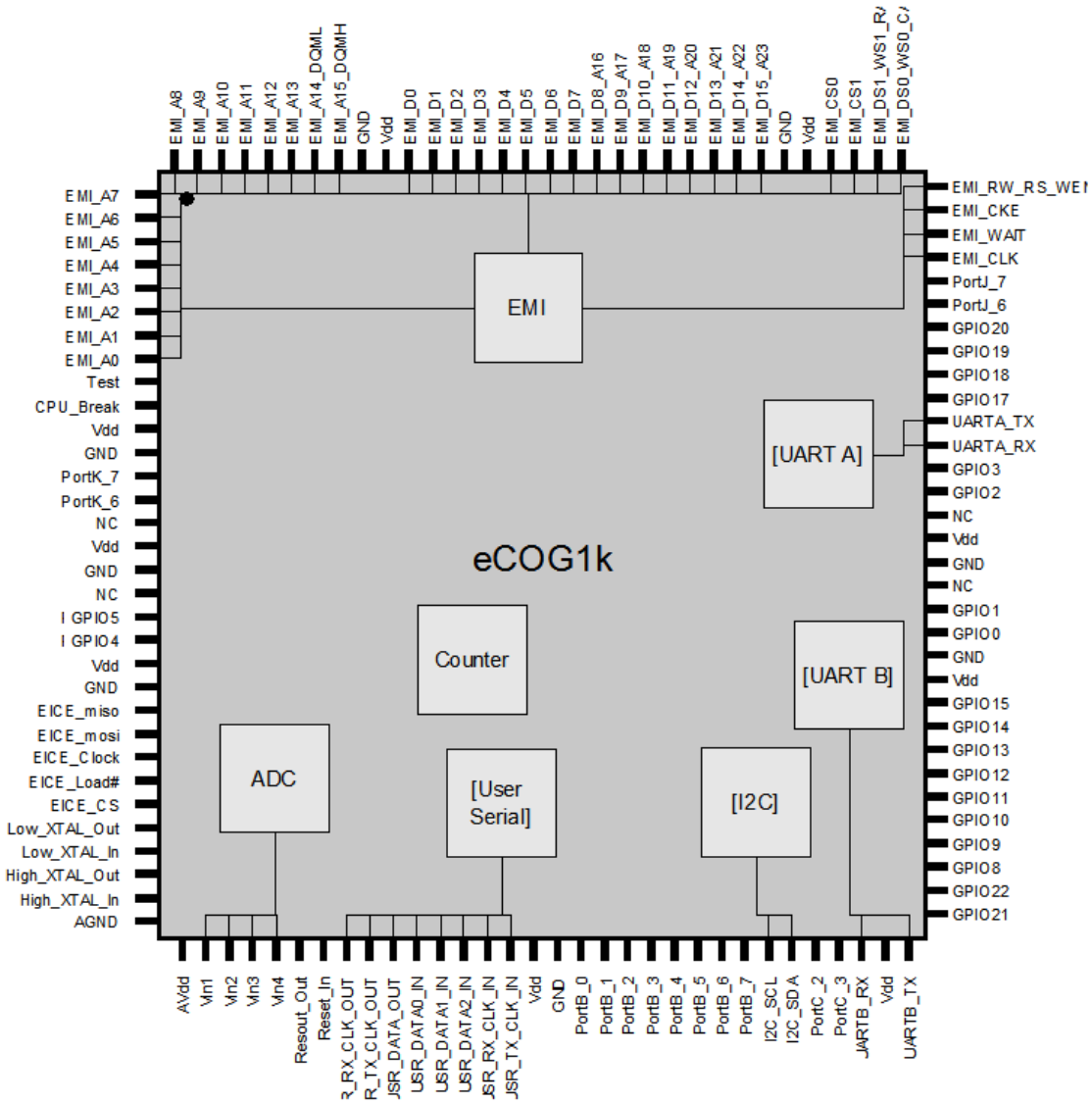
AN039 – Installing and Using µC/OS-II

Version 1.0

µC/OS-II is a highly portable, ROMable, scalable, pre-emptive real-time, multitasking kernel (RTOS) with support for the eCOG1k micro processor.

This application note serves as a description of the compilation and installation method for µC/OS-II on the eCOG1k development board.



Confidential and Proprietary Information

Revision History

Version	Date	Notes
V1.0	05/05/2006	Initial release

Contents

1 Introduction

µC/OS-II is a highly portable, ROMable, scalable, pre-emptive real-time, multitasking kernel (RTOS) with support for the eCOG1k micro processor.

This application note serves as a description of the compilation and installation method for µC/OS-II. Additionally, this application note is intended to serve as a guide for µC/OS-II on the eCOG1k development board, and to verify correct compilation, installation and operation through the use of simple test code. The software is compiled and downloaded into the eCOG1k on the development board using the tools provided in the CyanIDE software development package.

The majority of the instructions specific to this application are directly applicable to other applications and form a basic method for compiling code and downloading it to the eCOG1 micro controller.

2 Glossary

A table of abbreviations used in this document.

eCOG1k	Cyan Technology target micro controller
ISR	Interrupt Service Routine
LCD	Liquid Crystal Display
µC/OS-II	Micrium operating system ported to the eCOG1k
ROM	Read-Only Memory
RTOS	Real-Time Operating System

3 Requirements

- PC running CyanIDE V1.3 or later.
- eCOG1k development board, connected to the PC with a download cable.
- Licensed copy of µC/OS-II, version 2.82.

It is assumed that the reader is familiar with the µC/OS-II interfaces and their usage. For operating system related questions, the reader should refer to the operating system reference manual:

```
MicroC/OS-II: The Real-Time Kernel (Second Edition)
Jean J. Labrosse
CMP Books
ISBN 1-57820-103-9
```

4 µC/OS-II Installation

The µC/OS-II operating system software is neither freeware nor Open Source software. A licensed copy of version 2.82 of the operating system should be obtained directly from Micrium, http://www.micrium.com/. Delivery of the source code should take the form of a .ZIP file, which should be extracted to the CyanIDE install directory. Normally this gives the following directory structure:

```
...\CyanIDE\Micrium
...\CyanIDE\Micrium\Software
...\CyanIDE\Micrium\Software\uCOS-II
...\CyanIDE\Micrium\Software\uCOS-II\Source
...\CyanIDE\Micrium\Software\uCOS-II\Source\os_cfg_r.h
...\CyanIDE\Micrium\Software\uCOS-II\Source\os_core.c
...\CyanIDE\Micrium\Software\uCOS-II\Source\os_flag.c
...\CyanIDE\Micrium\Software\uCOS-II\Source\os_mbox.c
...\CyanIDE\Micrium\Software\uCOS-II\Source\os_mem.c
...\CyanIDE\Micrium\Software\uCOS-II\Source\os_q.c
...\CyanIDE\Micrium\Software\uCOS-II\Source\os_sem.c
...\CyanIDE\Micrium\Software\uCOS-II\Source\os_task.c
...\CyanIDE\Micrium\Software\uCOS-II\Source\os_time.c
...\CyanIDE\Micrium\Software\uCOS-II\Source\ucos_ii.c
...\CyanIDE\Micrium\Software\uCOS-II\Source\os_dbg_r.c
...\CyanIDE\Micrium\Software\uCOS-II\Source\os_mutex.c
...\CyanIDE\Micrium\Software\uCOS-II\Source\os_tmr.c
...\CyanIDE\Micrium\Software\uCOS-II\Source\ucos_ii.h
...\CyanIDE\Micrium\Software\uCOS-II\Doc
...\CyanIDE\Micrium\Software\uCOS-II\Doc\uCOS-II-CfgMan.PDF
...\CyanIDE\Micrium\Software\uCOS-II\Doc\TaskAssignmentWorksheet.PDF
...\CyanIDE\Micrium\Software\uCOS-II\Doc\QuickRefChart-Color.PDF
...\CyanIDE\Micrium\Software\uCOS-II\Doc\QuickRefChart-Color.xls
...\CyanIDE\Micrium\Software\uCOS-II\Doc\README.TXT
...\CyanIDE\Micrium\Software\uCOS-II\Doc\ReleaseNotes.PDF
...\CyanIDE\Micrium\Software\uCOS-II\Doc\TaskAssignmentWorksheet.XLS
...\CyanIDE\Micrium\Software\uCOS-II\Doc\uCOS-II-RefMan.PDF
...\CyanIDE\Micrium\Software\uCOS-II\Doc\WhatsNewSince-V200.PDF
...\CyanIDE\Micrium\Software\uCOS-II\Doc\uCOS-II-RAM-Calc.xls
```

These files comprise the architecture-independent files for µC/OS-II (not associated with any specific microprocessor). The architecture-specific files for µC/OS-II (associated with the eCOG1k microcontroller), together with the CyanIDE example applications and project templates, can be obtained in two ways:

- By downloading "AN039SW.zip" and extracting it to the CyanIDE install directory.
- By selecting the µC/OS-II component when installing CyanIDE.

Regardless of the method chosen, normally this gives the following additional directory structure:

```
...\CyanIDE\Micrium
...\CyanIDE\Micrium\Software
...\CyanIDE\Micrium\Software\uCOS-II
...\CyanIDE\Micrium\Software\uCOS-II\Ports
...\CyanIDE\Micrium\Software\uCOS-II\Ports\CyanTechnology
...\CyanIDE\Micrium\Software\uCOS-II\Ports\CyanTechnology\eCOG1k
...\CyanIDE\Micrium\Software\uCOS-II\Ports\CyanTechnology\eCOG1k\NCC
...\CyanIDE\Micrium\Software\uCOS-II\Ports\CyanTechnology\eCOG1k\NCC\os_cpu_a.asm
...\CyanIDE\Micrium\Software\uCOS-II\Ports\CyanTechnology\eCOG1k\NCC\os_cpu_c.c
...\CyanIDE\Micrium\Software\uCOS-II\Ports\CyanTechnology\eCOG1k\NCC\os_cpu.h
...\CyanIDE\Micrium\Software\uCOS-II\Ports\CyanTechnology\eCOG1k\NCC\os_dbg.c
...\CyanIDE\examples\eCOG1 dev board\ucos-ii\umatrix\umatrix.cyp
...\CyanIDE\examples\eCOG1 dev board\ucos-ii\umatrix\cstartup.asm
...\CyanIDE\examples\eCOG1 dev board\ucos-ii\umatrix\devboard.cfg
...\CyanIDE\examples\eCOG1 dev board\ucos-ii\umatrix\app_cfg.h
...\CyanIDE\examples\eCOG1 dev board\ucos-ii\umatrix\os_cfg.h
...\CyanIDE\examples\eCOG1 dev board\ucos-ii\umatrix\internal.map
...\CyanIDE\examples\eCOG1 dev board\ucos-ii\umatrix\irq.asm
...\CyanIDE\examples\eCOG1 dev board\ucos-ii\umatrix\umatrix.c
...\CyanIDE\examples\eCOG1 dev board\ucos-ii\umatrix\cpu_hooks.c
```

```
...\CyanIDE\examples\eCOG1 dev board\ucos-ii\umatrix\isr.asm
...\CyanIDE\examples\eCOG1 dev board v2\ucos-ii\umatrix\umatrix.cyp
...\CyanIDE\examples\eCOG1 dev board v2\ucos-ii\umatrix\cstartup.asm
...\CyanIDE\examples\eCOG1 dev board v2\ucos-ii\umatrix\devboard.cfg
...\CyanIDE\examples\eCOG1 dev board v2\ucos-ii\umatrix\app_cfg.h
...\CyanIDE\examples\eCOG1 dev board v2\ucos-ii\umatrix\os_cfg.h
...\CyanIDE\examples\eCOG1 dev board v2\ucos-ii\umatrix\internal.map
...\CyanIDE\examples\eCOG1 dev board v2\ucos-ii\umatrix\irq.asm
...\CyanIDE\examples\eCOG1 dev board v2\ucos-ii\umatrix\umatrix.c
...\CyanIDE\examples\eCOG1 dev board v2\ucos-ii\umatrix\cpu_hooks.c
...\CyanIDE\examples\eCOG1 dev board v2\ucos-ii\umatrix\isr.asm
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board Project\
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board Project\app_cfg.h
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board Project\cpu_hooks.c
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board Project\cstartup.asm
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board Project\devboard.cfg
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board Project\internal.map
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board Project\irq.asm
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board Project\main.c
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board Project\os_cfg.h
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board Project\project.cyp
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board Project\template.xml
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board V2 Project\
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board V2 Project\app_cfg.h
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board V2 Project\cpu_hooks.c
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board V2 Project\cstartup.asm
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board V2 Project\devboard.cfg
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board V2 Project\internal.map
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board V2 Project\irq.asm
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board V2 Project\main.c
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board V2 Project\os_cfg.h
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board V2 Project\project.cyp
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Dev Board V2 Project\template.xml
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Eval Board Project\
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Eval Board Project\app_cfg.h
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Eval Board Project\cpu_hooks.c
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Eval Board Project\cstartup.asm
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Eval Board Project\devboard.cfg
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Eval Board Project\internal.map
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Eval Board Project\irq.asm
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Eval Board Project\main.c
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Eval Board Project\os_cfg.h
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Eval Board Project\project.cyp
...\CyanIDE\templates\eCOG1MicroProjects\uCOS-II Eval Board Project\template.xml
```

5 Example Application: µMatrix

5.1 Installation

The example project is bundled together with the architecture specific files for µC/OS-II, which should already have been installed.

5.2 Purpose and Method

The µMatrix example application exists primarily to demonstrate a complete application for µC/OS-II. The eCOG1k µC/OS-II features showcased include:

- Multi-tasking, pre-emptive operation (including priority inversion cases).
- Critical sections (for timing-sensitive access to hardware).
- Memory management.
- Mutices (for arbitrating access to shared data).
- Semaphores (for task->task and ISR->task communication).
- Time delays.
- Interrupt muxing/demuxing with C ISRs.
- Power saving.

µMatrix scrolls digits across the LCD from left to right at a configurable rate, controlled by variable resistor VR1. To accomplish this, the application uses shared data, three tasks and three interrupt routines.

5.2.1 Shared Data

A single display data structure is shared by all three tasks. It contains:

- Information required for refreshing the LCD display.
- Mutex to moderate access to the above information.
- Scroll semaphore for communication between counter CNT2 ISR and the scroll task.
- Update semaphore for communication between the scroll and new character tasks and the display task.

5.2.2 Display Task

The LCD display is only refreshed by the display task when signalled to do so by either the scroll task or the new character task. Signalling is via the update semaphore.

Access to the information backing the LCD is protected by a mutex, to ensure consistent update.

The LCD is a timing sensitive device; to prevent interrupts from upsetting the timing, LCD updates are performed within a critical section.

5.2.3 Scroll Task

The data backing the LCD display is only scrolled one digit to the right by the scroll task when signalled to do so by the counter CNT2 ISR. Signalling is via the scroll semaphore.

Access to the information backing the LCD is protected by a mutex, to ensure consistent update.

Once the display has been scrolled, the display task is signalled (via the update semaphore) to update the LCD.

5.2.4 New Character Task

The new character task places a single, random digit into the display information at a random location once per µC/OS-II tick.

Access to the information backing the LCD is protected by a mutex, to ensure consistent update.

Once the digit has been placed, the display task is signalled (via the update semaphore) to update the LCD.

5.2.5 Counter cnt1 ISR

Counter CNT1 is used as the source of µC/OS-II tick interrupts. The counter is configured for 16Hz operation.

Counter CNT1 interrupts are muxed/demuxed in *isr.asm*. This ensures that interrupts are handled correctly within the context of the operating system, without excessive code duplication.

The ISR acknowledges the interrupt and performs the necessary µC/OS-II calls.

5.2.6 Counter cnt2 ISR

Since the scroll task is intended to function asynchronously with respect to µC/OS-II ticks, the scroll task is driven by a separate interrupt source, counter CNT2.

Counter CNT2 interrupts are muxed/demuxed in *isr.asm*. This ensures that interrupts are handled correctly within the context of the operating system, without excessive code duplication.

The ISR acknowledges the interrupt and signals the scroll task via the scroll semaphore.

5.2.7 ADC ISR

The ADC is used to sample the voltage across variable resistor VR1.

ADC interrupts are muxed/demuxed in *isr.asm*. This ensures that interrupts are handled correctly within the context of the operating system, without excessive code duplication.

The ISR acknowledges the interrupt and updates the counter CNT2 reload value. The overall effect is to alter the frequency of counter CNT2 interrupts, and hence the scroll rate.

5.3 Opening the Project

From the CyanIDE main menu, select *Project->Open*, and navigate to the directory in which the µMatrix example project is stored. Select the project file (*umatrix.cyp*) and click the *Open* button. CyanIDE opens the project and displays the source files in the navigator pane to the left of the main window. Double-click on any file in the navigator (or right-click and select *Open*) to open it in a code editor window in the main CyanIDE workspace.

5.4 Compiling the Application

From the CyanIDE main menu, select *Build->Build*, or *Build->Rebuild All*. CyanIDE compiles the C source files, then assembles and links all assembly files to generate the project output binary file. Any compilation or assembly errors are listed in the build tab of the output window. Double-click on any error to open the corresponding source file in the editor, with the cursor at the line that contains the error.

If changes are made to any .c, .h or .asm files within the project, then it is sufficient to use the Build command to recompile the project. CyanIDE checks the project for any dependencies on these files, and compiles and links only those files that need to be rebuilt. However, if any change is made to the files *app_cfg.h* and *os_cfg.h*, then use the *Build->Rebuild All* command to ensure that all project files are recompiled with the new µC/OS-II settings.

5.5 Loading and Running the Application

Having compiled the application, the resulting code is downloaded to the eCOG1 evaluation board using CyanIDE. From the main menu, select *Debug->Run*. CyanIDE downloads the file to target memory and starts execution.

The CyanIDE debugger allows the program to be stopped or paused and restarted. It supports single step by source line or by instruction and provides windows to display the CPU registers, memory and any watched variables.

5.6 Results

When the program is executed, the software scrolls digits across the LCD attached to the development board. The digits scroll from left to right, and the speed of the scrolling can be controlled by adjusting variable resistor VR1. This indicates correct operation and verifies that the installation was successful.

6 New Projects

CyanIDE templates for the eCOG1k Evaluation Board, eCOG1k Development Board and eCOG1k Development Board v2 are bundled together with the architecture-specific files for µC/OS-II. Installing these files is described earlier.

New projects based upon these templates comprise:

- Default µC/OS-II configuration.
- Code to setup and handle counter CNT1 as a source of µC/OS-II tick interrupts.
- Code to create a single initial task that loops indefinitely.
- A complete application that should build and run immediately.

Appendix A Project Files

The µMatrix example project includes the following files and directories:

- Micrium folder containing µC/OS-II source files
- app_cfg.h µC/OS-II application configuration
- cpu_hooks.c µC/OS-II hooks
- cstartup.asm C startup and initialisation code
- devboard.cfg peripheral configuration
- internal.map memory map
- irq.asm reset and interrupt vectors
- isr.asm interrupt service routines
- os_cfg.h µC/OS-II operating system configuration
- umatrix.c main program
- umatrix.cyp CyanIDE project file