User Manual

for S32K14X EEP Driver

Document Number: UM2EEPASR4.2 Rev0002R1.0.1
Rev. 1.0

Contents

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

Chapter 1
Revision History

Chapter 2
Introduction

Chapter 3
Driver

Chapter 4
Tresos Configuration Plug-in

Chapter 1
Revision History

Table 1-1. Revision History

Revision	Date	Author	Description
1.0	13/07/2018	NXP MCAL Team	Updated version for ASR 4.2.2S32K14X1.0.1 Release

Chapter 2
Introduction

This User Manual describes NXP Semiconductors AUTOSAR Eeprom (EEP) driver for S32K14X.

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

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
DEM	Diagnostic Event Manager
DET	Default Error Tracer
ECC	Error Correcting Code
VLE	Variable Length Encoding
N/A	Not Applicable
MCU	Micro Controller Unit
ECU	Electronic Control Unit
EEPROM	Electrically Erasable Programmable Read-Only Memory
EA	EEPROM Abstraction

Table continues on the next page...

Table 2-2. Acronyms and Definitions (continued)

Term	Definition
EEP	EEPROM driver
XML	Extensible Markup Language

2.5 Reference List

Table 2-3. Reference List

#	Title	Version
1	Specification of EEP Driver	AUTOSAR Release 4.2.2
2	S32K14X Reference Manual	Reference Manual, Rev. 7, 4/2018
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)	30/11/2017
7	S32K118 Mask Set Errata for Mask 0N97V (0N97V)	26/02/2018

Chapter 3
Driver

3.1 Requirements

Requirements for this driver are detailed in the AUTOSAR 4.2 Rev0002EEP Driver Software Specification document (See Table Reference List).

3.2 Driver Design Summary

The EEP driver provides services for reading, writing and erasing eeprom memory.

3.3 Deviation from Requirements

The driver deviates from the AUTOSAR EEP Driver software specification in some places.

There are also some additional requirements (on top of requirements detailed in AUTOSAR EEP Driver software specification) which need to be satisfied for correct operation.

Table 3-1. Deviations Status Column Description

Term	Definition
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

SW Requirement ID	Status	Description	Notes
EEP082	N/I	The source code of external EEPROM drivers shall be independent of the microcontroller platform.	Applicable only for external Eep driver
EEP159	N/I	The module Eep shall provide a file Eep_Lcfg.c containing all link time configurable parameters.	Link time file Eep_Lcfg.c is not implemented because there is no support for VARIANT-LINK-TIME (ASR inconsistency between EEP159 and chapter 10.2.1 Variants)
EEP228	N/I	If the module implementation uses custom interrupt processing, the interrupt service routines shall be placed in Eep_Irq.c	Interrupt mode is not supported
EEP230	N/I	In case of a driver for an external SPI EEPROM, Eep.h shall include Spi.h	EEP does not have the support for an external EEP driver.
EEP231	N/F	If present, Eep_Irq.c shall include Eep.h and Eep_MemMap.h.	Applicable only for external Eep driver
EEP232	N/I	If present, Eep_Lcfg.c shall include Eep.h and Eep_MemMap.h. It shall optionally include Eep_Cbk.h	Link time file Eep_Lcfg.c is not implemented because there is no support for VARIANT-LINK-TIME (ASR inconsistency between EEP159 and chapter 10.2.1 Variants)
EEP088	N/F	The Eep SWS shall be valid both for internal and external EEPROMs.The Eep SWS defines asynchronous services for EEPROM operations (read/write/erase/compare).	EEP does not have the support for an external EEP driver.
EEP106	N/F	The detection of production code errors cannot be switched off.	SMCAL architecture defines the possibility of disabling DEM errors(PR-MCAL-326), therefore this requirement is rejected.
EEP128	N/I	The Eep module shall allow to be configured for interrupt or polling controlled job processing (if this is supported by the EEPROM hardware) through the configuration parameter EepUseInterrupts (see ECUC_Eep_00163).	Interrupt mode is not supported.
EEP129	N/F	If interrupt controlled job processing is supported and enabled, the external interrupt service routine located in Eep_Irq.c shall call an additional job processing function.	Interrupt mode is not supported.
EEP056	N/I	For an Eep module driving an external EEPROM through SPI: If the SPI access fails, the Eep module shall behave as specified in SWS_Eep_00068.	EEP does not have the support for an external EEP driver.
EEP052	N/I	For an Eep module driving an external EEPROM through SPI: In normal EEPROM mode, the Eep module shall access the external EEPROM by usage of SPI channels that are configured for normal access to the SPI EEPROM.	EEP does not have the support for an external EEP driver.
EEP053	N/I	For an Eep module driving an external EEPROM through SPI: The Eep's configuration shall be such that the value of the configuration parameter	EEP does not have the support for an external EEP driver.

Table continues on the next page...

Table 3-2. Driver Deviations Table (continued)

SW Requirement ID	Status	Description	Notes
		EepNormalReadBlockSize fits to the number of bytes that are readable in normal SPI mode..	
EEP055	N/I	For an Eep module driving an external EEPROM through SPI: In fast EEPROM mode, the Eep module shall access the external EEPROM by usage of SPI channels that are configured for burst access to the SPI EEPROM.	EEP does not have the support for an external EEP driver.
EEP073	N/I	For an Eep module driving an external EEPROM through SPI: The Eep's configuration shall be such that the value of the configuration parameter EepFastReadBlockSize fits to the number of bytes that are readable in burst SPI mode..	EEP does not have the support for an external EEP driver.
EEP132	N/I	For an Eep module driving an external EEPROM: in case the external EEPROM does not support the burst mode, the Eep module shall accept a selection of fast read mode, but shall behave the same as in normal mode (don't care of mode parameter).	EEP does not have the support for an external EEP driver.
EEP134	N/I	For the case of an Eep module driving an external EEPROM: if the external EEPROMs does not provide burst mode, the Eep module shall accept a selection of fast mode, but shall behave the same as in normal mode (don't care of mode parameter).	EEP does not have the support for an external EEP driver.
EEP060	N/I	If the value to be written to an EEPROM cell is already contained in the EEPROM cell, the Eep module should[1] skip the programming of that cell if it is configured to do so through the configuration parameter EepWriteCycleReduction..	The EPP driver does not need to implement it because the feature is implemented in the S32K hw. See AAI-290..
EEP059	N/I	The Eep module shall erase an EEPROM cell before writing to it if this is not done automatically by the EEPROM hardware.	Not needed for the S32K hw.
EEP063	N/I	The Eep module shall preserve data of affected EEPROM cells by performing read - modify - write operations, if the number of bytes to be written are smaller than the erasable and/or writeable data units.	Not needed for the S32K hw, smallest write page is 1 bytes.for the non-autosar quick write mode there will be a limitation in the driver that only 4 bytes alligned data is allowed.
EEP090	N/I	The Eep module shall preserve data of affected EEPROM cells by performing read - modify - write operations, if the given parameters (EepromAddress and Length) do not align with the erasable/writeable data units.	Not needed for the S32K hw, smallest write page is 1 bytes.for the non-autosar quick write mode there will be a limitation in the driver that only 4 bytes alligned data is allowed..
EEP069	N/I	The Eep module shall erase only as many bytes to the EEPROM as supported by the	Erase operation is implemented as writing the erased value(0xFF), since the S32K

Table continues on the next page...

Table 3-2. Driver Deviations Table (continued)

SW Requirement ID	Status	Description	Notes
		EEPROM hardware within one job processing cycle.	FLEXRAM memory does not need/allow erase.
EEP070	N/I	The Eep module shall use block erase commands if supported by the EEPROM hardware and if the given parameters (EepromAddress and Length) are aligned to erasable blocks.	Not needed because the writing unit is 1 bytes.
EEP072	N/I	The Eep module shall preserve the contents of affected EEPROM cells by using read - modify - write operations, if the given erase parameters (EepromAddress and Length) do not align with the erasable data units.	Not needed because the writing unit is 1 bytes
EEP208	N/I	Each variable that shall be accessible by AUTOSAR Debugging shall be defined as global variable.	As per PR-MCAL-3330, debugging req are rejected
EEP209	N/I	All type definitions of variables which shall be debugged shall be accessible by the header file Eep.h.	As per PR-MCAL-3330, debugging req are rejected
EEP210	N/I	The declaration of variables in the header file shall be such that it is possible to calculate the size of the variables by C-“sizeof”.	As per PR-MCAL-3330, debugging req are rejected
EEP211	N/I	Variables available for debugging shall be described in the respective Basic Software Module Description.	As per PR-MCAL-3330, debugging req are rejected
EEP212	N/I	The EEPROM module state shall be available for debugging.	As per PR-MCAL-3330, debugging req are rejected.
EEP213	N/I	The job result shall be available for debugging.	As per PR-MCAL-3330, debugging req are rejected.
EEP214	N/I	The EEPROM operation mode shall be available for debugging.	As per PR-MCAL-3330, debugging req are rejected.
EEP161	N/I	For variants with no postbuild multiple selectable configuration parameters (Variant PC), the EEP module’s environment shall pass a NULL pointer to the function Eep_Init().	requirement for the EEP user, not EEP driver.
EEP115	N/I	The Eep’s user shall not call the function Eep_Init during a running operation.	requirement for the EEP user, not EEP driver.
EEP116	N/I	The Eep’s user shall not call the function Eep_SetMode during a running operation.	requirement for the EEP user, not EEP driver.
EEP117	N/I	The Eep’s user shall only call Eep_Read after the Eep module has been been initialized.	requirement for the EEP user, not EEP driver.
EEP118	N/I	The Eep’s user shall only call Eep_Read after the Eep module has been been initialized.	requirement for the EEP user, not EEP driver.
EEP119	N/I	The Eep module’s user shall only call the function Eep_Write after the Eep module has been initialized.	requirement for the EEP user, not EEP driver.

Table continues on the next page...

Table 3-2. Driver Deviations Table (continued)

SW Requirement ID	Status	Description	Notes
EEP120	N/I	The Eep module's user shall not call the function Eep_Write during a running Eep module job (read/write/erase/compare).	requirement for the EEP user, not EEP driver.
EEP121	N/I	The Eep module's user shall only call the function Eep_Erase after the Eep module has been initialized.	requirement for the EEP user, not EEP driver.
EEP122	N/I	The Eep module's user shall not call the function Eep_Erase during a running Eep job (read/write/erase/compare).	requirement for the EEP user, not EEP driver.
EEP124	N/I	The Eep module's user shall not call the function Eep_Compare during a running Eep job (read/write/erase/compare).	requirement for the EEP user, not EEP driver.
EEP123	N/I	The Eep module's user shall only call the function Eep_Compare after the Eep module has been initialized.	requirement for the EEP user, not EEP driver.
EEP165_Conf	N/I	The Eep module's user shall only call the function Eep_Compare after the Eep module has been initialized.	requirement for the EEP user, not EEP driver.
EEP190_Conf	N/F	EepExternalDriver Description This container is present for external EEPROM drivers only. Internal EEPROM drivers do not use the parameter listed in this container, hence its multiplicity is 0 for internal drivers.	not implemented, see AAI-290
EEP176_Conf	N/F	Name EepSpiReference Description...	not implemented, see AAI-290
EEP031	N/I	When a job has been initiated, the Eep's user shall call the function Eep_MainFunction cyclically until the job is finished.	Requirement for EEP user.
EEP084	N/I	The configuration parameter EepJobCallCycle (see ECUC_Eep_00170) shall be used for internal timing of the EEPROM driver (deadline monitoring, write and erase timing etc.) if needed by the implementation and/or the underlying hardware.	Not needed, the smcal drivers use software timers for timeout supervision. See AAI-290.
EEP187_Conf	N/I	EepSpecifiedEraseCycles {EEP_SPECIFIED_ERASE_CYCLES}	not implemented, FLEXRAM does not allow erase.
EEP170_Conf	N/I	EepJobCallCycle	Not implemented, see AAI-290.
EEP178_Conf	N/I	EepEraseTime	Not implemented, see AAI-290.

3.4 Det Error Description

Table 3-3. Det Error Description

Error Code	Value	Condition triggering the error
EEP_E_INIT_FAILED(For asr422)	0x10	API service called with init failed
EEP_E_PARAM_CONFIG(For asr403)	0x10	API service called with wrong parameter
EEP_E_BUSY	0x21	API service called while driver still busy
EEP_E_UNINIT	0x20	API service called without module initialization
EEP_E_PARAM_ADDRESS	0x11	TargetAddress is not in range and aligned to first byte of eeprom sector
EEP_E_PARAM_LENGTH	0x13	TargetAddress is not in range and aligned to last byte of eeprom sector
EEP_E_PARAM_DATA	0x12	NULL_PTR == SourceAddressPtr
EEP_E_TIMEOUT	0x22	The hardware operation did not finish before timeout expired.
EEP_E_PARAM_POINTER	0x23	NULL_PTR passed
EEP_E_CANCEL_REJECTED	0x24	Eep_Cancel was called during processing of a write job requested by Eep_QuickWrite, which is not allowed.

3.5 Software specification

The following sections contains driver software specifications.

3.5.1 Define Reference

Constants supported by the driver are as per AUTOSAR EEP Driver software specification Version 4.2 Rev0002 .

3.5.2 Enum Reference

Enumeration of all constants supported by the driver are as per AUTOSAR EEP Driver software specification Version 4.2 Rev0002 .

3.5.3 Function Reference

Functions of all functions supported by the driver are as per AUTOSAR EEP Driver software specification Version 4.2 Rev0002 .

3.5.3.1 Function Eep_Cancel

Cancel an ongoing eeprom read, write, erase or compare job.

Details:

Abort a running job synchronously so that directly after returning from this function a new job can be started.

Pre: The module must be initialized.

Post: Eep_Cancelchanges module status andEep_eJobResultinternal variable.

Implements: Eep_Cancel_Activity

Violates: All declarations and definitions of objects or functions at file scope shall have internal linkage unless external linkage required

Prototype: `void Eep_Cancel(void);`

Note

Aborting an operation ongoing at hardware level is not supported.

3.5.3.2 Function Eep_Compare

Compares a eeprom memory area with an application data buffer.

Details:

Starts a compare job asynchronously. The actual job is performed by Eep_MainFunction .

Return: Std_ReturnType.

Pre: The module has to be initialized and not busy.

Violates: All declarations and definitions of objects or functions at file scope shall have internal linkage unless external linkage required

Prototype: `Std_ReturnType Eep_Compare(Eep_AddressType u32SourceAddress, const uint8 *pTargetAddressPtr, Eep_LengthType u32Length);`

Table 3-4. Eep_Compare Arguments

Type	Name	Direction	Description
`Eep_AddressType`	u32SourceAddress	**input**	Source address in eeprom memory.
const uint8 *	pTargetAddressPtr	**input**	Pointer to source data buffer.
`Eep_LengthType`	u32Length	**input**	Number of bytes to compare.

Table 3-5. Eep_Compare Return Values

Name	Description
E_OK	Compare command has been accepted.
E_NOT_OK	Compare command has not been accepted.

3.5.3.3 Function Eep_Erase

Erase a memory area of size length. The EEP driver writes and reads from the FLEXRAM memory which does not need erase before re-writing. In order to comply with AUTOSAR interface requirements, the NXP EEP provides the Eep_Erase API which will perform a software erase(write ERASED_VALUE 0xFF), in case it is needed by the EEP user.

Details:

Starts an erase job asynchronously. The actual job is performed by theEep_MainFunction.

Return: Std_ReturnType.

Pre: The module has to be initialized and not busy.

Prototype: `Std_ReturnType Eep_Erase(Eep_AddressType u32TargetAddress, Eep_LengthType u32Length);`

Table 3-6. Eep_Erase Arguments

Type	Name	Direction	Description
`Eep_AddressType`	u32TargetAddress	**input**	Target address in eeprom memory.
`Eep_LengthType`	u32Length	**input**	Number of bytes to erase.

Table 3-7. Eep_Erase Return Values

Name	Description
E_OK	Erase command has been accepted.
E_NOT_OK	Erase command has not been accepted.

3.5.3.4 Function Eep_GetJobResult

Returns the result of the last job.

Details:

Returns synchronously the result of the last job.

Return: MemIf_JobResultType.

Implements: Eep_GetJobResult_Activity

Violates: All declarations and definitions of objects or functions at file scope shall have internal linkage unless external linkage required

Prototype: `MemIf_JobResultType Eep_GetJobResult(void);`

Table 3-8. Eep_GetJobResult Return Values

Name	Description
MEMIF_JOB_OK	Successfully completed job.
MEMIF_JOB_FAILED	Not successfully completed job.
MEMIF_JOB_PENDING	Still pending job (not yet completed).
MEMIF_JOB_CANCELED	Job has been canceled.
MEMIF_BLOCK_INCONSISTENT	Inconsistent block requested, it may contains corrupted data.
MEMIF_BLOCK_INVALID	Invalid block requested.

3.5.3.5 Function Eep_GetStatus

Returns the EEP module status.

Details:

Returns the EEP module status synchronously.

Return: MemIf_StatusType.

Implements: Eep_GetStatus_Activity

Violates: All declarations and definitions of objects or functions at file scope shall have internal linkage unless external linkage required

Prototype: `MemIf_StatusType Eep_GetStatus(void);`

Table 3-9. Eep_GetStatus Return Values

Name	Description
MEMIF_UNINIT	Module has not been initialized (yet).
MEMIF_IDLE	Module is currently idle.
MEMIF_BUSY	Module is currently busy.

3.5.3.6 Function Eep_GetVersionInfo

Returns version information about EEP module.

Details:

Version information includes:

- Module Id
- Vendor Id
- Vendor specific version numbers (BSW00407).

Implements: Eep_GetVersionInfo_Activity

Prototype: `void Eep_GetVersionInfo(Std_VersionInfoType *pVersionInfoPtr);`

Table 3-10. Eep_GetVersionInfo Arguments

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

3.5.3.7 Function Eep_Init

The function initializes Eep module. The FLEXRAM and FLEXNVM must be partitioned for eeprom emulation usage prior to using API Eep_Init. The partitioning is done by running the "Program partition command".

Details:

Implements: Eep_Init_Activity

Prototype: `void Eep_Init(const Eep_ConfigType *pConfigPtr);`

Table 3-11. Eep_Init Arguments

Type	Name	Direction	Description
const`Eep_ConfigType*`	pConfigPtr	**input**	Pointer to eeprom driver configuration set.

3.5.3.8 Function Eep_MainFunction

Performs actual eeprom read, write, erase and compare jobs.

Details:

Bytes number processed per cycle depends by job type (erase, write, read, compare) current EEP module's operating mode (normal, fast) and write, erase Mode of Execution (sync, async).

Pre: The module has to be initialized.

Implements: Eep_MainFunction_Activity

Note

This function have to be called ciclically by the Basic Software Module; it will do nothing if there aren't pending job.

Prototype: `void Eep_MainFunction(void);`

3.5.3.9 Function Eep_Read

Reads from eeprom memory.

Details:

Starts a read job asynchronously. The actual job is performed byEep_MainFunction.

Return: MemIf_JobResultType.

Pre: The module has to be initialized and not busy.

Prototype: `Std_ReturnType Eep_Read(Eep_AddressType u32SourceAddress, uint8 *pTargetAddressPtr, Eep_LengthType u32Length);`

Table 3-12. Eep_Read Arguments

Type	Name	Direction	Description
	SourceAddress	**input**	Source address in eeprom memory.
	Length	**input**	Number of bytes to read.
	TargetAddressPtr	**output**	Pointer to target data buffer.

Table 3-13. Eep_Read Return Values

Name	Description
MEMIF_JOB_OK	Successfully completed job.
MEMIF_JOB_FAILED	Not successfully completed job.
MEMIF_JOB_PENDING	Still pending job (not yet completed).
MEMIF_JOB_CANCELED	Job has been canceled.
MEMIF_BLOCK_INCONSISTENT	Inconsistent block requested, it may contains corrupted data.
MEMIF_BLOCK_INVALID	Invalid block requested.

3.5.3.10 Function Eep_SetMode

Sets the EEP module's operation mode to the given Mode.

Details:

Every given mode determinates maximum bytes for read-write operations. Every mode has a set of pre-configured values.

Pre: The module has to be initialized and not busy.

Post Eep_SetMode nchanges internal variables Eep_u32MaxRead and Eep_u32MaxWrite.

Implements: Eep_SetMode_Activity

Violates: All declarations and definitions of objects or functions at file scope shall have internal linkage unless external linkage required

Prototype: `void Eep_SetMode(MemIf_ModeType eMode);`

Table 3-14. Eep_SetMode Arguments

Type	Name	Direction	Description
MemIf_ModeType	Mode	**input**	MEMIF_MODE_FAST or MEMIF_MODE_SLOW.

3.5.3.11 Function Eep_Write

Write one or more complete eeprom pages to the eeprom device.

Details:

Starts a write job asynchronously. The actual job is performed by Eep_MainFunction.

Return: Std_ReturnType.

Pre: The module has to be initialized and not busy.

Prototype: `Std_ReturnType Eep_Write(Eep_AddressType u32TargetAddress, const uint8 *pSourceAddressPtr, Eep_LengthType u32Length);`

Table 3-15. Eep_Write Arguments

Type	Name	Direction	Description
	TargetAddress	**input**	Target address in eeprom memory.
	SourceAddressPtr	**input**	Pointer to source data buffer.
	Length	**input**	Number of bytes to write.

Table 3-16. Eep_Write Return Values

Name	Description
E_OK	Write command has been accepted.
E_NOT_OK	Write command has not been accepted.

3.5.3.12 Function Eep_QuickWrite

Write to the eeprom device using the hardware quick write mode.

Details:

Starts a write job asynchronously. The actual job is performed by Eep_MainFunction.

Return: Std_ReturnType.

Pre: The module has to be initialized and not busy.

Prototype: `Std_ReturnType Eep_QuickWrite(Eep_AddressType u32TargetAddress, const uint8 *pSourceAddressPtr, Eep_LengthType u32Length, uint16 u16QuickWritesLength);`

Table 3-17. Eep_QuickWrite Arguments

Type	Name	Direction	Description
	TargetAddress	**input**	Target address in eeprom memory.
	SourceAddressPtr	**input**	Pointer to source data buffer.
	Length	**input**	Number of bytes to write.
	u16QuickWritesLength	**input**	Number of quick write bytes to setup in the hardware.

Table 3-18. Eep_QuickWrite Return Values

Name	Description
E_OK	Write command has been accepted.
E_NOT_OK	Write command has not been accepted.

3.5.4 Structs Reference

Data structures supported by the driver are as per AUTOSAR EEP Driver software specification Version 4.2 Rev0002 .

3.5.5 Types Reference

Types supported by the driver are as per AUTOSAR EEP Driver software specification Version 4.2 Rev0002 .

3.5.6 Variables Reference

Variables supported by the driver are as per AUTOSAR EEP Driver software specification Version 4.2 Rev0002 .

3.6 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 EEP Driver. The most of the parameters are described below.

4.1 Configuration elements of Eep

Included forms :

- IMPLEMENTATION_CONFIG_VARIANT
- NonAutosar
- EepGeneral
- EepPublishedInformation
- CommonPublishedInformation
- EepInitConfiguration

Table 4-1. Revision table

Revision	Date
4.1.0	2010-12-03

4.2 Form IMPLEMENTATION_CONFIG_VARIANT

VariantPostBuild: Mix of precompile and postbuild time configuration parameters.

If Config Variant = VariantPostBuild, the files Eep_Cfg.h and Eep_PBcfg.c should be used.

If Config Variant = VariantPreCompile, the files Eep_Cfg.h and Eep_Cfg.c should be used.

Figure 4-1. Tresos Plugin snapshot for IMPLEMENTATION_CONFIG_VARIANT form.

Table 4-2. Attribute IMPLEMENTATION_CONFIG_VARIANT detailed description

Property	Value
Label	Config Variant
Type	ENUMERATION
Default	VariantPostBuild
Range	VariantPostBuild VariantPreCompile

4.3 Form NonAutosar

Vendor specific: This container contains the global Non-Autosar configuration parameters of the Eep driver. This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set.

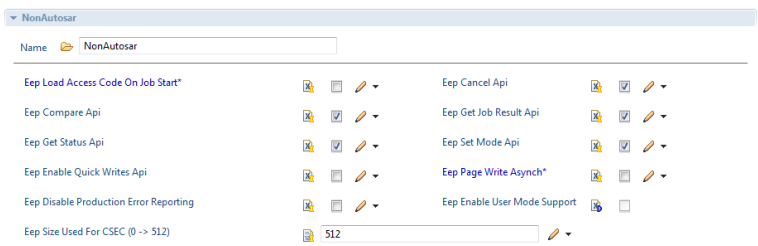

Figure 4-2. Tresos Plugin snapshot for NonAutosar form.

4.3.1 EepAcLoadOnJobStart (NonAutosar)

The EEP driver shall load the FLEXRAM access code to RAM whenever an erase, write job is started and unload (overwrite) it after that job has been finished or canceled.The EEP driver shall also load the FLEXRAM access code during the init phase, before using Set as Flexram command, and unload it after the command completes.

true: Eeprom access code loaded on job start / unloaded on job end or error.

false: Eeprom access code not loaded to / unloaded from RAM.

Table 4-3. Attribute EepAcLoadOnJobStart (NonAutosar) detailed description

Property	Value
Label	Eep Load Access Code On Job Start
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

4.3.2 EepCancelApi (NonAutosar)

Compile switch to enable and disable the Eep_Cancel function.

true: API supported / function provided.

false: API not supported / function not provided

Table 4-4. Attribute EepCancelApi (NonAutosar) detailed description

Property	Value
Label	Eep Cancel Api
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	true

4.3.3 EepCompareApi (NonAutosar)

Compile switch to enable and disable the Eep_Compare function.

true: API supported / function provided.

false: API not supported / function not provided

Table 4-5. Attribute EepCompareApi (NonAutosar) detailed description

Property	Value
Label	Eep Compare Api
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	true

4.3.4 EepGetJobResultApi (NonAutosar)

Compile switch to enable and disable the Eep_GetJobResult function.

true: API supported / function provided.

false: API not supported / function not provided

Table 4-6. Attribute EepGetJobResultApi (NonAutosar) detailed description

Property	Value
Label	Eep Get Job Result Api
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	true

4.3.5 EepGetStatusApi (NonAutosar)

Compile switch to enable and disable the Eep_GetStatus function.

true: API supported / function provided.

false: API not supported / function not provided

Table 4-7. Attribute EepGetStatusApi (NonAutosar) detailed description

Property	Value
Label	Eep Get Status Api
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	true

4.3.6 EepSetModeApi (NonAutosar)

Compile switch to enable and disable the Eep_SetMode function.

true: API supported / function provided.

false: API not supported / function not provided

Table 4-8. Attribute EepSetModeApi (NonAutosar) detailed description

Property	Value
Label	Eep Set Mode Api
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	true

4.3.7 EepPageAsynchBehaviorEn (NonAutosar)

Vendor specific: Enable asynchronous execution of the write or erase job in the Eep_MainFunction function which doesn't wait (block) for completion of the page write or erase operation(s). The eeprom driver doesn't use the access code in asynchronous mode. Also, if EEP is used in the same application with CSEC or FLS, the asynchronous mode must not be used.

Table 4-9. Attribute EepPageAsynchBehaviorEn (NonAutosar) detailed description

Property	Value
Label	Eep Page Write Asynch
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

4.3.8 EepEnableQuickWriteApi (NonAutosar)

Vendor specific: Pre-processor switch to enable / disable Eep_QuickWrite API usage.

true: Eep_QuickWrite will be available

false: Eep_QuickWrite will not be available

Table 4-10. Attribute EepEnableQuickWriteApi (NonAutosar) detailed description

Property	Value
Label	Enable Eep_QuickWrite API usage
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	true

4.3.9 EepSizeUsedForCSEC (NonAutosar)

This parameter is needed only if CSEC driver is used.It represents the size of FLEXRAM used by CSEC in bytes. For example if the maximum number of CSEC keys is used, then the integrator should configure EepSizeUsedForCSEC to 512 bytes.

Table 4-11. Attribute EepSizeUsedForCSEC (NonAutosar) detailed description

Property	Value
Label	Eep Size Used For CSEC
Type	INTEGER
Origin	Custom
Symbolic Name	false
Default	0

4.3.10 EepDisableDemReportErrorStatus (NonAutosar)

Vendor specific: Switches the Diagnostic Error Reporting and Notification OFF.

Table 4-12. Attribute EepDisableDemReportErrorStatus (NonAutosar) detailed description

Property	Value
Label	Eep Disable Production Error Reporting
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

4.3.11 EepEnableUserModeSupport (NonAutosar)

When this parameter is enabled, the EEP module will adapt to run from User Mode, with the following measures:

configuring REG_PROT for Eep IPs so that the registers under protection can be accessed from user mode by setting UAA bit in REG_PROT_GCR to 1

for more information and availability on this platform, please see chapter User Mode Support in IM

Table 4-13. Attribute EepEnableUserModeSupport (NonAutosar) detailed description

Property	Value
Label	Eep Enable User Mode Support
Type	BOOLEAN
Origin	Custom
Symbolic Name	false
Default	false

4.4 Form EepGeneral

Container for general parameters of the eeprom driver. These parameters are always pre-compile.

Included forms :

- Form EepTimeouts

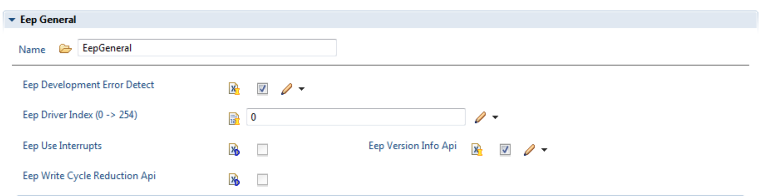

Figure 4-3. Tresos Plugin snapshot for EepGeneral form.

4.4.1 EepDevErrorDetect (EepGeneral)

Pre-processor switch to enable and disable development error detection (see EEP077).

true: Development error detection enabled.

false: Development error detection disabled.

Table 4-14. Attribute EepDevErrorDetect (EepGeneral) detailed description

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

4.4.2 EepDriverIndex (EepGeneral)

Index of the driver, used by FEE.

Table 4-15. Attribute EepDriverIndex (EepGeneral) detailed description

Property	Value
Label	Eep Driver Index
Type	INTEGER
Origin	AUTOSAR_ECUC
Symbolic Name	true
Default	0
Invalid	Range <=254 >=0

4.4.3 EepUseInterrupts (EepGeneral)

Job processing triggered by hardware interrupt.

true: Job processing triggered by interrupt (hardware controlled)

false: Job processing not triggered by interrupt (software controlled)

Note:

Not used by the EEP driver.

Table 4-16. Attribute EepUseInterrupts (EepGeneral) detailed description

Property	Value
Label	Eep Use Interrupts
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	false

4.4.4 EepVersionInfoApi (EepGeneral)

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

true: Version info API enabled.

false: Version info API disabled.

Table 4-17. Attribute EepVersionInfoApi (EepGeneral) detailed description

Property	Value
Label	Eep Version Info Api
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	true

4.4.5 EepWriteCycleReduction (EepGeneral)

Switches to activate or deactivate write cycle reduction (EEPROM value is read and compared before being overwritten). true: writecycle reduction enabled. false: Write cycle reduction disabled. The parameter is currently not used as the support for the write cycle reduction is implemented in hardware.

Table 4-18. Attribute EepWriteCycleReduction (EepGeneral) detailed description

Property	Value
Label	Eep Write Cycle Reduction Api
Type	BOOLEAN
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	false

4.5 Form EepTimeouts

Container for hardware timeout handling.

Is included by form : Form EepGeneral

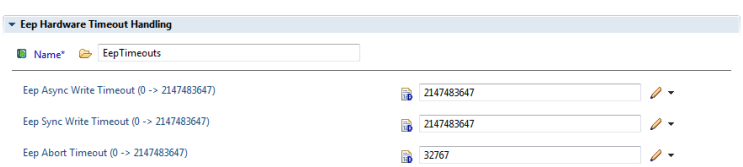

Figure 4-4. Tresos Plugin snapshot for EepTimeouts form.

4.5.1 EepAsyncWriteTimeout (EepTimeouts)

Vendor specific: Eep Async Write Timeout is the timeout value for write operation in asynchronous mode.

Table 4-19. Attribute EepAsyncWriteTimeout (EepTimeouts) detailed description

Property	Value
Label	Eep Async Write Timeout
Type	INTEGER
Origin	Custom
Symbolic Name	false
Default	2147483647
Invalid	Range <=2147483647 >=0

4.5.2 EepSyncWriteTimeout (EepTimeouts)

Vendor specific: Eep Sync Write Timeout is the timeout value for write operation in synchronous mode.

Table 4-20. Attribute EepSyncWriteTimeout (EepTimeouts) detailed description

Property	Value
Label	Eep Sync Write Timeout
Type	INTEGER
Origin	Custom
Symbolic Name	false
Default	2147483647
Invalid	Range <=2147483647 >=0

4.5.3 EepAbortTimeout (EepTimeouts)

Vendor specific: Eep Abort Timeout is the timeout value for aborting an ongoing operation.

Table 4-21. Attribute EepAbortTimeout (EepTimeouts) detailed description

Property	Value
Label	Eep Async Abort Timeout
Type	INTEGER
Origin	Custom
Symbolic Name	false
Default	32767
Invalid	Range <=2147483647 >=0

4.6 Form EepPublishedInformation

Additional published parameters not covered by CommonPublishedInformation container. Note that these parameters do not have any configuration class setting, since they are published information.

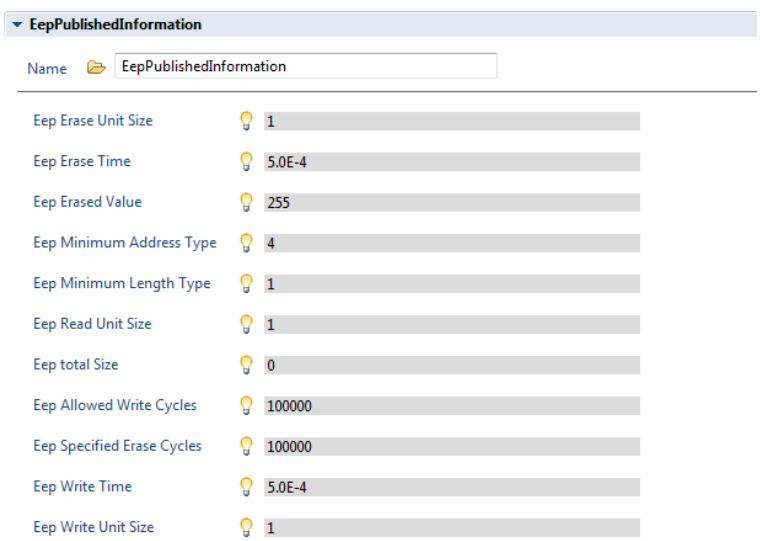

Figure 4-5. Tresos Plugin snapshot for EepPublishedInformation form.

4.6.1 EepEraseUnitSize (EepPublishedInformation)

Size of smallest erasable EEPROM data unit in bytes.FLEXRAM does not allow erase.

Table 4-22. Attribute EepEraseUnitSize (EepPublishedInformation) detailed description

Property	Value
Label	Eep Erase Unit Size
Type	INTEGER_LABEL
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	1
Invalid	Range <=4294967295 >=0

4.6.2 EepEraseTime (EepPublishedInformation)

Maximum time to erase one complete eeprom data unit.

Note:

This value can be found on DS as the maximum erase time occurs after the specified number of program/erase cycles. FLEXRAM does not allow erase, only DFLASH, PFLASH allow erase.

Table 4-23. Attribute EepEraseTime (EepPublishedInformation) detailed description

Property	Value
Label	Eep Erase Time
Type	FLOAT_LABEL
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	0.0005
Invalid	Range <=0.0005 >=0

4.6.3 EepEraseValue (EepPublishedInformation)

The contents of an erased eeprom memory cell.

Table 4-24. Attribute EepEraseValue (EepPublishedInformation) detailed description

Property	Value
Label	Eep Erased Value

Table continues on the next page...

Table 4-24. Attribute EepEraseValue (EepPublishedInformation) detailed description (continued)

Property	Value
Type	INTEGER_LABEL
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	4294967295
Invalid	Range <=4294967295 >=0

4.6.4 EepMinimumAddressType (EepPublishedInformation)

Minimum expected size of Eep_AddressType.

Table 4-25. Attribute EepMinimumAddressType (EepPublishedInformation) detailed description

Property	Value
Label	Eep Minimum Address Type
Type	INTEGER_LABEL
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	4
Invalid	Range <=4294967295 >=0

4.6.5 EepMinimumLengthType (EepPublishedInformation)

Minimum expected size of Eep_LengthType.

Table 4-26. Attribute EepMinimumLengthType (EepPublishedInformation) detailed description

Property	Value
Label	Eep Minimum Length Type
Type	INTEGER_LABEL
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	1

Table continues on the next page...

Table 4-26. Attribute EepMinimumLengthType (EepPublishedInformation) detailed description (continued)

Property	Value
Invalid	Range <=4294967295 >=0

4.6.6 EepReadUnitSize (EepPublishedInformation)

Size of smallest readable EEPROM data unit in bytes.

Table 4-27. Attribute EepReadUnitSize (EepPublishedInformation) detailed description

Property	Value
Label	Eep Read Unit Size
Type	INTEGER_LABEL
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	1
Invalid	Range <=4294967295 >=0

4.6.7 EepTotalSize (EepPublishedInformation)

This parameter is the used size of EEPROM device in bytes.

Table 4-28. Attribute EepTotalSize (EepPublishedInformation) detailed description

Property	Value
Label	Eep Size
Type	INTEGER_LABEL
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	4096
Invalid	Range <=4294967295 >=0

4.6.8 EepAllowedWriteCycles (EepPublishedInformation)

Specified maximum number of write cycles under worst case conditions of specific EEPROM hardware.

Table 4-29. Attribute EepAllowedWriteCycles (EepPublishedInformation) detailed description

Property	Value
Label	Eep Allowed Write Cycles
Type	INTEGER_LABEL
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	100000
Invalid	Range <=4294967295 >=0

4.6.9 EepSpecifiedEraseCycles (EepPublishedInformation)

Number of erase cycles specified for the EEP device (usually given in the device data sheet). FLEXRAM does not allow erase.

Table 4-30. Attribute EepSpecifiedEraseCycles (EepPublishedInformation) detailed description

Property	Value
Label	Eep Specified Erase Cycles
Type	INTEGER_LABEL
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	100000
Invalid	Range <=4294967295 >=0

4.6.10 EepWriteTime (EepPublishedInformation)

Maximum time to program one complete eeprom page.

Table 4-31. Attribute EepWriteTime (EepPublishedInformation) detailed description

Property	Value
Label	Eep Write Time
Type	FLOAT_LABEL
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	0.0005
Invalid	Range <=0.0005 >=0

4.6.11 EepWriteUnitSize (EepPublishedInformation)

Size of smallest writeble EEPROM data unit in bytes.

Table 4-32. Attribute EepWriteUnitSize (EepPublishedInformation) detailed description

Property	Value
Label	Eep Write Unit Size
Type	INTEGER_LABEL
Origin	AUTOSAR_ECUC
Symbolic Name	false
Default	1
Invalid	Range <=4294967295 >=0

4.7 Form CommonPublishedInformation

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

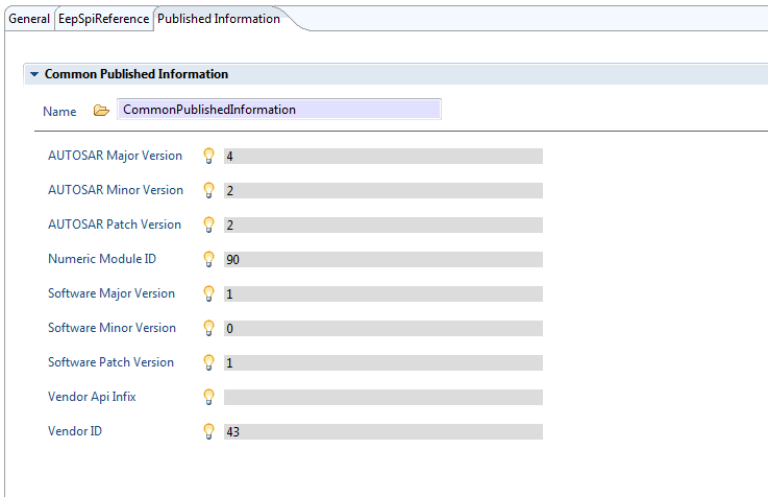

Figure 4-6. Tresos Plugin snapshot for CommonPublishedInformation form.

4.7.1 ArReleaseMajorVersion (CommonPublishedInformation)

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

Table 4-33. 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

4.7.2 ArReleaseMinorVersion (CommonPublishedInformation)

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

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

Property	Value
Label	AUTOSAR Minor Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	2
Invalid	Range >=2 <=2

4.7.3 ArReleaseRevisionVersion (CommonPublishedInformation)

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

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

Property	Value
Label	AUTOSAR Release Revision Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	2
Invalid	Range >=2 <=2

4.7.4 ModuleId (CommonPublishedInformation)

Module ID of this module from Module List.

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

Property	Value
Label	Module Id
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false

Table continues on the next page...

Table 4-36. Attribute ModuleId (CommonPublishedInformation) detailed description (continued)

Property	Value
Default	90
Invalid	Range >=90 <=90

4.7.5 SwMajorVersion (CommonPublishedInformation)

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

Table 4-37. 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

4.7.6 SwMinorVersion (CommonPublishedInformation)

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

Table 4-38. Attribute SwMinorVersion (CommonPublishedInformation) detailed description

Property	Value
Label	Software Minor Version
Type	INTEGER_LABEL
Origin	Custom
Symbolic Name	false
Default	0
Invalid	Range >=0 <=0

4.7.7 SwPatchVersion (CommonPublishedInformation)

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

Table 4-39. 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

4.7.8 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 4-40. Attribute VendorApiInfix (CommonPublishedInformation) detailed description

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

4.7.9 VendorId (CommonPublishedInformation)

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

Table 4-41. Attribute VendorId (CommonPublishedInformation) detailed description

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

Chapter 5
Platform

Chapter 6
Async/sync mode

6.1 Avoiding RWW problem

To avoid RWW (Read While Write) problems the eeprom driver provide the EepAcLoadOnJobStart configuration parameter. If it is set to true the Eep driver will load the eeprom access code routine to RAM whenever an erase or write job is started and unload (overwrite) it after that job has been finished or cancelled.

EepAcLoadOnJobStart functionality can be used only in case of Synchronuous Mode, in which case the eeprom access code is loaded to RAM and therefore the eeprom driver shouldn't have RWW problems; if EepAcLoadOnJobStart is set to false then the EEP code must not be placed in the same RWW partition as the EFLASH(data flash used as eeprom emulation).

In case of Async operations it is only possible to load EEP code to a different RWW partition that the one containing the EFLASH(data flash used for eeprom emulation)

Chapter 7
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 8
How to Configure

This chapter describes the configure Tresos plug-in for the Wdg Driver.

8.1 EEP Settings Config

This chapter describle how to configure a channel in EB Tresos plug-in EepInitConfiguration_0

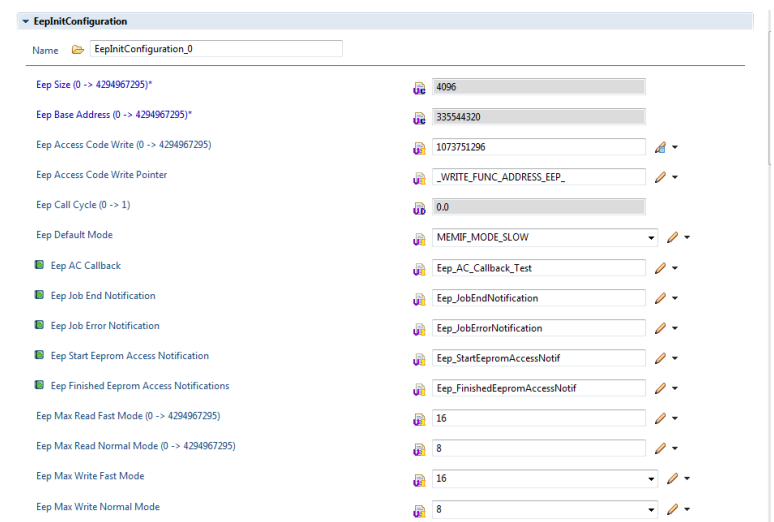

Figure 8-1. Config EEP Settings EepInitConfiguration_0

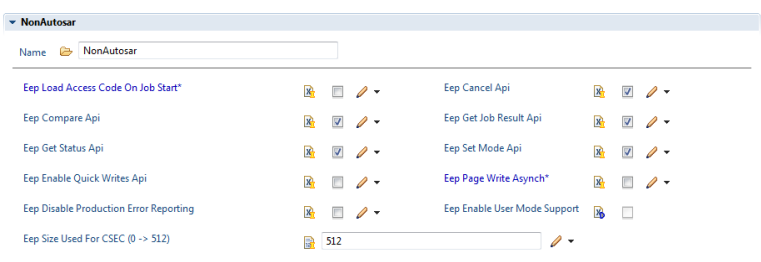

Figure 8-2. EEP NonAutosar Config Set.

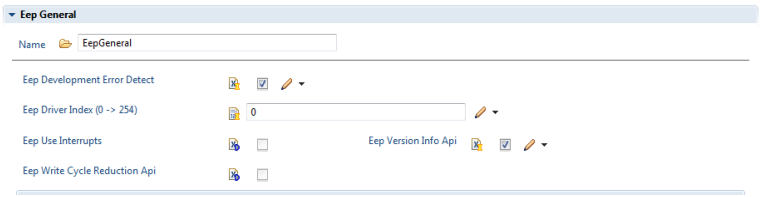

Figure 8-3. EEP General Config Set

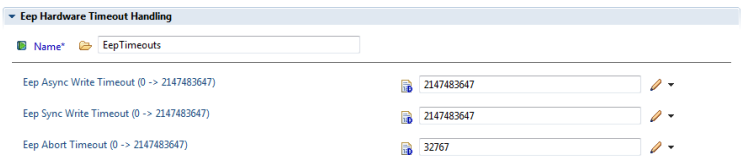

Figure 8-4. EEP Hardware Timeout Handling Config Set

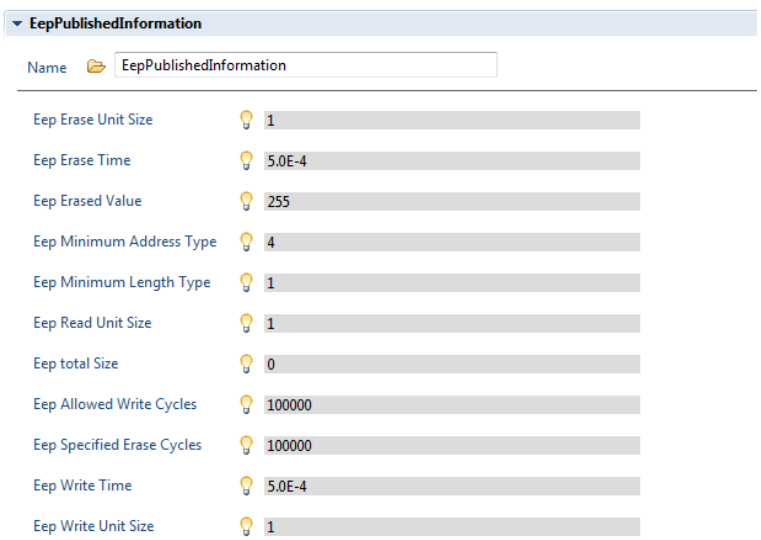

Figure 8-5. EEP Published Information Config Set

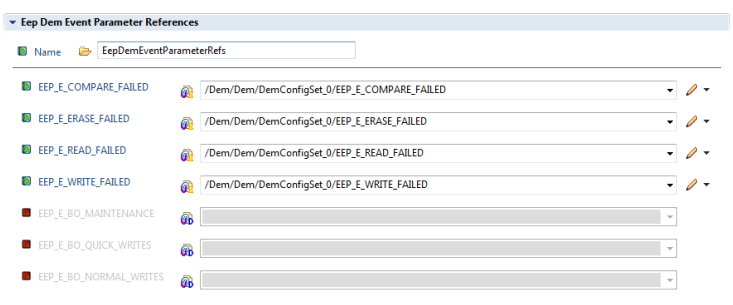

Figure 8-6. EEP Dem Event Parameter References

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number UM2EEPASR4.2 Rev0002R1.0.1
Revision 1.0