

Integration Manual

for MPC5634M DIO Driver

Document Number: IM14DIOASR3.0R2.0.0

Rev. 1.1





Contents

Section Number	Title	Page
Chapter 1		
Revision History		
Chapter 2		
Introduction		
2.1	Supported Derivatives.....	7
2.2	Overview.....	7
2.3	About this Manual.....	8
2.4	Acronyms and Definitions.....	8
2.5	Reference List.....	9
Chapter 3		
Building the Driver		
3.1	Build Options.....	11
3.1.1	CW Compiler/Linker/Assembler Options.....	11
3.1.2	DIAB Compiler/Linker/Assembler Options.....	13
3.1.3	GHS Compiler/Linker/Assembler Options.....	15
3.1.4	CSMC Compiler/Linker/Assembler Options.....	16
3.2	Files required for Compilation.....	17
3.3	Setting up the Plug-ins.....	19
Chapter 4		
Function calls to module		
4.1	Function Calls during Start-up.....	21
4.2	Function Calls during Shutdown.....	21
4.3	Function Calls during Wake-up.....	21
Chapter 5		
Module requirements		
5.1	Exclusive areas to be defined in BSW scheduler.....	23
5.2	Peripheral Hardware Requirements.....	23
5.3	ISR to configure within OS – dependencies.....	23

Section Number	Title	Page
5.4	ISR Macro.....	23
5.5	Other AUTOSAR modules - dependencies.....	24

Chapter 6 Memory Allocation

6.1	Sections to be defined in MemMap.h.....	25
6.2	Linker command file.....	26

Chapter 7 Configuration parameters considerations

7.1	Configuration Parameters.....	27
-----	-------------------------------	----

Chapter 8 Integration Steps

Chapter 1

Revision History

Table 1-1. Revision History

Revision	Date	Author	Description
1.0	07-Feb-2011	Giovanni Di Martino	Document generation
1.1	19-Dec-2011	Robin Gupta	Updated for Monaco RTM 2.0.0



Chapter 2

Introduction

This integration manual describes the integration requirements for DIO Driver for MPC5634M microcontrollers.

2.1 Supported Derivatives

The software described in this document is intended to be used with the following microcontroller devices of Freescale Semiconductor.

Table 2-1. MPC5634M Derivatives

Freescale Semiconductor	mpc5634m_bga208, mpc5634m_qfp144, mpc5634m_qfp176
-------------------------	--

All of the above microcontroller devices are collectively named as MPC5634M.

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
ASM	Assembler
AUTOSAR	Automotive Open System Architecture
BSMI	Basic Software Make file Interface
C/CPP	C and C++ Source Code
DEM	Diagnostic Event Manager
DET	Development Error Tracer
DIO	Digital Input Output
EcuM	ECU state Manager
N/A	Not Applicable
OS	Operating System
VLE	Variable Length Encoding

2.5 Reference List

Table 2-3. Reference List

#	Title	Version
1	AUTOSAR 3.0DIO Driver Software Specification Document.	V2.2.0 R3.0 Rev 0001
2	MPC5634M Reference Manual	Rev. 6, 4 October 2011

Chapter 3

Building the Driver

This section describes the source files and various compilers, linker options used for building the Autosar DIO driver for Freescale SemiconductorMPC5634M. It also explains the EB Tresos Studio plugin setup procedure.

3.1 Build Options

The DIO driver files are compiled using

- GHS 5.2.4
- DIAB 5_8_0_02 wind00198363 20100511 123238
- CW Version 4.3 build 182

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

Note

The TS_T2D14M20I0R0 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 = 2 identifies PowerPC architecture and
Derivative_Id = 14 identifies the MPC5634M)

3.1.1 CW Compiler/Linker/Assembler Options

Table 3-1. Compiler Options

Option	Description
-proc Zen	Generates and links object code for Zen processor. The compiler uses unsigned as the default parameter for the -char switch
-lang c	Expects source code to conform to the language specified by the ISO/IEC 9899-1990 ("C90") standard
-opt all	This option is selected all optimization (the same as -opt speed,level=4,intrinsics,noframe)
-common off	Disables moving uninitialized data into a common section
-sdatathreshold 0	Specifies the threshold size (in bytes) for an item considered by the linker to be small data. (The linker stores small data items in the Small Data address space. The compiler can generate faster code to access this data.)
-sdata2threshold 0	Specifies the threshold size (in bytes) for an item considered by the linker to be small constant data. (The linker stores small constant data items in the Small Constant Data address space.)
-vle	Tells the compiler and linker to generate and lay out Variable Length Encoded (VLE) instructions, available on Zen variants of Power Architecture processors
-use_lmw_stmw on	Enables the use of multiple load and store instructions for function prologues and epilogues
-ir	Include the debug information
-ppc_asm_to_vle	Converts regular Power Architecture assembler mnemonics to equivalent VLE (Variable Length Encoded) assembler mnemonics in the inline assembler
-cpp_exceptions off	When on, generates executable code for C++ exceptions. When off, generates smaller, faster executable code
-func_align 4	Specifies alignment of functions in executable code
-sym dwarf-2,full	Generate DWARF-2-conforming debugging information (Debug With Arbitrary Record Format)
-gdwarf-2	Generate DWARF-2-conforming debugging information (Debug With Arbitrary Record Format). The linker ignores debugging information that is not in the Dwarf 1, Dwarf 2 format
-w on	Turns on most warning messages
-r	Compiler should expect function prototypes
-w undefmacro	Issues warning messages on the use of undefined macros in #if and #elif conditionals
-char unsigned	Controls the default sign of the char data type: char data items are unsigned
-nosyspath	Performs a search of both the user and system paths, treating #include statements of the form #include xyz the same as the form #include "xyz"
-fp none	No floating point code generation
-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
-DEU_DISABLE_ANSILIB_CALLS	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the EU_DISABLE_ANSILIB_CALLS preprocessor symbol.
-DMCAL_CER_VALIDATION	-D defines a preprocessor symbol for CER Report

Table continues on the next page...

Table 3-1. Compiler Options (continued)

Option	Description
-DMCAL_VERSION_CHECK	-D defines enable the cross check between the AutoSar component Version Numbers
-DMWERKS	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the CWpreprocessor symbol.

Table 3-2. Assembler Options

Option	Description
-proc Zen	Generates and links object code for Zen processor. The compiler uses unsigned as the default parameter for the -char switch
-vle	Tells the compiler and linker to generate and lay out Variable Length Encoded (VLE) instructions, available on Zen variants of Power Architecture processors
-sym dwarf-2,full	Generate DWARF-2-conforming debugging information (Debug With Arbitrary Record Format)
-gdwarf-2	Generate DWARF-2-conforming debugging information (Debug With Arbitrary Record Format). The linker ignores debugging information that is not in the Dwarf 1, Dwarf 2 format.

Table 3-3. Linker Options

Option	Description
-proc Zen	Generates and links object code for Zen processor. The compiler uses unsigned as the default parameter for the -char switch
-code_merging all	Removes duplicated functions to reduce object code size
-far_near_addressing	Simplifies address computations to reduce object code size and improve performance
-vle_enhance_merging	Removes duplicated functions that are called by functions that use VLE instructions to reduce object code size
-listdwarf	DWARF debugging information in the linker's map file
-sym dwarf-2,full	Generate DWARF-2-conforming debugging information (Debug With Arbitrary Record Format)
-char unsigned	Controls the default sign of the char data type: char data items are unsigned.

3.1.2 DIAB Compiler/Linker/Assembler Options

Table 3-4. Compiler Options

Option	Description
-tPPCE200Z3VEG:simple	Sets target processor to PPCE200Z3, generates ELF using EABI conventions, All Single Hardware Floating Point (Single precision uses hardware, double precision is mapped to single precision), selects simple environment settings for Startup Module and Libraries
-Xdialect-ansi	Follow the ANSI C standard with some additions
-XO	Enables extra optimizations to produce highly optimized code
-Xsize-opt	Optimize for size rather than speed when there is a choice

Table continues on the next page...

Table 3-4. Compiler Options (continued)

Option	Description
-Xsmall-data=0	Set Size Limit for “small data” Variables to zero.
-Xsmall-const=0	Set Size Limit for “small const” Variables to zero.
-Xno-common	Disable use of the “COMMON” feature so that the compiler or assembler will allocate each uninitialized public variable in the .bss section for the module defining it, and the linker will require exactly one definition of each public variable
-Xnested-interrupts	Allow nested interrupts
-Xalign-functions=4	Align each function on an address boundary divisible by 4
-g	Generate symbolic debugger information. Do most target-independent optimizations. Also, disable most target-dependent optimizations: option -g2 also disables basic reordering and all peephole optimizations.
-Xdebug-dwarf2	Generate symbolic debug information in dwarf2 format
-Xdebug-local-all	Force generation of type information for all local variables
-Xdebug-local-cie	Create common information entry per module
-Xdebug-struct-all	Force generation of type information for all typedefs, struct, union and class types
-Xforce-declarations	Generates warnings if a function is used without a previous declaration
-ee1481	Generate an error when the function was used before it has been declared
-Xforce-prototypes	Generate warnings if a function is used without a previous prototype declaration
-Xmacro-undefined-warn	Generates a warning when an undefined macro name occurs in a #if preprocessor directive
-Xlink-time-lint	Enable the checking of object and function declarations across compilation units, as well as the consistency of compiler options used to compile source files
-Xlint	Generate warnings when suspicious and non-portable C code is encountered. Enables all warnings
-ei1604	Suppress the warning messages 1604.
-W:as;,-l	Pass the option “-l” (lower case letter L) to the assembler to get an assembler listing file
-Wa,-Xisa-vle	Instruct the assembler to expect and assemble VLE (Variable Length Encoding) instructions rather than BookE instructions.
-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
-DDIAB	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the DIAB preprocessor symbol.
-DEU_DISABLE_ANSILIB_CALLS	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the EU_DISABLE_ANSILIB_CALLS preprocessor symbol.
-DMCAL_CER_VALIDATION	-D defines a preprocessor symbol for CER Report

Table 3-5. Assembler Options

Option	Description
-tPPCE200Z3VEN:simple	Selects target processor: PPCE200Z3, generates ELF using EABI conventions, NO floating point support, selects simple environment settings for Startup Module and Libraries.
-g	Dump the symbols in the global symbol table in each archive file.
-Xisa-vle	Expect and assemble VLE (Variable Length Encoding) instructions rather than Book E instructions. The default code section is named .text_vle instead of .text, and the default code section fill "character" is set to 0x44444444 instead of 0. The .text_vle code section will have ELF section header flags marking it as VLE code, not Book E code.
-Xasm-debug-on	Generate debug line and file information

Table 3-6. Linker Options

Option	Description
-tPPCE200Z3VEN:simple	Selects target processor: PPCE200Z3, generates ELF using EABI conventions, NO floating point support, selects simple environment settings for Startup Module and Libraries.
-Xelf	Generates ELF object format for output file
-m6	Generates a detailed link map and cross reference table
-lc	Specifies to linker to search for libc.a
-Xlink-time-lint	Enable the checking of object and function declarations across compilation units, as well as the consistency of compiler options used to compile source files.
-Xlibc-old	Enables usage of legacy (pre-release 5.6) libraries

3.1.3 GHS Compiler/Linker/Assembler Options

Table 3-7. Compiler Options

Option	Description
-cpu=ppc563xm	Selects target processor: ppc563xm
-ansi	Enforces strict ANSI mode (C89 standard)
-noSPE	Disables the use of SPE and vector floating point instructions by the compiler.
-Ospace	Optimize for size
-sda=0	Enables the Small Data Area optimization with a threshold of 0.
--no_commons	Allocates uninitialized global variables to a section and initializes them to zero at program startup. This may improve optimizations by giving the compiler optimizer more information about the location of the variable.
-vle	Enables VLE code generation
-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

Table continues on the next page...

Table 3-7. Compiler Options (continued)

Option	Description
-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
--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
-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
-DGHS	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the GHS preprocessor symbol.
-DEU_DISABLE_ANSILIB_CALLS	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the EU_DISABLE_ANSILIB_CALLS preprocessor symbol.
-DMCAL_CER_VALIDATION	-D defines a preprocessor symbol for CER Report
-DMCAL_VERSION_CHECK	-D defines enable the cross check between the AutoSar component Version Numbers

Table 3-8. Assembler Options

Option	Description
-cpu=ppc563xm	Selects target processor: ppc563xm

Table 3-9. Linker Options

Option	Description
-cpu=ppc563xm	Selects target processor: ppc563xm
-nostartfiles	Do not use Start files.
-vle	Enables VLE code generation
-linker_warnings	Display linker warnings

3.1.4 CSMC Compiler/Linker/Assembler Options

Table 3-10. Compiler Options

Option	Description
-l	Create listing file; this option directs the compiler to produce an assembly language file with C source line interspersed in it. Please note that the C source lines are commented in the assembly language file: they start with ";".
+modvc	Memory model with "medium size" application, in detail: "data" less than 64kb, "constants" less than 64kb, no code size limit
+rev	Tells the compiler to reverse the order of bits in the bitfields. You need this option in order to use most non-Cosmic header files.
-pc99	authorize the repetition of the const and volatile modifiers in the declaration either directly or indirectly in the typedef.
-pxf	prefix filenames in the debug information with absolute full path name.
+debug	produce debug information to be used by the debug utilities provided with the compiler and by any external debugger.
-DCSMC	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the CSMC preprocessor symbol.
-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
-DEU_DISABLE_ANSILIB_CALLS	-D defines a preprocessor symbol and optionally can set it to a value. This one defines the EU_DISABLE_ANSILIB_CALLS preprocessor symbol.
-DMCAL_CER_VALIDATION	-D defines a preprocessor symbol for CER Report
-DMCAL_VERSION_CHECK	-D defines enable the cross check between the AutoSar component Version Numbers

Table 3-11. Assembler Options

Option	Description
-l	create a listing file. The name of the listing file is derived from the input file name by replacing the suffix by the ".ls" extension

Table 3-12. Linker Options

Option	Description
-p	display symbols with physical address instead of logical address in the map file.

3.2 Files required for Compilation

This section describes the include files required to compile, assemble (if assembler code) and link the DIO driver for MPC5634M 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.

DIO Files

- ..\Dio_TS_T2D14M20I0R0\src\Dio.c
- ..\Dio_TS_T2D14M20I0R0\src\Siu_LLD_dio.c
- ..\Dio_TS_T2D14M20I0R0\include\Dio.h
- ..\Dio_TS_T2D14M20I0R0\include\Dio_LLD.h
- ..\Dio_TS_T2D14M20I0R0\include\Reg_eSys_SIU.h
- ..\Dio_TS_T2D14M20I0R0\include\Siu_LLD.h
- ..\Dio_TS_T2D14M20I0R0\include\Siu_LLD_CfgEx.h

DIO Generated Files

- ..\Dio_TS_T2D14M20I0R0\generate_PB\src\Dio_PBcfg.c (For PB Variant) - This file should be generated by the user using a configuration tool for compilation
- ..\Dio_TS_T2D14M20I0R0\generate_PC\src\Dio_Cfg.c (For PC Variant) - This file should be generated by the user using a configuration tool for compilation
- ..\Dio_TS_T2D14M20I0R0\generate_PC\include\Dio_Cfg.h - This file should be generated by the user using a configuration tool for compilation

Files from Base common folder

- ..\Base_TS_T2D14M20I0R0\include\Cer.h
- ..\Base_TS_T2D14M20I0R0\include\Compiler.h
- ..\Base_TS_T2D14M20I0R0\include\Compiler_Cfg.h
- ..\Base_TS_T2D14M20I0R0\include\ComStack_Types.h
- ..\Base_TS_T2D14M20I0R0\include\Mcal.h
- ..\Base_TS_T2D14M20I0R0\include\MemMap.h
- ..\Base_TS_T2D14M20I0R0\include\Platform_Types.h
- ..\Base_TS_T2D14M20I0R0\include\Reg_eSys.h
- ..\Base_TS_T2D14M20I0R0\include\Reg_Macros.h
- ..\Base_TS_T2D14M20I0R0\include\Std_Types.h

Base Generated Files

- ..\Base_TS_T2D14M20I0R0\generate_PC\include\modules.h
- ..\Base_TS_T2D14M20I0R0\generate_PC\include\Soc_Ips.h

Files from Dem folder:

- ..\Dem_TS_T2D14M20I0R0\include\Dem.h
- ..\Dem_TS_T2D14M20I0R0\include\Dem_IntErrId.h
- ..\Dem_TS_T2D14M20I0R0\include\Dem_Types.h

Files from Det folder:

- ..\Det_TS_T2D14M20I0R0\include\Det.h

3.3 Setting up the Plug-ins

The DIO driver was designed to be configured by using the EB Tresos Studio (version Tresos 2010a.sr4 20100415-release2010a-sr4 or later.)

Location of various files inside the DIO module folder:

- VSMD (Vendor Specific Module Definition) file in EB tresos Studio XDM format:
 - ..\Dio_TS_T2D14M20I0R0\config\Dio.xdm
- VSMD (Vendor Specific Module Definition) file(s) in AUTOSAR compliant EPD format:
 - ..\Dio_TS_T2D14M20I0R0\autosar\Dio.epd
- Code Generation Templates for Pre-Compile time configuration parameters:
 - ..\Dio_TS_T2D14M20I0R0\generate_PC\src\Dio_Cfg.c
 - ..\Dio_TS_T2D14M20I0R0\generate_PC\include\Dio_Cfg.h
- Code Generation Templates for Post-Build time configuration parameters:
 - ..\Dio_TS_T2D14M20I0R0\generate_PB\src\Dio_PBcfg.c

Steps to generate the configuration:

1. Copy the module folders Dio_TS_T2D14M20I0R0 , Base_TS_T2D14M20I0R0 , Resource_TS_T2D14M20I0R0 , EcuM_TS_T2D14M20I0R0 into the Tresos 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.

Dependencies

- **RESOURCE** is required to select processor derivative. Current Dio driver has support for the following derivatives, everyone having attached a Resource file: mpc5634m_bga208, mpc5634m_qfp144, mpc5634m_qfp176.
- **ECUM** is required for a reference to the Wakeup source for this controller as defined in the ECU State Manager.
- **DET** is required for signaling the development error detection (parameters out of range, null pointers, etc).
- **DEM** is required for signaling the production error detection (hardware failure, etc).



Chapter 4

Function calls to module

4.1 Function Calls during Start-up

None.

4.2 Function Calls during Shutdown

None.

4.3 Function Calls during Wake-up

None.

Chapter 5

Module requirements

5.1 Exclusive areas to be defined in BSW scheduler

None.

5.2 Peripheral Hardware Requirements

The DIO driver uses DIO's peripheral.

5.3 ISR to configure within OS – dependencies

None.

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:

a. OS is not used - AUTOSAR_OS_NOT_USED is defined:

i. If USE_SW_VECTOR_MODE is defined:

```
#define ISR(IsrName) void IsrName(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(IsrName) INTERRUPT_FUNC void IsrName(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(IsrName) void OS_isr_##IsrName()
```

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

- **Det:** This module is necessary for enabling Development error detection. The API function used is Det_ReportError(). The activation/deactivation of Development error detection is configurable using 'DioDevErrorDetect' configuration parameter.
- **Dem:** This module is necessary for enabling reporting of production relevant error status. The API function used is Dem_ReportErrorStatus().
- **EcuM:** This module is necessary for a reference to the Wakeup source for this controller as defined in the ECU State Manager.
- **Resource:** Sub-Derivative model is selected from Resource configuration.

Chapter 6

Memory Allocation

6.1 Sections to be defined in MemMap.h

For Post Build data:

```
#ifndef DIO_START_CONFIG_DATA_UNSPECIFIED
#define DIO_START_CONFIG_DATA_UNSPECIFIED
#define MEMMAP_ERROR
/*Memory Section for Post Build Data to be defined here.
   Example given in the next line*/
#pragma ghs section const=".pbdio_cfg"
#endif
#ifndef DIO_STOP_CONFIG_DATA_UNSPECIFIED
#define DIO_STOP_CONFIG_DATA_UNSPECIFIED
#define MEMMAP_ERROR
/*End of section to be mentioned here. Example given in the
   next line.*/
#pragma ghs section
#endif
```

For Code:

```
#ifndef DIO_START_SEC_CODE
#define DIO_START_SEC_CODE
#define MEMMAP_ERROR
/*Memory Section for Code to be defined here.*/
#endif
#ifndef DIO_STOP_SEC_CODE
#define DIO_STOP_SEC_CODE
#define MEMMAP_ERROR
/*End of section to be mentioned here*/
#endif
```

For Variables:

```
#ifndef DIO_START_SEC_VAR_UNSPECIFIED
#define DIO_START_SEC_VAR_UNSPECIFIED
#define MEMMAP_ERROR
/*Memory Section for Variables to be defined here.*/
#endif
#ifndef DIO_STOP_SEC_VAR_UNSPECIFIED
```

Linker command file

```
#undef DIO_STOP_SEC_VAR_UNSPECIFIED
#undef MEMMAP_ERROR
/*End of section to be mentioned here*/
#endif
```

For Constant data:

```
#ifdef DIO_START_SEC_CONST_UNSPECIFIED
#undef DIO_START_SEC_CONST_UNSPECIFIED
#undef MEMMAP_ERROR
/*Memory Section for Constants to be defined here.*/
#endif
#ifdef DIO_STOP_SEC_CONST_UNSPECIFIED
#undef DIO_STOP_SEC_CONST_UNSPECIFIED
#undef MEMMAP_ERROR
/*End of section to be mentioned here*/
#endif
```

6.2 Linker command file

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

Chapter 7

Configuration parameters considerations

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

7.1 Configuration Parameters

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

Table 7-1. Configuration Parameters

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
DioGeneral			
	DioDevErrorDetect	Pre Compile parameter for all Variants of Configuration	Pre Compile
	DioVersionInfoApi	Pre Compile parameter for all Variants of Configuration	Pre Compile
	DioReversePortBits	Pre Compile parameter for all Variants of Configuration	Pre Compile
DioPort			
	DioPortId	VariantPC or VariantPB	Post Build
DioChannel			
	DioChannelId	VariantPC or VariantPB	Post Build
DioChannelGroup			
	DioChannelGroupIdentification	Pre Compile parameter for all Variants of Configuration	Pre Compile
	DioPortMask	VariantPC or VariantPB	Post Build
	DioPortOffset	VariantPC or VariantPB	Post Build

Table continues on the next page...

Table 7-1. Configuration Parameters (continued)

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
CommonPublishedInformation			
	VendorId	VariantPC or VariantPB	Post Build
	ModuleId	VariantPC or VariantPB	Post Build
	ArMajorVersion	VariantPC or VariantPB	Post Build
	ArMinorVersion	VariantPC or VariantPB	Post Build
	ArPatchVersion	VariantPC or VariantPB	Post Build
	SwMajorVersion	VariantPC or VariantPB	Post Build
	SwMinorVersion	VariantPC or VariantPB	Post Build
	SwPatchVersion	VariantPC or VariantPB	Post Build
	VendorApiInfix	VariantPC or VariantPB	Post Build

Chapter 8

Integration Steps

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

- Generate the required DIO configurations. For more details refer to [Setting up the Plug-ins](#)
- Allocate proper memory sections in MemMap.h and linker command file. For more details refer to section [Sections to be defined in MemMap.h](#)
- Make sure all include files for compilation. For more details refer to section [Files required for Compilation](#)
- Compile and build the DIO with all the dependent modules. For more details refer to section [Building the Driver](#)



How to Reach Us:

Home Page:

www.freescale.com

Web Support:

<http://www.freescale.com/support>

USA/Europe or Locations Not Listed:

Freescale Semiconductor
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:

Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:

Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:

Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:

Freescale Semiconductor Literature Distribution Center
1-800-441-2447 or +1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductors products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale Semiconductor 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. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claims alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

RoHS-compliant and/or Pb-free versions of Freescale products have the functionality and electrical characteristics as their non-RoHS-complaint and/or non-Pb-free counterparts. For further information, see <http://www.freescale.com> or contact your Freescale sales representative.

For information on Freescale's Environmental Products program, go to <http://www.freescale.com/epp>.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.

© 2011 Freescale Semiconductor, Inc.



AUTOSAR and AUTOSAR logo are registered trademarks of AUTOSAR GbR (www.autosar.org)