Integration Manual

for MPC563XM SPI Driver

Document Number: IM14SPIASR3.0R2.0.0
Rev. 1.2

freescale™

Contents

Section Number **Title** **Page**

Chapter 1
Revision History

Chapter 2
Introduction

Chapter 3
Building the Driver

Chapter 4
Function Calls to Module

Chapter 5
Module Requirements

Chapter 6
Main API Requirements

Chapter 7
Memory Allocation

Chapter 8
Configuration Parameter Considerations

Chapter 9
Integration Steps

Chapter 1
Revision History

Table 1-1. Document Change History

Date	Version	Changed by	Change description
10-Feb-2011	1.0	Srikanth M.S	Initial version
13-Apr-2011	1.1	Rutuja Bichile	Updated for Timed Serial Bus Support
20-Dec-2011	1.2	Subramanya M Naik	Updated for MPC5634M RTM 2.0.0 Release

Chapter 2
Introduction

This Integration Manual describes the integration requirements for Autosar SPI Driver for Freescale Semiconductor's MPC5634M microcontrollers .

The roadmap for the document is as follows:

Building the Driver : This section gives a brief overview of the build procedure (compiler,linker options and source files) and Plugins setup.

Function Calls to Module : This section lists the various function calls to modules during Start-up, Shutdown and Wake-up.

Module Requirements : This section specifies the various module requirements related to

- Exclusive areas to be defined in BSW scheduler
- Peripheral Hardware Requirements
- Specific interface to other modules
- ISR to configure within OS
- Dependencies with other AUTOSAR modules

Main API Requirements : This section specifies the requirements related to to the main SPI_main API and gives a brief overview of the main functions calls within BSW scheduler, API_Name Requirements and calls to notification functions, callbacks, callouts.

Memory Allocation : This section describes the memory allocation requirements namely the sections to be defined in MemMap.h and the linker command file.

Configuration Parameter Considerations : This section covers the various Pre Compile, Link Time and Post Build time configuration parameters.

Integration Steps : This section describes in brief the steps for integrating SPI module.

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. Supported 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.

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.

2.4 Acronyms and Definitions

Table 2-2. Acronyms and Definitions

Term	Definition
BSW	Basic Software
DEM	Diagnostic Event Manager
DET	Development Error Tracer
ECU	Electronic Control Unit

Table continues on the next page...

Table 2-2. Acronyms and Definitions (continued)

Term	Definition
SPI	Serial Peripheral Interface
ISR	Interrupt Service Routine
OS	Operating System
MCU	Microcontroller Unit
GUI	Graphical User Interface
API	Application Programming Interface
TSB	Timed Serial Bus
MSC	Micro Second Channel
PB Variant	Post Build Variant
PC Variant	Pre Compile Variant

2.5 Reference List

Table 2-3. Reference List

#	Title	Version
2	MPC5634M Reference Manual	Rev. 6, 4 October 2011
2	AUTOSAR_SWS_SPI_HandlerDriver.pdf Reference Manual	V2.2.0 R3.0 Rev 0001

Chapter 3
Building the Driver

This section describes the source files and various compiler, linker options used for building the Autosar SPIdriver for Freescale Semiconductor's MPC5634M microcontrollers. It also explains the Plugins setup procedure.

NOTE

The Spi_TS_T2D14M20I0R0 is composed as follow: TS_T<Target_Id>D<Derivative_Id>M<SW_Version_Major>I <SW_Version_Minor>R0 (i.e. Target_Id = 2 identifies PowerPC architecture and Derivative_Id = 14 identifies the MPC5634M

3.1 Build Options

The driver files are compiled using GHS 5.1.7, DIAB 5_8_0_02 wind00198363 20100511 123238, COSMIC Software PPC C Cross Compiler V4.3.4 - 16 Nov 2011 - Win32-F and CW Version 4.3 build 182. The compiler, linker flags used for building the driver are explained below:

3.1.1 GHS Compiler/Linker/Assembler Options

Table 3-1. 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.

Table continues on the next page...

Table 3-1. Compiler Options (continued)

Option	Description
--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
-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 3-2. Assembler Options

Option	Description
-cpu=ppc563xm	Selects target processor: ppc563xm

Table 3-3. Linker Options

Option	Description
-cpu=ppc563xm	Selects target processor: ppc563xm
-nostartfiles	Do not use Start files.

Table continues on the next page...

Table 3-3. Linker Options (continued)

Option	Description
-vle	Enables VLE code generation
-linker_warnings	Display linker warnings

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
-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

Table continues on the next page...

Table 3-4. Compiler Options (continued)

Option	Description
-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 CW Compiler/Linker/Assembler Options

Table 3-7. 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 3-7. 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 3-8. 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 3-9. 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.

3.1.4 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

Table continues on the next page...

Table 3-10. Compiler Options (continued)

Option	Description
+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.
-odB5	disable the optimization B5.
-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 Autosar SPI driver for Freescale Semiconductor’s 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.

SPI Files:

Table 3-13. Include Files

..\Spi_TS_T2D14M20I0R0\include\	Dma_LLD.h
	Dspi_LLD.h
	Spi.h
	Spi_LLD.h
	Dspi_LLD_CfgEx.h
	Mcu_Cfg.h
	Reg_eSys_DMA.h
	Reg_eSys_DSPI.h
	Reg_eSys_PIT.h
	Reg_eSys_SIUL.h

Table 3-14. Source Files

..\Spi_TS_T2D14M20I0R0\src\	Spi_Irq.c
	Dma_Spi_LLD.c
	Dspi_LLD.c
	Spi.c

Spi_Cfg.c (For PC Variant) - This file should be generated by the user using a configuration tool for compilation

Spi_Lcfg.c (For LT Variant) - This file should be generated by the user using a configuration tool for compilation

Spi_PBcfg.c (For PB Variant) - This file should be generated by the user using a configuration tool for compilation

Spi_Cfg.h - This file should be generated by the user using a configuration tool for compilation

Other include files:

Table 3-15. Files from Base folder:

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

Table 3-16. Files from Dem folder:

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

Table 3-17. Files from Det folder:

..\Det_TS_T2D14M20I0R0\generc\include\	Det.h

Table 3-18. Files from SchM folder:

..\SchM_TS_T2D14M20I0R0\include\	SchM_Spi.h

3.3 Setting up the Plug-ins

All the Autosar MCAL drivers for MPC5634M were designed to be configured using Tresos® Studio configuration and code generation tool from tresos Tresos 2010a.sr4 20100415-release2010a-sr4.

Location of various files inside the plugin folder is explained below.

Module Parameter Definition File:

..\Spi_TS_T2D14M20I0R0\config\Spi.xdm

Code Generation Templates for Pre-Compile time configuration parameters:

..\Spi_TS_T2D14M20I0R0\generate_PC\src\Spi_Cfg.c

..\Spi_TS_T2D14M20I0R0\generate_PC\include\Spi_Cfg.h

Code Generation Templates for Post-Build time configuration parameters:

..\Spi_TS_T2D14M20I0R0\generate_PB\src\Spi_PBcfg.c

Code Generation Templates for Link time configuration parameters:

..\Spi_TS_T2D14M20I0R0\generate_LT\src\Spi_Lcfg.c

Steps to generate configurations:

1. Copy the module folders Resource<plugin_name>, Spi_ TS_T2D14M20I0R0 into the Tresos plug-in folder.
2. Set the desired Tresos Output location folder for the generated sources and header files.
3. Use the Tresos GUI to modify configuration parameters values.
4. Generate the Pre-Compile, Link time and Post-Build files.

Chapter 4
Function Calls to Module

4.1 Function Calls during Start-up

The SPI Handler & Driver shall be initialized before the SPI peripherals are used. The API to be called for this purpose is Spi_Init(). The MCU and PORT modules shall be initialized before SPI is initialized.

4.2 Function Calls during Shutdown

SPI can be silenced by calling Spi_DeInit().

4.3 Function Calls during Wake-up

N/A

Chapter 5
Module Requirements

5.1 Exclusive areas to be defined in BSW scheduler

In the current implementation, SPI is using the services of Schedule Manager (SchM) for entering and exiting the critical regions.

SchM implementation is done by the integrators of the MCAL using OS or non-OS services.

For testing the SPI, stubs are used for SchM.

Some SPI driver global variables updates are performed by ISRs before calling the user notification functions. In order to avoid the scenario where an executing SPI driver function is preempted by an SPI ISR, which is modifying some of the global variables, some exclusive areas are defined

The ISR critical regions must not block the other critical regions to avoid deadlocks. This is ensured by exiting the ISR critical region before calling the user notification functions.

The following critical regions are used in the SPI driver:

5.2 SPI_EXCLUSIVE_AREA_01

Used in function Spi_SyncTransmit, to protect the status of the given sequence result. Also it protects the global variable which contains the status of the Spi_SyncTransmit service. As stated by the Autosar, this service cannot be called when another sequence is during transmission, using this service.

5.3 SPI_EXCLUSIVE_AREA_02

Used in function Spi_SyncTransmit, to protect the status of the given sequence result. Also it protects the global variable which contains the status of the Spi_SyncTransmit service. As stated by the Autosar, this service cannot be called when another sequence is during transmission, using this service.

5.4 SPI_EXCLUSIVE_AREA_03

Used in the internal function Spi_ScheduleJob, protects the schedule mechanism for the situation when a scheduling operation determined by a pending Spi_AsyncTransmit() call may be preempted by a job scheduling requested by an ISR event. It also protect concurrent Spi_AsyncTransmit() calls to schedule in the same time different jobs on the same DSPI unit.

5.5 SPI_EXCLUSIVE_AREA_04

Used in the internal function Spi_ScheduleNextJob, protects the schedule mechanism for the situation when a scheduling operation determined by a pending Spi_AsyncTransmit() call may be preempted by a job scheduling requested by an ISR event.

5.6 SPI_EXCLUSIVE_AREA_05

Used in the internal function Spi_LockJobs, guaranties the atomicity of locking for the entire set of jobs belonging to an asynchronous sequence.

5.7 SPI_EXCLUSIVE_AREA_06

Used in the internal function Spi_UnlockRemainingJobs, guaranties the atomicity of unlocking for the entire set of jobs belonging to an asynchronous sequence.

5.8 Critical region exclusivity matrix

Below is the table depicting the exclusivity between different critical region IDs from the SPI driver. If there is an “X” in a table, it means that those 2 critical regions cannot interrupt each other. The critical regions from interrupts are grouped in “ISR Critical Regions”. If an exclusive area is “exclusive” with the composed “ISR Critical Regions” group, it means that it is exclusive with each one of the ISR critical regions.

Table 5-1. Critical region exclusivity matrix

	SPI_EA_01	SPI_EA_02	SPI_EA_03	SPI_EA_04	SPI_EA_05	SPI_EA_06	ISR Critical Regions
SPI_EA_01	X	X					
SPI_EA_02	X	X					
SPI_EA_03			X	X			X
SPI_EA_04			X	X			X
SPI_EA_05					X	X	
SPI_EA_06					X	X	
ISR Critical Regions			X	X			X

Note

- SPI_EA_xx means SPI_EXCLUSIVE_AREA_xx

5.9 Peripheral Hardware Requirements

N/A

5.10 ISR to configure within OS – dependencies

The following ISRs are used by the SPI driver and need to be assigned to a priority level. The interrupt vector numbers corresponding to the DMA channel configuration is as shown in Table 3 and interrupt vector numbers corresponding to PIO_FIFO is as shown in Table 4. The interrupt occurs each time the EOQ bit in SR register arises.

(Note: Unused interrupts shouldn't be configured in the OS.)

Table 3 shows an example Dma Vector location for the corresponding channels configured. :

Table 5-2. SPI ISRs for DMA channel

DMA Name	DMA Interrupt Vector	DMA Channel
Spi_LLD_IsrRxDma_DSPI_0	24	13
Spi_LLD_IsrRxDma_DSPI_1	26	15

Table 5-3. SPI ISRs for PIO_FIFO

ISR Name	Hardware interrupt vector
Spi_LLD_IsrEOQ_DSPI_0	132
Spi_LLD_IsrEOQ_DSPI_1	137

Note: In case of AUTOSAR_OS_NOT_USED, the compiler option "-DUSE_SW_VECTOR_MODE" must be added to the list of compiler options to be used with interrupt controller configured to be in software vector mode.

5.11 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:

1. OS is not used - AUTOSAR_OS_NOT_USED is defined:
   - 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.

   - 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.

2. Freescale Semiconductor 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.

3. 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.

Please refer to the OS documentation for description of the ISR macro.

5.12 Other AUTOSAR modules - dependencies

Development Error Tracer:

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 the

'SpiDevErrorDetect' configuration parameter.

Diagnostic Event Manager:

This module is necessary for enabling Production error detection. The API function used is Dem_ReportErrorStatus ().

Mcu:

The SPI reference clock is provided by MCU plugin. For each DSPI in use, a transmit and a receive DMA channel need to be defined and routed through the DMA Multiplexer.

The Table5 shows an example DMA configuration. For more information, refer to section 4.4

Table 5-4. SPI DMA Channel Multiplexer

DMA Name	DMA Source
DSPI 0 Transmit DMA	12 (DSPI0.SpiPhyTxDmaChannel)
DSPI 0 Receive DMA	13 (DSPI0.SpiPhyRxDmaChannel)
DSPI 1 Transmit DMA	14 (DSPI1.SpiPhyTxDmaChannel)
DSPI 1 Receive DMA	15 (DSPI1.SpiPhyRxDmaChannel)

PORT module: For each DSPI, the SCK, SOUT, SIN and CSx_y signals need to be configured. The following table shows an example configuration for DSPI 0 :

Table 5-5. SPI Pins

Signal	PortPin Direction	Port Control Register (PCR)	PortPinLevelValue	PortPinMux
CS1_0	Out	105	High	Option1
CS0_0	Out	106	High	Option1
SCK	Out	102	High	Option1
SOUT	Out	103	High	Option1
SIN	In	104	High	Option1

Chapter 6
Main API Requirements

6.1 Main functions calls within BSW scheduler

The function Spi_MainFunction_Driving() should be called periodically only if polling mode is enabled for Spi_AsyncTransmit() .

6.2 Calls to notification functions, callbacks, callouts

Call-back Notifications:

None.

User Notification:

The SPI Handler & Driver provides notifications per job and sequence in asynchronous mode. The notifications can be configured as pointers to user defined functions. If notification is not desired, the appropriate EndNotification field shall be left blank.

For asynchronous transmissions, job and sequences notifications are performed before the scheduling of the next job (contrary to the recommendation given by SPI088) . In this way, calls like Spi_SetupIB() or Spi_WriteIB() can be targeted on the next schedulable jobs, before the starting of the job transfer.

Chapter 7
Memory Allocation

7.1 Sections to be defined in MemMap.h

For Post Build data:

```
#ifdef SPI_START_CONFIG_DATA_UNSPECIFIED

#undef SPI_START_CONFIG_DATA_UNSPECIFIED

#undef MEMMAP_ERROR

/*Memory Section for Post Build Data to be defined here. Example given in the next line*/

#pragma ghs section const=".pbspi_cfg"

#endif


#ifdef SPI_STOP_CONFIG_DATA_UNSPECIFIED

#undef SPI_STOP_CONFIG_DATA_UNSPECIFIED

#undef MEMMAP_ERROR

/*End of section to be mentioned here. Example given in the next line.*/

#pragma ghs section

#endif
```

For Code:

```
#ifdef SPI_START_SEC_CODE

#undef SPI_START_SEC_CODE
```

```
#undef MEMMAP_ERROR
```

/*Memory Section for Code to be defined here.*/

```
#endif
```

```
#ifdef SPI_STOP_SEC_CODE

#undef SPI_STOP_SEC_CODE

#undef MEMMAP_ERROR
```

/*End of section to be mentioned here*/

```
#endif
```

For Variables:

```
#ifdef SPI_START_SEC_VAR_UNSPECIFIED

#undef SPI_START_SEC_VAR_UNSPECIFIED

#undef MEMMAP_ERROR
```

/*Memory Section for Variables to be defined here.*/

```
#endif
```

```
#ifdef SPI_STOP_SEC_VAR_UNSPECIFIED

#undef SPI_STOP_SEC_VAR_UNSPECIFIED

#undef MEMMAP_ERROR
```

/*End of section to be mentioned here*/

```
#endif
```

For Constant data:

```
#ifdef SPI_START_SEC_CONST_UNSPECIFIED

#undef SPI_START_SEC_CONST_UNSPECIFIED
```

```
#undef MEMMAP_ERROR
```

*/*Memory Section for Constants to be defined here.*/*

```
#endif
```

```
#ifdef SPI_STOP_SEC_CONST_UNSPECIFIED

#undef SPI_STOP_SEC_CONST_UNSPECIFIED

#undef MEMMAP_ERROR
```

*/*End of section to be mentioned here*/*

```
#endif
```

7.2 Linker command file

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

Chapter 8
Configuration Parameter Considerations

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

Table 8-1. Configuration Parameters

Configuration Container	Configuration Parameters	Configuration Variant	Current Implementation
SpiDriver			
	SPI_MAX_CHANNEL	PC, LT or PB	Pre Compile (1)
	SPI_MAX_JOB	PC, LT or PB	Pre Compile (1)
	SPI_MAX_SEQUENCE	PC, LT or PB	Pre Compile (1)
SpiChannel			
	TSBModeEnable	Pre-Compile all Variants	Pre Compile
	SpiChannelId	Pre-Compile all Variants	Pre Compile
	SpiChannelType	PC, LT or PB	Pre Compile (2)
	SpiIbNBuffers	PC, LT or PB	Pre Compile (3)
	SpiDataWidth	PC, LT or PB	Post Build
	SpiDefaultData	PC, LT or PB	Post Build
	SpiEbMaxlength	PC, LT or PB	Post Build
	SpiTransferStart	PC, LT or PB	Post Build
SpiExternalDevice			
	SpiBaudRate	PC, LT or PB	Post Build
	SpiCs	PC, LT or PB	Post Build
	SpiCsPolarity	PC, LT or PB	Post Build
	SpiDataShiftEdge	PC, LT or PB	Post Build
	SpiEnableCs	PC, LT or PB	Post Build
	SpiShiftClockIdleLevel	PC, LT or PB	Post Build
	SpiTimeClk2Cs	PC, LT or PB	Post Build
	SpiTImeCs2Clk	Vendor specific	Post Build
	SpiTimeCs2Cs	Vendor specific	Post Build

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

	SpiCsContinuous	Vendor specific	Post Build
SpiJob			
	TSBModeEnable	Pre-Compile all Variants	Pre Compile
	DualReceiverSupport	Pre-Compile all Variants	Pre Compile
	SpiJobId	Pre-Compile all Variants	Pre Compile
	SpiHwUnit	PC, LT or PB	Post Build
	SpiJobEndNotification	PC, LT or PB	Post Build
	SpiJobStartNotification	PC, LT or PB	Post Build
	SpiJobPriority	PC, LT or PB	Post Build
	ChannelAssignment	PC, LT or PB	Post Build
	DeviceAssignment	PC, LT or PB	Post Build
	TSBFrameSize	PC, LT or PB	Post Build
	DsiCsIdentifier	PC, LT or PB	Post Build
	TransmitDataSource	PC, LT or PB	Post Build
	ChangeInDataTransfer	PC, LT or PB	Post Build
	SecondaryFrameSize	PC, LT or PB	Post Build
	SecondaryDsiCsIdentifier	PC, LT or PB	Post Build
SpiSequence			
	SpiSequenceId	Pre-Compile all Variants	Pre Compile
	SpiInterruptibleSequence	PC, LT or PB	Post Build
	SpiSeqEndNotification	PC, LT or PB	Post Build
	JobAssignment	PC, LT or PB	Post Build
SpiGeneral			
	SpiCancelApi	Pre-Compile all Variants	Pre Compile
	SpiChannelBuffersAllowed	Pre-Compile all Variants	Pre Compile
	SpiDevErrorDetect	Pre-Compile all Variants	Pre Compile
	SpiHwStatusApi	Pre-Compile all Variants	Pre Compile
	SpiInterruptibleSeqAllowed	Pre-Compile all Variants	Pre Compile
	SpiLevelDelivered	Pre-Compile all Variants	Pre Compile
	SpiVersionInfoApi	Pre-Compile all Variants	Pre Compile
	SpiClockReference	Vendor specific	Pre Compile (4)
	SpiGlobalDmaEnable	Vendor specific	Pre Compile
	SpiSyncTransmitTimeout	Vendor specific	Pre Compile
	SpiOptimizeOneJobSequences	Vendor specific	Pre Compile
	SpiOptimizedSeqNumber	Vendor specific	Pre Compile

Table continues on the next page...

Table 8-1. Configuration Parameters (continued)

	SpiOptimizedChannelsNumber	Vendor specific	Pre Compile
SpiNonAUTOSAR			
	SpiEnableMultiSyncTransmit	Vendor specific	Pre Compile
	SpiEnableHWUnitAsyncMode	Vendor specific	Pre Compile
	SpiEnableDualClockMode	Vendor specific	Pre Compile
	SpiAlternateClockRef	Vendor specific	Pre Compile
	SpiJobStartNotificationenable	Vendor specific	Pre Compile
	SpiTSBModeSupport	Vendor Specific	Pre Compile
	SpiForceDataType	Vendor Specific	Pre Compile
SpiPhyUnit			
	SpiPhyUnitMapping	Vendor specific	Pre Compile
	SpiPhyUnitSync	Vendor specific	Post Build
	SpiPhyUnitAsyncMethod	Vendor specific	Post Build
	SpiPhyTxDmaChannel	Vendor specific	Post Build
	SpiPhyTxDmaChannelAux	Vendor specific	Post Build
	SpiPhyRxDmaChannel	Vendor specific	Post Build

1. Adding or removing Channels, Jobs or Sequences typically requires updating the application, rendering those parameters useless as PB option.
2. Changing the buffer type of a channel requires updating the application, rendering this parameter useless as PB option.
3. Changing the size for internal buffers post build requires a “PostBuild RAM” concept.
4. Please note that this is the peripheral clock frequency supplied to the DSPI.

Chapter 9
Integration Steps

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

1. Generate the required SPI configurations. For more details refer to the section "Setting up the Plug-ins"
2. Allocate proper memory sections in MemMap.h and linker command file. For more details refer to the section "Memory Allocation"
3. Make sure all include files for compilation are as per the section "Files required for Compilation"
4. Map the ISRs to their vector locations. For more details refer to the section "ISR to configure within OS – dependencies"
5. Compile & build the SPI with all the dependent modules. For more details refer to the sections "Building the Driver" & "ISR to configure within OS – dependencies"

   Note:MCU shall be initialized with desired global Pre-scalar and system frequency before initializing the SPIdriver. PORT shall be initialized with desired signal settings for DSPI.

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™