MCAL User Manual for Bfx

32-bit TriCore™ AURIX™ TC3xx microcontroller

About this document

Scope and purpose

This User Manual is intended to enable users to integrate the Microcontroller Abstraction Layer (MCAL) software for the TriCore™ AURIX™ family of 32-bit microcontrollers.

This document describes responsibilities of integrator in-charge of integrating MCAL software with the basic software (BSW) stack. This document also provides detailed information on safety, configuration and functions along with examples of usage of significant features.

Note: *Detailed information about package installation, safety and other generic information that are common across all modules are provided in MCAL User Manual General.*

Intended audience

This document is intended for anyone using the Bfx module of the TC3xx MCAL software.

Document conventions

Table 1 Conventions

Convention	Explanation
Bold	Emphasizes heading levels, column headings, table and figure captions, screen names, windows, dialog boxes, menus, sub-menus
Italics	Denotes variable(s) and reference(s)
`Courier New`	Denotes APIs, functions, interrupt handlers, events, data types, error handlers, file/folder names, directories, command line inputs, code snippets
>	Indicates that a cascading sub-menu opens when you select a menu item
[cover parentID=<alpha numeric value>]	Used for traceability completeness. Reader should ignore these.

Reference documents

This User Manual should be read in conjunction with the following documents:

- AURIX™ TC3xx MCAL User Manual General
- Specification of BFX Driver, AUTOSAR_SWS_BFX_Driver, AUTOSAR Release 4.2.2
- Specification of BFX Driver, AUTOSAR_SWS_BFX_Driver, AUTOSAR Release 4.4.0

Table of contents

Table of contents

Table of contents

1 Bfx driver

1.1 User information

1.1.1 Description

The BFX library provides bit handling functionality for fixed-point data specified by AUTOSAR. The library provides support for 8-bit, 16-bit, 32-bit and 64-bit data. The library provides all its functionality, independent of any underlying hardware IP.

1.1.2 Hardware-software mapping

This section is not applicable for BFX library as it does not have any associated hardware IP.

1.1.3 File structure

1.1.3.1 C file structure

This section provides details of the C files of the BFX library.

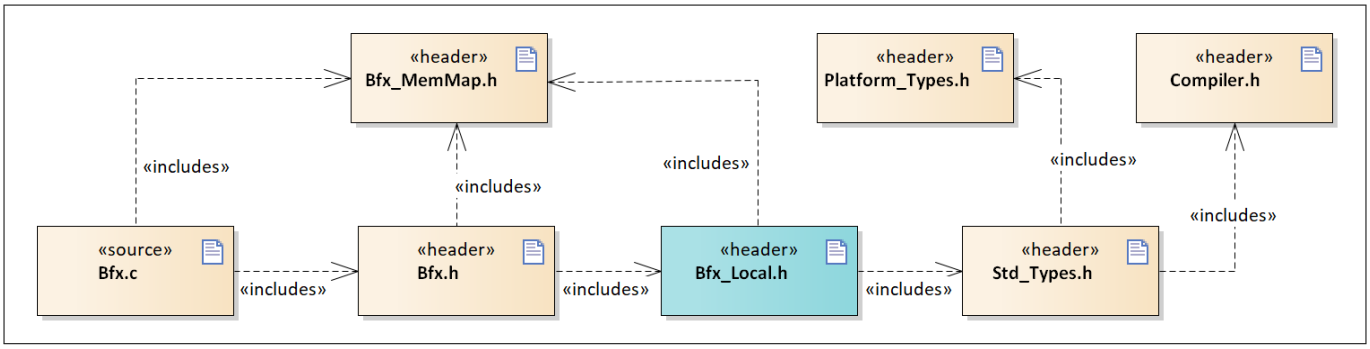

Figure 1 **Bfx_C_File_Structure-1.png**

Table 2 **C file structure**

File name	Description
`Bfx.c`	File (Static) containing the Bfx_GetVersionInfo API definition.
`Bfx.h`	Header file (Static) contains inline implementation of all functions of the BFX library exposed to the upper layer.
`Bfx_Local.h`	Header file (Static) contains the inline local function definitions of BFX library.
`Bfx_MemMap.h`	File (Static) containing the memory section definitions used by the BFX library
`Compiler.h`	Provides abstraction from compiler-specific keywords
`Platform_Types.h`	Platform-specific type declaration file as defined by AUTOSAR
`Std_Types.h`	Standard type declaration file as defined by AUTOSAR. It is independent of compiler or platform.

1.1.3.2 Code generator plugin files

This section provides details of the code generator plugin files of the BFX library.

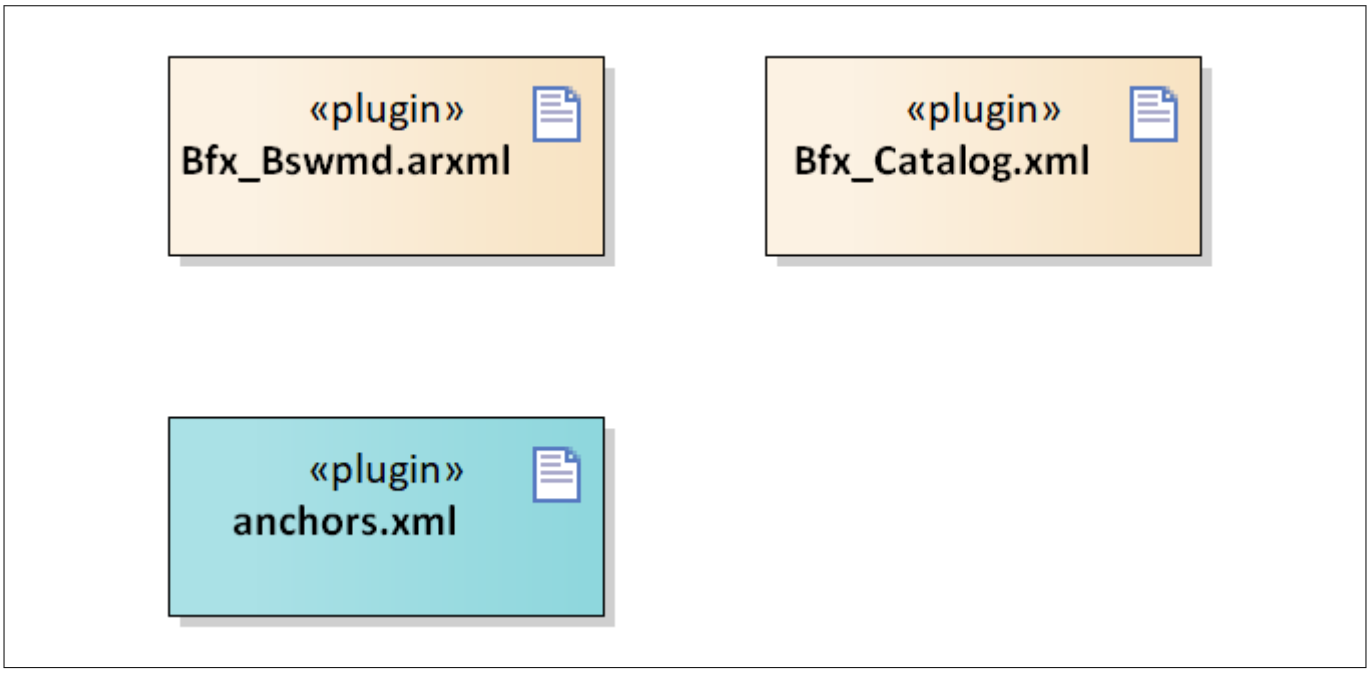

Figure 2 **Bfx_Code_Generator_Plugin_Files-1.png**

Table 3 **Code generator plugin files**

File name	Description
`Bfx_Bswmd.arxml`	AUTOSAR format module description file for the BFX library
`Bfx_Catalog.xml`	AUTOSAR format catalog file as per catalog_V3_0_0.ml.xsd for the BFX library
`anchors.xml`	Tresos anchors support file for the BFX library

1.1.4 Integration hints

This section lists the key points that an integrator or user of the BFX library must consider.

The BFX library does not require initialization phase to provide its intended functionality as the library does not have any associated hardware IP, SFRs or global variables, which need to be initialized. Shutdown phase is also not required for the BFX library as there is no initialization phase. The BFX library does not depend on any other module for its functionality.

The APIs of the BFX library may be invoked from several CPU cores simultaneously. However, the access to the shared resources, passed as API parameters, must be serialized.

1.1.4.1 Integration with AUTOSAR stack

This section lists the modules, which are not part of the MCAL, but are required to integrate the BFX library.

- **EcuM**

The EcuM module is not required for integrating the BFX library.

- **Memory mapping**

Memory mapping is a concept from AUTOSAR that allows relocation of text, variables, constants and configuration data to user-specific memory regions. To achieve this, all the relocatable elements of the driver are encapsulated in different memory-section macros. These macros are defined in the `Bfx_MemMap.h` file.

The `Bfx_MemMap.h` file is provided in the MCAL package as a stub code. The integrator must place appropriate compiler pragmas within the memory-section macros. The pragmas ensure that the elements are relocated to the correct memory region. A sample implementation listing the memory-section macros is shown as follows:

```
/* Code Section */
/*
To be used for mapping code to application block, boot block, external flash
etc.
{codePeriod} is the typical period time value and unit of the
ExecutableEntitys in this MemorySection.
The name part _{codePeriod} is optional.
Units are:
- US microsecond
- MS millisecond
- S second
For example 100US, 400US, 1MS, 5MS, 10MS, 20MS, 100MS, 1S
*/
#if defined BFX_START_SEC_CODE_ASIL_B_GLOBAL
/* User Pragma for PF[x] */
#undef BFX_START_SEC_CODE_ASIL_B_GLOBAL
#undef MEMMAP_ERROR
#elif defined BFX_STOP_SEC_CODE_ASIL_B_GLOBAL
/* User Pragma for PF[x] */
#undef BFX_STOP_SEC_CODE_ASIL_B_GLOBAL
#undef MEMMAP_ERROR
#endif
#if defined MEMMAP_ERROR
#error "BFX_MemMap.h, wrong pragma command"
#endif
```

- **DET**

The DET module is not required for integrating the BFX library.

- **DEM**

The DEM module is not required for integrating the BFX library.

- **SchM**

The SchM is not required for integrating the BFX library.

- **Safety error**

The BFX library does not report any safety errors.

- **Notification and callbacks**

The BFX library does not provide any notifications or callbacks.

- **OS**

The OS is not required for integrating the BFX library.

1.1.4.2 Multicore and Resource Manager

The BFX library supports execution of its APIs simultaneously from all CPU cores as long as the access to the shared resources, passed as parameters to the BFX APIs, is serialized. The following are the key points to be considered with respect to multicore in the BFX library:

- The BFX library does not access any SFRs or any shared resource, except in case where a shared resource is passed as parameter to a BFX API. AoU is provided to the user to serialize the access to such shared resources, which are passed as parameters to the BFX APIs.
- Locating text to correct memory space should be done by the user. All memory sections for BFX library are marked GLOBAL (common to all cores). The following should be considered by the user to ensure better performance of the driver:

  Code section:

  The executable code of the BFX library is placed under single MemMap section. This MemMap section can be relocated to any PFlash region. It is possible to access the code from all CPU cores.

  Data section:

  The BFX library does not define any RAM variables. Hence, data section is not required for the BFX library.

  Configuration data and constants:

  The BFX library does not define any configuration data or constants. Hence, configuration data section is not required for the BFX library.

Note: Relocating of code to a distant memory space would impact execution timings.

Note: If the driver operates from single core, the code section may be relocated to the PFlash/DSPR of the same CPU core.

1.1.4.3 MCU support

The BFX library does not use any services provided by the MCU driver.

1.1.4.4 Port support

The BFX library does not use any services provided by the PORT driver.

1.1.4.5 DMA support

The BFX library does not use any services provided by the DMA driver.

1.1.4.6 Interrupt connections

The BFX library does not use any interrupt source.

1.1.4.7 Example usage

The BFX is a library module. All the BFX APIs can be called independently of each other; therefore, there is no example usage for the BFX library.

1.1.5 Key architectural considerations

1.2 Assumptions of Use (AoU)

The AoU for Bfx driver are as follows.

- **Proper memory alignment and valid pointer as parameter**

User shall ensure that a valid pointer parameter is passed to the BFX APIs and address to be written must adhere with the memory alignment as per HW UM because the driver does not have any mechanism to validate the pointer parameter and report error.

[cover parentID BFX={360E5AAD-9083-4e2d-BD5B-10DA92664475}]

- **Serialized access to shared resource**

The user shall implement an appropriate mechanism to serialize the access to the shared resources, which are passed as parameter to the BFX APIs by using SchM functions or spinlocks.

[cover parentID BFX={5A3CAADA-6377-43e1-8AE5-C5F043BC9CE6}]

- **Valid permission level**

The user shall not pass any SFR, for which the user application does not have appropriate access rights, as a parameter of any BFX API.

[cover parentID BFX={CBA42528-D0E6-400c-92F5-F1F8DE36D4A1}]

- **Parameter range check**

The user shall ensure all parameters are within the specified valid range as the input range checks are not performed by the BFX APIs.

[cover parentID BFX={FCE75C8A-8252-4996-87B2-E66CE25EEEC7}]

1.3 Reference information

1.3.1 Configuration interfaces

The BFX library does not support any configuration options that may affect the functional behavior of the routines.

1.3.2 Functions - Type definitions

The BFX library does not define any data types.

1.3.3 Functions - APIs

This section lists all the APIs of the BFX library.

1.3.3.1 Bfx_SetBit_u8u8

Table 4 Specification for `Bfx_SetBit_u8u8` API

Syntax	`void Bfx_SetBit_u8u8 ( uint8 * const Data, const uint8 BitPn )`	
Service ID	0x01	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	BitPn	Bit position (Valid range: 0 to 7)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_SetBit_u8u8 function sets the logical status of the bit at BitPn bit position of the Data parameter to 1.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying a bit of the 8-bit Data parameter. Hence, the valid range for the BitPn parameter, which indicates the position of the bit to be modified, is 0 to 7.	
SFR accessed	None	

Table 4 Specification for `Bfx_SetBit_u8u8` API (continued)

Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.2 Bfx_SetBit_u16u8

Table 5 Specification for `Bfx_SetBit_u16u8` API

Syntax	`void Bfx_SetBit_u16u8 ( uint16 * const Data, const uint8 BitPn )`	
Service ID	0x02	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	BitPn	Bit position (Valid range: 0 to 15)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_SetBit_u16u8 function sets the logical status of the bit at BitPn bit position of the Data parameter to 1.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying a bit of the 16-bit Data parameter. Hence, the valid range for the BitPn parameter, which indicates the position of the bit to be modified, is 0 to 15.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.3 Bfx_SetBit_u32u8

Table 6 Specification for `Bfx_SetBit_u32u8` API

Syntax	`void Bfx_SetBit_u32u8 ( uint32 * const Data,`

Table 6 Specification for `Bfx_SetBit_u32u8` API (continued)

	`const uint8 BitPn` `)`	
Service ID	0x03	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	BitPn	Bit position (Valid range: 0 to 31)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_SetBit_u32u8 function sets the logical status of the bit at BitPn bit position of the Data parameter to 1.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying a bit of the 32-bit Data parameter. Hence, the valid range for the BitPn parameter, which indicates the position of the bit to be modified, is 0 to 31.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.4 Bfx_SetBit_u64u8

Table 7 Specification for `Bfx_SetBit_u64u8` API

Syntax	`void Bfx_SetBit_u64u8`
	`(`
	`uint64 * const Data,`
	`const uint8 BitPn`
	`)`	

Service ID	0x04	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	BitPn	Bit position (Valid range: 0 to 63)

Table 7 Specification for `Bfx_SetBit_u64u8` API (continued)

Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_SetBit_u64u8 function sets the logical status of the bit at BitPn bit position of the Data parameter to 1.	
Source	IFX for AS4.2.2 variant and AUTOSAR for AS4.4.0 variant	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying a bit of the 64-bit Data parameter. Hence, the valid range for the BitPn parameter, which indicates the position of the bit to be modified, is 0 to 63.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.5 Bfx_ClrBit_u8u8

Table 8 Specification for `Bfx_ClrBit_u8u8` API

Syntax	`void Bfx_ClrBit_u8u8 ( uint8 * const Data, const uint8 BitPn )`	
Service ID	0x06	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	BitPn	Bit position (Valid range: 0 to 7)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_ClrBit_u8u8 function clears the logical status of the bit at BitPn bit position of the Data parameter to 0.	
Source	AUTOSAR	

Table 8 Specification for `Bfx_ClrBit_u8u8` API (continued)

Error handling	-
Configuration dependencies	-
User hints	The API is used for modifying a bit of the 8-bit Data parameter. Hence, the valid range for the BitPn parameter, which indicates the position of the bit to be modified, is 0 to 7.
SFR accessed	None
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.6 Bfx_ClrBit_u16u8

Table 9 Specification for `Bfx_ClrBit_u16u8` API

Syntax	`void Bfx_ClrBit_u16u8 ( uint16 * const Data, const uint8 BitPn )`	
Service ID	0x07	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	BitPn	Bit position (Valid range: 0 to 15)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_ClrBit_u16u8 function clears the logical status of the bit at BitPn bit position of the Data parameter to 0.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying a bit of the 16-bit Data parameter. Hence, the valid range for the BitPn parameter, which indicates the position of the bit to be modified, is 0 to 15.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.7 Bfx_ClrBit_u32u8

Table 10 Specification for `Bfx_ClrBit_u32u8` API

Syntax	`void Bfx_ClrBit_u32u8 ( uint32 * const Data, const uint8 BitPn )`	
Service ID	0x08	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	BitPn	Bit position (Valid range: 0 to 31)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_ClrBit_u32u8 function clears the logical status of the bit at BitPn bit position of the Data parameter to 0.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying a bit of the 32-bit Data parameter. Hence, the valid range for the BitPn parameter, which indicates the position of the bit to be modified, is 0 to 31.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.8 Bfx_ClrBit_u64u8

Table 11 Specification for `Bfx_ClrBit_u64u8` API

Syntax	`void Bfx_ClrBit_u64u8 ( uint64 * const Data, const uint8 BitPn )`
Service ID	0x09
Sync/Async	Synchronous
ASIL Level	B

Table 11 Specification for `Bfx_ClrBit_u64u8` API (continued)

Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	BitPn	Bit position (Valid Range: 0 to 63)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_ClrBit_u64u8 function clears the logical status of the bit at BitPn bit position of the Data parameter to 0.	
Source	IFX for AS4.2.2 variant and AUTOSAR for AS4.4.0 variant	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying a bit of the 64-bit Data parameter. Hence, the valid range for the BitPn parameter, which indicates the position of the bit to be modified, is 0 to 63.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.9 Bfx_GetBit_u8u8_u8

Table 12 Specification for `Bfx_GetBit_u8u8_u8` API

Syntax	`boolean Bfx_GetBit_u8u8_u8`
	`(`
	`  const uint8 Data,`
	`  const uint8 BitPn`
	`)`	

Service ID	0x0a	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	Data BitPn	Input data Bit position (Valid range: 0 to 7)
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	boolean	Status as per the extracted bit

Table 12 Specification for `Bfx_GetBit_u8u8_u8` API (continued)

		TRUE : Extracted bit is 1 FALSE : Extracted bit is 0
Description	The Bfx_GetBit_u8u8_u8 function returns TRUE when the logical status of the bit at BitPn bit position of the Data input parameter is 1, otherwise the function returns FALSE.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for extracting a bit from the 8-bit Data parameter. Hence, the valid range for the BitPn parameter, which indicates the position of the bit to be extracted, is 0 to 7.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.10 Bfx_GetBit_u16u8_u8

Table 13 Specification for `Bfx_GetBit_u16u8_u8` API

Syntax	`boolean Bfx_GetBit_u16u8_u8`
	`(`
	`  const uint16 Data,`
	`  const uint8 BitPn`
	`)`	

Service ID	0x0b	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	Data BitPn	Input data Bit position (Valid range: 0 to 15)
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	boolean	Status as per the extracted bit TRUE : Extracted bit is 1 FALSE : Extracted bit is 0
Description	The Bfx_GetBit_u16u8_u8 function returns TRUE when the logical status of the bit at BitPn bit position of the Data input parameter is 1, otherwise the function returns FALSE.	
Source	AUTOSAR	
Error handling	-	

Table 13 Specification for `Bfx_GetBit_u16u8_u8` API (continued)

Configuration dependencies	-
User hints	The API is used for extracting a bit from the 16-bit Data parameter. Hence, the valid range for the BitPn parameter, which indicates the position of the bit to be extracted, is 0 to 15.
SFR accessed	None
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.11 Bfx_GetBit_u32u8_u8

Table 14 Specification for `Bfx_GetBit_u32u8_u8` API

Syntax	`boolean Bfx_GetBit_u32u8_u8 ( const uint32 Data, const uint8 BitPn )`	
Service ID	0x0c	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	Data BitPn	Input data Bit position (Valid range: 0 to 31)
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	boolean	Status as per the extracted bit TRUE : Extracted bit is 1 FALSE : Extracted bit is 0
Description	The Bfx_GetBit_u32u8_u8 function returns TRUE when the logical status of the bit at BitPn bit position of the Data input parameter is 1, otherwise the function returns FALSE.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for extracting a bit of the 32-bit Data parameter. Hence, the valid range for the BitPn parameter, which indicates the position of the bit to be extracted, is 0 to 31.	
SFR accessed	None	

Table 14	Specification for `Bfx_GetBit_u32u8_u8` API (continued)
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.12 Bfx_GetBit_u64u8_u8

Table 15	Specification for `Bfx_GetBit_u64u8_u8` API	
Syntax	`boolean Bfx_GetBit_u64u8_u8 ( const uint64 Data, const uint8 BitPn )`	
Service ID	0x0d	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	Data BitPn	Input data Bit position (Valid range: 0 to 63)
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	boolean	Status as per the extracted bit TRUE : Extracted bit is 1 FALSE : Extracted bit is 0
Description	The Bfx_GetBit_u64u8_u8 function returns TRUE when the logical status of the bit at BitPn bit position of the Data input parameter is 1, otherwise the function returns FALSE.	
Source	IFX for AS4.2.2 variant and AUTOSAR for AS4.4.0 variant	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for extracting a bit of the 64-bit Data parameter. Hence, the valid range for the BitPn parameter, which indicates the position of the bit to be extracted, is 0 to 63.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.13 Bfx_SetBits_u8u8u8u8

Table 16 Specification for `Bfx_SetBits_u8u8u8u8` API

Syntax	`void Bfx_SetBits_u8u8u8u8 ( uint8 * const Data, const uint8 BitStartPn, const uint8 BitLn, const uint8 Status )`	
Service ID	0x20	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	BitStartPn BitLn Status	Start bit position (Valid range: 0 to 7) Bit field length (Valid range: 1 to (8 - BitStartPn)) Status value to be set
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_SetBits_u8u8u8u8 function clears the logical status of the bits of the Data parameter starting from BitStartPn bit position for BitLn number of bits to 0 when the value of Status parameter is zero. Otherwise, for non-zero value of Status parameter, the function sets the logical status of the bits of the Data parameter starting from BitStartPn bit position for BitLn number of bits to 1.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying some bits of the 8-bit Data parameter. Hence, the valid ranges for input parameters are as follows: 1. The valid range for the BitStartPn parameter, which indicates the start position of the bits to be modified, is 0 to 7. 2. The valid range for the BitLn parameter, which indicates the number of bits to be modified, is 1 to (8 - BitStartPn).	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.14 Bfx_SetBits_u16u8u8u8

Table 17 Specification for `Bfx_SetBits_u16u8u8u8` API

Syntax	`void Bfx_SetBits_u16u8u8u8 ( uint16 * const Data, const uint8 BitStartPn, const uint8 BitLn, const uint8 Status )`	
Service ID	0x21	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	BitStartPn BitLn Status	Start bit position (Valid range: 0 to 15) Bit field length (Valid range: 1 to (16 - BitStartPn)) Status value to be set
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_SetBits_u16u8u8u8 function clears the logical status of the bits of the Data parameter starting from BitStartPn bit position for BitLn number of bits to 0 when the value of Status parameter is zero. Otherwise, for non-zero value of Status parameter, the function sets the logical status of the bits of the Data parameter starting from BitStartPn bit position for BitLn number of bits to 1.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying some bits of the 16-bit Data parameter. Hence, the valid ranges for input parameters are as follows: 1. The valid range for the BitStartPn parameter, which indicates the start position of the bits to be modified, is 0 to 15. 2. The valid range for the BitLn parameter, which indicates the number of bits to be modified, is 1 to (16 - BitStartPn).	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.15 Bfx_SetBits_u32u8u8u8

Table 18 Specification for `Bfx_SetBits_u32u8u8u8` API

Syntax	`void Bfx_SetBits_u32u8u8u8 ( uint32 * const Data, const uint8 BitStartPn, const uint8 BitLn, const uint8 Status )`	
Service ID	0x22	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	BitStartPn BitLn Status	Start bit position (Valid range: 0 to 31) Bit field length (Valid range: 1 to (32 - BitStartPn)) Status value to be set
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_SetBits_u32u8u8u8 function clears the logical status of the bits of the Data parameter starting from BitStartPn bit position for BitLn number of bits to 0 when the value of Status parameter is zero. Otherwise, for non-zero value of Status parameter, the function sets the logical status of the bits of the Data parameter starting from BitStartPn bit position for BitLn number of bits to 1.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying some bits of the 32-bit Data parameter. Hence, the valid ranges for input parameters are as follows: 1. The valid range for the BitStartPn parameter, which indicates the start position of the bits to be modified, is 0 to 31. 2. The valid range for the BitLn parameter, which indicates the number of bits to be modified, is 1 to (32 - BitStartPn).	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.16 Bfx_SetBits_u64u8u8u8

Table 19 Specification for `Bfx_SetBits_u64u8u8u8` API

Syntax	`void Bfx_SetBits_u64u8u8u8 ( uint64 * const Data, const uint8 BitStartPn, const uint8 BitLn, const uint8 Status )`	
Service ID	0x23	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	BitStartPn BitLn Status	Start bit position (Valid range: 0 to 63) Bit field length (Valid range: 1 to (64 - BitStartPn)) Status value to be set
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_SetBits_u64u8u8u8 function clears the logical status of the bits of the Data parameter starting from BitStartPn bit position for BitLn number of bits to 0 when the value of Status parameter is zero. Otherwise, for non-zero value of Status parameter, the function sets the logical status of the bits of the Data parameter starting from BitStartPn bit position for BitLn number of bits to 1.	
Source	IFX for AS4.2.2 variant and AUTOSAR for AS4.4.0 variant	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying some bits of the 64-bit Data parameter. Hence, the valid ranges for input parameters are as follows: 1. The valid range for the BitStartPn parameter, which indicates the start position of the bits to be modified, is 0 to 63. 2. The valid range for the BitLn parameter, which indicates the number of bits to be modified, is 1 to (64 - BitStartPn).	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.17 Bfx_GetBits_u8u8u8_u8

Table 20 Specification for `Bfx_GetBits_u8u8u8_u8` API

Syntax	`uint8 Bfx_GetBits_u8u8u8_u8 ( const uint8 Data, const uint8 BitStartPn, const uint8 BitLn )`	
Service ID	0x26	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	Data BitStartPn BitLn	Input data Start bit position (Valid range: 0 to 7) Bit field length (Valid range: 1 to (8 - BitStartPn))
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	uint8	Bits extracted from the input parameter
Description	The Bfx_GetBits_u8u8u8_u8 function returns the bits of the Data input parameter starting from BitStartPn bit position for BitLn number of bits.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for extracting some bits of the 8-bit Data parameter. Hence, the valid ranges for input parameters are as follows: 1. The valid range for the BitStartPn parameter, which indicates the start position of the bits to be extracted, is 0 to 7. 2. The valid range for the BitLn parameter, which indicates the number of bits to be extracted, is 1 to (8 - BitStartPn).	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.18 Bfx_GetBits_u16u8u8_u16

Table 21 Specification for `Bfx_GetBits_u16u8u8_u16` API

Syntax	`uint16 Bfx_GetBits_u16u8u8_u16 (`

Table 21 Specification for `Bfx_GetBits_u16u8u8_u16` API (continued)

	`const uint16 Data,`
	`const uint8 BitStartPn,`
	`const uint8 BitLn`
	`)`	

Service ID	0x27	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	Data BitStartPn BitLn	Input data Start bit position (Valid range: 0 to 15) Bit field length (Valid range: 1 to (16 - BitStartPn))
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	uint16	Bits extracted from the input parameter
Description	The Bfx_GetBits_u16u8u8_u16 function returns the bits of the Data input parameter starting from BitStartPn bit position for BitLn number of bits.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for extracting some bits of the 16-bit Data parameter. Hence, the valid ranges for input parameters are as follows: 1. The valid range for the BitStartPn parameter, which indicates the start position of the bits to be extracted, is 0 to 15. 2. The valid range for the BitLn parameter, which indicates the number of bits to be extracted, is 1 to (16 - BitStartPn).	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.19 Bfx_GetBits_u32u8u8_u32

Table 22 Specification for `Bfx_GetBits_u32u8u8_u32` API

Syntax	`uint32 Bfx_GetBits_u32u8u8_u32`
	`(`
	`const uint32 Data,`
	`const uint8 BitStartPn,`
	`const uint8 BitLn`
	`)`

Table 22 Specification for `Bfx_GetBits_u32u8u8_u32` API (continued)

Service ID	0x28	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	Data BitStartPn BitLn	Input data Start bit position (Valid range: 0 to 31) Bit field length (Valid range: 1 to (32 - BitStartPn))
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	uint32	Bits extracted from the input parameter
Description	The Bfx_GetBits_u32u8u8_u32 function returns the bits of the Data input parameter starting from BitStartPn bit position for BitLn number of bits.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying some bits of the 32-bit Data parameter. Hence, the valid ranges for input parameters are as follows: 1. The valid range for the BitStartPn parameter, which indicates the start position of the bits to be extracted, is 0 to 31. 2. The valid range for the BitLn parameter, which indicates the number of bits to be extracted, is 1 to (32 - BitStartPn).	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.20 Bfx_GetBits_u64u8u8_u64

Table 23 Specification for `Bfx_GetBits_u64u8u8_u64` API

Syntax	`uint64 Bfx_GetBits_u64u8u8_u64`
	`(`
	`  const uint64 Data,`
	`  const uint8 BitStartPn,`
	`  const uint8 BitLn`
	`)`

Service ID	0x29
Sync/Async	Synchronous
ASIL Level	B

Table 23 Specification for `Bfx_GetBits_u64u8u8_u64` API (continued)

Re-entrancy	Reentrant	
Parameters (in)	Data BitStartPn BitLn	Input data Start bit position (Valid range: 0 to 63) Bit field length (Valid range: 1 to (64 - BitStartPn))
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	uint64	Bits extracted from the input parameter
Description	The Bfx_GetBits_u64u8u8_u64 function returns the bits of the Data input parameter starting from BitStartPn bit position for BitLn number of bits.	
Source	IFX for AS4.2.2 variant and AUTOSAR for AS4.4.0 variant	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying some bits of the 64-bit Data parameter. Hence, the valid ranges for input parameters are as follows: 1. The valid range for the BitStartPn parameter, which indicates the start position of the bits to be extracted, is 0 to 63. 2. The valid range for the BitLn parameter, which indicates the number of bits to be extracted, is 1 to (64 - BitStartPn).	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.21 Bfx_SetBitMask_u8u8

Table 24 Specification for `Bfx_SetBitMask_u8u8` API

Syntax	`void  Bfx_SetBitMask_u8u8`
	`(`
	`  uint8 * const Data,`
	`  const uint8 Mask`
	`)`	

Service ID	0x2a	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	Mask	Mask value

Table 24 Specification for `Bfx_SetBitMask_u8u8` API (continued)

Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_SetBitMask_u8u8 function sets the logical status of the bits of the Data parameter to 1, for all the bit positions for which the logical status of bit in the Mask parameter is set to 1. The remaining bits of the Data parameter will retain their original values.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.22 Bfx_SetBitMask_u16u16

Table 25 Specification for `Bfx_SetBitMask_u16u16` API

Syntax	`void Bfx_SetBitMask_u16u16 ( uint16 * const Data, const uint16 Mask )`	
Service ID	0x2b	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	Mask	Mask value
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_SetBitMask_u16u16 function sets the logical status of the bits of the Data parameter to 1, for all the bit positions for which the logical status of bit in the Mask parameter is set to 1. The remaining bits of the Data parameter will retain their original values.	
Source	AUTOSAR	

Table 25 Specification for `Bfx_SetBitMask_u16u16` API (continued)

Error handling	-
Configuration dependencies	-
User hints	None
SFR accessed	None
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.23 Bfx_SetBitMask_u32u32

Table 26 Specification for `Bfx_SetBitMask_u32u32` API

Syntax	`void Bfx_SetBitMask_u32u32 ( uint32 * const Data, const uint32 Mask )`	
Service ID	0x2c	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	Mask	Mask value
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_SetBitMask_u32u32 function sets the logical status of the bits of the Data parameter to 1, for all the bit positions for which the logical status of bit in the Mask parameter is set to 1. The remaining bits of the Data parameter will retain their original values.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.24 Bfx_SetBitMask_u64u64

Table 27 Specification for `Bfx_SetBitMask_u64u64` API

Syntax	`void Bfx_SetBitMask_u64u64 ( uint64 * const Data, const uint64 Mask )`	
Service ID	0x2d	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	Mask	Mask value
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_SetBitMask_u64u64 function sets the logical status of the bits of the Data parameter to 1, for all the bit positions for which the logical status of bit in the Mask parameter is set to 1. The remaining bits of the Data parameter will retain their original values.	
Source	IFX for AS4.2.2 variant and AUTOSAR for AS4.4.0 variant	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.25 Bfx_ClrBitMask_u8u8

Table 28 Specification for `Bfx_ClrBitMask_u8u8` API

Syntax	`void Bfx_ClrBitMask_u8u8 ( uint8 * const Data, const uint8 Mask )`
Service ID	0x30
Sync/Async	Synchronous
ASIL Level	B

Table 28 Specification for `Bfx_ClrBitMask_u8u8` API (continued)

Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	Mask	Mask value
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_ClrBitMask_u8u8 function clears the logical status of the bits of the Data parameter to 0, for all the bit positions for which the logical status of bit in the Mask parameter is set to 1. The remaining bits of Data parameter will retain their original values.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.26 Bfx_ClrBitMask_u16u16

Table 29 Specification for `Bfx_ClrBitMask_u16u16` API

Syntax	`void Bfx_ClrBitMask_u16u16 ( uint16 * const Data, const uint16 Mask )`	
Service ID	0x31	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	Mask	Mask value
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-

Table 29 Specification for `Bfx_ClrBitMask_u16u16` API (continued)

Description	The Bfx_ClrBitMask_u16u16 function clears the logical status of the bits of the Data parameter to 0, for all the bit positions for which the logical status of bit in the Mask parameter is set to 1. The remaining bits of Data parameter will retain their original values.
Source	AUTOSAR
Error handling	-
Configuration dependencies	-
User hints	None
SFR accessed	None
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.27 Bfx_ClrBitMask_u32u32

Table 30 Specification for `Bfx_ClrBitMask_u32u32` API

Syntax	`void Bfx_ClrBitMask_u32u32 ( uint32 * const Data, const uint32 Mask )`	
Service ID	0x32	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	Mask	Mask value
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_ClrBitMask_u32u32 function clears the logical status of the bits of the Data parameter to 0, for all the bit positions for which the logical status of bit in the Mask parameter is set to 1. The remaining bits of Data parameter will retain their original values.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	None	

Table 30 Specification for `Bfx_ClrBitMask_u32u32` API (continued)

SFR accessed	None
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.28 Bfx_ClrBitMask_u64u64

Table 31 Specification for `Bfx_ClrBitMask_u64u64` API

Syntax	`void Bfx_ClrBitMask_u64u64 ( uint64 * const Data, const uint64 Mask )`	
Service ID	0x33	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	Mask	Mask value
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_ClrBitMask_u64u64 function clears the logical status of the bits of the Data parameter to 0, for all the bit positions for which the logical status of bit in the Mask parameter is set to 1. The remaining bits of Data parameter will retain their original values.	
Source	IFX for AS4.2.2 variant and AUTOSAR for AS4.4.0 variant	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.29 Bfx_TstBitMask_u8u8_u8

Table 32 Specification for `Bfx_TstBitMask_u8u8_u8` API

Syntax	`boolean Bfx_TstBitMask_u8u8_u8 ( const uint8 Data, const uint8 Mask )`	
Service ID	0x36	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	Data Mask	Input data Mask value
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	boolean	Test result TRUE : All bits defined in mask are set in input parameter FALSE: At least one bit defined in mask is not set in input parameter
Description	The Bfx_TstBitMask_u8u8_u8 function returns TRUE when the logical status of all the bits defined in the Mask parameter are also set at the same bit position in the Data input parameter, otherwise the function returns FALSE.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.30 Bfx_TstBitMask_u16u16_u8

Table 33 Specification for `Bfx_TstBitMask_u16u16_u8` API

Syntax	`boolean Bfx_TstBitMask_u16u16_u8 ( const uint16 Data, const uint16 Mask )`

Table 33 Specification for `Bfx_TstBitMask_u16u16_u8` API (continued)

Service ID	0x37	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	Data Mask	Input data Mask value
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	boolean	Test result TRUE : All bits defined in mask are set in input parameter FALSE: At least one bit defined in mask is not set in input parameter
Description	The Bfx_TstBitMask_u16u16_u8 function returns TRUE when the logical status of all the bits defined in the Mask parameter are also set at the same bit position in the Data input parameter, otherwise the function returns FALSE.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.31 Bfx_TstBitMask_u32u32_u8

Table 34 Specification for `Bfx_TstBitMask_u32u32_u8` API

Syntax	`boolean  Bfx_TstBitMask_u32u32_u8`
	`(`
	`  const uint32 Data,`
	`  const uint32 Mask`
	`)`

Service ID	0x38
Sync/Async	Synchronous
ASIL Level	B
Re-entrancy	Reentrant

Table 34 Specification for `Bfx_TstBitMask_u32u32_u8` API (continued)

Parameters (in)	Data Mask	Input data Mask value
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	boolean	Test result TRUE : All bits defined in mask are set in input parameter FALSE: At least one bit defined in mask is not set in input parameter
Description	The Bfx_TstBitMask_u32u32_u8 function returns TRUE when the logical status of all the bits defined in the Mask parameter are also set at the same bit position in the Data input parameter, otherwise the function returns FALSE.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.32 Bfx_TstBitMask_u64u64_u8

Table 35 Specification for `Bfx_TstBitMask_u64u64_u8` API

Syntax	`boolean Bfx_TstBitMask_u64u64_u8`
	`(`
	`  const uint64 Data,`
	`  const uint64 Mask`
	`)`	

Service ID	0x39	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	Data Mask	Input data Mask value
Parameters (out)	-	-
Parameters (in - out)	-	-

Table 35 Specification for `Bfx_TstBitMask_u64u64_u8` API (continued)

Return	boolean	Test result TRUE : All bits defined in mask are set in input parameter FALSE: At least one bit defined in mask is not set in input parameter
Description	The Bfx_TstBitMask_u64u64_u8 function returns TRUE when the logical status of all the bits defined in the Mask parameter are also set at the same bit position in the Data input parameter, otherwise the function returns FALSE.	
Source	IFX for AS4.2.2 variant and AUTOSAR for AS4.4.0 variant	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.33 Bfx_TstBitLnMask_u8u8_u8

Table 36 Specification for `Bfx_TstBitLnMask_u8u8_u8` API

Syntax	`boolean Bfx_TstBitLnMask_u8u8_u8`
	`(`
	`  const uint8 Data,`
	`  const uint8 Mask`
	`)`	

Service ID	0x3a	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	Data Mask	Input data Mask value
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	boolean	Test result TRUE : At least one bit defined in mask is set in input parameter FALSE: No bit defined in mask is set in input parameter
Description	The Bfx_TstBitLnMask_u8u8_u8 function returns TRUE when the logical status of at least one bit defined in the Mask parameter is also set at the same bit position in the Data input parameter, otherwise the function returns FALSE.	

Table 36 Specification for `Bfx_TstBitLnMask_u8u8_u8` API (continued)

Source	AUTOSAR
Error handling	-
Configuration dependencies	-
User hints	None
SFR accessed	None
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.34 Bfx_TstBitLnMask_u16u16_u8

Table 37 Specification for `Bfx_TstBitLnMask_u16u16_u8` API

Syntax	`boolean Bfx_TstBitLnMask_u16u16_u8 ( const uint16 Data, const uint16 Mask )`	
Service ID	0x3b	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	Data Mask	Input data Mask value
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	boolean	Test result TRUE : At least one bit defined in mask is set in input parameter FALSE: No bit defined in mask is set in input parameter
Description	The Bfx_TstBitLnMask_u16u16_u8 function returns TRUE when the logical status of at least one bit defined in the Mask parameter is also set at the same bit position in the Data input parameter, otherwise the function returns FALSE.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	

Table 37	Specification for `Bfx_TstBitLnMask_u16u16_u8` API (continued)
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.35 Bfx_TstBitLnMask_u32u32_u8

Table 38 Specification for `Bfx_TstBitLnMask_u32u32_u8` API

Syntax	`boolean Bfx_TstBitLnMask_u32u32_u8 ( const uint32 Data, const uint32 Mask )`	
Service ID	0x3c	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	Data Mask	Input data Mask value
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	boolean	Test result TRUE : At least one bit defined in mask is set in input parameter FALSE: No bit defined in mask is set in input parameter
Description	The Bfx_TstBitLnMask_u32u32_u8 function returns TRUE when the logical status of at least one bit defined in the Mask parameter is also set at the same bit position in the Data input parameter, otherwise the function returns FALSE.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.36 Bfx_TstBitLnMask_u64u64_u8

Table 39 Specification for `Bfx_TstBitLnMask_u64u64_u8` API

Syntax	`boolean Bfx_TstBitLnMask_u64u64_u8 ( const uint64 Data, const uint64 Mask )`	
Service ID	0x3d	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	Data Mask	Input data Mask value
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	boolean	Test result TRUE : At least one bit defined in mask is set in input parameter FALSE: No bit defined in mask is set in input parameter
Description	The Bfx_TstBitLnMask_u64u64_u8 function returns TRUE when the logical status of at least one bit defined in the Mask parameter is also set at the same bit position in the Data input parameter, otherwise the function returns FALSE.	
Source	IFX for AS4.2.2 variant and AUTOSAR for AS4.4.0 variant	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.37 Bfx_TstParityEven_u8_u8

Table 40 Specification for `Bfx_TstParityEven_u8_u8` API

Syntax	`boolean Bfx_TstParityEven_u8_u8 ( const uint8 Data )`
Service ID	0x40

Table 40 Specification for `Bfx_TstParityEven_u8_u8` API (continued)

Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	Data	Input Data
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	boolean	Test result TRUE : Parity of input parameter is even FALSE: Parity of input parameter is odd
Description	The Bfx_TstParityEven_u8_u8 function returns TRUE when the number of bits whose logical status is set to 1 in the Data input parameter is even, otherwise the function returns FALSE.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.38 Bfx_TstParityEven_u16_u8

Table 41 Specification for `Bfx_TstParityEven_u16_u8` API

Syntax	`boolean  Bfx_TstParityEven_u16_u8`
	`(`
	`  const uint16 Data`
	`)`	

Service ID	0x41	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	Data	Input Data
Parameters (out)	-	-

Table 41 Specification for `Bfx_TstParityEven_u16_u8` API (continued)

Parameters (in - out)	-	-
Return	boolean	Test result TRUE : Parity of input parameter is even FALSE: Parity of input parameter is odd
Description	The Bfx_TstParityEven_u16_u8 function returns TRUE when the number of bits whose logical status is set to 1 in the Data input parameter is even, otherwise the function returns FALSE.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.39 Bfx_TstParityEven_u32_u8

Table 42 Specification for `Bfx_TstParityEven_u32_u8` API

Syntax	`boolean Bfx_TstParityEven_u32_u8 ( const uint32 Data )`	
Service ID	0x42	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	Data	Input Data
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	boolean	Test result TRUE : Parity of input parameter is even FALSE: Parity of input parameter is odd
Description	The Bfx_TstParityEven_u32_u8 function returns TRUE when the number of bits whose logical status is set to 1 in the Data input parameter is even, otherwise the function returns FALSE.	
Source	AUTOSAR	

Table 42 Specification for `Bfx_TstParityEven_u32_u8` API (continued)

Error handling	-
Configuration dependencies	-
User hints	None
SFR accessed	None
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.40 Bfx_TstParityEven_u64_u8

Table 43 Specification for `Bfx_TstParityEven_u64_u8` API

Syntax	`boolean Bfx_TstParityEven_u64_u8 ( const uint64 Data )`	
Service ID	0x43	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	Data	Input Data
Parameters (out)	-	-
Parameters (in - out)	-	-
Return	boolean	Test result TRUE : Parity of input parameter is even FALSE: Parity of input parameter is odd
Description	The Bfx_TstParityEven_u64_u8 function returns TRUE when the number of bits whose logical status is set to 1 in the Data input parameter is even, otherwise the function returns FALSE.	
Source	IFX for AS4.2.2 variant and AUTOSAR for AS4.4.0 variant	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.41 Bfx_ToggleBits_u8

Table 44 Specification for `Bfx_ToggleBits_u8` API

Syntax	`void Bfx_ToggleBits_u8 ( uint8 * const Data )`	
Service ID	0x46	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	-	-
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_ToggleBits_u8 function toggles all the bits of the Data parameter (1's complement of the Data parameter).	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.42 Bfx_ToggleBits_u16

Table 45 Specification for `Bfx_ToggleBits_u16` API

Syntax	`void Bfx_ToggleBits_u16 ( uint16 * const Data )`
Service ID	0x47
Sync/Async	Synchronous
ASIL Level	B
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API

Table 45 Specification for `Bfx_ToggleBits_u16` API (continued)

Parameters (in)	-	-
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_ToggleBits_u16 function toggles all the bits of the Data parameter (1's complement of the Data parameter).	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.43 Bfx_ToggleBits_u32

Table 46 Specification for `Bfx_ToggleBits_u32` API

Syntax	`void Bfx_ToggleBits_u32 ( uint32 * const Data )`	
Service ID	0x48	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	-	-
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_ToggleBits_u32 function toggles all the bits of the Data parameter (1's complement of the Data parameter).	
Source	AUTOSAR	

Table 46 Specification for `Bfx_ToggleBits_u32` API (continued)

Error handling	-
Configuration dependencies	-
User hints	None
SFR accessed	None
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.44 Bfx_ToggleBits_u64

Table 47 Specification for `Bfx_ToggleBits_u64` API

Syntax	`void Bfx_ToggleBits_u64 ( uint64 * const Data )`	
Service ID	0x49	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	-	-
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_ToggleBits_u64 function toggles all the bits of the Data parameter (1's complement of the Data parameter).	
Source	IFX for AS4.2.2 variant and AUTOSAR for AS4.4.0 variant	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.45 Bfx_ToggleBitMask_u8u8

Table 48 Specification for `Bfx_ToggleBitMask_u8u8` API

Syntax	`void Bfx_ToggleBitMask_u8u8 ( uint8 * const Data, const uint8 Mask )`	
Service ID	0x4a	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	Mask	Mask value
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_ToggleBitMask_u8u8 function toggles the logical status of the bits of the Data parameter, for all the bit positions for which the logical status of bit in the Mask parameter is set to 1. The remaining bits of the Data parameter will retain their original values.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.46 Bfx_ToggleBitMask_u16u16

Table 49 Specification for `Bfx_ToggleBitMask_u16u16` API

Syntax	`void Bfx_ToggleBitMask_u16u16 ( uint16 * const Data, const uint16 Mask )`
Service ID	0x4b
Sync/Async	Synchronous
ASIL Level	B

Table 49 Specification for `Bfx_ToggleBitMask_u16u16` API (continued)

Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	Mask	Mask value
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_ToggleBitMask_u16u16 function toggles the logical status of the bits of the Data parameter, for all the bit positions for which the logical status of bit in the Mask parameter is set to 1. The remaining bits of the Data parameter will retain their original values.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.47 Bfx_ToggleBitMask_u32u32

Table 50 Specification for `Bfx_ToggleBitMask_u32u32` API

Syntax	`void Bfx_ToggleBitMask_u32u32 ( uint32 * const Data, const uint32 Mask )`	
Service ID	0x4c	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	Mask	Mask value
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-

Table 50 Specification for `Bfx_ToggleBitMask_u32u32` API (continued)

Description	The Bfx_ToggleBitMask_u32u32 function toggles the logical status of the bits of the Data parameter, for all the bit positions for which the logical status of bit in the Mask parameter is set to 1. The remaining bits of the Data parameter will retain their original values.
Source	AUTOSAR
Error handling	-
Configuration dependencies	-
User hints	None
SFR accessed	None
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.48 Bfx_ToggleBitMask_u64u64

Table 51 Specification for `Bfx_ToggleBitMask_u64u64` API

Syntax	`void Bfx_ToggleBitMask_u64u64 ( uint64 * const Data, const uint64 Mask )`	
Service ID	0x4d	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	Mask	Mask value
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_ToggleBitMask_u64u64 function toggles the logical status of the bits of the Data parameter, for all the bit positions for which the logical status of bit in the Mask parameter is set to 1. The remaining bits of the Data parameter will retain their original values.	
Source	IFX for AS4.2.2 variant and AUTOSAR for AS4.4.0 variant	
Error handling	-	
Configuration dependencies	-	
User hints	None	

Table 51 Specification for `Bfx_ToggleBitMask_u64u64` API (continued)

SFR accessed	None
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.49 Bfx_ShiftBitRt_u8u8

Table 52 Specification for `Bfx_ShiftBitRt_u8u8` API

Syntax	`void Bfx_ShiftBitRt_u8u8 ( uint8 * const Data, const uint8 ShiftCnt )`	
Service ID	0x50	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	ShiftCnt	Shift right count (Valid range: 0 to 7)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_ShiftBitRt_u8u8 function shifts the bits of the Data parameter to the right by ShiftCnt count. The most significant bit (left-most bit) is replaced by a 0 bit and the least significant bit (right-most bit) is discarded for every single bit shift cycle.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for shifting bits of the 8-bit Data parameter. Hence, the valid range for the ShiftCnt parameter, which indicates the count by which the bits are to be shifted, is 0 to 7.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.50 Bfx_ShiftBitRt_u16u8

Table 53 Specification for `Bfx_ShiftBitRt_u16u8` API

Syntax	`void Bfx_ShiftBitRt_u16u8 ( uint16 * const Data, const uint8 ShiftCnt )`	
Service ID	0x51	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	ShiftCnt	Shift right count (Valid range: 0 to 15)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_ShiftBitRt_u16u8 function shifts the bits of the Data parameter to the right by ShiftCnt count. The most significant bit (left-most bit) is replaced by a 0 bit and the least significant bit (right-most bit) is discarded for every single bit shift cycle.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for shifting bits of the 16-bit Data parameter. Hence, the valid range for the ShiftCnt parameter, which indicates the count by which the bits are to be shifted, is 0 to 15.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.51 Bfx_ShiftBitRt_u32u8

Table 54 Specification for `Bfx_ShiftBitRt_u32u8` API

Syntax	`void Bfx_ShiftBitRt_u32u8 ( uint32 * const Data, const uint8 ShiftCnt )`
Service ID	0x52
Sync/Async	Synchronous

Table 54 Specification for `Bfx_ShiftBitRt_u32u8` API (continued)

ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	ShiftCnt	Shift right count (Valid range: 0 to 31)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_ShiftBitRt_u32u8 function shifts the bits of the Data parameter to the right by ShiftCnt count. The most significant bit (left-most bit) is replaced by a 0 bit and the least significant bit (right-most bit) is discarded for every single bit shift cycle.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for shifting bits of the 32-bit Data parameter. Hence, the valid range for the ShiftCnt parameter, which indicates the count by which the bits are to be shifted, is 0 to 31.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.52 Bfx_ShiftBitRt_u64u8

Table 55 Specification for `Bfx_ShiftBitRt_u64u8` API

Syntax	`void Bfx_ShiftBitRt_u64u8 ( uint64 * const Data, const uint8 ShiftCnt )`	
Service ID	0x53	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	ShiftCnt	Shift right count (Valid range: 0 to 63)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified

Table 55 Specification for `Bfx_ShiftBitRt_u64u8` API (continued)

Return	void	-
Description	The Bfx_ShiftBitRt_u64u8 function shifts the bits of the Data parameter to the right by ShiftCnt count. The most significant bit (left-most bit) is replaced by a 0 bit and the least significant bit (right-most bit) is discarded for every single bit shift cycle.	
Source	IFX for AS4.2.2 variant and AUTOSAR for AS4.4.0 variant	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for shifting bits of the 64-bit Data parameter. Hence, the valid range for the ShiftCnt parameter, which indicates the count by which the bits are to be shifted, is 0 to 63.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.53 Bfx_ShiftBitLt_u8u8

Table 56 Specification for `Bfx_ShiftBitLt_u8u8` API

Syntax	`void Bfx_ShiftBitLt_u8u8 ( uint8 * const Data, const uint8 ShiftCnt )`	
Service ID	0x56	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	ShiftCnt	Shift left count (Valid range: 0 to 7)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_ShiftBitLt_u8u8 function shifts the bits of the Data parameter to the left by ShiftCnt count. The least significant bit (right-most bit) is replaced by a 0 bit and the most significant bit (left-most bit) is discarded for every single bit shift cycle.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	

Table 56 Specification for `Bfx_ShiftBitLt_u8u8` API (continued)

User hints	The API is used for shifting bits of the 8-bit Data parameter. Hence, the valid range for the ShiftCnt parameter, which indicates the count by which the bits are to be shifted, is 0 to 7.
SFR accessed	None
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.54 Bfx_ShiftBitLt_u16u8

Table 57 Specification for `Bfx_ShiftBitLt_u16u8` API

Syntax	`void Bfx_ShiftBitLt_u16u8 ( uint16 * const Data, const uint8 ShiftCnt )`	
Service ID	0x57	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	ShiftCnt	Shift left count (Valid range: 0 to 15)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_ShiftBitLt_u16u8 function shifts the bits of the Data parameter to the left by ShiftCnt count. The least significant bit (right-most bit) is replaced by a 0 bit and the most significant bit (left-most bit) is discarded for every single bit shift cycle.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for shifting bits of the 16-bit Data parameter. Hence, the valid range for the ShiftCnt parameter, which indicates the count by which the bits are to be shifted, is 0 to 15.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.55 Bfx_ShiftBitLt_u32u8

Table 58 Specification for `Bfx_ShiftBitLt_u32u8` API

Syntax	`void Bfx_ShiftBitLt_u32u8 ( uint32 * const Data, const uint8 ShiftCnt )`	
Service ID	0x58	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	ShiftCnt	Shift left count (Valid range: 0 to 31)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_ShiftBitLt_u32u8 function shifts the bits of the Data parameter to the left by ShiftCnt count. The least significant bit (right-most bit) is replaced by a 0 bit and the most significant bit (left-most bit) is discarded for every single bit shift cycle.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for shifting bits of the 32-bit Data parameter. Hence, the valid range for the ShiftCnt parameter, which indicates the count by which the bits are to be shifted, is 0 to 31.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.56 Bfx_ShiftBitLt_u64u8

Table 59 Specification for `Bfx_ShiftBitLt_u64u8` API

Syntax	`void Bfx_ShiftBitLt_u64u8 ( uint64 * const Data, const uint8 ShiftCnt )`
Service ID	0x59
Sync/Async	Synchronous

Table 59 Specification for `Bfx_ShiftBitLt_u64u8` API (continued)

ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	ShiftCnt	Shift left count (Valid range: 0 to 63)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_ShiftBitLt_u64u8 function shifts the bits of the Data parameter to the left by ShiftCnt count. The least significant bit (right-most bit) is replaced by a 0 bit and the most significant bit (left-most bit) is discarded for every single bit shift cycle.	
Source	IFX for AS4.2.2 variant and AUTOSAR for AS4.4.0 variant	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for shifting bits of the 64-bit Data parameter. Hence, the valid range for the ShiftCnt parameter, which indicates the count by which the bits are to be shifted, is 0 to 63.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.57 Bfx_RotBitRt_u8u8

Table 60 Specification for `Bfx_RotBitRt_u8u8` API

Syntax	`void Bfx_RotBitRt_u8u8 ( uint8 * const Data, const uint8 ShiftCnt )`	
Service ID	0x5a	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	ShiftCnt	Rotate right count (Valid range: 0 to 7)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified

Table 60 Specification for `Bfx_RotBitRt_u8u8` API (continued)

Return	void	-
Description	The Bfx_RotBitRt_u8u8 function rotates the bits of the Data parameter to the right by ShiftCnt count. The least significant bit (right-most bit) is rotated to the most significant bit (left-most bit) location for every single bit shift cycle.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for rotating bits of the 8-bit Data parameter. Hence, the valid range for the ShiftCnt parameter, which indicates the count by which the bits are to be rotated, is 0 to 7.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.58 Bfx_RotBitRt_u16u8

Table 61 Specification for `Bfx_RotBitRt_u16u8` API

Syntax	`void Bfx_RotBitRt_u16u8 ( uint16 * const Data, const uint8 ShiftCnt )`	
Service ID	0x5b	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	ShiftCnt	Rotate right count (Valid range: 0 to 15)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_RotBitRt_u16u8 function rotates the bits of the Data parameter to the right by ShiftCnt count. The least significant bit (right-most bit) is rotated to the most significant bit (left-most bit) location for every single bit shift cycle.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	

Table 61 Specification for `Bfx_RotBitRt_u16u8` API (continued)

User hints	The API is used for rotating bits of the 16-bit Data parameter. Hence, the valid range for the ShiftCnt parameter, which indicates the count by which the bits are to be rotated, is 0 to 15.
SFR accessed	None
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.59 Bfx_RotBitRt_u32u8

Table 62 Specification for `Bfx_RotBitRt_u32u8` API

Syntax	`void Bfx_RotBitRt_u32u8 ( uint32 * const Data, const uint8 ShiftCnt )`	
Service ID	0x5c	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	ShiftCnt	Rotate right count (Valid range: 0 to 31)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_RotBitRt_u32u8 function rotates the bits of the Data parameter to the right by ShiftCnt count. The least significant bit (right-most bit) is rotated to the most significant bit (left-most bit) location for every single bit shift cycle.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for rotating bits of the 32-bit Data parameter. Hence, the valid range for the ShiftCnt parameter, which indicates the count by which the bits are to be rotated, is 0 to 31.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.60 Bfx_RotBitRt_u64u8

Table 63 Specification for `Bfx_RotBitRt_u64u8` API

Syntax	`void Bfx_RotBitRt_u64u8 ( uint64 * const Data, const uint8 ShiftCnt )`	
Service ID	0x5d	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	ShiftCnt	Rotate right count (Valid range: 0 to 63)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_RotBitRt_u64u8 function rotates the bits of the Data parameter to the right by ShiftCnt count. The least significant bit (right-most bit) is rotated to the most significant bit (left-most bit) location for every single bit shift cycle.	
Source	IFX for AS4.2.2 variant and AUTOSAR for AS4.4.0 variant	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for rotating bits of the 64-bit Data parameter. Hence, the valid range for the ShiftCnt parameter, which indicates the count by which the bits are to be rotated, is 0 to 63.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.61 Bfx_RotBitLt_u8u8

Table 64 Specification for `Bfx_RotBitLt_u8u8` API

Syntax	`void Bfx_RotBitLt_u8u8 ( uint8 * const Data, const uint8 ShiftCnt )`
Service ID	0x60
Sync/Async	Synchronous

Table 64 Specification for `Bfx_RotBitLt_u8u8` API (continued)

ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	ShiftCnt	Rotate left count (Valid range: 0 to 7)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_RotBitLt_u8u8 function rotates the bits of the Data parameter to the left by ShiftCnt count. The most significant bit (left-most bit) is rotated to the least significant bit (right-most bit) location for every single bit shift cycle.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for rotating bits of the 8-bit Data parameter. Hence, the valid range for the ShiftCnt parameter, which indicates the count by which the bits are to be rotated, is 0 to 7.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.62 Bfx_RotBitLt_u16u8

Table 65 Specification for `Bfx_RotBitLt_u16u8` API

Syntax	`void Bfx_RotBitLt_u16u8 ( uint16 * const Data, const uint8 ShiftCnt )`	
Service ID	0x61	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	ShiftCnt	Rotate left count (Valid range: 0 to 15)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified

Table 65 Specification for `Bfx_RotBitLt_u16u8` API (continued)

Return	void	-
Description	The Bfx_RotBitLt_u16u8 function rotates the bits of the Data parameter to the left by ShiftCnt count. The most significant bit (left-most bit) is rotated to the least significant bit (right-most bit) location for every single bit shift cycle.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for rotating bits of the 16-bit Data parameter. Hence, the valid range for the ShiftCnt parameter, which indicates the count by which the bits are to be rotated, is 0 to 15.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.63 Bfx_RotBitLt_u32u8

Table 66 Specification for `Bfx_RotBitLt_u32u8` API

Syntax	`void Bfx_RotBitLt_u32u8 ( uint32 * const Data, const uint8 ShiftCnt )`	
Service ID	0x62	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	ShiftCnt	Rotate left count (Valid range: 0 to 31)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_RotBitLt_u32u8 function rotates the bits of the Data parameter to the left by ShiftCnt count. The most significant bit (left-most bit) is rotated to the least significant bit (right-most bit) location for every single bit shift cycle.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	

Table 66 Specification for `Bfx_RotBitLt_u32u8` API (continued)

User hints	The API is used for rotating bits of the 32-bit Data parameter. Hence, the valid range for the ShiftCnt parameter, which indicates the count by which the bits are to be rotated, is 0 to 31.
SFR accessed	None
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.64 Bfx_RotBitLt_u64u8

Table 67 Specification for `Bfx_RotBitLt_u64u8` API

Syntax	`void Bfx_RotBitLt_u64u8 ( uint64 * const Data, const uint8 ShiftCnt )`	
Service ID	0x63	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	ShiftCnt	Rotate left count (Valid range: 0 to 63)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_RotBitLt_u64u8 function rotates the bits of the Data parameter to the left by ShiftCnt count. The most significant bit (left-most bit) is rotated to the least significant bit (right-most bit) location for every single bit shift cycle.	
Source	IFX for AS4.2.2 variant and AUTOSAR for AS4.4.0 variant	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for rotating bits of the 64-bit Data parameter. Hence, the valid range for the ShiftCnt parameter, which indicates the count by which the bits are to be rotated, is 0 to 63.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.65 Bfx_CopyBit_u8u8u8u8

Table 68 Specification for `Bfx_CopyBit_u8u8u8u8` API

Syntax	`void Bfx_CopyBit_u8u8u8u8 ( uint8 * const DestinationData, const uint8 DestinationPosition, const uint8 SourceData, const uint8 SourcePosition )`	
Service ID	0x66	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	DestinationPosition SourceData SourcePosition	Destination bit position (Valid range: 0 to 7) Source data Source bit position (Valid range: 0 to 7)
Parameters (out)	-	-
Parameters (in - out)	DestinationData	Pointer to destination data which is to be modified
Return	void	-
Description	The Bfx_CopyBit_u8u8u8u8 function copies a bit at SourcePosition bit position of the SourceData parameter to DestinationPosition bit position of the DestinationData parameter.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying a bit of the 8-bit SourceData parameter. Hence, the valid range for input parameters are as follows: 1. The valid range for the SourcePosition parameter, which indicates the position of the bit to be copied, is 0 to 7. 2. The valid range for the DestinationPosition parameter, which indicates the position of the bit to be modified, is 0 to 7.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.66 Bfx_CopyBit_u16u8u16u8

Table 69 Specification for `Bfx_CopyBit_u16u8u16u8` API

Syntax	`void Bfx_CopyBit_u16u8u16u8 ( uint16 * const DestinationData, const uint8 DestinationPosition, const uint16 SourceData, const uint8 SourcePosition )`	
Service ID	0x67	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	DestinationPosition SourceData SourcePosition	Destination bit position (Valid range: 0 to 15) Source data Source bit position (Valid range: 0 to 15)
Parameters (out)	-	-
Parameters (in - out)	DestinationData	Pointer to destination data which is to be modified
Return	void	-
Description	The Bfx_CopyBit_u16u8u16u8 function copies a bit at SourcePosition bit position of the SourceData parameter to DestinationPosition bit position of the DestinationData parameter.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying a bit of the 16-bit SourceData parameter. Hence, the valid range for input parameters are as follows: 1. The valid range for the SourcePosition parameter, which indicates the position of the bit to be copied, is 0 to 15. 2. The valid range for the DestinationPosition parameter, which indicates the position of the bit to be modified, is 0 to 15.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.67 Bfx_CopyBit_u32u8u32u8

Table 70 Specification for `Bfx_CopyBit_u32u8u32u8` API

Syntax	`void Bfx_CopyBit_u32u8u32u8 ( uint32 * const DestinationData, const uint8 DestinationPosition, const uint32 SourceData, const uint8 SourcePosition )`	
Service ID	0x68	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	DestinationPosition SourceData SourcePosition	Destination bit position (Valid range: 0 to 31) Source data Source bit position (Valid range: 0 to 31)
Parameters (out)	-	-
Parameters (in - out)	DestinationData	Pointer to destination data which is to be modified
Return	void	-
Description	The Bfx_CopyBit_u32u8u32u8 function copies a bit at SourcePosition bit position of the SourceData parameter to DestinationPosition bit position of the DestinationData parameter.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying a bit of the 32-bit SourceData parameter. Hence, the valid range for input parameters are as follows: 1. The valid range for the SourcePosition parameter, which indicates the position of the bit to be copied, is 0 to 31. 2. The valid range for the DestinationPosition parameter, which indicates the position of the bit to be modified, is 0 to 31.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.68 Bfx_CopyBit_u64u8u64u8

Table 71 Specification for `Bfx_CopyBit_u64u8u64u8` API

Syntax	`void Bfx_CopyBit_u64u8u64u8 ( uint64 * const DestinationData, const uint8 DestinationPosition, const uint64 SourceData, const uint8 SourcePosition )`	
Service ID	0x69	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	DestinationPosition SourceData SourcePosition	Destination bit position (Valid range: 0 to 63) Source data Source bit position (Valid range: 0 to 63)
Parameters (out)	-	-
Parameters (in - out)	DestinationData	Pointer to destination data which is to be modified
Return	void	-
Description	The Bfx_CopyBit_u64u8u64u8 function copies a bit at SourcePosition bit position of the SourceData parameter to DestinationPosition bit position of the DestinationData parameter.	
Source	IFX for AS4.2.2 variant and AUTOSAR for AS4.4.0 variant	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying a bit of the 64-bit SourceData parameter. Hence, the valid range for input parameters are as follows: 1. The valid range for the SourcePosition parameter, which indicates the position of the bit to be copied, is 0 to 63. 2. The valid range for the DestinationPosition parameter, which indicates the position of the bit to be modified, is 0 to 63.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.69 Bfx_PutBits_u8u8u8u8

Table 72 Specification for `Bfx_PutBits_u8u8u8u8` API

Syntax	`void Bfx_PutBits_u8u8u8u8 ( uint8 * const Data, const uint8 BitStartPn, const uint8 BitLn, const uint8 Pattern )`	
Service ID	0x70	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	BitStartPn BitLn Pattern	Start bit position (Valid range: 0 to 7) Bit field length (Valid range: 1 to (8 - BitStartPn)) Bit pattern to be set
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to destination data which is to be modified
Return	void	-
Description	The Bfx_PutBits_u8u8u8u8 function copies the bit pattern from the Pattern parameter starting from 0 bit position for BitLn number of bits into the Data parameter at the bit positions starting from BitStartPn bit position for BitLn number of bits.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying some bits of the 8-bit Data parameter. Hence, the valid ranges for input parameters are as follows: 1. The valid range for the BitStartPn parameter, which indicates the start position of the bits to be modified, is 0 to 7. 2. The valid range for the BitLn parameter, which indicates the number of bits to be modified, is 1 to (8 - BitStartPn).	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.70 Bfx_PutBits_u16u8u8u16

Table 73 Specification for `Bfx_PutBits_u16u8u8u16` API

Syntax	`void Bfx_PutBits_u16u8u8u16 ( uint16 * const Data, const uint8 BitStartPn, const uint8 BitLn, const uint16 Pattern )`	
Service ID	0x71	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	BitStartPn BitLn Pattern	Start bit position (Valid range: 0 to 15) Bit field length (Valid range: 1 to (16 - BitStartPn)) Bit pattern to be set
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to destination data which is to be modified
Return	void	-
Description	The Bfx_PutBits_u16u8u8u16 function copies the bit pattern from the Pattern parameter starting from 0 bit position for BitLn number of bits into the Data parameter at the bit positions starting from BitStartPn bit position for BitLn number of bits.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying some bits of the 16-bit Data parameter. Hence, the valid ranges for input parameters are as follows: 1. The valid range for the BitStartPn parameter, which indicates the start position of the bits to be modified, is 0 to 15. 2. The valid range for the BitLn parameter, which indicates the number of bits to be modified, is 1 to (16 - BitStartPn).	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.71 Bfx_PutBits_u32u8u8u32

Table 74 Specification for `Bfx_PutBits_u32u8u8u32` API

Syntax	`void Bfx_PutBits_u32u8u8u32 ( uint32 * const Data, const uint8 BitStartPn, const uint8 BitLn, const uint32 Pattern )`	
Service ID	0x72	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	BitStartPn BitLn Pattern	Start bit position (Valid range: 0 to 31) Bit field length (Valid range: 1 to (32 - BitStartPn)) Bit pattern to be set
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to destination data which is to be modified
Return	void	-
Description	The Bfx_PutBits_u32u8u8u32 function copies the bit pattern from the Pattern parameter starting from 0 bit position for BitLn number of bits into the Data parameter at the bit positions starting from BitStartPn bit position for BitLn number of bits.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying some bits of the 32-bit Data parameter. Hence, the valid ranges for input parameters are as follows: 1. The valid range for the BitStartPn parameter, which indicates the start position of the bits to be modified, is 0 to 31. 2. The valid range for the BitLn parameter, which indicates the number of bits to be modified, is 1 to (32 - BitStartPn).	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.72 Bfx_PutBits_u64u8u8u64

Table 75 Specification for `Bfx_PutBits_u64u8u8u64` API

Syntax	`void Bfx_PutBits_u64u8u8u64 ( uint64 * const Data, const uint8 BitStartPn, const uint8 BitLn, const uint64 Pattern )`	
Service ID	0x73	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	BitStartPn BitLn Pattern	Start bit position (Valid range: 0 to 63) Bit field length (Valid range: 1 to (64 - BitStartPn)) Bit pattern to be set
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to destination data which is to be modified
Return	void	-
Description	The Bfx_PutBits_u64u8u8u64 function copies the bit pattern from the Pattern parameter starting from 0 bit position for BitLn number of bits into the Data parameter at the bit positions starting from BitStartPn bit position for BitLn number of bits.	
Source	IFX for AS4.2.2 variant and AUTOSAR for AS4.4.0 variant	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying some bits of the 64-bit Data parameter. Hence, the valid ranges for input parameters are as follows: 1. The valid range for the BitStartPn parameter, which indicates the start position of the bits to be modified, is 0 to 63. 2. The valid range for the BitLn parameter, which indicates the number of bits to be modified, is 1 to (64 - BitStartPn).	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.73 Bfx_PutBitsMask_u8u8u8

Table 76 Specification for `Bfx_PutBitsMask_u8u8u8` API

Syntax	`void Bfx_PutBitsMask_u8u8u8 ( uint8 * const Data, const uint8 Pattern, const uint8 Mask )`	
Service ID	0x80	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	Pattern Mask	Bit pattern to be set Mask value
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to destination data which is to be modified
Return	void	-
Description	The Bfx_PutBitsMask_u8u8u8 function copies the bit pattern from the Pattern parameter into the Data parameter, for all the bit positions for which the logical status of bit in the Mask parameter is set to 1. The remaining bits of the Data parameter retain their original values.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.74 Bfx_PutBitsMask_u16u16u16

Table 77 Specification for `Bfx_PutBitsMask_u16u16u16` API

Syntax	`void Bfx_PutBitsMask_u16u16u16 ( uint16 * const Data, const uint16 Pattern, const uint16 Mask )`
Service ID	0x81

Table 77 Specification for `Bfx_PutBitsMask_u16u16u16` API (continued)

Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	Pattern Mask	Bit pattern to be set Mask value
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to destination data which is to be modified
Return	void	-
Description	The Bfx_PutBitsMask_u16u16u16 function copies the bit pattern from the Pattern parameter into the Data parameter, for all the bit positions for which the logical status of bit in the Mask parameter is set to 1. The remaining bits of the Data parameter retain their original values.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.75 Bfx_PutBitsMask_u32u32u32

Table 78 Specification for `Bfx_PutBitsMask_u32u32u32` API

Syntax	`void  Bfx_PutBitsMask_u32u32u32`
	`(`
	`  uint32 * const Data,`
	`  const uint32 Pattern,`
	`  const uint32 Mask`
	`)`	

Service ID	0x82	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	Pattern Mask	Bit pattern to be set Mask value
Parameters (out)	-	-

Table 78 Specification for `Bfx_PutBitsMask_u32u32u32` API (continued)

Parameters (in - out)	Data	Pointer to destination data which is to be modified
Return	void	-
Description	The Bfx_PutBitsMask_u32u32u32 function copies the bit pattern from the Pattern parameter into the Data parameter, for all the bit positions for which the logical status of bit in the Mask parameter is set to 1. The remaining bits of the Data parameter retain their original values.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.76 Bfx_PutBitsMask_u64u64u64

Table 79 Specification for `Bfx_PutBitsMask_u64u64u64` API

Syntax	`void Bfx_PutBitsMask_u64u64u64 ( uint64 * const Data, const uint64 Pattern, const uint64 Mask )`	
Service ID	0x83	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	Pattern Mask	Bit pattern to be set Mask value
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to destination data which is to be modified
Return	void	-
Description	The Bfx_PutBitsMask_u64u64u64 function copies the bit pattern from the Pattern parameter into the Data parameter, for all the bit positions for which the logical status of bit in the Mask parameter is set to 1. The remaining bits of the Data parameter retain their original values.	
Source	IFX for AS4.2.2 variant and AUTOSAR for AS4.4.0 variant	

Table 79	Specification for `Bfx_PutBitsMask_u64u64u64` API (continued)
Error handling	-
Configuration dependencies	-
User hints	None
SFR accessed	None
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.

1.3.3.77 Bfx_PutBit_u8u8u8

Table 80	Specification for `Bfx_PutBit_u8u8u8` API	
Syntax	`void Bfx_PutBit_u8u8u8 ( uint8 * const Data, const uint8 BitPn, const boolean Status )`	
Service ID	0x85	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	BitPn Status	Bit position (Valid range: 0 to 7) Status value (Valid values: TRUE or FALSE)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_PutBit_u8u8u8 function updates the logical status of the bit at BitPn bit position of the Data parameter to 1 when the value of Status parameter is TRUE; otherwise, the function updates the logical status of the bit at BitPn bit position of the Data parameter to 0.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying a bit of the 8-bit Data parameter. Hence, the valid range for the BitPn parameter, which indicates the position of the bit to be modified, is 0 to 7.	
SFR accessed	None	

Table 80	Specification for `Bfx_PutBit_u8u8u8` API (continued)	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.78 Bfx_PutBit_u16u8u8

Table 81	Specification for `Bfx_PutBit_u16u8u8` API	
Syntax	`void Bfx_PutBit_u16u8u8 ( uint16 * const Data, const uint8 BitPn, const boolean Status )`	
Service ID	0x86	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	BitPn Status	Bit position (Valid range: 0 to 15) Status value (Valid values: TRUE or FALSE)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_PutBit_u16u8u8 function updates the logical status of the bit at BitPn bit position of the Data parameter to 1 when the value of Status parameter is TRUE; otherwise, the function updates the logical status of the bit at BitPn bit position of the Data parameter to 0.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying a bit of the 16-bit Data parameter. Hence, the valid range for the BitPn parameter, which indicates the position of the bit to be modified, is 0 to 15.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.79 Bfx_PutBit_u32u8u8

Table 82 Specification for `Bfx_PutBit_u32u8u8` API

Syntax	`void Bfx_PutBit_u32u8u8 ( uint32 * const Data, const uint8 BitPn, const boolean Status )`	
Service ID	0x87	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	BitPn Status	Bit position (Valid range: 0 to 31) Status value (Valid values: TRUE or FALSE)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_PutBit_u32u8u8 function updates the logical status of the bit at BitPn bit position of the Data parameter to 1 when the value of Status parameter is TRUE; otherwise, the function updates the logical status of the bit at BitPn bit position of the Data parameter to 0.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying a bit of the 32-bit Data parameter. Hence, the valid range for the BitPn parameter, which indicates the position of the bit to be modified, is 0 to 31.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.80 Bfx_PutBit_u64u8u8

Table 83 Specification for `Bfx_PutBit_u64u8u8` API

Syntax	`void Bfx_PutBit_u64u8u8 ( uint64 * const Data, const uint8 BitPn, const boolean Status )`

Table 83 Specification for `Bfx_PutBit_u64u8u8` API (continued)

Service ID	0x88	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant for pointer to distinct memory location passed as parameter to the API	
Parameters (in)	BitPn Status	Bit position (Valid range: 0 to 63) Status value (Valid values: TRUE or FALSE)
Parameters (out)	-	-
Parameters (in - out)	Data	Pointer to data which is to be modified
Return	void	-
Description	The Bfx_PutBit_u64u8u8 function updates the logical status of the bit at BitPn bit position of the Data parameter to 1 when the value of Status parameter is TRUE; otherwise, the function updates the logical status of the bit at BitPn bit position of the Data parameter to 0.	
Source	IFX for AS4.2.2 variant and AUTOSAR for AS4.4.0 variant	
Error handling	-	
Configuration dependencies	-	
User hints	The API is used for modifying a bit of the 64-bit Data parameter. Hence, the valid range for the BitPn parameter, which indicates the position of the bit to be modified, is 0 to 63.	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.3.81 Bfx_GetVersionInfo

Table 84 Specification for `Bfx_GetVersionInfo` API

Syntax	`void Bfx_GetVersionInfo ( Std_VersionInfoType * const Versioninfo )`	
Service ID	0xff	
Sync/Async	Synchronous	
ASIL Level	B	
Re-entrancy	Reentrant	
Parameters (in)	-	-
Parameters (out)	Versioninfo	Pointer to memory location where the version information of this module is to be stored

Table 84 Specification for `Bfx_GetVersionInfo` API (continued)

Parameters (in - out)	-	-
Return	void	-
Description	The Bfx_GetVersionInfo function returns the version information of BFX library.	
Source	AUTOSAR	
Error handling	-	
Configuration dependencies	-	
User hints	None	
SFR accessed	None	
Autosar Version	Applicable for Autosar versions 4.2.2 and 4.4.0.	

1.3.4 Notifications and Callbacks

The BFX library does not provide any notifications or callbacks.

1.3.5 Scheduled functions

The BFX library does not provide any scheduled functions.

1.3.6 Interrupt service routines

The BFX library does not provide any interrupt handlers.

1.3.7 Callout

The driver does not support any callout functions.

1.3.8 Errors Handling

The BFX library does not report any errors.

1.3.9 Deviations and limitations

This section describes the deviations and limitations of the Bfx Library.

1.3.9.1 Deviations

This section describes the deviation of the Bfx Library.

1.3.9.1.1 Software specification deviations

The Bfx Library does not have any deviations.

1.3.9.1.2 AMDC Violations

The Bfx Library does not have any AMDC violations.

1.3.9.1.3 VSMD Violations

The Bfx Library does not have any VSMD violations.

1.3.9.2 Limitations

The section describes the limitations of Bfx Library.

Table 85 Known limitations

Reference	Limitation
Multicore capability and reentrancy of the BFX APIs with pointer parameters	The BFX library does not have any mechanism to serialize the access to a shared resource, which is passed as parameter to a BFX API. Therefore, the BFX APIs are multicore capable and reentrant only for distinct pointer instances as parameters. The onus is on the user to implement an appropriate mechanism to serialize the access to such shared resources, which are passed as parameters to BFX APIs.

Revision history

Table 86 Revision history

Date	Version	Description
2020-08-13	1.0	Released
2020-08-06	0.1	• Initial Version • Bfx driver chapter moved from MC-ISAR_TC3xx_UM_Basic to this document • AoU "Status Assumption" removed and the AoU "Valid pointer as parameter" updated • 64-bit variants of all APIs added

Trademarks

All referenced product or service names and trademarks are the property of their respective owners.

Edition 2020-08-13

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about any aspect of this document?

Email: erratum@infineon.com

Document reference
IFX-ocr1484806431059

IMPORTANT NOTICE

The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics (“Beschaffenheitsgarantie”) .

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party.

In addition, any information given in this document is subject to customer’s compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer’s products and any use of the product of Infineon Technologies in customer’s applications.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer’s technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

WARNINGS

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies’ products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.