PortPin_JTAG_TCK having PortPinPcr set to 68 and PortPinMode set to JTAG_TCLK_SWD_CLK
- PortPin_JTAG_TMS having PortPinPcr set to 4 and PortPinMode set to JTAG_TMS_SWD_DIO
- PortPin_Reset_b having PortPinPcr set to 5 and PortPinMode set to RESET_b

In order to be easier to add the above pins into the configuration, no need to manually add each pin in the plugin, the Port configuration must be selected along with Default recommended configuration as: PortRecConfiguration_JtagPins.


Module Configurations

Module Configurations

Edit the table below to change the set of module configurations for this project.

Available Modules

- All Modules
- No Cluster
  - Base (V1.0.1, AS4.3.1)
  - Dem (V1.0.1, AS4.3.1)
  - Det (V1.0.1, AS4.3.1)
  - Dio (V1.0.1, AS4.3.1)
  - EcuC (V1.0.1, AS4.3.1)
  - Port (V1.0.1, AS4.3.1)
  - Resource (V1.0.1, AS4.3.1)
  - Rte (V1.0.1, AS4.3.1)

Default Pre-Configuration:

Default Recommended Configuration:

PortRecConfiguration_JtagPins

Module Configurations

Enable	Generate	Module	Name
✓	✓	Resource (V1.0.1, AS4.3.1)	Resource

Info

Please select either one Module or one Configuration.

Figure 3-1. How to configure JTAG pins

Autosar extension functionality

- Support to run driver’s code from User Mode. This option is configurable on/off per entire driver, using the checkbox ‘Enable Port User Mode Support’ in PortGeneral container. When this parameter is enabled, the Port module will adapt to run from user mode so that the registers under protection can be accessed from user mode. For more information, please see the IM chapter 'User Mode Support'.
- Port SetPinMode Does Not Touch GPIO Levels. This option is configurable on/off and it affects the functionality of the Port_SetPinMode() API. When not checked, the function Port_SetPinMode() will set the output level of the pin to the value configured in the PortPinLevelValue combo when called at run time to change mode of a pin from alternate function to GPIO. When checked, the function Port_SetPinMode() will not touch the output level of the pin when called at run time to change mode of a pin from alternate function to GPIO.

3.7 Runtime Errors

This driver doesn't generate any runtime error.

3.8 Software specification

The following sections contains driver software specifications.

3.8.1 Define Reference

Constants supported by the driver are as per AUTOSAR Port Driver software specification Version 4.3 Rev0001 .

3.8.1.1 Define PORT_E_DIRECTION_UNCHANGEABLE

Port Pin Direction not configured as changeable.

Details:

Det Error value, returned by Port_SetPinDirection if the passed PortPin have unchangeable direction.

Table 3-3. Define PORT_E_DIRECTION_UNCHANGEABLE Description

Name	PORT_E_DIRECTION_UNCHANGEABLE
Initializer	(uint8)0x0B

3.8.1.2 Define PORT_INSTANCE_ID

Instance IDs

Details:

Instance ID of port driver.

Table 3-4. Define PORT_INSTANCE_ID Description

Name	PORT_INSTANCE_ID
Initializer	(uint8)0x0

3.8.1.3 Define PORT_E_MODE_UNCHANGEABLE

API `Port_SetPinMode()` service called when mode is unchangeable.

Details:

Det Error value, returned by Port_SetPinMode function if the passed PortPin have a unchangeable Mode.

Table 3-5. Define PORT_E_MODE_UNCHANGEABLE Description

Name	PORT_E_MODE_UNCHANGEABLE
Initializer	(uint8)0x0E

3.8.1.4 Define PORT_E_INIT_FAILED

API `Port_Init()` service called with wrong parameter.

Details:

Det Error value, returned by Port_Init function if Port_Init is called with wrong parameter.

Table 3-6. Define PORT_E_INIT_FAILED Description

Name	PORT_E_INIT_FAILED
Initializer	(uint8)0x0C

3.8.1.5 Define PORT_E_PARAM_INVALID_MODE

API `Port_SetPinMode()` service called when mode is invalid.

Details:

Det Error value, returned by Port_SetPinMode function if the passed PortPinMode is invalid.

Table 3-7. Define PORT_E_PARAM_INVALID_MODE Description

Name	PORT_E_PARAM_INVALID_MODE
Initializer	(uint8)0x0D

3.8.1.6 Define PORT_E_PARAM_PIN

Invalid Port Pin ID requested.

Details:

Det Error value, returned by Port_SetPinDirection and Port_SetPinMode if a wrong PortPin ID is passed.

Table 3-8. Define PORT_E_PARAM_PIN Description

Name	PORT_E_PARAM_PIN
Initializer	(uint8)0x0A

3.8.1.7 Define PORT_E_PARAM_POINTER

API service called with NULL Pointer Parameter.

Details:

Det Error value, returned by Port_GetVersionInfo function if API is called with NULL Pointer Parameter.

Table 3-9. Define PORT_E_PARAM_POINTER Description

Name	PORT_E_PARAM_POINTER
Initializer	(uint8)0x10

3.8.1.8 Define PORT_E_UNINIT

API service called without module initialization.

Details:

Det Error value, returned by a function if API service called prior to module initialization.

Table 3-10. Define PORT_E_UNINIT Description

Name	PORT_E_UNINIT
Initializer	(uint8)0x0F

3.8.1.9 Define PORT_GETVERSIONINFO_ID

API service ID for PORT get version info function.

Details:

Parameters used when raising an error/exception.

Table 3-11. Define PORT_GETVERSIONINFO_ID Description

Name	PORT_GETVERSIONINFO_ID
Initializer	(uint8)0x03

3.8.1.10 Define PORT_INIT_ID

API service ID for PORT Init function.

Details:

Parameters used when raising an error/exception.

Table 3-12. Define PORT_INIT_ID Description

Name	PORT_INIT_ID
Initializer	(uint8)0x00

3.8.1.11 Define PORT_SETPINDIRECTION_ID

API service ID for PORT set pin direction function.

Details:

Parameters used when raising an error/exception.

Table 3-13. Define PORT_SETPINDIRECTION_ID Description

Name	PORT_SETPINDIRECTION_ID
Initializer	(uint8)0x01

3.8.1.12 Define PORT_SETPINMODE_ID

API service ID for PORT set pin mode.

Details:

Parameters used when raising an error/exception.

Table 3-14. Define PORT_SETPINMODE_ID Description

Name	PORT_SETPINMODE_ID

Table continues on the next page...

Table 3-14. Define PORT_SETPINMODE_ID Description (continued)

Initializer	(uint8)0x04

3.8.1.13 Define PORT_REFRESHPINDIRECTION_ID

API service ID for PORT refresh pin direction function.

Details:

Parameters used when raising an error/exception.

Table 3-15. Define PORT_REFRESHPINDIRECTION_ID Description

Name	PORT_REFRESHPINDIRECTION_ID
Initializer	(uint8)0x02

3.8.1.14 Define PORT_ALT0_FUNC_MODE

Port Alternate 0 Mode.

Table 3-16. Define PORT_ALT0_FUNC_MODE Description

Name	PORT_ALT0_FUNC_MODE
Initializer	((Port_PinModeType)0)

3.8.1.15 Define PORT_GPIO_MODE

Port GPIO Mode.

Table 3-17. Define PORT_GPIO_MODE Description

Name	PORT_GPIO_MODE
Initializer	((Port_PinModeType)1)

3.8.1.16 Define PORT_ALT2_FUNC_MODE

Port Alternate 2 Mode.

Table 3-18. Define PORT_ALT2_FUNC_MODE Description

Name	PORT_ALT2_FUNC_MODE
Initializer	((Port_PinModeType)2)

3.8.1.17 Define PORT_ALT3_FUNC_MODE

Port Alternate 3 Mode.

Table 3-19. Define PORT_ALT3_FUNC_MODE Description

Name	PORT_ALT3_FUNC_MODE
Initializer	((Port_PinModeType)3)

3.8.1.18 Define PORT_ALT4_FUNC_MODE

Port Alternate 4 Mode.

Table 3-20. Define PORT_ALT4_FUNC_MODE Description

Name	PORT_ALT4_FUNC_MODE
Initializer	((Port_PinModeType)4)

3.8.1.19 Define PORT_ALT5_FUNC_MODE

Port Alternate 5 Mode.

Table 3-21. Define PORT_ALT5_FUNC_MODE Description

Name	PORT_ALT5_FUNC_MODE
Initializer	((Port_PinModeType)5)

3.8.1.20 Define PORT_ALT6_FUNC_MODE

Port Alternate 6 Mode.

Table 3-22. Define PORT_ALT1_FUNC_MODE Description

Name	PORT_ALT6_FUNC_MODE
Initializer	((Port_PinModeType)6)

3.8.1.21 Define PORT_ALT7_FUNC_MODE

Port Alternate 7 Mode.

Table 3-23. Define PORT_ALT7_FUNC_MODE Description

Name	PORT_ALT7_FUNC_MODE
Initializer	(Port_PinModeType)7

3.8.2 Enum Reference

Enumeration of all constants supported by the driver are as per AUTOSAR Port Driver software specification Version 4.3 Rev0001 .

3.8.2.1 Structure Port_PinDirectionType

Structure needed by`Port_SetPinDirection()`.

Details:

The structure`Port_PinDirectionType` Possible directions of a port pin.

Declaration:

```
typedef enum
{
    PORT_PIN_DISABLED = 0,
    PORT_PIN_IN,
    PORT_PIN_OUT,
    PORT_PIN_HIGH_Z
}
```

Table 3-24. Enumeration Port_PinDirectionType member description

Member	Description
PORT_PIN_DISABLED	No settings: the pin is not available.
PORT_PIN_IN	Sets port pin as input.

Table continues on the next page...

Table 3-24. Enumeration Port_PinDirectionType member description (continued)

Member	Description
PORT_PIN_OUT	Sets port pin as output.
PORT_PIN_HIGH_Z	Sets port pin as high-Z.

3.8.3 Function Reference

Functions of all functions supported by the driver are as per AUTOSAR Port Driver software specification Version 4.3 Rev0001 .

3.8.3.1 Function Port_Init

Initializes the Port Driver module.

Details:

The function `Port_Init()`will initialize ALL ports and port pins with the configuration set pointed to by the parameter ConfigPtr. It always requires an input as a valid pointer.

Pre:

Function `Port_Init()`should not have been called before.

Post: `Port_Init()` must be called before all other Port Driver module's functions otherwise no operation can occur on the MCU ports and port pins.

Prototype: `void Port_Init(const Port_ConfigType *ConfigPtr);`

Table 3-25. Port_Init Arguments

Type	Name	Direction	Description
const `Port_ConfigType*`	ConfigPtr	**input**	A pointer to the structure which contains initialization parameters.

3.8.3.2 Function Port_SetPinDirection

Sets the port pin direction.

Details:

The function `Port_SetPinDirection()` will set the port pin direction during runtime.

Pre: `Port_Init()`must have been called first. In order to change the pin direction the PortPinDirectionChangeable flag must have been set to TRUE.

Prototype: `void Port_SetPinDirection(Port_PinType Pin, Port_PinDirectionType Direction);`

Table 3-26. Port_SetPinDirection Arguments

Type	Name	Direction	Description
Port_PinType	Pin	**input**	Pin ID number.
Port_PinDirectionType	Direction	**input**	Port Pin direction.

3.8.3.3 Function Port_SetPinMode

Sets the port pin mode.

Details:

The function `Port_SetPinMode()` will set the port pin mode of the referenced pin during runtime.

Pre: `Port_Init()`must have been called first.

Prototype: `void Port_SetPinMode(Port_PinType Pin, Port_PinModeType Mode);`

Table 3-27. Port_SetPinMode Arguments

Type	Name	Direction	Description
Port_PinType	Pin	**input**	Pin ID number.
Port_PinModeType	Mode	**input**	New Port Pin mode to be set on port pin.

3.8.3.4 Function Port_RefreshPortDirection

Refreshes port direction.

Details:

This function will refresh the direction of all configured ports to the configured direction. The PORT driver will exclude from refreshing those port pins that are configured as "pin direction changeable during runtime".

Pre: `Port_Init()`must have been called first.

Prototype: `void Port_RefreshPortDirection(void);`

3.8.3.5 Function Port_GetVersionInfo

Returns the version information of this module.

Details:

The function `Port_GetVersionInfo()` will return the version information of this module. The version information includes:

- Module Id,
- Vendor Id,
- Vendor specific version numbers.

Pre: None

Prototype: `void Port_GetVersionInfo(Std_VersionInfoType *versioninfo);`

Table 3-28. Port_GetVersionInfo Arguments

Type	Name	Direction	Description
Std_VersionInfoType *	versioninfo	**input, output**	Pointer to where to store the version information of this module.

3.8.4 Structs Reference

Data structures supported by the driver are as per AUTOSAR Port Driver software specification Version 4.3 Rev0001 .

3.8.4.1 Structure Port_ConfigType

Structure needed by`Port_Init()`.

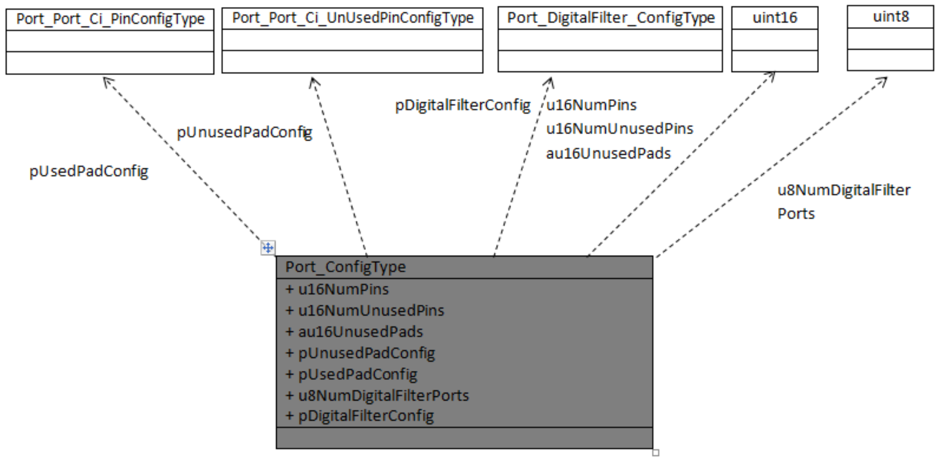


Figure 3-2. Struct Port_ConfigType

Details:

The structure`Port_ConfigType` is a type for the external data structure containing the initialization data for the PORT Driver.

Note

The user must use the symbolic names defined in the configuration tool.

Declaration:

```
typedef struct
{
    VAR(uint16, AUTOMATIC) u16NumPins;
    VAR(uint16, AUTOMATIC) u16NumUnusedPins;
    P2CONST(uint16, AUTOMATIC, PORT_APPL_CONST) pau16UnusedPads;
    P2CONST(Port_Port_Ci_UnUsedPinConfigType, AUTOMATIC, PORT_APPL_CONST)
pUnusedPadConfig;
    P2CONST(Port_Port_Ci_PinConfigType,       AUTOMATIC, PORT_APPL_CONST)
pUsedPadConfig;
    VAR(uint8, AUTOMATIC)  u8NumDigitalFilterPorts;
    P2CONST(Port_DigitalFilter_ConfigType, AUTOMATIC, PORT_APPL_CONST) pDigitalFilterConfig;
 } Port_ConfigType;
```

Table 3-29. Structure Port_ConfigType member description

Member	Description
u16NumPins	Number of used pads (to be configured).
u16NumUnusedPins	Number of unused pads.
pau16UnusedPads	Unused pad id's array.
pUnusedPadConfig	Unused pad configuration.
pUsedPadConfig	Used pads data configuration

Table continues on the next page...

Table 3-29. Structure Port_ConfigType member description (continued)

Member	Description
u8NumDigitalFilterPorts	Number of configured digital filter ports
pDigitalFilterConfig	Digital filter ports configuration

3.8.4.2 Structure Port_Port_Ci_PinConfigType

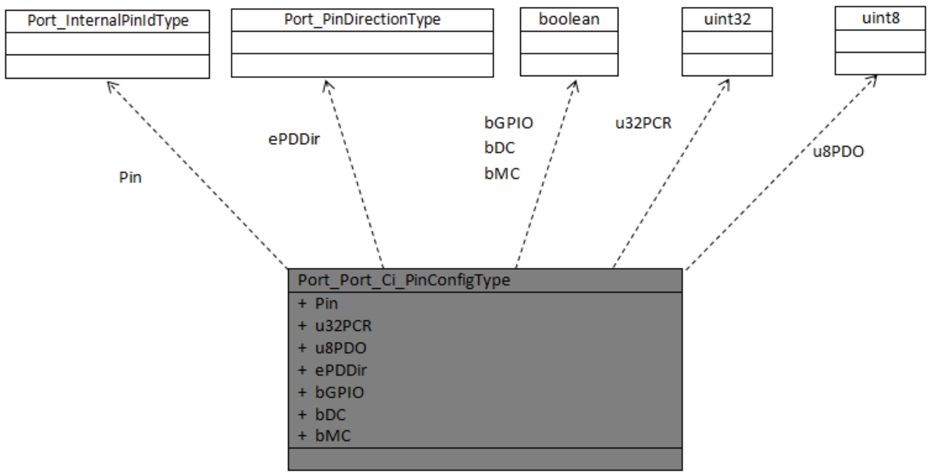


Figure 3-3. Struct Port_Port_Ci_PinConfigType

Details:

The structure `Port_Port_Ci_PinConfigType` contains all configuration parameters of a single pin identified by @p PORT Pin..

Note

The user must use the symbolic names defined in the configuration tool.

Declaration:

```
typedef struct
{
    VAR(Port_InternalPinIdType, AUTOMATIC)   Pin;
    VAR(uint32,                 AUTOMATIC)   u32PCR;
    VAR(uint8,                  AUTOMATIC)   u8PDO;
    VAR(Port_PinDirectionType,  AUTOMATIC)   ePDDir;
    VAR(boolean,                AUTOMATIC)   bGPIO;
    VAR(boolean,                AUTOMATIC)   bDC;
    VAR(boolean,                AUTOMATIC)   bMC;
 } Port_Port_Ci_PinConfigType;
```

Table 3-30. Structure Port_Port_Ci_PinConfigType member description

Member	Description
Pin	Pin Defined on PORT.
u32PCR	Pad Control Register.
u8PDO	Pad Data Output.
ePDDir	Pad Data Direction.
bGPIO	GPIO initial mode.
bDC	Direction changebility.
bMC	Mode changebility.

3.8.4.3 Structure Port_Port_Ci_UnUsedPinConfigType

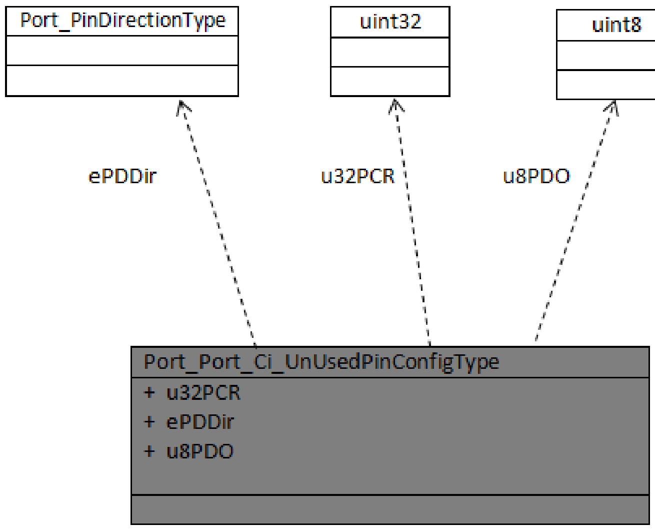


Figure 3-4. Struct Port_Port_Ci_UnUsedPinConfigType

Details:

The structure `Port_Port_Ci_UnUsedPinConfigType` contains all configuration parameters of a Default pin.

Note

The user must use the symbolic names defined in the configuration tool.

Declaration:

```
typedef struct
{
    VAR(uint32,                 AUTOMATIC)   u32PCR;
    VAR(Port_PinDirectionType,  AUTOMATIC)   ePDDir;
    VAR(uint8,                  AUTOMATIC)   u8PDO;
 } Port_Port_Ci_UnUsedPinConfigType;
```

Table 3-31. Structure Port_Port_Ci_UnUsedPinConfigType member description

Member	Description
u32PCR	Pad Control Register.
ePDDir	Pad Data Direction.
u8PDO	Pad Data Output.

3.8.4.4 Structure Port_DigitalFilter_ConfigType

Structure needed by `Port_Init()`.

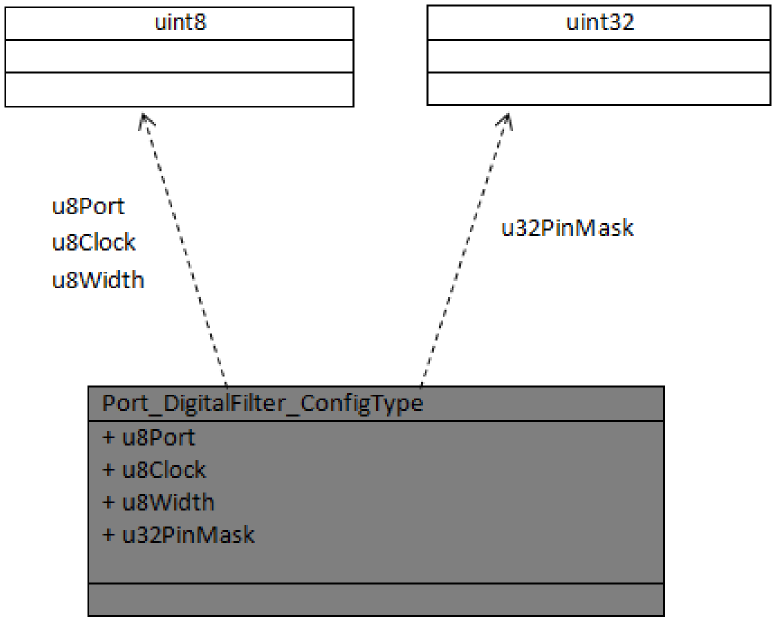


Figure 3-5. Struct Port_DigitalFilter_ConfigType

Details:

The structure `Port_DigitalFilter_ConfigType` contains all configuration parameters of a digital filter port.

Note

The user must use the symbolic names defined in the configuration tool.

Declaration:

```
typedef struct
{
    VAR(uint8,  AUTOMATIC)   u8Port;
    VAR(uint8,  AUTOMATIC)   u8Clock;
    VAR(uint8,  AUTOMATIC)   u8Width;
    VAR(uint32, AUTOMATIC)   u32PinMask;
 } Port_DigitalFilter_ConfigType;
```

Table 3-32. Structure Port_DigitalFilter_ConfigType member description

Member	Description
u8Port	Digital Filter Port.
u8Clock	Digital Filter Clock.
u8Width	Digital Filter Width.
u32PinMask	Mask of pins for which digital filter is enabled.

3.8.5 Types Reference

Types supported by the driver are as per AUTOSAR Port Driver software specification Version 4.3 Rev0001 .

3.8.5.1 Typedef Port_InternalPinIdType

It is the same with the index of the PCR register.

Type:`uint16`

3.8.5.2 Typedef Port_PinModeType

A port pin shall be configurable with a number of port pin modes (type Port_PinModeType). The type Port_PinModeType shall be used with the function call Port_SetPinMode

Type:`uint8`

3.8.5.3 Typedef Port_PinType

Data type for the symbolic name of a port pin.

Type:`uint32`

3.8.5.4 Typedef Port_Port_Ci_PadSelConfigType

Data type used for Pad Selection Multiplexed Configuration.

Type:`uint8`

3.8.5.5 Typedef Port_RegValueType

A port register shall be written with a 32 bits value (type Port_RegValueType). The type Port_RegValueType shall be used with the function call Port_SetPinMode

Type:`uint16`

3.9 Symbolic Names Disclaimer

All containers having the symbolic name tag set as true in the Autosar schema will generate defines like:

#define <Container_Short_Name> <Container_ID>

For this reason it is forbidden to duplicate the name of such containers across the MCAL configuration, or to use names that may trigger other compile issues (e.g. match existing #ifdefs arguments).

Chapter 4
Tresos Configuration Plug-in

This chapter describes the Tresos configuration plug-in for the Port Driver. The most of the parameters are described below.

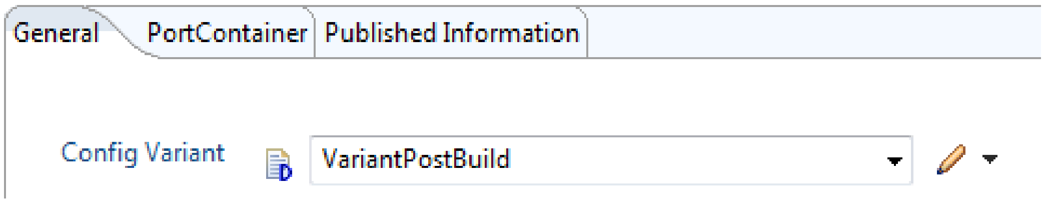
4.1 Configuration elements of Port

Included forms :

- IMPLEMENTATION_CONFIG_VARIANT
- PortGeneral
- PortConfigSet
- CommonPublishedInformation

4.2 Form IMPLEMENTATION_CONFIG_VARIANT

VariantPreCompile: Only precompile time configuration parameters. Only one set of parameters. VariantPostBuild: Mix of precompile and postbuild time configuration parameters. More sets of parameters. If Config Variant = VariantPreCompile, the files Port_Cfg.h and Port_Cfg.c should be used. If Config Variant = VariantPostBuild, the files Port_Cfg.h and Port_PBcfg.c should be used.



Config Variant: VariantPostBuild

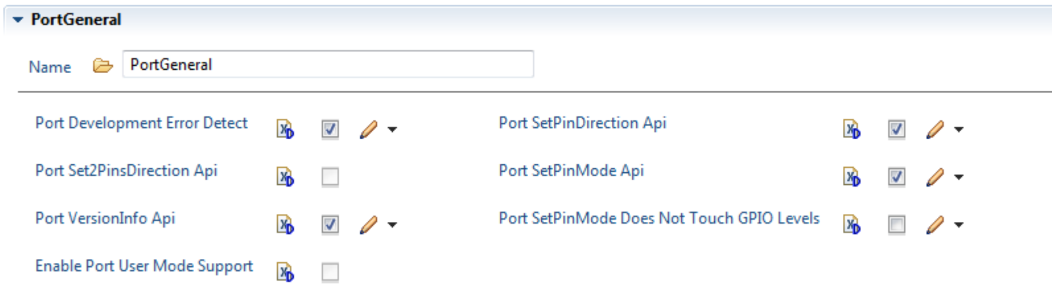
Figure 4-1. Tresos Plugin snapshot for IMPLEMENTATION_CONFIG_VARIANT

Table 4-1. Attribute IMPLEMENTATION_CONFIG_VARIANT detailed description

Property	Value
Label	Config Variant
Default	VariantPostBuild
Range	VariantPostBuild VariantPreCompile

4.3 PortGeneral

Module wide configuration parameters of the PORT driver.



Name	PortGeneral		
Port Development Error Detect	✓	Port SetPinDirection Api	✓
Port Set2PinsDirection Api	☐	Port SetPinMode Api	✓
Port VersionInfo Api	✓	Port SetPinMode Does Not Touch GPIO Levels	☐
Enable Port User Mode Support	☐		

Figure 4-2. Tresos Plugin snapshot for PortGeneral form.

4.3.1 PortDevErrorDetect (PortGeneral)

Switches the Development Error Detection and Notification ON or OFF.

Table 4-2. Attribute PortDevErrorDetect (PortGeneral) detailed description

Property	Value
Label	Port Development Error Detect
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	true

4.3.2 PortSetPinDirectionApi (PortGeneral)

Pre-processor switch to enable/disable the use of the function Port_SetPinDirection().

Table 4-3. Attribute PortSetPinDirectionApi (PortGeneral) detailed description

Property	Value
Label	Port SetPinDirection Api
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	true

4.3.3 PortSet2PinsDirectionApi (PortGeneral)

Pre-processor switch to enable/disable the use of the function Port_Set2PinsDirection().

Table 4-4. Attribute PortSet2PinsDirectionApi (PortGeneral) detailed description

Property	Value
Label	Port Set2PinsDirection Api
Type	BOOLEAN
Origin	NXP
Symbolic Name	false
Default	true

4.3.4 PortSetPinModeApi (PortGeneral)

Pre-processor switch to enable/disable the use of the function Port_SetPinMode().

Table 4-5. Attribute PortSetPinModeApi (PortGeneral) detailed description

Property	Value
Label	Port SetPinMode Api
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	true

4.3.5 PortVersionInfoApi (PortGeneral)

Pre-processor switch to enable/disable the API to read out the modules version information.

Table 4-6. Attribute PortVersionInfoApi (PortGeneral) detailed description

Property	Value
Label	Port VersionInfo Api
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	true

4.3.6 PortSetPinModeDoesNotTouchGpioLevel (PortGeneral)

Pre-processor switch. When not checked, the function Port_SetPinMode() will set the output level of the pin to the value configured in the PortPinLevelValue combo when called at run time to change mode of a pin from alternate function to GPIO. When checked, the function Port_SetPinMode() will not touch the output level of the pin when called at run time to change mode of a pin from alternate function to GPIO.

Table 4-7. Attribute PortSetPinModeDoesNotTouchGpioLevel (PortGeneral) detailed description

Property	Value
Label	Port SetPinMode Does Not Touch GPIO Levels
Type	BOOLEAN
Origin	NXP
Symbolic Name	false
Default	False

4.3.7 PortEnableUserModeSupport (PortGeneral)

This parameter is added in Port configuration in order to keep a consistent design over the entire set of MCAL drivers. It cannot be configured by the user and is always set to 'false'. There are no registers used by the driver which require special measures in order to be accessed from user mode, so Port driver can be run from either user or supervisor mode.

Table 4-8. Attribute PortEnableUserModeSupport (PortGeneral) detailed description

Property	Value
Label	Port Enable User Mode Support
Type	BOOLEAN

Table continues on the next page...

Table 4-8. Attribute PortEnableUserModeSupport (PortGeneral) detailed description (continued)

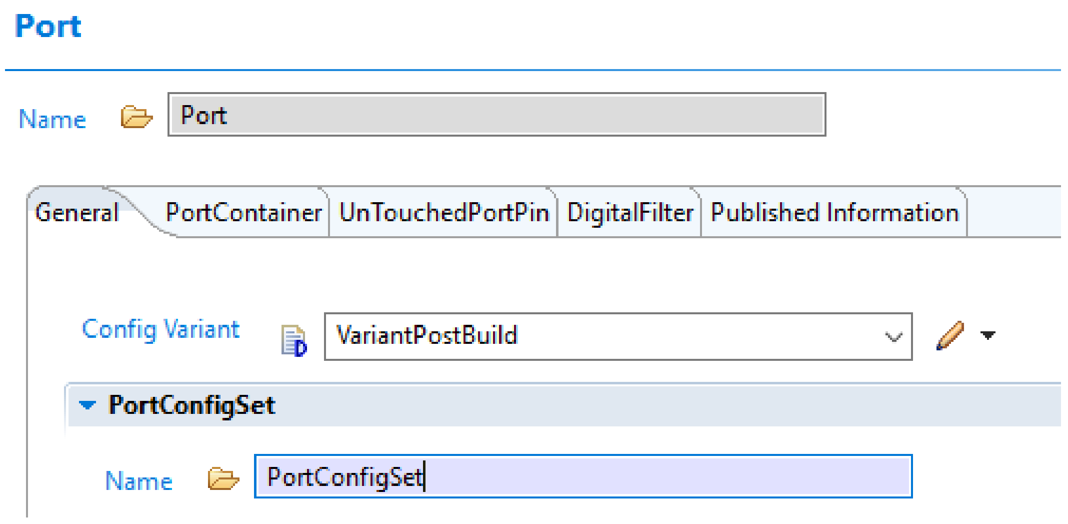
Property	Value
Origin	NXP
Symbolic Name	false
Default	False

4.4 PortConfigSet

This container contains a configuration of the PORT driver / PORT module.

Includes:

- PortContainer
- UnTouchedPortPin
- NotUsedPortPin



Name: Port

Config Variant: VariantPostBuild

Name: PortConfigSet

Figure 4-3. Tresos Plugin snapshot for PortConfigSet.

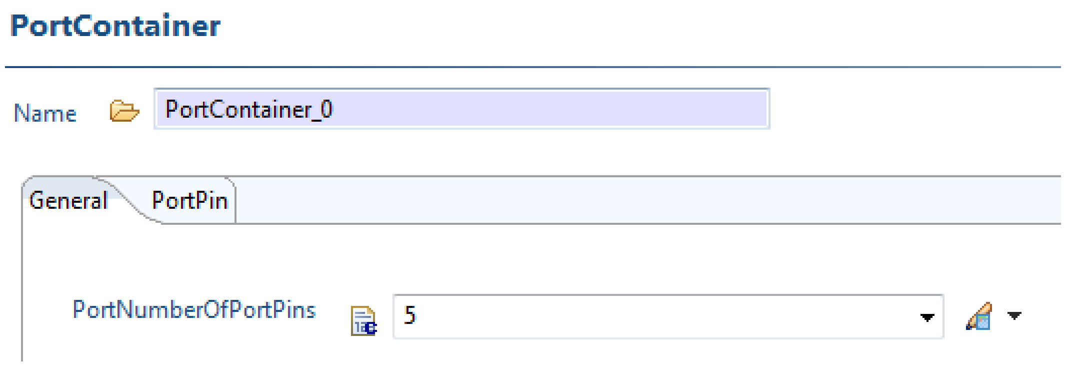
4.4.1 PortContainer

Container collecting the PortPins.

Is included by : PortConfigSet

Includes :

- PortPin


PortContainer

Name: PortContainer_0

PortNumberOfPortPins: 5

Figure 4-4. Tresos Plugin snapshot for PortContainer form.

4.4.1.1 PortNumberOfPortPins (PortContainer)

The number of specified PortPins in this PortContainer.

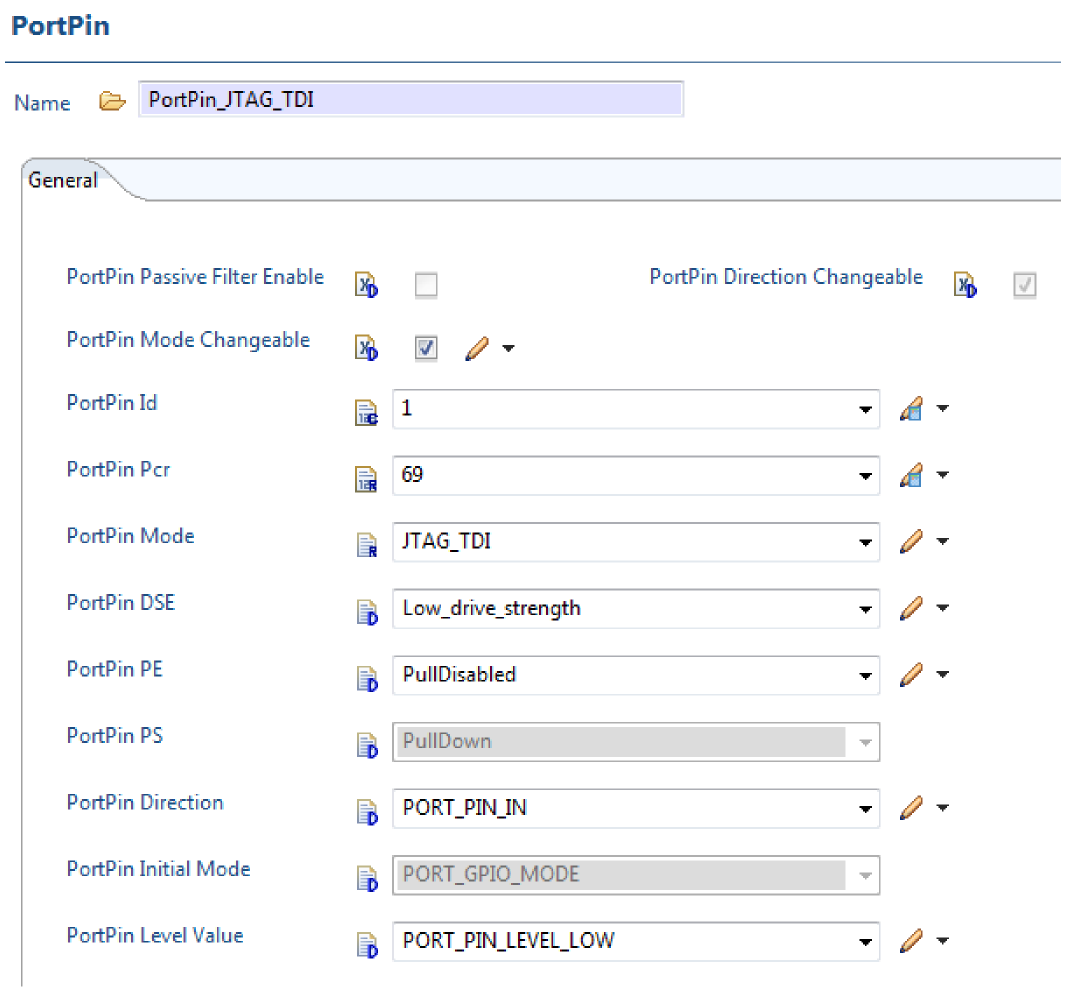
Table 4-9. Attribute PortNumberOfPortPins (PortContainer) detailed description

Property	Value
Label	PortNumberOfPortPins
Type	INTEGER
Origin	AUTOSAR_ECUC
Symbolic Name	false
Invalid	Range >=1 <=156

4.4.1.2 PortPin

Configuration of the individual port pins.

Is included by : PortContainer


PortPin

Name: PortPin_JTAG_TDI

General

PortPin Passive Filter Enable: ☐
PortPin Direction Changeable: ✓
PortPin Mode Changeable: ✓
PortPin Id: 1
PortPin Pcr: 69
PortPin Mode: JTAG_TDI
PortPin DSE: Low_drive_strength
PortPin PE: PullDisabled
PortPin PS: PullDown
PortPin Direction: PORT_PIN_IN
PortPin Initial Mode: PORT_GPIO_MODE
PortPin Level Value: PORT_PIN_LEVEL_LOW

Figure 4-5. Tresos Plugin snapshot for PortPin

4.4.1.2.1 PortPinDirectionChangeable (PortPin)

Enable/Disable the changeability for the configured Pin. Checked box means the Direction Changeability is enabled. This is an implementation specific parameter.

Table 4-10. Attribute PortPinDirectionChangeable (PortPin) detailed description

Property	Value
Label	PortPin Direction Changeable
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	true

4.4.1.2.2 PortPinModeChangeable (PortPin)

Parameter to indicate if the mode of a port pin is changeable during runtime. True: Port Pin mode changeable allowed. False: Port Pin mode changeable not permitted

Table 4-11. Attribute PortPinModeChangeable (PortPin) detailed description

Property	Value
Label	PortPin Mode Changeable
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	true

4.4.1.2.3 PortPinPassiveFilterEnable (PortPin)

Passive Filter Enable Passive filter configuration is valid in all digital pin muxing modes

Table 4-12. Attribute PortPin Passive Filter Enable(PortPin) detailed description

Property	Value
Label	PortPin Passive Filter Enable
Type	BOOLEAN
Origin	NXP
Symbolic Name	false
Default	false

4.4.1.2.4 PortPinId (PortPin)

Pin Id of the port pin. This value will be assigned to the symbolic name derived from the port pin container short name.

Table 4-13. Attribute PortPinId (PortPin) detailed description

Property	Value
Label	PortPin Id
Type	INTEGER
Origin	AUTOSAR_ECUC
Symbolic Name	true
Invalid	Range >=1 <=156

4.4.1.2.5 PortPinPcr (PortPin)

Used to specify the PCR (Port Configuration Register) for the configured pin.

Table 4-14. Attribute PortPinPcr (PortPin) detailed description

Property	Value
Label	PortPinPcr
Type	INTEGER
Origin	NXP
Symbolic Name	false
Invalid	Range >=0 <=155

4.4.1.2.6 PortPinMode (PortPin)

Selects the PORT pin mode from the modes list. By default more than one mode are allowed. That way it is e.g. possible to combine DIO with another mode such as ICU. For the Alternative Function modes (not a GPIO mode) the OUT direction is hw selected for that pin. NOTE: To set the IN direction take care, please, that all the possible module inputs, possible as Alternative Functions for the pad mode, are hw connected together, if IN direction is enabled, to the pad.

Table 4-15. Attribute PortPinMode (PortPin) detailed description

Property	Value
Label	PortPin Mode
Type	ENUMERATION
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	GPIO

4.4.1.2.7 PortPinDSE (PortPin)

Selects the drive strength value for the configured Pin.

Table 4-16. Attribute PortPinDSE (PortPin) detailed description

Property	Value
Label	PortPin DSE
Type	ENUMERATION
Origin	NXP

Table continues on the next page...

Table 4-16. Attribute PortPinDSE (PortPin) detailed description (continued)

Property	Value
Symbolic Name	false
Default	Low_Drive_Strength
Range	Low_Drive_Strength Hight_Drive_Strength

4.4.1.2.8 PortPinPE (PortPin)

Selects if the pull-up or pull-down resistors are enabled.

Table 4-17. Attribute PortPinPE (PortPin) detailed description

Property	Value
Label	PortPin PE
Type	ENUMERATION
Origin	NXP
Symbolic Name	false
Default	PullDisabled
Range	PullDisabled PullEnabled

4.4.1.2.9 PortPinPS (PortPin)

Selects between the pull-up and pull-down resistors. Only valid when PortPin PE is set to 'PullEnabled'.

Table 4-18. Attribute PortPinPS (PortPin) detailed description

Property	Value
Label	PortPin PS
Type	ENUMERATION
Origin	NXP
Symbolic Name	false
Default	PullDown
Range	PullDown PullUp

4.4.1.2.10 PortPinDirection (PortPin)

Selects the direction of the pin (IN, OUT, HIGH_Z) that will be configured by Port_Init() function if the pin is configured as GPIO. If the direction is not changeable, the value configured here is fixed. For the Alternative Function modes (PortPinMode is different than GPIO), the setting in this enumeration control is kept in the port configuration structure and it is used when Port_SetPinMode() is called at runtime to change the mode of the pin to GPIO. If HIGH_Z direction is enabled, it is not available in S32K11X (see chapter PORT Driver limitations).

Table 4-19. Attribute PortPinDirection (PortPin) detailed description

Property	Value
Label	PortPin Direction
Type	ENUMERATION
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	PORT_PIN_IN
Range	PORT_PIN_IN PORT_PIN_OUT PORT_PIN_HIGH_Z

4.4.1.2.11 PortPinInitialMode (PortPin)

Port pin mode from mode list for use with Port_Init() function. NOTE: This parameter is not used in the current implementation and is retained as per std AUTOSAR_EcucParamDef.arxml file.

Table 4-20. Attribute PortPinInitialMode (PortPin) detailed description

Property	Value
Label	PortPin Initial Mode
Type	ENUMERATION
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	PORT_GPIO_MODE
Enable	false
Range	PORT_GPIO_MODE PORT_ALT1_FUNC_MODE PORT_ALT2_FUNC_MODE PORT_ALT3_FUNC_MODE PORT_ALT4_FUNC_MODE PORT_ALT5_FUNC_MODE PORT_ALT6_FUNC_MODE PORT_ALT7_FUNC_MODE

4.4.1.2.12 PortPinLevelValue (PortPin)

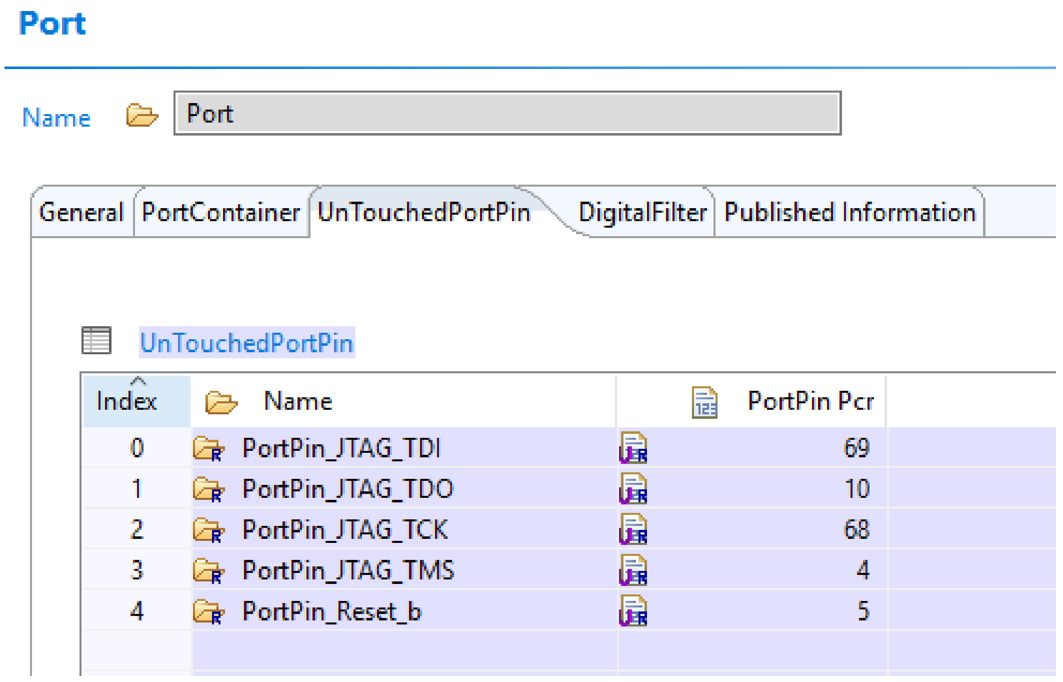
Port Pin Level value from Port pin list.

Table 4-21. Attribute PortPinLevelValue (PortPin) detailed description

Property	Value
Label	PortPin Level Value
Type	ENUMERATION
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	PORT_PIN_LEVEL_LOW
Range	PORT_PIN_LEVEL_HIGH PORT_PIN_LEVEL_LOW PORT_PIN_LEVEL_NOTCHANGED

4.4.2 UnTouchedPortPin

List with pins that will not be touched in any way by the Port driver.



Name: Port

Index	Name	PortPin Pcr
0	PortPin_JTAG_TDI	69
1	PortPin_JTAG_TDO	10
2	PortPin_JTAG_TCK	68
3	PortPin_JTAG_TMS	4
4	PortPin_Reset_b	5

Figure 4-6. Tresos Plugin snapshot for UnTouchedPortPin.

4.4.2.1 PortPin PCR

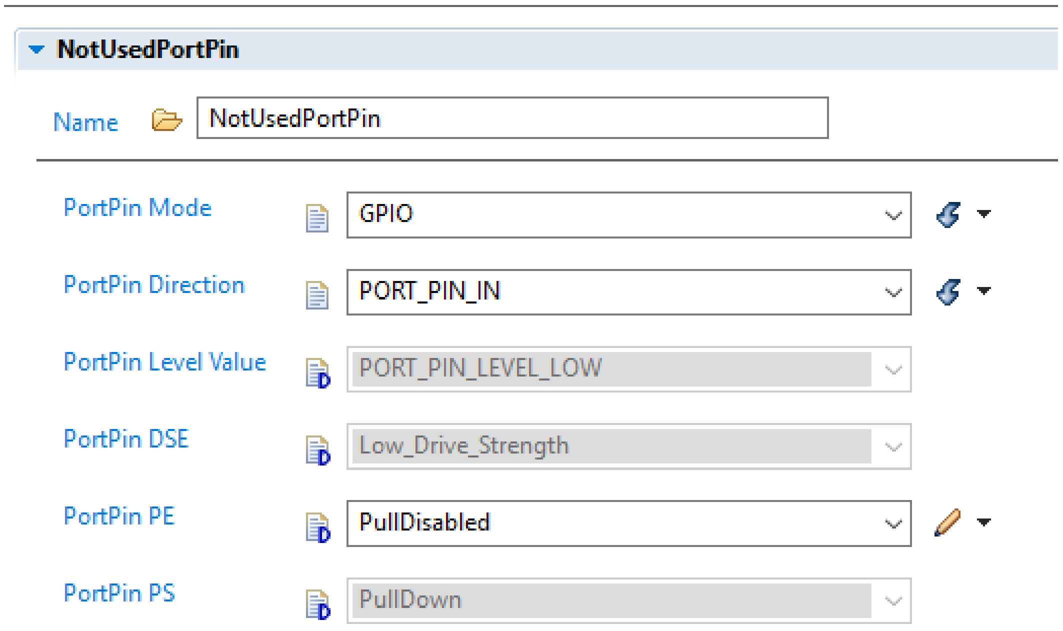
Selects the PCR of the pin that will not be touched by Port driver.

Table 4-22. Attribute PortPin PCR detailed description

Property	Value
Label	PortPin PCR
Type	INTEGER
Origin	Custom
Symbolic Name	false
Default	0
Range	0 Max num PCRs on the platform

4.4.3 NotUsedPortPin

Module wide configuration parameters of the PORT driver.


NotUsedPortPin

Name: NotUsedPortPin

PortPin Mode: GPIO

PortPin Direction: PORT_PIN_IN

PortPin Level Value: PORT_PIN_LEVEL_LOW

PortPin DSE: Low_Drive_Strength

PortPin PE: PullDisabled

PortPin PS: PullDown

Figure 4-7. Tresos Plugin snapshot for NotUsedPortPin.

4.4.3.1 PortPinMode(NotUsedPortPin)

Selects the PORT pin mode from the modes list. This is an implementation specific parameter

Table 4-23. Attribute PortPinMode(NotUsedPortPin) detailed description

Property	Value
Label	PortPin Mode
Type	ENUMERATION
Origin	NXP
Symbolic Name	false
Default	GPIO
Range	GPIO Disabled

4.4.3.2 PortPinDSE (PortPin)

Selects the drive strength value for the configured Pin. This is an implementation specific parameter.

Table 4-24. Attribute PortPinDSE (NotUsedPortPin) detailed description

Property	Value
Label	PortPin DSE
Type	ENUMERATION
Origin	NXP
Symbolic Name	false
Default	Low_Drive_Strength
Range	Low_Drive_Strength High_Drive_Strength

4.4.3.3 PortPinPE (PortPin)

Selects if the pull-up or pull-down resistors are enabled.

Table 4-25. Attribute PortPinPE (NotUsedPortPin) detailed description

Property	Value
Label	PortPin PE
Type	ENUMERATION

Table continues on the next page...

Table 4-25. Attribute PortPinPE (NotUsedPortPin) detailed description (continued)

Property	Value
Origin	NXP
Symbolic Name	false
Default	PullDisabled
Range	PullDisabled PullEnabled

4.4.3.4 PortPinPS (PortPin)

Selects between the pull-up and pull-down resistors. Only valid when PortPin PE is set to 'PullEnabled'.

Table 4-26. Attribute PortPinPS (NotUsedPortPin) detailed description

Property	Value
Label	PortPin PKE
Type	ENUMERATION
Origin	NXP
Symbolic Name	false
Default	PullDown
Range	PullDown PullUp

4.4.3.5 PortPinDirection (NotUsedPortPin)

Selects the initial direction of the pin (IN, OUT, HIGH_Z). If the direction is not changeable, the value configured here is fixed. The pin direction can be set only for the GPIO pins. For the Alternative Function modes the OUT pin direction is hw selected. If the IN direction is needed too, it can be set at runtime. NOTE: To set the IN direction take care, please, that all the possible module inputs, possible as Alternative Functions for the pad mode, are hw connected together, if IN direction is enabled, to the pad. If HIGH_Z direction is enabled, it is not available in S32K11X (see chapter PORT Driver limitations).

Table 4-27. Attribute PortPinDirection(NotUsedPortPin) detailed description

Property	Value
Label	PortPin Direction
Type	ENUMERATION

Table continues on the next page...

Table 4-27. Attribute PortPinDirection(NotUsedPortPin) detailed description (continued)

Property	Value
Origin	NXP
Symbolic Name	false
Default	PORT_PIN_IN
Range	PORT_PIN_IN PORT_PIN_OUT PORT_PIN_HIGH_Z

4.4.3.6 PortPinLevelValue(NotUsedPortPin)

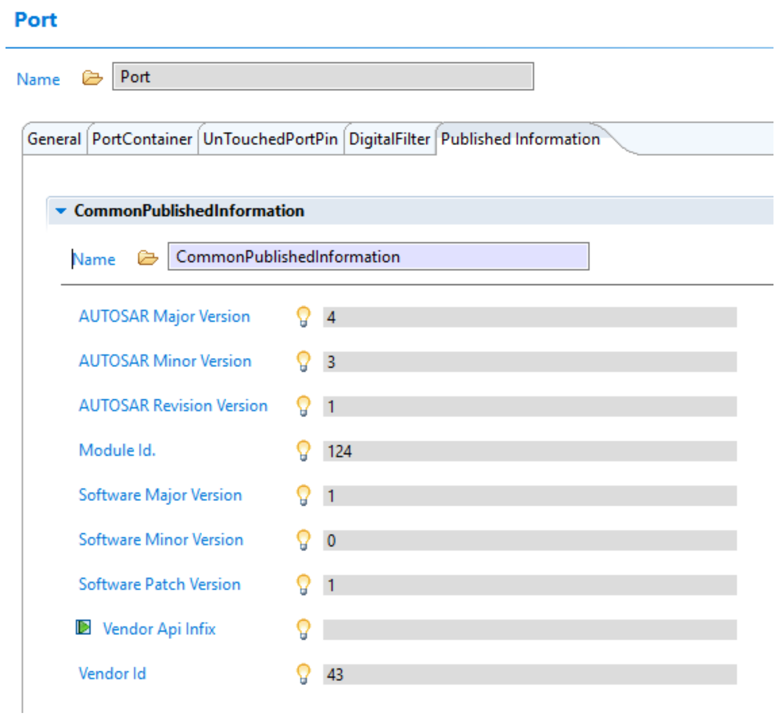
Port Pin Level value from Port pin list.

Table 4-28. Attribute PortPinLevelValue(NotUsedPortPin) detailed description

Property	Value
Label	PortPin Level Value
Type	ENUMERATION
Origin	NXP
Symbolic Name	false
Default	PORT_PIN_LEVEL_LOW
Range	PORT_PIN_LEVEL_HIGH PORT_PIN_LEVEL_LOW

4.5 Form CommonPublishedInformation

Common container, aggregated by all modules. It contains published information about vendor and versions.


Port

Name: Port

General | PortContainer | UnTouchedPortPin | DigitalFilter | Published Information

CommonPublishedInformation

Name: CommonPublishedInformation

AUTOSAR Major Version	4
AUTOSAR Minor Version	3
AUTOSAR Revision Version	1
Module Id.	124
Software Major Version	1
Software Minor Version	0
Software Patch Version	1
Vendor Api Infix	
Vendor Id	43

Figure 4-8. Tresos Plugin snapshot for CommonPublishedInformation form.

4.5.1 ArReleaseMajorVersion (CommonPublishedInformation)

Major version number of AUTOSAR specification on which the appropriate implementation is based on.

Table 4-29. Attribute ArReleaseMajorVersion (CommonPublishedInformation) detailed description

Property	Value
Label	AUTOSAR Major Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	4
Invalid	Range >=4 <=4

4.5.2 ArReleaseMinorVersion (CommonPublishedInformation)

Minor version number of AUTOSAR specification on which the appropriate implementation is based on.

Table 4-30. Attribute ArReleaseMinorVersion (CommonPublishedInformation) detailed description

Property	Value
Label	AUTOSAR Minor Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	3
Invalid	Range >=3 <=3

4.5.3 ArReleaseRevisionVersion (CommonPublishedInformation)

Revision version number of AUTOSAR specification on which the appropriate implementation is based on.

Table 4-31. Attribute ArReleaseRevisionVersion (CommonPublishedInformation) detailed description

Property	Value
Label	AUTOSAR Release Revision Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	1
Invalid	Range >=1 <=1

4.5.4 ModuleId (CommonPublishedInformation)

Module ID of this module from Module List.

Table 4-32. Attribute ModuleId (CommonPublishedInformation) detailed description

Property	Value
Label	Module Id
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	124
Invalid	Range >=124 <=124

4.5.5 SwMajorVersion (CommonPublishedInformation)

Major version number of the vendor specific implementation of the module. The numbering is vendor specific.

Table 4-33. Attribute SwMajorVersion (CommonPublishedInformation) detailed description

Property	Value
Label	Software Major Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	1
Invalid	Range >=1 <=1

4.5.6 SwMinorVersion (CommonPublishedInformation)

Minor version number of the vendor specific implementation of the module. The numbering is vendor specific.

Table 4-34. Attribute SwMinorVersion (CommonPublishedInformation) detailed description

Property	Value
Label	Software Minor Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false

Table continues on the next page...

Table 4-34. Attribute SwMinorVersion (CommonPublishedInformation) detailed description (continued)

Property	Value
Default	0
Invalid	Range >=0 <=0

4.5.7 SwPatchVersion (CommonPublishedInformation)

Patch level version number of the vendor specific implementation of the module. The numbering is vendor specific.

Table 4-35. Attribute SwPatchVersion (CommonPublishedInformation) detailed description

Property	Value
Label	Software Patch Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	1
Invalid	Range >=1 <=1

4.5.8 VendorApiInfix (CommonPublishedInformation)

In driver modules which can be instantiated several times on a single ECU, BSW00347 requires that the name of APIs is extended by the VendorId and a vendor specific name. This parameter is used to specify the vendor specific name. In total, the implementation specific name is generated as follows: <ModuleName>_<VendorId>_<VendorApiInfix><Api name from SWS>. E.g. assuming that the VendorId of the implementor is 123 and the implementer chose a VendorApiInfix of "v11r456" a api name Can_Write defined in the SWS will translate to Can_123_v11r456Write. This parameter is mandatory for all modules with upper multiplicity > 1. It shall not be used for modules with upper multiplicity =1.

Table 4-36. Attribute VendorApiInfix (CommonPublishedInformation) detailed description

Property	Value
Label	Vendor Api Infix
Type	STRING_LABEL
Origin	Custom
Symbolic Name	false
Default	
Enable	false

4.5.9 VendorId (CommonPublishedInformation)

Vendor ID of the dedicated implementation of this module according to the AUTOSAR vendor list.

Table 4-37. Attribute VendorId (CommonPublishedInformation) detailed description

Property	Value
Label	Vendor Id
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	43
Invalid	Range >=43 <=43

Document Number UM2PORTASR4.3 Rev0001R1.0.1
Revision 1.0