Micriμm

Empowering Embedded Systems

μC/LIB

V1.28

Release Notes

www.Micrium.com

Revision History

Version	Date	Description
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 1.28
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 1.28

V1.28-001
Added new configuration to (optionally) specify the heap memory base address :

`LIB_MEM_CFG_HEAP_BASE_ADDR`

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 :

`ASCII_IsDigOct()` **`ASCII_IS_DIG_OCT()`**	indicates whether a character is an octal digit

See also ‘New Features V1.25-002’.

V1.26-003
Added new string compare functions :

`Str_CmpIgnoreCase()`	compares two strings, ignoring case
`Str_CmpIgnoreCase_N()`	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 :

Str_FmtNbr_Int32U()	formats an unsigned number into a string
Str_FmtNbr_Int32S()	formats a signed number into a string

V1.26-004b
Added new string parse functions :

Str_ParseNbr_Int32U()	parses an unsigned number from a string
Str_ParseNbr_Int32S()	parses a signed number from a string

Version 1.25

V1.25-001
Added new memory allocation functions :

Mem_PoolCreate()	create a memory pool
Mem_PoolBlkGet()	get a memory block from a memory pool
Mem_PoolBlkFree()	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 :

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

Version 1.19

N/A

Version 1.18

N/A

Improvements

Version 1.28

V1.28-001

Replaced all '**cpu_sr**' local variable declarations with µC/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 1.28

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 :

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

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 1.28
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 1.28
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

Micriμm
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