Integration Manual

for S32K14X ETH Driver

Document Number: IM2ETHASR4.3 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 integration manual describes the integration requirements for ETH Driver for S32K14X microcontrollers.

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
DEM	Diagnostic Event Manager
DET	Development Error Tracer
ENET	Ethernet MAC (Ethernet Controller)
ETH	Ethernet
ETHIF	Ethernet Interface
FEC	Fast Ethernet Controlled (Ethernet Controller)
FIFO	First-In First-Out Reception Storage
N/A	Not Applicable
MCU	Micro Controller Unit
MII	Media Independent Interface
RAM	Random Access Memory

Table continues on the next page...

Table 2-2. Acronyms and Definitions (continued)

Term	Definition
RMII	Reduced Media Independent Interface
VLE	Variable Length Encoding

- The term "Ethernet Controller" is related to the hardware module providing the Ethernet functionality.
- The term "Ethernet Driver" is related to the software handling the Ethernet Controller.
- The term "Application" is used for the software utilizing the Ethernet Driver.

2.5 Reference List

Table 2-3. Reference List

#	Title	Version
1	Specification of ETH 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
Building the Driver

This section describes the source files and various compilers, linker options used for building the Autosar ETH driver for NXP SemiconductorsS32K14X . It also explains the EB Tresos Studio plugin setup procedure.

3.1 Build Options

The ETH driver files are compiled using

- Green Hills Multi 7.1.4 / Compiler 2017.1.4
- (Linaro GCC 6.3-2017.06~dev) 6.3.1 20170509 (Wed Jan 24 16:21:45 CST 2018 build.sh rev=g27a1317 s=L631 Earmv7 -V release_g27a1317_build_Fed_Earmv7)
- IAR: V8.11.2

The compiler, linker flags used for building the driver are explained below:

Note

The TS_T40D2M10I1R0 plugin name is composed as follow:

TS_T = Target_Id

D = Derivative_Id

M = SW_Version_Major

I = SW_Version_Minor

R = Revision

(i.e. Target_Id = 40 identifies CORTEXM architecture and Derivative_Id = 2 identifies the S32K14X)

3.1.1 GHS Compiler/Linker/Assembler Options

Table 3-1. Compiler Options

Option	Description
-cpu=cortexm4	Selects target processor: Arm Cortex M4
-cpu=cortexm0plus	Selects target processor: Arm Cortex M0+
-ansi	Specifies ANSI C with extensions. This mode extends the ANSI X3.159-1989 standard with certain useful and compatible constructs.
-Osize	Optimize for size.
-dual_debug	Enables the generation of DWARF, COFF, or BSD debugging information in the object file
-G	Generates source level debugging information and allows procedure call from debugger's command line.
--no_exceptions	Disables support for exception handling
-Wundef	Generates warnings for undefined symbols in preprocessor expressions
-Wimplicit-int	Issues a warning if the return type of a function is not declared before it is called
-Wshadow	Issues a warning if the declaration of a local variable shadows the declaration of a variable of the same name declared at the global scope, or at an outer scope
-Wtrigraphs	Issues a warning for any use of trigraphs
-Wall	Enables all the warnings about constructions that some users consider questionable, and that are easy to avoid even in conjunction with macros.
--prototype_errors	Generates errors when functions referenced or called have no prototype
--incorrect_pragma_warnings	Valid #pragma directives with wrong syntax are treated as warnings
-noslashcomment	C++ like comments will generate a compilation error
-preprocess_assembly_files	Preprocesses assembly files
-nostartfile	Do not use Start files
--short_enum	Store enumerations in the smallest possible type
-c	Produces an object file (called input-file.o) for each source file.
--no_commons	Allocates uninitialized global variables to a section and initializes them to zero at program startup.
-keeptempfiles	Prevents the deletion of temporary files after they are used. If an assembly language file is created by the compiler, this option will place it in the current directory instead of the temporary directory. Produces an object file (called input-file.o) for each source file.
-list	Creates a listing by using the name of the object file with the .lst extension. Assembler option
-DAUTOSAR_OS_NOT_USED	-D defines a preprocessor symbol and optionally can set it to a value. AUTOSAR_OS_NOT_USED: By default in the package, the drivers are compiled to be used without Autosar OS. If the drivers are used with Autosar OS, the compiler option '-DAUTOSAR_OS_NOT_USED' must be removed from project options
-DDISABLE_MCAL_INTERMODULE_ASR_CHECK	-D defines a preprocessor symbol to disable the inter-module version check for AR_RELEASE versions. DISABLE_MCAL_INTERMODULE_ASR_CHECK: By default in the package, drivers are compiled to perform the inter-module version check as per Autosar BSW004. When the inter-module version check needs to be disabled then the DISABLE_MCAL_INTERMODULE_ASR_CHECK global define must be added to the list of compiler options.
-DGHS	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the GHS preprocessor symbol.

Table 3-2. Assembler Options

Option	Description
-cpu=cortexm4	Selects target processor: Arm Cortex M4
-cpu=cortexm0plus	Selects target processor: Arm Cortex M0+
-c	Produces an object file (called input-file.o) for each source file.
-preprocess_assembly_files	Preprocesses assembly files
-asm=list	Creates a listing by using the name of the object file with the .lst extension. Assembler option

Table 3-3. Linker Options

Option	Description
-Mn	Map file numeric ordering
-delete	Removal from the executable of functions that are unused and unreferenced
-v	Display removed unused functions
-ignore_debug_references	Ignores relocations from DWARF debug sections when using -delete.
-map	Creates a detailed map file
-keepmap	Keep the map file in the event of a link error
-lstartup	Link libstartup library -Run-time environment startup routines
-lsys	Link libsys library -Run-time environment system routines
-larch	Link libarch library -Target-specific run-time support. Any file produced by the Green Hills Compiler may depend on symbols in this library.
-lansi	Link libansi library -the standard C library
-L(/lib/thumb2)	Link thumb2 library
-lutf8_s32	Include utf8_s32.a to use the Wide Character Functions

3.1.2 IAR Compiler/Linker/Assembler Options

Table 3-4. Compiler Options

Option	Description
--cpu=Cortex-M4	Selects target processor: Arm Cortex M4
--cpu=Cortex-M0+	Selects target processor: Arm Cortex M0+
--cpu_mode=thumb	Selects generating code that executes in Thumb state.
--endian=little	Specifies the endianess of core: little endian.
-Ohz	Sets the optimization level to High, favoring size.
-c	Produces an object file (called input-file.o) for each source file.
--no_clustering	Disables static clustering optimizations.
--no_mem_idioms	Makes the compiler to not optimize code sequences that clear, set, or copy a memory region.
--no_explicit_zero_opt	Places the zero initialized variables in data section instead of bss.
--debug	Makes the compiler include information in the object modules.

Table continues on the next page...

Table 3-4. Compiler Options (continued)

Option	Description
--diag_suppress=Pa050	Suppresses diagnostic messages (warnings) about non-standard line endings.
-DAUTOSAR_OS_NOT_USED	-D defines a preprocessor symbol and optionally can set it to a value. AUTOSAR_OS_NOT_USED: By default in the package, the drivers are compiled to be used without Autosar OS. If the drivers are used with Autosar OS, the compiler option '-DAUTOSAR_OS_NOT_USED' must be removed from project options
-DIAR	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the IAR preprocessor symbol.
--require_prototypes	Forces the compiler to verify that all functions have proper prototypes.
--no_wrap_diagnostics	Disables line wrapping of diagnostic messages issued by compiler.
--no_system_include	Disables the automatic search for system include files.
-e	Enables language extensions. This option is needed by FLS driver which uses _packed structures.

Table 3-5. Assembler Options

Option	Description
--cpu=Cortex-M4	Selects target processor: Arm Cortex M4
--cpu=Cortex-M0+	Selects target processor: Arm Cortex M0+
--cpu_mode=thumb	Selects generating code that executes in Thumb state.
-g	Use this option to disable the automatic search for system include files.

Table 3-6. Linker Options

Option	Description
--cpu=Cortex-M4	Selects target processor: Arm Cortex M4
--cpu=Cortex-M0+	Selects target processor: Arm Cortex M0+
--map filename	Produces a map file.
--no_library_search	Disables automatic runtime library search.
--entry _start	Treats the symbol _start as a root symbol and as the start of the application.
--enable_stack_usage	Enables stack usage analysis.
--skip_dynamic_initialization	Suppress dynamic initialization during system startup.
--no_wrap_diagnostics	Disables line wrapping of diagnostic messages issued by linker.
--config	Specifies the configuration file to be used by the linker.

3.1.3 GCC Compiler/Linker/Assembler Options

Table 3-7. Compiler Options

Option	Description
-c	Produces an object file (called input-file.o) for each source file.
-Os	Use optimization for size.
-ggdb3	Produce debugging information for use by GDB. Level 3 includes extra information, such as all the macro definitions present in the program.
-mcpu=cortex-m4	Selects target processor: Arm Cortex M4
-mcpu=cortex-m0plus	Selects target processor: Arm Cortex M0+
-mthumb	Selects generating code that executes in Thumb state.
-ansi	Specifies ANSI C with extensions.
-mlittle-endian	Generate code for a processor running in little-endian mode.
-fomit-frame-pointer	Removes the frame pointer for all functions, which might make debugging harder.
-msoft-float	Use software floating-point instructions.
-fno-common	Specifies that the compiler should place uninitialized global variables in the data section of the object file, rather than generating them as common blocks.
-Wall	Enables all the warnings about constructions that some users consider questionable, and that are easy to avoid even in conjunction with macros.
-Wextra	Enables some extra warning flags that are not enabled by '-Wall'.
-Wstrict-prototypes	Warn if a function is declared or defined without specifying the argument types.
-Wno-sign-compare	Do not warn when a comparison between signed and unsigned values could produce an incorrect result when the signed value is converted to unsigned.
-fstack-usage	Geneates an extra file that specifies the maximum amount of stack used, on a per-function basis.
-fdump-ipa-all	Enables all inter-procedural analysis dumps.
-Werror=implicit-function-declaration	Generates an error when the prototype of the function is not defined..
-DAUTOSAR_OS_NOT_USED	-D defines a preprocessor symbol and optionally can set it to a value. AUTOSAR_OS_NOT_USED: By default in the package, the drivers are compiled to be used without Autosar OS. If the drivers are used with Autosar OS, the compiler option '-DAUTOSAR_OS_NOT_USED' must be removed from project options
-DGCC	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the GCC preprocessor symbol.
-std=c99	C programming language standard version c99

Table 3-8. Assembler Options

Option	Description
-mcpu=cortex-m4	Selects target processor: Arm Cortex M4
-mcpu=cortex-m0plus	Selects target processor: Arm Cortex M0+
-c	Produces an object file (called input-file.o) for each source file.
-mthumb	This option specifies that the assembler should start assembling Thumb instructions.
-x assembler-with-cpp	Indicates that the assembly code contains C directives and the C preprocessor must be run.

Table 3-9. Linker Options

Option	Description
-Map=filename	Print a link map to the file mapfile.
-T scriptfile	Use scriptfile as the linker script. This script replaces ld's default linker script(rather than adding to it), so commandfile must specify everything necessary to describe the output file.
--disable-newlib-supplied-syscalls -specs=nosys.specs	These options support for using newlib on core M0+
-u _printf_float -u _scanf_float	These options support generating profile report.
-nostartfiles	Do not use the standard system startup files when linking
-e _start	Specify that the program entry point is _start
-static	The --static flag tells the linker to link a static, not a dynamically linked
-lc	The -lc flag tells the linker to link this binary against the C library, which is newlib in our case.
-lnosys	The -lnosys flag tells the linker to link this binary against the "nosys" library
$(TOOLCHAIN_DIR)/arm-none-eabi/newlib/lib/thumb/v6-m $(TOOLCHAIN_DIR)/lib/gcc/arm-none-eabi/6.3.1/thumb/v6-m	Library for core M0+, added with -L and -B option
$(TOOLCHAIN_DIR)/arm-none-eabi/newlib/lib/thumb $(TOOLCHAIN_DIR)/arm-none-eabi/newlib/lib)	Library for core M4, added with -L and -B option

3.2 Files Required for the Compilation

This section describes the include files required to compile, assemble (if assembler code) and link the Autosar Ethernet driver for S32K14X microcontrollers.

To avoid integration of incompatible files, all the include files from other modules shall have the same AR_MAJOR_VERSION and AR_MINOR_VERSION, i.e. only files with the same Autosar major and minor versions can be compiled.

Eth Driver Files

- Eth_TS_T40D2M10I1R0\src\Eth.c
- Eth_TS_T40D2M10I1R0\src\Eth_Irq.c
- Eth_TS_T40D2M10I1R0\src\Eth_Buffers.c
- Eth_TS_T40D2M10I1R0\src\Eth_Ipw.c
- Eth_TS_T40D2M10I1R0\src\Eth_Enet.c
- Eth_TS_T40D2M10I1R0\include\Eth.h
- Eth_TS_T40D2M10I1R0\include\Eth_Enet_Counters.h
- Eth_TS_T40D2M10I1R0\include\Eth_Irq.h

- Eth_TS_T40D2M10I1R0\include\Eth_Ipw.h
- Eth_TS_T40D2M10I1R0\include\Eth_Enet.h

Eth Generated Files

- Eth_<VariantName>_PBcfg.c - This file should be generated by the user using a configuration tool. The file contains the definition of the init pointer for the respective variant.
- Eth_Cfg.h - This file should be generated by the user using a configuration tool for compilation
- Eth_Cfg.c - This file should be generated by the user using a configuration tool for compilation

As a deviation from standard:

- Eth_<VariantName>_PBcfg.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)
- Eth_Cfg.c file will contain the definition for all configuration structures containing only variables that are not variant aware, configured and generated only once. This file alone does not contain the whole structure needed by Eth_Init function to configure the driver. Based on the number of variants configured in the EcuC, there can be more than one configuration structure for one module even for PreCompile variant.

Files from Base common folder

- Base_TS_T40D2M10I1R0\include*.h

Files from Det folder:

- Det_TS_T40D2M10I1R0\include\Det.h
- Det_TS_T40D2M10I1R0\src\Det.c

Files from Dem folder:

- Dem_TS_T40D2M10I1R0\include\Dem.h
- Dem_TS_T40D2M10I1R0\src\Dem.c

Files from EthIf folder:

- EthIf_TS_T40D2M10I1R0\include\EthIf_Cbk.h
- EthIf_TS_T40D2M10I1R0\src\EthIf_Cbk.c

Files from EthTrcv folder:

- EthTrcv_TS_T40D2M10I1R0\include\EthTrcv.h
- EthTrcv_TS_T40D2M10I1R0\src\EthTrcv.c

Files from Rte folder:

- Rte_TS_T40D2M10I1R0\include\SchM_Eth.h

3.3 Setting up the Plugins

The Ethermet driver was designed to be configured by using the EB tresos Studio (version EB tresos Studio 24.0.1 b180321-0610.)

Location of various files inside the Eth module plug-in folder is explained below.

- Module Parameter Definition:
  - Eth_TS_T40D2M10I1R0\config\Eth.xdm
- Code Generation Templates for Pre-Compile, Link-Time, PostBuild configuration parameters:
  - Eth_TS_T40D2M10I1R0\generate\include\Eth_Cfg.h
  - Eth_TS_T40D2M10I1R0\generate\src\Eth_variant_PBcfg.c
  - Eth_TS_T40D2M10I1R0\generate\src\Eth_Cfg.c

Steps to generate the configuration:

1. Copy both the module folder (Eth_TS_T40D2M10I1R0) into EB tresos Studio installation path under "\plugins" folder.
2. Set the desired Tresos Output location folder for the generated sources and header files.
3. Use the EB tresos Studio GUI to modify ECU configuration parameters values.
4. Generate the configuration files.

Using alternative plugins directory:

1. Copy the folder (Eth_TS_T40D2M10I1R0) into desired alternative directory like "<MyDirectoryPath>\<MyDirectory>" under the folders "<MyDirectoryPath> \<MyDirectory>\eclipse\plugins".
2. Create a text file with the extension ".link" into the EB tresos installation in a folder called "links" (e.g. <MyTresosIntallation>\links\Eth_TS_T40D2M10I1R0.link).
3. Put the following text in that Eth_TS_T40D2M10I1R0.link file: "path=<MyDirectoryPath>/<MyDirectoryName>". Please make sure the path is described with forward slashes.

Chapter 4
Function calls to module

4.1 Function Calls during Start-up

The ETH module shall be initialized by the Eth_Init() function call during the start-up. Please note that GPIO pins used to connect the Ethernet Transceiver have to be properly configured to the appropriate mode prior the ETH module initialization. Configure fast slew-rate for all ethernet pins (otherwise packet losts may occure).

4.2 Function Calls during Shutdown

The application should ensure that all the Ethernet frame transmissions have been completed before the module is shut down by the Eth_SetControllerMode(0, ETH_MODE_DOWN) function call. Note that all not transmitted buffers are flushed and all locked transmit buffers are unlocked when the controller is disabled. All receive buffers are also discarded when the controller is stopped so the Eth_Receive function should be called before the shut down.

4.3 Function Calls during Wake-up

None.

Chapter 5
Module requirements

5.1 Exclusive Areas to be defined in BSW Scheduler

The ETH_EXCLUSIVE_AREA_00 is used in Eth_Transmit to protect Eth_u8LockedTxBufCount (counter to store the number of messages sent need confirmation). When one of the functions using the shared resources enters the exclusive area, the other function must not enter the exclusive area until the first function leaves it.

The ETH_EXCLUSIVE_AREA_01 is used in Eth_TxConfirmation to protect Eth_Enet_ERR006358() (Process the errata 6358 workaround), Eth_au8TxBufFlags (software flag), Eth_u8LockedTxBufCount (counter to store the number of messages sent need confirmation). When one of the functions using the shared resources enters the exclusive area, the other function must not enter the exclusive area until the first function leaves it.

The ETH_EXCLUSIVE_AREA_02 is used in Eth_SetGlobalTime to protect Eth_LocalTime (store the value for current time). When one of the functions using the shared resources enters the exclusive area, the other function must not enter the exclusive area until the first function leaves it.

5.1.1 Critical Region Exclusivity Matrix

Below is the table depicting the exclusivity between different critical region IDs from the ETH driver. If there is an “X” in a table, it means that those 2 critical regions cannot interrupt each other.

Table 5-1. Exclusive Areas

	ETH_EXCLUSIVE_AREA_00	ETH_EXCLUSIVE_AREA_01
ETH_EXCLUSIVE_AREA_00		x

Table continues on the next page...

Table 5-1. Exclusive Areas (continued)

	ETH_EXCLUSIVE_AREA_00	ETH_EXCLUSIVE_AREA_01
ETH_EXCLUSIVE_AREA_01	x	

5.2 Peripheral Hardware Requirements

The Ethernet Transceiver should be connected to the Ethernet Controller module pins which should be configured to be used by the Ethernet Controller. In addition these pins should have configured the highest possible slew rate.

5.3 ISR to Configure within OS - Dependencies

The following ISRs (see Table 5-2 table) are used by the ETH Driver and they need to be assigned to a priority level when the interrupts are switched on (the Ethernet Driver can be run in poll-driven mode).

Table 5-2. Ethernet Driver ISR

ISR Name	Hardware Interrupt Vector
`Eth_RxIrqHdlr_0`	74
`Eth_TxIrqHdlr_0`	73
`Eth_TimerWrapIsr_0`	72

5.4 ISR Macro

MCAL drivers use the ISR macro to define the functions that will process hardware interrupts. Depending on whether the OS is used or not, this macro can have different definitions:

OS is not used (AUTOSAR_OS_NOT_USED is defined):

i. If USE_SW_VECTOR_MODE is defined:

```
#define ISR(Eth_TxIrqHdlr_x) void Eth_TxIrqHdlr_x(void)

#define ISR(Eth_RxIrqHdlr_x) void Eth_RxIrqHdlr_x(void)
```

In this case, drivers' interrupt handlers are normal C functions and the prolog/epilog handle the context save and restore.

ii. If USE_SW_VECTOR_MODE is not defined:

```
#define ISR(Eth_TxIrqHdlr_x) INTERRUPT_FUNC Eth_TxIrqHdlr_x(void)

#define ISR(Eth_RxIrqHdlr_x) INTERRUPT_FUNC Eth_RxIrqHdlr_x(void)
```

In this case, drivers' interrupt handlers must save and restore the execution context.

Custom OS is used (AUTOSAR_OS_NOT_USED is not defined)

```
#define ISR(Eth_TxIrqHdlr_x) void OS_isr_##Eth_TxIrqHdlr_x()

#define ISR(Eth_RxIrqHdlr_x) void OS_isr_##Eth_RxIrqHdlr_x()
```

In this case, OS is handling the execution context when an interrupt occurs. Drivers' interrupt handlers are normal C functions.

Other vendor's OS is used - AUTOSAR_OS_NOT_USED is not defined. Please refer to the OS documentation for description of the ISR macro.

5.5 Other AUTOSAR Modules - Dependencies

- **Port:**The PORT module is used to configure the port pins with the needed modes, before they are used by the Ethernet module.
- **Det** (only if EthDevErrorDetect=true): The DET module is used for enabling Development error detection. The API function used is Det_ReportError(). The activation/deactivation of the Development Error Detection is configurable using the "EthDevErrorDetect" configuration parameter.
- **Dem:**The DEM module is used for enabling reporting of production relevant error status.
- **Rte:**The RTE module is needed for implementing data consistency of exclusive areas that are used by Ethernet module.
- **EthIf:** This module callbacks EthIf_RxIndication and EthIf_TxConfirmation are used by the ETH Driver to notify the upper layers about the Ethernet frame transmission and/or reception.
- **EthTrcv:** This module is used for report indication to transceiver layer by functions: EthTrcv_ReadMiiIndication or EthTrcv_WriteMiiIndication
- **Mcu:**The MCU driver provides services for basic microcontroller initialization, power down functionality, reset and microcontroller specific functions required by other MCAL software modules. The clocks need to be initialized prior to using the Ethernet driver.
- **Mcl:** This module is used for calling cache workaround instruction.

- **Resource:**Resource module is used to select microcontroller derivative.
- **EcuC:**The ECUC module is used for ECU configuration. MCAL modules need ECUC to retrieve the variant information.
- **Base:**The BASE module contains the common files/definitions needed by all MCAL modules.

5.6 Data Cache Restriction

None.

5.7 User Mode Support

Eth drivers can run in both supervisor mode and user mode. For this platform, there is not any specific measure required for running in user mode.

Chapter 6
Main API Requirements

6.1 Main functions calls within BSW scheduler

ETH Driver support main functions that can be configured to be scheduled by BSW scheduler:

- `FUNC(void, ETH_CODE) Eth_MainFunction(VAR(void, AUTOMATIC))`

The function checks for controller errors and lost frames. Used for polling state changes. Calls EthIf_CtrlModeIndication when the controller mode changed.

6.2 Api Requirements

None.

6.3 Calls to Notification Functions, Callbacks, Callouts

Notification functions

- None

Call-backs

- `EthIf_RxIndication` This EthIf interface is called when one or more unread Ethernet frames are waiting in the receive buffers to be processed. Call is made from the `Eth_Receive` function or the `Eth_RxIrqHdlr_0` interrupt handler.
- `EthIf_TxConfirmation` This EthIf interface is called when one or more Ethernet frames were transmitted and had been set to confirm the transmission. Call is made from the `Eth_TxConfirmation` function or the `Eth_TxIrqHdlr_0` interrupt handler.

Callouts

- None

Chapter 7
Memory Allocation

7.1 Sections to Be Defined in MemMap.h

Table 7-1. Sections to be defined in MemMap.h

Section name	Type of section	Description
ETH_START_SEC_CONFIG_DATA_UNSPECIFIED	Configuration Data	Start of Memory Section for Config Data.
ETH_STOP_SEC_CONFIG_DATA_UNSPECIFIED	Configuration Data	End of Memory Section for Config Data.
ETH_START_SEC_CODE	Code	Start of memory Section for Code.
ETH_STOP_SEC_CODE	Code	End of memory Section for Code.
ETH_START_SEC_VAR_NO_INIT_UNSPECIFIED	Variables	Used for variables, structures, arrays when the SIZE (alignment) does not fit the criteria of 8,16 or 32 bit. These variables are never cleared and never initialized by start-up code.
ETH_STOP_SEC_VAR_NO_INIT_UNSPECIFIED	Variables	End of above section.
ETH_START_SEC_VAR_NO_INIT_8	Variables	Used for variables which have to be aligned to 8 bit. For instance used for variables of size 8 bit or used for composite data types: arrays, structs containing elements of maximum 8 bits. These variables are never cleared and never initialized by start-up code.
ETH_STOP_SEC_VAR_NO_INIT_8	Variables	End of above section.
ETH_START_SEC_VAR_INIT_UNSPECIFIED	Variables	Used for variables, structures, arrays, when the SIZE (alignment) does not fit the criteria of 8,16 or 32 bit. These variables are initialized with values after every reset.
ETH_STOP_SEC_VAR_INIT_UNSPECIFIED	Variables	End of above section.
ETH_START_SEC_VAR_INIT_32	Variables	Used for variables which have to be aligned to 32 bit. For instance used for variables of size 32 bit or used

Table continues on the next page...

Table 7-1. Sections to be defined in MemMap.h (continued)

Section name	Type of section	Description
		for composite data types: arrays, structs containing elements of maximum 32 bits. These variables are initialized with values after every reset.
ETH_STOP_SEC_VAR_INIT_32	Variables	End of above section.
ETH_START_SEC_VAR_INIT_16	Variables	Used for variables which have to be aligned to 16 bit. For instance used for variables of size 16 bit or used for composite data types: arrays, structs containing elements of maximum 16 bits. These variables are initialized with values after every reset.
ETH_STOP_SEC_VAR_INIT_16	Variables	End of above section.
ETH_START_SEC_VAR_INIT_8	Variables	Used for variables which have to be aligned to 8 bit. For instance used for variables of size 8 bit or used for composite data types: arrays, structs containing elements of maximum 8 bits. These variables are initialized with values after every reset.
ETH_STOP_SEC_VAR_INIT_8	Variables	End of above section.
ETH_START_SEC_VAR_INIT_BOOLEAN	Variables	Used for initialized boolean variables.
ETH_STOP_SEC_VAR_INIT_BOOLEAN	Variables	End of above section.
ETH_START_SEC_VAR_NO_INIT_UNSPECIFIED_NO_CACHEABLE	Non-Cacheable Variables	Used for variables, structures, arrays when the SIZE (alignment) does not fit the criteria of 8,16 or 32 bit. Normally, this section is used for store Eth Driver buffer memory. **CAUTION:** This section must be cache inhibited.
ETH_STOP_SEC_VAR_NO_INIT_UNSPECIFIED_NO_CACHEABLE	Non-Cacheable Variables	End of above section.
ETH_START_SEC_CONST_32	Constant	Used for constant which have to be aligned to 32 bit. For instance used for constant of size 32 bit or used for composite data types: arrays, structs containing elements of maximum 32 bits. These constant are initialized with values after every reset.
ETH_STOP_SEC_CONST_32	Constant	End of above section.
ETH_START_SEC_CONST_UNSPECIFIED	Constant	Used for constant when the SIZE (alignment) does not fit the criteria of 8,16 or 32 bit. For instance used for constant for composite data types: arrays, structs, etc . These

Table continues on the next page...

Table 7-1. Sections to be defined in MemMap.h (continued)

Section name	Type of section	Description
		constant are initialized with values after every reset.
ETH_STOP_SEC_CONST_UNSPECIFIED	Constant	End of above section.

7.2 Linker command file

Memory shall be allocated for every section defined in ETH_MemMap.h

Chapter 8
Configuration parameters considerations

Configuration parameter class for Autosar ETH driver fall into the following variants as defined below:

8.1 Configuration parameters

Table 8-1. Configuration Parameters

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
EthGeneral			
	EthDevErrorDetect	PC	PC
	EthIndex	PC	PC
	EthMaxCtrlsSupported	PC	PC
	EthVersionInfoApi	PC	PC
	EthVersionInfoApiMacro	PC	PC
	EthUpdatePhysAddrFilter	PC	PC
	EthEnableUserModeSupport	PC	PC
EthGeneral/EthVendorSpecific			
	EthMulticastPoolSize	PC	PC
	EthUseMultiBufferRxFrames	PC	PC
	EthEnableRxFrameWrap	PC	PC
	EthUseMultiBufferTxFrames	PC	PC
	EthDisableDemEventDetect	PC	PC
	EthMaxTXBuffersSupported	PC	PC
EthConfigSet_n/EthCtrlConfig/ EthCtrlConfig_n			
	EthCtrlEnableMii	PC/PB/LT	PC
	EthCtrlEnableRxInterrupt	PC/PB/LT	PC
	EthCtrlEnableTxInterrupt	PC/PB/LT	PC
	EthCtrlIdx	PC/PB/LT	PC
	EthCtrlRxBufLenByte	PC/PB/LT	PC/PB/LT
	EthCtrlTxBufLenByte	PC/PB/LT	PC/PB/LT

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
	EthRxBufTotal	PC/PB/LT	PC/PB/LT
	EthTxBufTotal	PC/PB/LT	PC/PB/LT
	EthCtrlPhyAddress	PC/PB/LT	PC/PB/LT
EthConfigSet_n/EthCtrlConfig/ EthCtrlConfig_n/ EthDemEventParameterRefs			
	ETH_E_ACCESS	PC/PB/LT	PC/PB/LT
EthConfigSet_n/EthCtrlConfig/ EthCtrlConfig_n/EthVendorSpecific			
	EthPhyInterface	PC/PB/LT	PC/PB/LT
	EthCtrlSupportMDIO	PC/PB/LT	PC/PB/LT
	EthMIISpeedControl	PC/PB/LT	PC/PB/LT
	EthFullDuplexEnable	PC/PB/LT	PC/PB/LT
	EthReceiveBroadcast	PC/PB/LT	PC/PB/LT
	EthEnablePromiscuousMode	PC/PB/LT	PC/PB/LT
	EthDropInvalidMAC	PC/PB/LT	PC/PB/LT
	EthInterPacketGap	PC/PB/LT	PC/PB/LT
	EthMDIOHoldTime	PC/PB/LT	PC/PB/LT
EthConfigSet_n/EthCtrlConfig/ EthCtrlConfig_n/EthVendorSpecific/ EthEnableLoopbackMode		PC/PB/LT	PC/PB/LT
	EthInternalLoopbackMode	PC/PB/LT	PC/PB/LT

Chapter 9
Integration Steps

9.1 Integration Steps

This section gives a brief overview of the steps needed for integrating Ethernet :

- Generate the required ETH configurations. For more details refer to section Files Required for the Compilation
- Allocate proper memory sections in ETH_MemMap.h and linker command file. For more details refer to section Sections to Be Defined in MemMap.h
- Compile & build the ETH with all the dependent modules. For more details refer to section Building the Driver

Chapter 10
ISR Reference

10.1 ISR Reference

`void Eth_RxIrqHdlr_0(void);` Reception interrupt handler;

`void Eth_TxIrqHdlr_0(void);` Transmission interrupt handler;

Chapter 11
External Assumptions for ETH driver

11.1 External Assumptions for ETH driver

The section presents requirements that must be complied with when integrating ETH driver into the application.

[SMCAL_CPR_EXT163]

<< If interrupts are locked a centralized function pair to lock and unlock interrupts shall be used. >>

[SMCAL_CPR_EXT177]

<< When caches are enabled and data buffers are allocated in cachable memory regions the buffers involved in DMA transfer shall be aligned with both start and end to cache line size.

>>

NOTE

Rationale: This ensures that no other buffers/variables to compete for the same cache lines.

[SWS_Eth_00149]

<< The types specified in SWS_EthernetDriver shall be declared in Eth_GeneralTypes.h. >>

NOTE

Under control of BASE module

[SWS_Eth_00157]

<< Name: Eth_ReturnType

Type: Enumeration

Range: ETH_OK success

ETH_E_NOT_OK general failure

ETH_E_NO_ACCESS Ethernet hardware access failure

Description: Ethernet Driver specific return type.

>>

NOTE

Under control of BASE module

[SWS_Eth_00158]

<< Name: Eth_ModeType

Type: Enumeration

Range: ETH_MODE_DOWN Controller disabled

ETH_MODE_ACTIVE Controller enabled

Description: This type defines the controller modes

>>

NOTE

Under control of BASE module

[SWS_Eth_00159]

<< Name: Eth_StateType

Type: Enumeration

Range: ETH_STATE_UNINIT Driver is not yet configured

ETH_STATE_INIT Driver is configured

Description: Status supervision used for Development Error Detection. The state shall be available for debugging.

>>

NOTE

Under control of BASE module

[SWS_Eth_00160]

<< Name: Eth_FrameType

Type: --

Range: uint16 0x0000 - 0xFFFF See [21]

Description: This type defines the Ethernet frame type used in the Ethernet frame header

>>

NOTE

Under control of BASE module

[SWS_Eth_00161]

<< Name: Eth_DataType

Type: --

Range: uint8 0x00 - 0xFF 8, 16 or 32 bit CPU

uint16 0x0000 - 0xFFFF 8 or 16 bit CPU

uint32 0x00000000 - 0xFFFFFFFF 32 bit CPU

Description: This type defines the Ethernet data type used for data transmission. Its definition depends on the used CPU.

>>

NOTE

Under control of BASE module

[SWS_Eth_00175]

<< Ethernet buffer identifier type >>

NOTE

Under control of BASE module

[SWS_Eth_00162]

<< Name: Eth_RxStatusType

Type: Enumeration

Range: ETH_RECEIVED Ethernet frame has been received, no further frames available

ETH_NOT_RECEIVED Ethernet frame has not been received, no further frames available

ETH_RECEIVED_MORE_DATA_AVAILABLE Ethernet frame has been received, more frames are available

Description: Used as out parameter in Eth_Receive() indicates whether a frame has been received and if so, whether more frames are available or frames got lost.

>>

NOTE

Under control of BASE module

[SWS_Eth_00163]

<< Name: Eth_FilterActionType

Type: Enumeration

Range: ETH_ADD_TO_FILTER add the MAC address to the filter, meaning allow reception

ETH_REMOVE_FROM_FILTER remove the MAC address from the filter, meaning reception is blocked in the lower layer

Description: The Enumeration Type Eth_FilterActionType describes the action to be taklen for the MAC address given in *PhysAddrPtr.

>>

NOTE

Under control of BASE module

[SWS_Eth_00177]

<< Name: Eth_TimeStampQualType

Type: Enumeration

Range: ETH_VALID 0

ETH_INVALID 1

ETH_UNCERTAIN 2

Description: Depending on the HW, quality information regarding the evaluated time stamp might be supported. If not supported, the value shall be always Valid. For Uncertain and Invalid values, the upper layer shall discard the time stamp.

>>

NOTE

Under control of BASE module

[SWS_Eth_00178]

<< Name: Eth_TimeStampType

Type: Structure

Element: uint32 nanoseconds Nanoseconds part of the time

uint32 seconds 32 bit LSB of the 48 bits Seconds part of the time

uint16 secondsHi 16 bit MSB of the 48 bits Seconds part of the time

Description: Variables of this type are used for expressing time stamps including relative time and absolute calendar time. The absolute time starts acc. to "[5], Annex C/C1" specification at 1970-01-01.

0 to 281474976710655s

== 3257812230d

[0xFFFF FFFF FFFF]

0 to 999999999ns

[0x3B9A C9FF]

invalid value in nanoseconds: [0x3B9A CA00] to [0x3FFF FFFF]

Bit 30 and 31 reserved, default: 0

>>

NOTE

Under control of BASE module

[SWS_Eth_00179]

<< Name: Eth_TimeIntDiffType

Type: Structure

Element: Eth_TimeStampType diff time difference

boolean sign Positive (True) / negative (False) time

Description: Variables of this type are used to express time differences in a usual way. The described "TimeInterval" type referenced in "[5], chapter 6.3.3.3" will not be used and hereby slightly simplified.

>>

NOTE

Under control of BASE module

[SWS_Eth_00180]

<< Name: Eth_RateRatioType

Type: Structure

Element: Eth_TimeIntDiffType IngressTimeStampDelta IngressTimeStampSync2 - IngressTimeStampSync1

Eth_TimeIntDiffType OriginTimeStampDelta OriginTimeStampSync2[FUP2] - OriginTimeStampSync1[FUP1]

Description: Variables of this type are used to express frequency ratios.

>>

NOTE

Under control of BASE module

[SWS_Eth_00120]

<< This tabe defines all interfaces required to fulfill the core functionality of the DET

module. >>

NOTE

This is not Eth requirement

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer’s technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer's applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, COOLFLUX, EMBRACE, GREENCHIP, HITAG, I2C BUS, ICODE, JCOP, LIFE VIBES, MIFARE, MIFARE CLASSIC, MIFARE DESFire, MIFARE PLUS, MIFARE FLEX, MANTIS, MIFARE ULTRALIGHT, MIFARE4MOBILE, MIGLO, NTAG, ROADLINK, SMARTLX, SMARTMX, STARPLUG, TOPFET, TRENCHMOS, UCODE, Freescale, the Freescale logo, AltiVec, C-5, CodeTEST, CodeWarrior, ColdFire, ColdFire+, C-Ware, the Energy Efficient Solutions logo, Kinetis, Layerscape, MagniV, mobileGT, PEG, PowerQUICC, Processor Expert, QorIQ, QorIQ Qonverge, Ready Play, SafeAssure, the SafeAssure logo, StarCore, Symphony, VortiQa, Vybrid, Airfast, BeeKit, BeeStack, CoreNet, Flexis, MXC, Platform in a Package, QUICC Engine, SMARTMOS, Tower, TurboLink, and UMEMS are trademarks of NXP B.V. All other product or service names are the property of their respective owners. AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro, µVision, Versatile are trademarks or registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. The Power Architecture and Power.org word marks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org.

© 2019 NXP B.V.

Document Number IM2ETHASR4.3 Rev0001R1.0.1
Revision 1.0