

Micrium

Empowering Embedded Systems

μC/LIB

V1.30

Release Notes

www.Micrium.com

Revision History

Version	Date	Description
V1.30	2009 Jun	New features, bug fixes, & improvements
V1.29	2009 Apr	New features & improvements
V1.28	2009 Mar	New features & improvements
V1.27	2009 Jan	New features, bug fixes, & improvements
V1.26	2008 Nov	New features, bug fixes, & improvements
V1.25	2008 Jul	New features & improvements
V1.24	2007 May	Improvements
V1.23	2007 Mar	Bug fixes & improvements
V1.22	2006 Sep	Improvements
V1.21	2006 Aug	New features & improvements
V1.20	2006 Jun	New features & improvements
V1.19	2006 Apr	Improvements
V1.18	2005 Oct	Bug fixes & improvements First version with release history
V1.17	2005 Jul	Improvements
V1.16	2005 Jun	Improvements
V1.15	2005 May	Improvements
V1.14	2005 Apr	Improvements
V1.13	2005 Feb	Improvements
V1.12	2004 Dec	Improvements
V1.11	2004 Nov	Improvements
V1.10	2004 Sep	Improvements
V1.00	2004 Feb	First release

Requires the following versions of needed Modules

Version 1.30

μC/CPU Version 1.22

Version 1.29

μC/CPU Version 1.22

Version 1.28

μC/CPU Version 1.22

Version 1.27

μC/CPU Version 1.20

Version 1.26

μC/CPU Version 1.19

Version 1.25

μC/CPU Version 1.18

Version 1.24

μC/CPU Version 1.17

Version 1.23

μC/CPU Version 1.16

Version 1.22

μC/CPU Version 1.15

Version 1.21

μC/CPU Version 1.14

Version 1.20

μC/CPU Version 1.14

Version 1.19

μC/CPU Version 1.14

Version 1.18

μC/CPU Version 1.13

Version 1.17

μC/CPU Version 1.12

Version 1.16

μC/CPU Version 1.12

New Features

Version 1.30

V1.30-001

Added new template configuration file `lib_cfg.h`.

V1.30-002

Added `LIB_MEM_CFG_OPTIMIZE_ASM_EN` to enable/disable assembly-optimized memory functions. See also ‘Changes V1.30-001’.

V1.30-003

Added new math module functions :

<code>Math_Init()</code>	initializes mathematical library
<code>Math_Rand()</code>	generates a (pseudo-) random number
<code>Math_RandSeed()</code>	generates the next (pseudo-) random number after a specified seed value
<code>Math_RandSetSeed()</code>	sets the next (pseudo-) random number seed value

V1.30-004

Added new string functions :

<code>Str_Len_N()</code>	calculates a string’s length up to a maximum number of characters
--------------------------	--

See also ‘New Features V1.20-001’.

Version 1.29

V1.29-001

Added new time-related defines :

```
DEF_TIME_NBR_DAY_PER_WK
DEF_TIME_NBR_DAY_PER_YR
DEF_TIME_NBR_DAY_PER_YR_LEAP

DEF_TIME_NBR_HR_PER_WK
DEF_TIME_NBR_HR_PER_YR
DEF_TIME_NBR_HR_PER_YR_LEAP

DEF_TIME_NBR_MIN_PER_WK
DEF_TIME_NBR_MIN_PER_YR
DEF_TIME_NBR_MIN_PER_YR_LEAP

DEF_TIME_NBR_SEC_PER_WK
DEF_TIME_NBR_SEC_PER_YR
DEF_TIME_NBR_SEC_PER_YR_LEAP
```

Version 1.28

V1.28-001

Added **LIB_MEM_CFG_HEAP_BASE_ADDR** to (optionally) specify the heap memory base address.

Version 1.27

V1.27-001

Added new memory allocation function :

Mem_PoolClr() clear a memory pool

See also 'Changes V1.26-001' & 'New Features V1.25-001'.

Version 1.26

V1.26-001

Added new memory allocation function :

Mem_HeapAlloc() get memory from the heap

See also 'Changes V1.26-001' & 'New Features V1.25-001'.

V1.26-002

Added new ASCII module functions & macro's :

<code>ASCII_IsDigOct()</code>	indicates whether a character is an octal digit
<code>ASCII_IS_DIG_OCT()</code>	

See also 'New Features V1.25-002'.

V1.26-003

Added new string compare functions :

<code>Str_CmpIgnoreCase()</code>	compares two strings, ignoring case
<code>Str_CmpIgnoreCase_N()</code>	compares two strings, ignoring case, up to a maximum number of characters

See also 'New Features V1.20-001'.

V1.26-004a

Added new string format functions :

<code>Str_FmtNbr_Int32U()</code>	formats an unsigned number into a string
<code>Str_FmtNbr_Int32S()</code>	formats a signed number into a string

V1.26-004b

Added new string parse functions :

<code>Str_ParseNbr_Int32U()</code>	parses an unsigned number from a string
<code>Str_ParseNbr_Int32S()</code>	parses a signed number from a string

Version 1.25

V1.25-001

Added new memory allocation functions :

<code>Mem_PoolCreate()</code>	create a memory pool
<code>Mem_PoolBlkGet()</code>	get a memory block from a memory pool
<code>Mem_PoolBlkFree()</code>	free a memory block back to a memory pool

See also 'Changes V1.26-001'.

V1.25-002

Added new ASCII module functions & macro's :

<code>ASCII_IsAlpha()</code>	indicates whether a character is alphabetic
<code>ASCII_IS_ALPHA()</code>	
<code>ASCII_IsAlnum()</code>	indicates whether a character is alphanumeric
<code>ASCII_IS_ALNUM()</code>	(see also 'Changes V1.27-001')
<code>ASCII_IsLower()</code>	indicates whether a character is lowercase
<code>ASCII_IS_LOWER()</code>	
<code>ASCII_IsUpper()</code>	indicates whether a character is uppercase
<code>ASCII_IS_UPPER()</code>	
<code>ASCII_IsDig()</code>	indicates whether a character is a decimal digit
<code>ASCII_IS_DIG()</code>	
<code>ASCII_IsDigHex()</code>	indicates whether a character is a hexadecimal digit
<code>ASCII_IS_DIG_HEX()</code>	
<code>ASCII_IsBlank()</code>	indicates whether a character is blank
<code>ASCII_IS_BLANK()</code>	
<code>ASCII_IsSpace()</code>	indicates whether a character is a space
<code>ASCII_IS_SPACE()</code>	
<code>ASCII_IsPrint()</code>	indicates whether a character is printable
<code>ASCII_IS_PRINT()</code>	
<code>ASCII_IsGraph()</code>	indicates whether a character is graphic
<code>ASCII_IS_GRAPH()</code>	
<code>ASCII_IsPunct()</code>	indicates whether a character is punctuation
<code>ASCII_IS_PUNCT()</code>	
<code>ASCII_IsCtrl()</code>	indicates whether a character is a control
<code>ASCII_IS_CTRL()</code>	
<code>ASCII_ToLower()</code>	converts uppercase to lowercase
<code>ASCII_TO_LOWER()</code>	
<code>ASCII_ToUpper()</code>	converts lowercase to uppercase
<code>ASCII_TO_UPPER()</code>	
<code>ASCII_Cmp()</code>	compares two characters (case insensitive)

See also 'Changes V1.25-001'.

Version 1.24

V1.24-001

Added new CPU-related integer defines :

```
DEF_INT_CPU_NBR_BITS  
DEF_INT_CPU_MASK  
DEF_INT_CPU_U_MIN_VAL  
DEF_INT_CPU_U_MAX_VAL  
DEF_INT_CPU_S_MIN_VAL  
DEF_INT_CPU_S_MAX_VAL  
DEF_INT_CPU_S_MIN_VAL_ONES_CPL  
DEF_INT_CPU_S_MAX_VAL_ONES_CPL
```

Version 1.23

N/A

Version 1.22

N/A

Version 1.21

V1.21-001

Added new memory data value macro's :

MEM_VAL_GET_???()	decode data values from any memory address
MEM_VAL_SET_???()	encode data values to any memory address
MEM_VAL_COPY_GET_???()	copy & decode data values from any memory address to any other memory address
MEM_VAL_COPY_SET_???()	copy & encode data values from any memory address to any other memory address
MEM_VAL_COPY_???()	copy data values from any memory address to any other memory address

Version 1.20

V1.20-001

Added new string functions :

Str_Copy_N() copies a string up to a maximum number of characters

Str_Cat_N() concatenates two strings up to a maximum number of characters

Str_Char_N() searches a string up to a maximum number of characters

See also 'New Features V1.30-004'.

Version 1.19

N/A

Version 1.18

N/A

Improvements

Version 1.30

V1.30-001

Improved the following bit macro's to be called from within conditional expressions :

```
DEF_BIT_SET( )  
DEF_BIT_CLR( )
```

Version 1.29

V1.29-001

Improved the configuration of optional memory allocation argument checking.

Version 1.28

V1.28-001

Replaced all '**cpu_sr**' local variable declarations with μ C/CPU's new **CPU_SR_ALLOC()** macro.

Version 1.27

N/A

Version 1.26

V1.26-001

Improved the following string functions to call their related multi-character functions :

```
Str_Copy( )    calls Str_Copy_N( )  
Str_Cat( )     calls Str_Cat_N( )  
Str_Cmp( )     calls Str_Cmp_N( )  
Str_Char( )    calls Str_Char_N( )
```

See also 'New Features V1.20-001'.

V1.26-002a

Improved unsigned integer macro definitions by explicitly declaring unsigned constant.

V1.26-002b

Improved signed integer macro definitions by avoiding twos-complement arithmetic underflow.

Version 1.25

N/A

Version 1.24

V1.24-001

Added **LIB_VERSION** to indicate current library module software version number.

V1.24-002

Improved several **DEF_BIT_???()** macro's to handle overflow boundary conditions.

V1.24-003

Added several **LIB_STR_???** common string defines.

Version 1.23

V1.23-001

Removed **malloc()** & all other references to standard library memory functions.

Version 1.22

N/A

Version 1.21

N/A

Version 1.20

V1.20-001

Improved ARM assembly port files to be compatible for both ARM & Thumb modes.

Version 1.19

N/A

Version 1.18

V1.18-001

Added macro function headers for all **lib_def.h** macros.

V1.18-002

Improved consistency for all **lib_str.c** functions.

Changes

Version 1.30

V1.30-001

Replaced assembly-optimized configuration from generic '**uC_CFG_OPTIMIZE_ASM_EN**' to library-specific '**LIB_MEM_CFG_OPTIMIZE_ASM_EN**'. See also 'New Features V1.30-002'.

Version 1.29

N/A

Version 1.28

N/A

Version 1.27

V1.27-001

Renamed the following **lib_ascii.h** macro's & functions :

ASCII_IsAlnum()	renamed to ASCII_IsAlphaNum()
ASCII_IS_ALNUM()	renamed to ASCII_IS_ALPHA_NUM()

V1.27-002

Modified **Str_FmtNbr_???()** leading character parameter from a Boolean ('**lead_zeros**') that specified whether leading zeros were prepended to the formatted number string when necessary, to the desired ASCII character ('**lead_char**') to prepend to the formatted number string :

```
CPU_CHAR  *Str_FmtNbr_Int32U(CPU_INT32U    nbr,
                               CPU_INT08U    nbr_dig,
                               CPU_INT08U    nbr_base,
                               CPU_CHAR      lead_char,
                               CPU_BOOLEAN    lower_case,
                               CPU_BOOLEAN    nul,
                               CPU_CHAR      *pstr);

CPU_CHAR  *Str_FmtNbr_Int32S(CPU_INT32S    nbr,
                               CPU_INT08U    nbr_dig,
                               CPU_INT08U    nbr_base,
                               CPU_CHAR      lead_char,
                               CPU_BOOLEAN    lower_case,
                               CPU_BOOLEAN    nul,
                               CPU_CHAR      *pstr);
```

```

CPU_CHAR  *Str_FmtNbr_32      (CPU_FP32      nbr,
                                CPU_INT08U     nbr_dig,
                                CPU_INT08U     nbr_dp,
                                CPU_CHAR       lead_char,
                                CPU_BOOLEAN    nul,
                                CPU_CHAR       *pstr);

```

Version 1.26

V1.26-001

Changed memory pool configuration to memory allocation configuration — 'LIB_MEM_CFG_POOL_EN' to 'LIB_MEM_CFG_ALLOC_EN'.

V1.26-002

Changed the following `lib_mem.h` error codes :

```

LIB_MEM_ERR_INVALID_ADDR      changed to
LIB_MEM_ERR_INVALID_BLK_ADDR

```

V1.26-003

Changed the following `lib_def.h` macro constants :

```

DEF_INACTIVE      redefined to 0
DEF_ACTIVE        redefined to 1

```

Version 1.25

V1.25-001

The following macro's in `lib_str.h` have been deprecated & replaced with new macro's & functions in `lib_ascii.h` :

```

Str_IsAlpha()   replaced with ASCII_IsAlpha() / _IS_ALPHA()
Str_IsDigit()   replaced with ASCII_IsDig()   / _IS_DIG()
Str_IsSpace()   replaced with ASCII_IsSpace() / _IS_SPACE()
Str_IsPrint()   replaced with ASCII_IsPrint() / _IS_PRINT()
Str_IsUpper()   replaced with ASCII_IsUpper() / _IS_UPPER()
Str_IsLower()   replaced with ASCII_IsLower() / _IS_LOWER()

Str_ToUpper()   replaced with ASCII_ToUpper() / _TO_UPPER()
Str_ToLower()   replaced with ASCII_ToLower() / _TO_LOWER()

```

See also 'New Features V1.25-002'.

Version 1.24

N/A

Version 1.23

N/A

Version 1.22

N/A

Version 1.21

N/A

Version 1.20

V1.20-001

The following macro names in `lib_str.h` have been changed to comply with standard naming conventions :

<code>Is_Alpha()</code>	changed to <code>Str_IsAlpha()</code>
<code>Is_Digit()</code>	changed to <code>Str_IsDigit()</code>
<code>Is_Space()</code>	changed to <code>Str_IsSpace()</code>
<code>Is_Print()</code>	changed to <code>Str_IsPrint()</code>
<code>Is_Upper()</code>	changed to <code>Str_IsUpper()</code>
<code>Is_Lower()</code>	changed to <code>Str_IsLower()</code>
<code>To_Upper()</code>	changed to <code>Str_ToUpper()</code>
<code>To_Lower()</code>	changed to <code>Str_ToLower()</code>
<code>Str_To_Long()</code>	changed to <code>Str_ToLong()</code>
<code>Str_Format_Print()</code>	changed to <code>Str_FmtPrint()</code>
<code>Str_Format_Scan()</code>	changed to <code>Str_FmtScan()</code>

Version 1.19

V1.19-001

Macros `Str_Format_Print()` & `Str_Format_Scan()` in `lib_str.h` have been corrected to be compatible with some compilers.

Version 1.18

V1.18-001

`DEF_BIT_MASK()` macro & `DEF_BIT_FIELD()` macro switched names.

V1.18-002

Renamed `Str_Char_R()` to `Str_Char_Last()`.

Corrections

Version 1.30

N/A

Version 1.29

N/A

Version 1.28

N/A

Version 1.27

V1.27-001

Str_ParseNbr_Int32() failed to always set negative sign ('**neg**') during validation. Fixed by always setting '**neg**' for all conditions.

Version 1.26

V1.26-001

Mem_PoolCreate() incorrectly calculated the number of additional octets required to successfully allocate all requested memory (returned by '**p_octets_reqd**') for certain fault conditions. Fixed by calculating & returning the actual additional octets required to successfully allocate all requested memory for all error/fault conditions.

Version 1.25

N/A

Version 1.24

N/A

Version 1.23

V1.23-001

ARM assembly port files were not completely compatible for both ARM & Thumb modes (see 'Improvements V1.20-001'). Corrected by using only ARM & Thumb mode instructions.

Version 1.22

N/A

Version 1.21

N/A

Version 1.20

N/A

Version 1.19

N/A

Version 1.18

V1.18-001

Str_Str() incorrectly assigned unsigned string lengths to signed variables. Corrected by assigning string lengths to unsigned variables.

V1.18-002

lib_mem_a.asm did not correctly terminate the memory copy during the **Pre_Copy_1** label if no more data octets to copy. Corrected by terminating the memory copy if no more data octets.

Known Problems

Version 1.30

V1.18-001b (Unresolved)

Version 1.29

V1.18-001b (Unresolved)

Version 1.28

V1.18-001b (Unresolved)

Version 1.27

V1.18-001b (Unresolved)

Version 1.26

V1.18-001b (Unresolved)

Version 1.25

V1.18-001b (Unresolved)

Version 1.24

V1.18-001b (Unresolved)

Version 1.23

V1.18-001b (Unresolved)

Version 1.22

V1.18-001a (Unresolved)

V1.18-001b (Unresolved)

Version 1.21

V1.18-001a (Unresolved)

V1.18-001b (Unresolved)

Version 1.20

V1.18-001a (Unresolved)

V1.18-001b (Unresolved)

Version 1.19

V1.18-001a (Unresolved)

V1.18-001b (Unresolved)

Version 1.18

V1.18-001a

`lib_mem.h` includes some standard library files and functions. **ALL** references to standard library files and functions **SHOULD** be removed once all custom library functions are implemented.

V1.18-001b

`lib_str.h` includes some standard library files and functions. **ALL** references to standard library files and functions **SHOULD** be removed once all custom library functions are implemented.

Limitations

001

Does not support variable argument library functions

Contacts

Micrium

949 Crestview Circle

Weston, FL 33327

USA

+1 954 217 2036

+1 954 217 2037 (FAX)

e-mail: Licensing@Micrium.com

WEB: www.Micrium.com