µC/CPU™

CPU-Specific Ports

Release Notes V1.29.01

Micriµm

For the Way Engineers Work

Revision History

Version	Date	Description
V1.29.01	2012 Feb	Improvements
V1.29.00	2011 Aug	New features & improvements
V1.28.01	2011 Jun	Improvements
V1.28.00	2011 Feb	Re-released µC/CPU V1.28 core files as V1.28.00 & port files as V1.28.00.00
V1.28	2010 Dec	Bug fixes and improvements
V1.27	2010 Oct	New features & improvements
V1.26	2010 Apr	Improvements
V1.25.01	2010 Apr	Port updates only—NO changes to core files
V1.25	2010 Jan	Bug fixes and improvements
V1.24	2009 Dec	New features, bug fixes, & improvements
V1.23	2009 Jul	CPU timestamp, timer, & time measurement features First version with release history & user's manual

900-uC-CPU-007

Required Modules

Version 1.29.01

μC/LIB version 1.35.00

Version 1.29.00

μC/LIB version 1.35.00

Version 1.28.01

μC/LIB version 1.35.00

Version 1.28.00

μC/LIB version 1.34

New Features

Version 1.29.01

N/A

Version 1.29.00

V1.29.00-001

Added new core CPU functions:

`CPU_CntLeadZeros08()`	counts the number of contiguous, leading zero bits in an 8-bit value
`CPU_CntLeadZeros16()`	counts the number of contiguous, leading zero bits in a 16-bit value
`CPU_CntLeadZeros32()`	counts the number of contiguous, leading zero bits in a 32-bit value
`CPU_CntLeadZeros64()`	counts the number of contiguous, leading zero bits in a 64-bit value
`CPU_CntTrailZeros()`	counts the number of contiguous, trailing zero bits in a value
`CPU_CntTrailZeros08()`	counts the number of contiguous, trailing zero bits in an 8-bit value
`CPU_CntTrailZeros16()`	counts the number of contiguous, trailing zero bits in a 16-bit value
`CPU_CntTrailZeros32()`	counts the number of contiguous, trailing zero bits in a 32-bit value
`CPU_CntTrailZeros64()`	counts the number of contiguous, trailing zero bits in a 64-bit value

V1.29.00-002

Added `CPU_CFG_DATA_SIZE_MAX` to each **`cpu.h`** to define the maximum integer data size supported by the CPU/compiler.

Version 1.28.01

N/A

Version 1.28.00

N/A

Version 1.27

V1.27-001

Added `CPU_SW_EXCEPTION()` / `CPU_SW_Exception()` to trap on unrecoverable exceptions, primarily `NULL` pointers to return errors (a condition which cannot be returned via the `NULL` return pointer). See also 'Improvements V1.27-001a'.

Version 1.26

N/A

Version 1.25.01

N/A

Version 1.25

N/A

Version 1.24

V1.24-001

Added `CPU_STK_SIZE` data type definition to each `cpu.h`.

V1.24-002a

Added (optional) CPU timestamp's timer frequency, `CPU_TS_TmrFreq_Hz`.

V1.24-002b

Added new CPU timestamp timer functions:

`CPU_TS_TmrFreqGet()`	gets the CPU timestamp's timer frequency (in Hertz)
`CPU_TS_TmrFreqSet()`	sets the CPU timestamp's timer frequency (in Hertz)

See also 'New Features V1.23-001c'.

Version 1.23

V1.23-001

Added new CPU timestamp, timer, and time measurement features. (Note that an application must call `CPU_Init()` to initialize CPU timestamp or time measurement features prior to any other calls to CPU time functions.)

V1.23-001a

Added `CPU_CFG_TS_EN` in `cpu_cfg.h` to enable/disable CPU timestamps:

`CPU_TS_Get()`	gets the current, real-time value of 64-bit CPU timestamp, returned via two 32-bit values
`CPU_TS_GetLo()`	gets only the lower 32-bits of 64-bit timestamp
`CPU_TS_Update()`	updates the real-time value of 64-bit CPU timestamp [see 'New Features V1.23-001c `CPU_TS_TmrRd()`']

See also 'Changes V1.25-001a1 & V1.25-001c'.

V1.23-001b

Added `CPU_CFG_INT_DIS_MEAS_EN` & `CPU_CFG_INT_DIS_MEAS_OVRHD_NBR` in `cpu_cfg.h` to enable/disable measuring interrupts disabled times:

`CPU_IntDisMeasMaxGet()`	gets the maximum time interrupts are disabled, returned via a 32-bit timestamp value; this maximum value is non-resetable
`CPU_IntDisMeasMaxCurGet()`	gets the current maximum time interrupts are disabled, returned via a 32-bit timestamp value; this maximum value is resetable
`CPU_IntDisMeasMaxCurReset()`	resets the current maximum time interrupts are disabled

See also 'Changes V1.25-002'.

V1.23-001c

The following timer functions must be implemented in an application if either CPU timestamps *or* interrupts disabled time measurements are enabled:

`CPU_TS_TmrInit()`	initializes & starts a hardware (or software) timer to update CPU timestamps & time measurements
`CPU_TS_TmrRd()`	gets current hardware (or software) timer value to update CPU timestamps or time measurements
`CPU_TS_to_uSec()`	convert (up to) 64 bits of a CPU timestamp value into microseconds, returned via two 32-bit values

See also 'Changes V1.25-001d & V1.25-001e' & 'New Features V1.24-002b'.

Improvements

Version 1.29.01

V1.29.01-001
Updated µC/CPU's CERT-C and MISRA-C compliance:

V1.29.01-001a
Cast all ~ and << operands to appropriate integer data sizes (MISRA 2004 Rule 10.5).

Version 1.29.00

V1.29.00-001
Updated µC/CPU's CERT-C and MISRA-C compliance:

V1.29.00-001a
Removed '`u`' qualifier from certain integer constants that may be used in both signed and unsigned expressions (MISRA 2004 Rule 10.6). See also 'Improvements V1.28.01-001a, V1.28.00-001a & V1.24-001a1'.

V1.29.00-002
Modified `CPU_CntLeadZeros??()`'s `ix` data type from `CPU_INT08U` to `CPU_DATA` (see also 'New Features V1.29.00-001').

Version 1.28.01

V1.28.01-001
Updated µC/CPU's CERT-C and MISRA-C compliance:

V1.28.01-001a
Added '`u`' qualifier back to certain unsigned integer constants (MISRA 2004 Rule 10.6). This reverts the removal of all unsigned integer constants. See also 'Improvements V1.28.00-001a & V1.24-001a1'.

Version 1.28.00

V1.28.00-001
Updated µC/CPU's CERT-C and MISRA-C compliance:

V1.28.00-001a
Removed '`u`' qualifier from certain integer constants (MISRA 2004 Rule 10.6). This reverts a previously implemented improvement only for certain integer constants that may be used in both signed and unsigned expressions. See also 'Improvements V1.24-001a1'.

V1.28.00-001b
Added `const` modifier to all appropriate API function pointer arguments (MISRA 2004 Rule 16.7). See also 'Changes V1.28-001'.

Version 1.27

V1.27-001

Updated µC/CPU's CERT-C and MISRA-C compliance:

V1.27-001a

Added `CPU_SW_EXCEPTION()` / `CPU_SW_Exception()` to trap on unrecoverable exceptions, primarily `NULL` '`p_err`' pointers to return errors (a condition which cannot be returned via the `NULL` return pointer).

V1.27-001a1

Modified functions to trap `NULL` '`p_err`' pointers.

Version 1.26

V1.26-001

Updated µC/CPU's CERT-C and MISRA-C compliance:

V1.26-001a

Added argument names to function pointer data types (MISRA 2004 Rule 16.3).

V1.26-001b

Encapsulated all macros defined as code blocks within `do..while(0)` conditions (MISRA 2004 Rule 19.4).

Version 1.25.01

N/A

Version 1.25

V1.25-001a

Improved CPU timestamp API & performance. See also 'Changes V1.25-001'.

V1.25-002a

Refactored `CPU_CntLeadZeros()` to improve performance.

V1.25-002b

Added 64-bit support to `CPU_CntLeadZeros()`.

V1.25-003

Added 64-bit data types to most `cpu.h`'s.

Version 1.24

V1.24-001

Updated µC/CPU's CERT-C and MISRA-C compliance:

V1.24-001a1

Appended unsigned '`u`' qualifier to all unsigned integer constants (MISRA 2004 Rule 10.6).

V1.24-001a2

Removed redundant '`L`' qualifier from all long integer constants.

V1.24-001b

Replaced all calls to unbounded µC/LIB string library functions [e.g. `Str_Copy()`] with calls to bounded functions [e.g. `Str_Copy_N()`].

Version 1.23

V1.23-001

Added `CPU_CFG_MODULE_PRESENT` header guard to ensure `cpu_cfg.h` is processed only once, regardless if `#include`'d by multiple source or header files.

Changes

Version 1.29.01

N/A

Version 1.29.00

V1.29.00-001

Moved `CPU_CORE_VERSION` from `cpu_core.h` to `cpu_def.h`.

V1.29.00-002

Removed prototype for `CPU_CntLeadZeros()` from `cpu.h`'s, where applicable. `CPU_CntLeadZeros()` now prototyped only in `cpu_core.h`.

Version 1.28.01

V1.28.01-001a

Changed template `cpu_cfg.h`'s default `CPU_CFG_NAME_EN` configuration from `DEF_ENABLED` to `DEF_DISABLED`.

V1.28.01-001b

Modified `cpu_core.h` to not include µC/LIB's memory or string header files unless `CPU_CFG_NAME_EN` is configured as `DEF_ENABLED` in `cpu_cfg.h`.

Version 1.28.00

V1.28.00-001

Added `const` modifier to all appropriate pointer arguments in `CPU_NameSet()`.

Version 1.27

N/A

Version 1.26

N/A

Version 1.25.01

V1.25.01-001a

Renamed `\Micrium\Software\uC-CPU\Win32\Microsoft` directory to `\Micrium\Software\uC-CPU\Win32\Visual Studio`.

V1.25.01-001b

Refactored `\Micrium\Software\uC-CPU\Win32\Visual Studio` port files' critical section initialization & implementation.

Version 1.25

V1.25-001

Refactored CPU timestamps configuration, API, & implementation to improve performance (see also 'µC/CPU's User's Manual Section 3.03'):

V1.25-001a1

Replaced `cpu_cfg.h` configuration constant `CPU_CFG_TS_EN` with new configuration constants:

`CPU_CFG_TS_32_EN`	enables 32-bit CPU timestamps
`CPU_CFG_TS_64_EN`	enables 64-bit CPU timestamps

V1.25-001a2

Added `cpu_cfg.h` configuration constant `CPU_CFG_TS_TMR_SIZE` to configure the word size of the CPU timestamp's hardware (or software) timer.

V1.25-001b1

Replaced `CPU_TS` data type with new CPU timestamp data types:

`CPU_TS32`	handles 32-bit CPU timestamps
`CPU_TS64`	handles 64-bit CPU timestamps

V1.25-001b2

Added `CPU_TS_TMR` data type to handle CPU timestamp timer values instead of `CPU_TS`.

V1.25-001c

Replaced `CPU_TS_Get()` & `CPU_TS_GetLo()` with new CPU timestamp functions:

`CPU_TS_Get32()`	gets 32-bit CPU timestamp
`CPU_TS_Get64()`	gets 64-bit CPU timestamp

V1.25-001d

Modified developer-defined CPU timestamp timer function prototypes:

```
void        CPU_TS_TmrInit(void);
CPU_TS_TMR  CPU_TS_TmrRd  (void);
```

V1.25-001e

Replaced (optional) developer-defined `CPU_TS_to_uSec()` with new CPU timestamp functions:

`CPU_TS32_to_uSec()`	converts 32-bit CPU timestamp to microseconds
`CPU_TS64_to_uSec()`	converts 64-bit CPU timestamp to microseconds

V1.25-002

Modified CPU interrupts disabled time measurement function prototypes:

```
CPU_TS_TMR  CPU_IntDisMeasMaxCurReset(void);
CPU_TS_TMR  CPU_IntDisMeasMaxCurGet  (void);
CPU_TS_TMR  CPU_IntDisMeasMaxGet     (void);
```

Version 1.24

N/A

Version 1.23

V1.23-001

Moved `CPU_ERR` data type definition from each `cpu_cfg.h` to `cpu_core.h`.

Corrections

Version 1.29.01

N/A

Version 1.29.00

N/A

Version 1.28.01

N/A

Version 1.28

N/A

Version 1.27

N/A

Version 1.26

N/A

Version 1.25.01

N/A

Version 1.25

V1.25-001

Previous `CPU_TS_Get()` failed to re-entrantly calculate the current CPU timestamp since the current CPU timestamp timer was read [via a call to `CPU_TS_TmrRd()`] with interrupts enabled but saved for the next timestamp calculation with interrupts disabled. Fixed in `CPU_TS_Get32()` & `CPU_TS_Get64()` [see 'Changes V1.25-001c'] by calling `CPU_TS_TmrRd()` with interrupts disabled.

Version 1.24

N/A

Version 1.23

N/A

Known Problems

Version 1.29.01

Version 1.29.00

Version 1.28.01

Version 1.28.00

Version 1.27

Version 1.26

Version 1.25.01

Version 1.25

Version 1.24

Version 1.23

N/A

Limitations

001

Support for 64-bit address/data not available for some CPUs

Contacts

Micriμm
1290 Weston Road, Suite 306
Weston, FL 33326
USA

Phone:	+1 954 217 2036
Fax:	+1 954 217 2037
E-mail:	Licensing@Micrium.com
Web:	www.Micrium.com