Micriµm, Inc.

# µC/OS-II

## Directory Structure

## Application Note

AN-2002

**Jean J. Labrosse**
Jean.Labrosse@Micrium.com
www.Micrium.com

## 1.00 Introduction

This document describes a new directory structure used by Micriμm for storing μC/OS-II ports. The previous directory structure was becoming messy because of the large number of files being contributed for μC/OS-II ports. As a minimum, this document should address the question:

> "Where do I place this μC/OS-II port that I just did or that I'm planning on doing?"

It is difficult to plan for all the possible combinations of processors, compilers, evaluation boards and so on. This document attempts to give you the information necessary to allow you to make a reasonable choice.

Please note that you are free to place files for your application(s) anywhere you would like and, in fact, you might want to copy files from the μC/OS-II distributions (processor independent source and ports) in other directories to better suit your project architecture. This directory structure is just meant as guideline for people who submit ports.

It is assumed that files are placed in a hierarchical file system.

The convention for directory names is to use 'camelback' and, use Acronyms, Abbreviations and Mnemonics as needed. For example, the 'Application Notes' directory would be called 'AppNotes'. Note how the first letter of each word starts with an upper case character.

## 2.00 Top Level Directories

Micriµm offers many different products and, as we continue to add more products we do not want to fill up the ‘root’ directory of your file system, all Micriµm related files will thus be placed in the `\Micrium` directory under the root. Note that for Unix or Linux based systems, you should replace the ‘\’ with a ‘/’.

Under the `\Micrium` directory, we currently have the following subdirectories:

```
\Micrium
   \AppNotes
   \Software
```

`\AppNotes`

This directory contains application notes written by Micriµm personnel or contributed by users. Application notes are assigned a unique ‘AN-XXXX’ number when published.

`\Software`

Most of Micriµm’s products are software related and would most likely end-up under this directory (more on this shortly). The `\Software` directory is also there for historical reasons.

The current software products are stored as follows:

```
\Micrium
   \AppNotes
   \Software
      \uC-FS
      \uC-GUI
      \uCOS-II
      \uCOS-II-KA
      \uCOSView
```

## 3.00 µC/OS-II Directories

The main directories in µC/OS-II are shown below.

```
\Micrium
   \Software
      \uCOS-II
         \Doc
         \Ports
         \Source
```

`\Doc`
Contains documentation of the processor independent code (i.e. µC/OS-II itself). Specifically, this directory contains `README.TXT`, µC/OS-II release notes and updates on the reference and configuration manuals.

`\Ports`
The `\Ports` directory is where all the ports will be placed. This is where the complexity of the directory structure starts. The reason is that there are hundreds of combinations of processors, compilers and evaluation boards that can work with µC/OS-II.

`\Source`
This directory contains the processor independent source files: `os_core.c`, `os_flag.c`, `os_sem.c`, `os_mutex.c`, `os_mbox.c`, `os_q.c`, `os_time.c`, `os_task.c`, `os_mem.c`, `ucos_ii.c` and `ucos_ii.h`.

In V2.70, we introduced two additional files: `os_cfg_r.h` and `os_dbg_r.c`:

| | |
|---|---|
| `os_cfg_r.h` | is a 'reference' file for `os_cfg.h`. In other words, you can use this file as a starting point for your own os_cfg.h and thus, you can simply copy `os_cfg_r.h` to the `os_cfg.h` file you will use in your project. The reason this file is provided is to help you get started with a new project or with an existing one because a new µC/OS-II version might introduce new `#define` constants. |
| `os_dbg_r.c` | is a 'reference' file for `os_dbg.c`. In other words, you can use this file as a starting point for your own `os_dbg.c` (if needed) and thus, you can simply copy `os_dbg_r.c` to the `os_dbg.c` file you will use in your project. The reason this file is provided is to help you get started with a new project or with an existing one because a new µC/OS-II version might introduce new code in `os_dbg.c`. |

## 3.01 µC/OS-II Ports Directory

Probably the most complicated directory structure to manage is that of µC/OS-II's ports. Managing this directory structure is difficult because of the large number of processors, compilers and evaluation boards that supports µC/OS-II. The structure described in this section should take care of current and future expansion. Below is a small list of CPU architectures that are capable of running µC/OS-II. The company name (i.e. chip manufacturer) listed to the right are not part of the directory name.

```
\Micrium
   \Software
      \uCOS-II
         \Ports            Manufacturer
            \680x0         Motorola
            \683xx         Motorola
            \68HC08        Motorola
            \68HC11        Motorola
            \68HC12        Motorola
            \68HC16        Motorola
            \78K           NEC
            \80196         Intel
            \80296         Intel
            \8051          Intel and many other sources
            \80C16x        Infineon
            \80x86         Intel, AMD and others
            \AD21xx        Analog Devices
            \ARM7          ARM7 licensees
            \AVR           Atmel
            \ColdFire      Motorola
            \CR16          National Semiconductor
            \DSP56F8xx     Motorola
            \eZ80          ZiLOG
            \FFMC-16       Fujitsu
            \H8            Hitachi
            \M16C          Mitsubishi
            \M32C          Mitsubishi
            \M7700         Mitsubishi
            \MCore         Motorola
            \MicroBlaze    Xilinx
            \MIPS          MIPS
            \MSP430        Texas Instruments
            \NIOS          Altera
            \PowerPC       Motorola and IBM
            \Rabbit        Rabbit Semiconductors
            \SH            Hitachi
            \ST72          ST Microelectronics
            \TriCore       Infineon
            \TMS           Texas Instruments
            \V8            VA Automation
            \V850          NEC
            \XA            Philips
            \Z180          ZiLOG and Hitachi 64180
            \Z80           ZiLOG
            \ZSP           LSI Logic
```

## IMPORTANT

Note that ports for all the different combinations of processors, operating modes and compilers are not currently available and our point is to simply present a directory structure that would take care of all the possibilities.  In fact, if you have a port for a processor (or combination) that doesn't exist, we would recommend that you follow this directory scheme.

## 3.02 Port Family Directories

As you are probably aware, there are generally a number of different chips manufactured for any specific architecture. Some of these chips have exactly the same register model and instruction set but others might not. Let's take for example the Hitachi H8 architecture. Hitachi introduced a number of different sub-families as shown below. Because of architectural differences, the actual µC/OS-II port might be different for each of these sub-families and thus, each should have its own directory branch.

```
\Micrium
   \Software
      \uCOS-II
         \Ports
            \H8
               \H8-300
               \H8-300H
               \H8-300L
               \H8S-21xx
               \H8S-22xx
               \H8S-23xx
               \H8S-26xx
               \H8-Tiny
```

## 3.03 Port Mode Directories

For the H8S family, Hitachi allows you to run the processor in two different mode: *Normal* and *Advanced*. In Normal mode, the CPU can only address 64K bytes of memory while in Advanced mode, the CPU may use up to 24 address bits. The port for μC/OS-II for each mode is different and thus, we need to account for this as shown below:

```
\Micrium
   \Software
      \uCOS-II
         \Ports
            \H8
               \H8-300
               \H8-300H
               \H8-300L
               \H8S-21xx
                  \Adv
                  \Norm
               \H8S-22xx
                  \Adv
                  \Norm
               \H8S-23xx
                  \Adv
                  \Norm
               \H8S-26xx
                  \Adv
                  \Norm
               \H8-Tiny
```

## 3.04 Port Compiler Directories

There are a number of compilers for the H8 architecture. For example, GNU's GCC, Hitachi's HEW and IAR's EW. There might be a port for each of the different compiler for each of the modes and thus, the directory tree would look as follows:

```
\Micrium
   \Software
      \uCOS-II
         \Ports
            \H8
               \H8-300
                  \GNU
                  \HEW
                  \IAR
               \H8-300H
                  \GNU
                  \HEW
                  \IAR
               \H8-300L
                  \GNU
                  \HEW
                  \IAR
               \H8S-21xx
                  \Adv
                     \GNU
                     \HEW
                     \IAR
                  \Norm
                     \GNU
                     \HEW
                     \IAR
               \H8S-22xx
                  \Adv
                     \GNU
                     \HEW
                     \IAR
                  \Norm
                     \GNU
                     \HEW
                     \IAR
               \H8S-23xx
                  \Adv
                     \GNU
                     \HEW
                     \IAR
                  \Norm
                     \GNU
                     \HEW
                     \IAR
               \H8S-26xx
                  \Adv
                     \GNU
                     \HEW
                     \IAR
                  \Norm
                     \GNU
                     \HEW
                     \IAR
               \H8-Tiny
                  \GNU
                  \HEW
                  \IAR
```

## 3.05 Port Directory Files

In the above directories you will find the following files: `OS_CPU_A.ASM`, `OS_CPU_C.C`, `OS_DBG.C` and `OS_CPU.H` as shown below. Note that the `.ASM` extension depends on the actual compiler – it could be `.S` or something else.

```
\Micrium
   \Software
      \uCOS-II
         \Ports
            \H8
               \H8S-26xx
                  \Adv
                     \GNU
                        os_cpu_a.asm
                        os_cpu_c.c
                        os_cpu.h
                        os_dbg.c
                     \HEW
                        os_cpu_a.asm
                        os_cpu_c.c
                        os_cpu.h
                        os_dbg.c
                     \IAR
                        os_cpu_a.asm
                        os_cpu_c.c
                        os_cpu.h
                        os_dbg.c
                  \Norm
                     \GNU
                        os_cpu_a.asm
                        os_cpu_c.c
                        os_cpu.h
                        os_dbg.c
                     \HEW
                        os_cpu_a.asm
                        os_cpu_c.c
                        os_cpu.h
                        os_dbg.c
                     \IAR
                        os_cpu_a.asm
                        os_cpu_c.c
                        os_cpu.h
                        os_dbg.c
```

## 3.06 Port Submissions

Note that there might be different ports submitted by different people. In those cases, it is recommended to place those ports in a sub-directory as follows:

```
\Micrium
   \Software
      \uCOS-II
         \Ports
            \H8
               \H8S-26xx
                  \Adv
                     \IAR
                        os_cpu_a.asm      Micriµm's port
                        os_cpu_c.c
                        os_cpu.h
                        os_dbg.c
                        \YourPort         Other ports
                        \MyNewPort
                        \<Company>_<Individual>
```

As shown above, Micriµm's port should be placed at the root of that compiler and would indicate a Micriµm supported port. When possible, other non-Micrium ports should be labeled with a "personalized" directory indicating the company or individual who created the port.

When you submit a port, please add a README.TXT file describing details about the port:

- Which compiler was used and what version
- What options you selected
- What evaluation board it was tested on
- Contact information (e-mail) in case other users have questions
- Etc.

## 3.07 Port Test Directories

Now the question is “where do we place the test code for the different evaluation boards?”. Since the code for an evaluation board will most likely be built using a specific compiler then it would make sense to associate it accordingly. The following directories would seem to make sense.

The `\EVB2674R-1` directory contains example #1 of some test code on an evaluation board called EVB2674R (the name was chosen as an example and may not be a real board). For μC/OS-II, you need an `INCLUDES.H` and an `OS_CFG.H` to configure the OS options that you will use with the test code. TEST.PEW is the IAR compiler project file that describes how the different files will be assembled, compiled and linked to form an executable image that is then downloaded to the target board.

Note that I also showed another example. In this case, the contents of the files in the directory would most likely be different even though they have the same name.

```
\Micrium
   \Software
      \uCOS-II
         \Ports
            \H8
               \H8S-26xx
                  \Adv
                     \IAR
                        os_cpu_a.asm
                        os_cpu_c.c
                        os_cpu.h
                        os_dbg.c
                  \EVB2674R-Ex1
                     \IAR
                        includes.h
                        os_cfg.h
                        test.c
                        test.pew
                  \EVB2674R-Ex2
                     \IAR
                        includes.h
                        os_cfg.h
                        test.c
                        test.pew
```

## 4.00 µC/OS-II Distribution Directories

The directory tree for the µC/OS-II distribution has changed as of release V2.70 and now looks as shown below. Comments have been added where appropriate.

Note that a µC/OS-II release now consist of **only** the processor independent source code since all the ports (including the 80x86 ports that used to be provided with the µC/OS-II distribution) are available from the Micriµm web site: www.Micrium.com.

There are two reasons why we did this:

1) µC/OS-II is highly processor independent and really doesn't need to be distributed with any specific processor port. We also wanted to remove the 'association' of the Intel 80x86 from µC/OS-II because of the large number of ports available from our web site.
2) This makes the distribution slightly smaller for downloads.

```
\Micrium
   \Software
      \uCOS-II
         \Doc                        Release notes and manuals
            readme.txt
            TaskAssignmentWorksheet.pdf
            TaskAssignmentWorksheet.xls
            QuickRefChart-Color.pdf
            ReleaseNotes.pdf
            uCOS-II-CfgMan.pdf
            uCOS-II-RefMan.pdf
            WhatsNewSince-V200.pdf
         \Source                     Processor independent source
            os_cfg_r.h               Example of 'os_cfg.h' file
            os_core.c
            os_dbg_r.c               Example of 'os_dbg.c' file
            os_flag.c
            os_mbox.c
            os_mem.c
            os_mutex.c
            os_q.c
            os_sem.c
            os_task.c
            os_time.c
            ucos_ii.c
            ucos_ii.h
```

**µC/OS-II Code Release Directory Tree for V2.70 and higher**

Note that the new distribution contains `os_cfg_r.h` which is a 'reference' version of `os_cfg.h` that you need in your project build to configure µC/OS-II . In other words, you would copy `os_cfg_r.h` (from the `Source` directory) to `os_cfg.h` in you target directory. You could then modify `os_cfg.h` to configure µC/OS-II according to your product requirements. This is done to ensure that you always have ALL the `#define` constants whenever a new version of µC/OS-II adds new `#define` constants.

The new distribution also contains `os_dbg.c` which is a 'reference' version of `os_dbg.c` that you need in your project build if you plan on using kernel awareness debuggers supported by Micriμm. In other words, you would copy `os_dbg_r.c` from the `Source` directory to `os_dbg.c` in you target directory. You could then modify `os_dbg.c` according to how the compiler treats initialized constants. This is done to ensure that you always have ALL the entries in `os_dbg.c` needed whenever a new version of μC/OS-II adds new debug variables.

The new distribution will no longer include the DOS utility `TO.EXE` because this is no longer relevant.

## 4.01 µC/OS-II 80x86 Directories

The directory tree below shows, as an example, the directory structure of the 80x86 port files for a DOS environment. The 80x86 port files are no longer distributed with the processor independent code for µC/OS-II but can be downloaded for free from www.Micrium.com (follow the links to µC/OS-II and its ports).

```
\Micrium
   \Software
      \Blocks
      |  \DOS
      |     \L                              Large Model
      |        \BC45                        Borland V4.5x compiler
      |           pc.c
                  pc.h
      \uCOS-II
         \Ports
            \80x86
               \DOS                         DOS Operating System
                  \L                        Large Model
                     \BC45                  Borland V4.5x compiler
                  |     os_cpu_a.asm
                  |     os_cpu_c.c
                  |     os_cpu.h
                  |     os_dbg.c
                  |     \Ex1                Was previously Ex1_x86L
                  |        includes.h
                  |        test.c
                  |        os_cfg.h
                  |        test.mak
                  |        maketest.bat
                  |        test.exe
                  |        test.map
                  |        test.rsp         Compiler command file for Borland V4.5x
                  |     \Ex2                Was previously Ex2_x86L
                  |        includes.h
                  |        test.c
                  |        os_cfg.h
                  |        test.mak
                  |        maketest.bat
                  |        test.exe
                  |        test.map
                  |        test.rsp
                  |     \Ex3                Was previously Ex3_x86L
                  |        includes.h
                  |        test.c
                  |        os_cfg.h
                  |        test.mak
                  |        maketest.bat
                  |        test.exe
                  |        test.map
                           test.rsp
                  \L-FP                     Large Model with Floating-Point support
                     \BC45
                        os_cpu_a.asm
                        os_cpu_c.c
                        os_cpu.h
                        os_dbg.c
                        \Ex1                Was previously Ex4_x86L.FP
                           includes.h
                           test.c
                           os_cfg.h
                           test.mak
                           maketest.bat
                           test.exe
                           test.map
                           test.rsp
```

## μC/OS-II 80x86 Release Directory Tree for V2.70 and higher

## 4.02 µC/OS-II ARM Directories

The directory tree below shows, as an example, the directory structure of the ARM port files. This is shown as an example and can be thus used as a starting point.

```
\Micrium
   \Software
      \uCOS-II
         \Ports
            \ARM7                          ARM7 and derivatives
               \AT91                       Atmel ARM7
                  \ARM                     ARM mode
                     \IAR                  IAR EW ARM Tools
                        os_cpu_a.asm
                        os_cpu_c.c
                        os_cpu.h
                        os_dbg.c
                        \EB40A-Ex1         Example #1 running on the EB40A eval. board
                           includes.h
                           test.c
                           os_cfg.h
                           EB40A-Ex1.pew
                           AT91_Lnk_8000.xcl
                  \Thumb                   Thumb mode
                     \IAR                  IAR EW ARM Tools
                        os_cpu_a.asm
                        os_cpu_c.c
                        os_cpu.h
                        os_dbg.c
                        \EB40A-Ex1         Example #1 running on the EB40A eval. board
                           includes.h
                           test.c
                           os_cfg.h
                           EB40A-Ex1.pew
                           AT91_Lnk_8000.xcl
                  \ARM-Thumb               ARM and Thumb modes
                     \IAR                  IAR EW ARM Tools
                        os_cpu_a.asm
                        os_cpu_c.c
                        os_cpu.h
                        os_dbg.c
                        \EB40A-Ex1         Example #1 running on the EB40A eval. board
                           includes.h
                           test.c
                           os_cfg.h
                           EB40A-Ex1.pew
                           AT91_Lnk_8000.xcl
               \LH79520                    Sharp ARM7
                  \ARM                     ARM mode
                     \IAR                  IAR EW ARM Tools
                        os_cpu_a.asm
                        os_cpu_c.c
                        os_cpu.h
                        os_dbg.c
                        \Micrium-Ex1       Example #1 running on the Micrium eval. board
                           includes.h
                           test.c
                           os_cfg.h
                           Micrium-Ex1.pew
                           Micrium_Lnk_8000.xcl
```

**µC/OS-II ARM Directory Tree**

## 4.03 μC/OS-II Motorola 68HC12 Directories

The directory tree below shows the directory structure of the Motorola 68HC12 port files. This is shown as an example and can be thus used as a starting point.

```
\Micrium
   \Software
      \uCOS-II
         \Ports
            \68HC12                     Motorola 68HC12 family of processors
               \DG128A                  DG128A processor
                  \NonPaged             Uses only 64K address space
                     \COSMIC            COSMIC compiler
                        os_cpu_a.asm
                        os_cpu_c.c
                        os_cpu.h
                        os_dbg.c
                        \Ex1
                           includes.h
                           test.c
                           os_cfg.h
                           maketest.bat
                           start.s
                           test.mak
                           vectors.c
                     \IAR               IAR compiler
                        os_cpu_a.asm
                        os_cpu_c.c
                        os_cpu.h
                        os_dbg.c
                        \Ex1
                           includes.h
                           test.c
                           os_cfg.h
                  \Paged
                     \COSMIC
                        os_cpu_a.asm
                        os_cpu_c.c
                        os_cpu.h
                        os_dbg.c
                        \Ex1
                           includes.h
                           test.c
                           os_cfg.h
                           maketest.bat
                           start.s
                           test.mak
                           vectors.c
                     \IAR
                        os_cpu_a.asm
                        os_cpu_c.c
                        os_cpu.h
                        os_dbg.c
                        \Ex1
                           includes.h
                           test.c
                           os_cfg.h
```

**μC/OS-II Motorola M68HC12 Directory Tree**

## References

***μC/OS-II, The Real-Time Kernel, 2nd Edition***
Jean J. Labrosse
R&D Technical Books, 2002
ISBN 1-57820-103-9

***Embedded Systems Building Blocks***
Jean J. Labrosse
R&D Technical Books, 2000
ISBN 0-87930-604-1

## Contacts

Micriμm, Inc.
949 Crestview Circle
Weston, FL 33327
954-217-2036
954-217-2037 (FAX)
e-mail: Jean.Labrosse@Micrium.com
WEB: www.Micrium.com

R&D Books, Inc.
1601 W. 23rd St., Suite 200
Lawrence, KS 66046-9950
(785) 841-1631
(785) 841-2624 (FAX)
WEB: http://www.rdbooks.com
e-mail: rdorders@rdbooks.com