Document Title	Specification of MSFLibrary
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	1085

Document Status	published
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	R25-11

Document Change History

Date	Release	Changed by	Description
2025-11-27	R25-11	AUTOSAR Release Management	• No content changes
2024-11-27	R24-11	AUTOSAR Release Management	• Initial release of Memory Specific Function library

Table of Contents

1 Introduction and functional overview

This specification defines the functionality and the API of the AUTOSAR Memory Standard Function Library (Msf).

The functionallity provided by this specification is similar to existing standard functions which are provided by C environments (see [1, ISO-IEC-9899-1999]). AUTOSAR provides this specification in order to harmonize existing similar functions and to not depend on the existing <string.h> provided by compiler vendors.

2 Acronyms and Abbreviations

The glossary below includes acronyms and abbreviations relevant to the Msf module that are not included in the [2, AUTOSAR glossary].

Abbreviation / Acronym:	Description:
Msf	Memory Standard Functions

3 Related documentation

3.1 Input documents & related standards and norms

[1] ISO/IEC 9899:1999
https://www.iso.org

[2] Glossary
AUTOSAR_FO_TR_Glossary

[3] General Specification of Basic Software Modules
AUTOSAR_CP_SWS_BSWGeneral

[4] Requirements on Libraries
AUTOSAR_CP_RS_Libraries

[5] General Requirements on Basic Software Modules
AUTOSAR_CP_RS_BSWGeneral

[6] Specification of Platform Types for Classic Platform
AUTOSAR_CP_SWS_PlatformTypes

3.2 Related specification

AUTOSAR provides a General Specification on Basic Software modules [3, SWS BSW General], which applies also to BSW modules and (in parts) to libraries.

4 Constraints and assumptions

4.1 Limitations

No limitations

4.2 Applicability to car domains

No restrictions

5 Dependencies to other modules

No dependencies to other modules.

6 Requirements Tracing

The following tables reference the requirements specified in [4] and [5] and links to the fulfillment of these.

Requirement	Description	Satisfied by
[SRS_BSW_00003]	All software modules shall provide version and identification information	[CP_SWS_Msf_00011]
[SRS_BSW_00318]	Each AUTOSAR Basic Software Module file shall provide version numbers in the header file	[CP_SWS_Msf_00011]
[SRS_BSW_00321]	The version numbers of AUTOSAR Basic Software Modules shall be enumerated according specific rules	[CP_SWS_Msf_00011]
[SRS_BSW_00407]	Each BSW module shall provide a function to read out the version information of a dedicated module implementation	[CP_SWS_Msf_00011]

Table 6.1: Requirements Tracing

7 Functional specification

7.1 API behavior

The library provides various runtime optimized memory handling functions.

The following functions shall be replacements for the ones of the C standard library (see [1, ISO-IEC-9899-1999]) <string.h>:

Memory block copy function: string.h/memcpy -> `Msf_MemCopy`

Memory block copy function for overlapping memory regions: string.h/memmove -> `Msf_MemMove`

Memory filling function: string.h/memset -> `Msf_MemSet_TypeMn`

Additionally there is an optimized memcopy for 32bit aligned areas:

`Msf_MemCopyAligned_u32`

7.2 Initialization and shutdown

As Msf is a library no initialization is required. There is also no shutdown functionality.

7.3 Using Library API

Msf API can be directly called from BSW modules or SWC. No port definition is required. It is a pure function call.

Using a library shall be documented. If a BSW module or a SWC uses a Library, the developer shall add an `Implementation.DependencyOnArtifact` in the `BSWMDT`/`SWCT`.

7.4 Error Classification

Chapter [3, General Specification of Basic Software Modules] 7.2 *"Error Handling"* describes the error handling of the Basic Software in detail. Above all, it constitutes a classification scheme consisting of five error types which may occur in BSW modules.

As libraries do not depend on other BSW modules, they do not report errors.

7.4.1 Development Errors

There are no development errors.

7.4.2 Runtime Errors

There are no runtime errors.

7.4.3 Production Errors

There are no production errors.

7.4.4 Extended Production Errors

There are no extended production errors.

7.5 Security Events

The module does not report security events.

8 API specification

8.1 Imported types

In this chapter all types included from the following files are listed.

[CP_SWS_Msf_00012] Definition of imported datatypes of module Msf

Status: DRAFT

⌈

Module	***Header File***	***Imported Type***
Std	Std_Types.h	Std_VersionInfoType

⌋

It is observed that since the sizes of the integer types provided by the C language are implementation-defined, the range of values that may be represented within each of the integer types will vary between implementations.

Thus, in order to improve the portability of the software, these types are defined in [6]. The following mnemonic are used in the library routine names.

Note:

The naming convention for the API's with boolean return type/parameter type is given as _u8 which shall be interpreted as _b. (Boolean)

If there is no boolean data type present in the return type/parameter type then _u8 shall be interpreted as _u8 only.

Size	**Platform Type**	**Mnemonic**
unsigned 8-Bit	boolean	u8
signed 8-Bit	sint8	s8
signed 16-Bit	sint16	s16
signed 32-Bit	sint32	s32
unsigned 8-Bit	uint8	u8
unsigned 16-Bit	uint16	u16
unsigned 32-Bit	uint32	u32

Table 8.1: Base Types

As a convention in the rest of the document:

- Mnemonics will be used in the name of the routines (using <InTypeMn1> that means Type Mnemonic for Input 1)
- The real type will be used in the description of the prototypes of the routines (using <InType> or <OutType>).

8.2 Type definitions

None

8.3 Function definitions

8.3.1 Memory Copy Routine

[CP_SWS_Msf_00003] Definition of API function Msf_MemCopy

Status: DRAFT

⌈

Service Name	Msf_MemCopy (draft)	
Syntax	`void* Msf_MemCopy ( void* Dest, const void* Src, uint32 NrOfBytes )`	
Service ID [hex]	0x0100	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	Src	Pointer to source input data
	NrOfBytes	Block size
Parameters (inout)	None	
Parameters (out)	Dest	Pointer to destination data
Return value	void*	Copy of Dest address
Description	Copies the values of NrOfBytes bytes from the location pointed to by source Src directly to the memory block pointed to by destination Dest. **Tags:** atp.Status=draft	
Available via	Msf.h	

⌋

The underlying type of the objects pointed to by both the source and destination pointers are irrelevant for this function. The result is a binary copy of the data. For faster copy operation of 32 bit aligned 32 bit double words use

`Msf_MemCopyAligned_u32`.

[CP_SWS_Msf_CONSTR_00001] Array size restrictions

Status: DRAFT

⌈To avoid undefined behavior, the size of the arrays pointed to by both the destination and source parameters shall be at least NrOfBytes bytes and shall not overlap. This applies to `Msf_MemCopy` and `Msf_MemCopyAligned_u32`.⌋

8.3.2 Aligned Memory Copy Routine

[CP_SWS_Msf_00005] Definition of API function Msf_MemCopyAligned_u32

Status: DRAFT

⌈

Service Name	Msf_MemCopyAligned_u32 (draft)	

Syntax	`uint32* Msf_MemCopyAligned_u32 (`
	`  uint32* Dest,`
	`  const uint32* Src,`
	`  uint32 NrOfDWords`
	`)`	

Service ID [hex]	0x0101	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	Src	Pointer to source input data
	NrOfDWords	Number of 32 bit double word elements
Parameters (inout)	None	
Parameters (out)	Dest	Pointer to destination data
Return value	uint32*	Pointer to destination data
Description	Copies the values of NrOfElements 32 bit words from the location pointed to by source pointer Src directly to the memory block pointed to by destination pointer Dest. The source and data memory addresses shall be aligned according to the data type. Source and destination data region shall not overlap. **Tags:** atp.Status=draft	
Available via	Msf.h	

⌋

The source and data memory addresses shall be aligned to 32 bit boundaries. The function shall not perform any run-in for unaligned pointer access. Use `Msf_MemCopy` for data, which are not aligned to 32 bit or have 8 or 16 bit word width.

Please also consider [CP_SWS_Msf_CONSTR_00001].

Contrary to `Msf_MemCopy` and `Msf_MemMove` the size parameter NrOfElements defines the 32 bit double words, not bytes.

8.3.3 Memory Move Routine

[CP_SWS_Msf_00007] Definition of API function Msf_MemMove

Status: DRAFT

⌈

Service Name	Msf_MemMove (draft)	
Syntax	`void* Msf_MemMove ( void* Dest, const void* Src, uint32 NrOfBytes )`	
Service ID [hex]	0x30	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	Src	Pointer to source input data
	NrOfBytes	Block size
Parameters (inout)	None	
Parameters (out)	Dest	Pointer to destination data
Return value	void*	Copy of Dest address
Description	Copies the values of NrOfBytes bytes from the location pointed to by source pointer Src directly to the memory block pointed to by destination pointer Dest. Copying takes place as if an intermediate buffer were used, allowing the destination and source to overlap. **Tags:** atp.Status=draft	
Available via	Msf.h	

⌋

The underlying type of the objects pointed to by both the source and destination pointers are irrelevant for this function. The result is a binary copy of the data. The function implementation shall copy without any temporary buffer. If the destination address is greater than the source address, the copy loop direction is from right to left, otherwise from left to right.

8.3.4 Memory Filling Routine

[CP_SWS_Msf_00009] Definition of API function Msf_MemSet_<TypeMn>

Status: DRAFT

⌈

Service Name	Msf_MemSet_<TypeMn> (draft)
Syntax	`<Type>* Msf_MemSet_<TypeMn> ( <Type>* Dest, <Type> Pattern, uint32 NrOfElements )`
Service ID [hex]	0x40 to 0x42
Sync/Async	Synchronous

▽

△

Reentrancy	Reentrant	
Parameters (in)	Pattern	Fill pattern
	NrOfElements	Number of elements
Parameters (inout)	None	
Parameters (out)	Dest	Pointer to destination data
Return value	<Type>*	Pointer to destination data
Description	Sets the first NrOfElements words with the API type <Type> of the block of memory pointed by Dest to the specified value. **Tags:** atp.Status=draft	
Available via	Msf.h	

⌋

List of functions:

Function ID[hex]	Function prototype
0x40	uint8 *Msf_MemSet_u8(uint8 *Dest, uint8 Pattern, uint32 NrOfElements)
0x41	uint16 *Msf_MemSet_u16(uint16 *Dest, uint16 Pattern, uint32 NrOfElements)
0x42	uint32 *Msf_MemSet_u32(uint32 *Dest, uint32 Pattern, uint32 NrOfElements)

8.4 Callback notifications

None

8.5 Scheduled functions

None

8.6 Expected interfaces

None

8.6.1 Mandatory interfaces

[CP_SWS_Msf_00013] Definition of mandatory interfaces required by module Msf

Status: DRAFT

⌈

API Function	***Header File***	***Description***
There are no mandatory interfaces.		

⌋

8.6.2 Optional interfaces

[CP_SWS_Msf_00014] Definition of optional interfaces requested by module Msf

Status: DRAFT

⌈

API Function	***Header File***	***Description***
There are no optional interfaces.		

⌋

8.6.3 Configurable interfaces

None

8.7 Version API

8.7.1 Msf_GetVersionInfo

[CP_SWS_Msf_00011] Definition of API function Msf_GetVersionInfo

Status: DRAFT

Upstream requirements: SRS_BSW_00407, SRS_BSW_00003, SRS_BSW_00318, SRS_BSW_00321

⌈

Service Name	Msf_GetVersionInfo (draft)	
Syntax	`void Msf_GetVersionInfo ( Std_VersionInfoType* versioninfo )`	
Service ID [hex]	0xff	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	None	
Parameters (inout)	None	
Parameters (out)	versioninfo	Pointer to where to store the version information of this module. Format according to [SRS_BSW_00321].
Return value	None	
Description	Returns the version information of this library. **Tags:** atp.Status=draft	
Available via	Msf.h	

⌋

9 Sequence diagrams

None

10 Configuration specification

In general, this chapter defines configuration parameters and their clustering into containers. As libraries do not have a configuration this chapter is empty.

A Not applicable requirements

[CP_SWS_Msf_NA_00999]

Status:	DRAFT
Upstream requirements:	SRS_BSW_00344, SRS_BSW_00404, SRS_BSW_00405, SRS_BSW_00345, SRS_BSW_00159, SRS_BSW_00167, SRS_BSW_00171, SRS_BSW_00170, SRS_BSW_00380, SRS_BSW_00419, SRS_BSW_00383, SRS_BSW_00384, SRS_BSW_00388, SRS_BSW_00389, SRS_BSW_00390, SRS_BSW_00392, SRS_BSW_00393, SRS_BSW_00395, SRS_BSW_00396, SRS_BSW_00403, SRS_BSW_00397, SRS_BSW_00398, SRS_BSW_00399, SRS_BSW_00400, SRS_BSW_00438, SRS_BSW_00375, SRS_BSW_00101, SRS_BSW_00416, SRS_BSW_00406, SRS_BSW_00467, SRS_BSW_00437, SRS_BSW_00168, SRS_BSW_00423, SRS_BSW_00424, SRS_BSW_00425, SRS_BSW_00426, SRS_BSW_00427, SRS_BSW_00428, SRS_BSW_00429, SRS_BSW_00432, SRS_BSW_00433, SRS_BSW_00450, SRS_BSW_00461, SRS_BSW_00451, SRS_BSW_00478, SRS_BSW_00336, SRS_BSW_00337, SRS_BSW_00369, SRS_BSW_00339, SRS_BSW_00422, SRS_BSW_00417, SRS_BSW_00323, SRS_BSW_00004, SRS_BSW_00409, SRS_BSW_00385, SRS_BSW_00386, SRS_BSW_00452, SRS_BSW_00458, SRS_BSW_00466, SRS_BSW_00488, SRS_BSW_00489, SRS_BSW_00490, SRS_BSW_00491, SRS_BSW_00492, SRS_BSW_00493, SRS_BSW_00469, SRS_BSW_00470, SRS_BSW_00471, SRS_BSW_00472

⌈The uptraces of this spec item are not applicable to this specification.⌋

B Change history of AUTOSAR traceable items

Please note that the lists in this chapter also include traceable items that have been removed from the specification in a later version. These items do not appear as hyperlinks in the document.

B.1 Traceable item history of this document according to AUTOSAR Release R24-11

B.1.1 Added Specification Items in R24-11

Number	Heading
[CP_SWS_Msf_00003]	Definition of API function Msf_MemCopy
[CP_SWS_Msf_00005]	Definition of API function Msf_MemCopyAligned_u32
[CP_SWS_Msf_00007]	Definition of API function Msf_MemMove
[CP_SWS_Msf_00009]	Definition of API function Msf_MemSet_<TypeMn>
[CP_SWS_Msf_00011]	Definition of API function Msf_GetVersionInfo
[CP_SWS_Msf_00012]	Definition of imported datatypes of module Msf
[CP_SWS_Msf_00013]	Definition of mandatory interfaces required by module Msf
[CP_SWS_Msf_00014]	Definition of optional interfaces requested by module Msf

Table B.1: Added Specification Items in R24-11

B.1.2 Changed Specification Items in R24-11

none

B.1.3 Deleted Specification Items in R24-11

none

B.1.4 Added Constraints in R24-11

Number	Heading
[CP_SWS_Msf_CONSTR_00001]	Array size restrictions

Table B.2: Added Constraints in R24-11

B.1.5 Changed Constraints in R24-11

none

B.1.6 Deleted Constraints in R24-11

none

B.2 Traceable item history of this document according to AUTOSAR Release R25-11

B.2.1 Added Specification Items in R25-11

none

B.2.2 Changed Specification Items in R25-11

none

B.2.3 Deleted Specification Items in R25-11

none

B.2.4 Added Constraints in R25-11

none

B.2.5 Changed Constraints in R25-11

none

B.2.6 Deleted Constraints in R25-11

none