Micriμm

Empowering Embedded Systems

μC/OS-II

μC/TCP-IP

μC/GUI

and the

NXP LPC2468 Processor

(Using the Embedded Artists LPC2468 OEM Evaluation Board)

Application Note

AN-9468

www.Micrium.com

About Micriμm

Micriμm provides high-quality embedded software components in the industry by way of engineer-friendly source code, unsurpassed documentation, and customer support. The company's world-renowned real-time operating system, the Micriμm **μC/OS-II**, features the highest-quality source code available for today's embedded market. Micriμm delivers to the embedded marketplace a full portfolio of embedded software components that complement **μC/OS-II**. A TCP/IP stack, USB stack, CAN stack, File System (FS), Graphical User Interface (GUI), as well as many other high quality embedded components. Micriμm's products consistently shorten time-to-market throughout all product development cycles. For additional information on Micriμm, please visit www.micrium.com.

About μC/OS-II

Thank you for your interest in **μC/OS-II**. **μC/OS-II** is a preemptive, real-time, multitasking kernel. **μC/OS-II** has been ported to over 45 different CPU architectures and now, has been ported to the Embedded Artists LPC2468 OEM evaluation board which includes the ARM-based NXP LPC2468 processor.

μC/OS-II is small yet provides all the services you would expect from an RTOS: task management, time and timer management, semaphore and mutex, message mailboxes and queues, event flags an much more.

You will find that **μC/OS-II** delivers on all your expectations and you will be pleased by its ease of use.

About μC/TCP-IP

μC/TCP-IP is a compact, reliable, high performance TCP/IP protocol stack. Built from the ground up with Micrium's renowned quality, scalability and reliability, **μC/TCP-IP** enables the rapid configuration of required network options to minimize your time to market. **μC/TCP-IP** provides you with the highest quality source code in the industry.

μC/TCP-IP is a clean-room design and is not derived from publicly available Unix stacks, yet still maintains compatibility with the Berkeley 4.4 socket layer interface. As with all Micrium products, **μC/TCP-IP** is written in ANSI C enabling its usage with a wide array of best-of-class cross-development tools. **μC/TCP-IP** can be used on 16, 32 and even some 64-bit CPUs.

μC/TCP-IP was designed specifically for the demanding requirements of embedded systems. Critical sections were kept to a minimum and selected run-time validations can be disabled to enhance performance. **μC/TCP-IP** implements zero copy buffer management for highest efficiency.

μC/TCP-IP allows you to adjust the memory footprint based on your requirements. **μC/TCP-IP** can be configured to only include only those network modules absolutely required by your system. When a module is not used, it's not included in the build to save valuable memory space for resource limited embedded systems.

Licensing

μC/OS-II and **μC/TCP-IP** are provided in source form for **FREE** for educational use or for peaceful research.

If you plan on using **μC/OS-II** in a commercial product you can evaluate **μC/OS-II** and **μC/TCP-IP** for FREE for **45 days** and within that period, need to contact Micriμm to properly license its use in your product.

We provide **ALL** the source code with this application note for your convenience and to help you experience **μC/OS-II** and **μC/TCP-IP**. The fact that the source is provided **DOES NOT** mean that you can use it without paying a licensing fee. Please help us continue to provide the Embedded community with the finest software available. Your honesty is greatly appreciated.

Manual Version

If you find any errors in this document, please inform us and we will make the appropriate corrections for future releases.

Version	Date	By	Description
V.1.00	2007/04/27	BAN	Initial version.

Software Versions

This document may or may not have been downloaded as part of an executable file, *Micrium-NXP-uCOS-II–TCPIP-GUI-Bin-LPC2468-EA.exe*, containing the pre-compiled projects described here. If so, then the versions of the Micriμm software modules in the table below are included in these binaries. In either case, the software port described in this document uses the module versions in the table below

Module	Version	Comment
μC/OS-II	V2.84	ARM Port V1.82
μC/OS-View	V1.33	
μC/TCP-IP	V1.89	
μC/GUI		

See Also

In addition to the pre-compiled binaries accompanying this appnote, several projects with full source code are available from Micrium. A **μC/OS-II** project which just uses Micriμm's **μC/OS-II** RTOS can also be found on the NXP LPC24xx page on the Micriμm website with application note AN-1468. Also, a **μC/OS-II** and **μC/TCP-IP** project which just uses both Micriμm's **μC/OS-II** RTOS and Micriμm's TCP-IP stack, **μC/TCP-IP**, can also be found on the NXP LPC24xx page on the Micriμm website with application note AN-3468.

This application omits all details of the **μC/OS-II** port, BSP, and **μC/TCP-IP** EMAC port. For more information about these, see AN-1468 and AN-3468.

Document Conventions

Numbers and Number Bases

- Hexadecimal numbers are preceded by the "0x" prefix and displayed in a monospaced font. Example: `0xFF886633`.
- Binary numbers are followed by the suffix "b"; for longer numbers, groups of four digits are separated with a space. These are also displayed in a monospaced font. Example: `0101 1010 0011 1100b`.
- Other numbers in the document are decimal. These are displayed in the proportional font prevailing where the number is used.

Typographical Conventions

- Hexadecimal and binary numbers are displayed in a monospaced font.
- Code excerpts, variable names, and function names are displayed in a monospaced font. Functions names are always followed by empty parentheses (e.g., `OS_Start()`). Array names are always followed by empty square brackets (e.g., `BSP_Vector_Array[]`).
- File and directory names are always displayed in an italicized serif font. Example: */Micrium/Sofware/uCOS-II/Source/*.
- A bold style may be layered on any of the preceding conventions—or in ordinary text—to more strongly emphasize a particular detail.
- Any other text is displayed in a sans-serif font.

Table of Contents

1. Introduction

This document, *AN-9468*, accompanies example projects with using **μC/OS-II**, **μC/OS-View**, **μC/TCP-IP** and **μC/GUI** with the Embedded Artists LPC2468 OEM Evaluation Board, as shown in Figure 1-1, which employs NXP's ARM7TDMI-based LPC2468 microcontroller. The processor includes 512 kB on-chip flash memory and 64-kB SRAM in addition to dedicated SRAM for the EMAC and DMA peripherals. The chip includes serial interfaces such as an internal 10/100 EMAC, USB device and host (with support for an external OTG transceiver), two CAN channels, a SPI controller, two SSP controllers, four UARTs, and several I^2C and I^2S interfaces . Additionally, the chip has a SD/MMC card interface, many general purpose I/O pins, and a 10-bit A/D converter.

The LPC2468 OEM daughterboard includes the processor, a Micrel PHY, and external memories. The baseboard on which this is situated provides the user peripherals, including five user push buttons (one of which is attached to an interrupt line), two potentiometers, and four LEDs. One RS-232 port (for the processor's UART1), a USB port used for a serial bridge (for UART0), one CAN port, one USB device port, one USB host port, a SD/MMC card holder, and an Ethernet port provide for external communication. The board includes a standard 20-pin JTAG connector for debugging and programming.

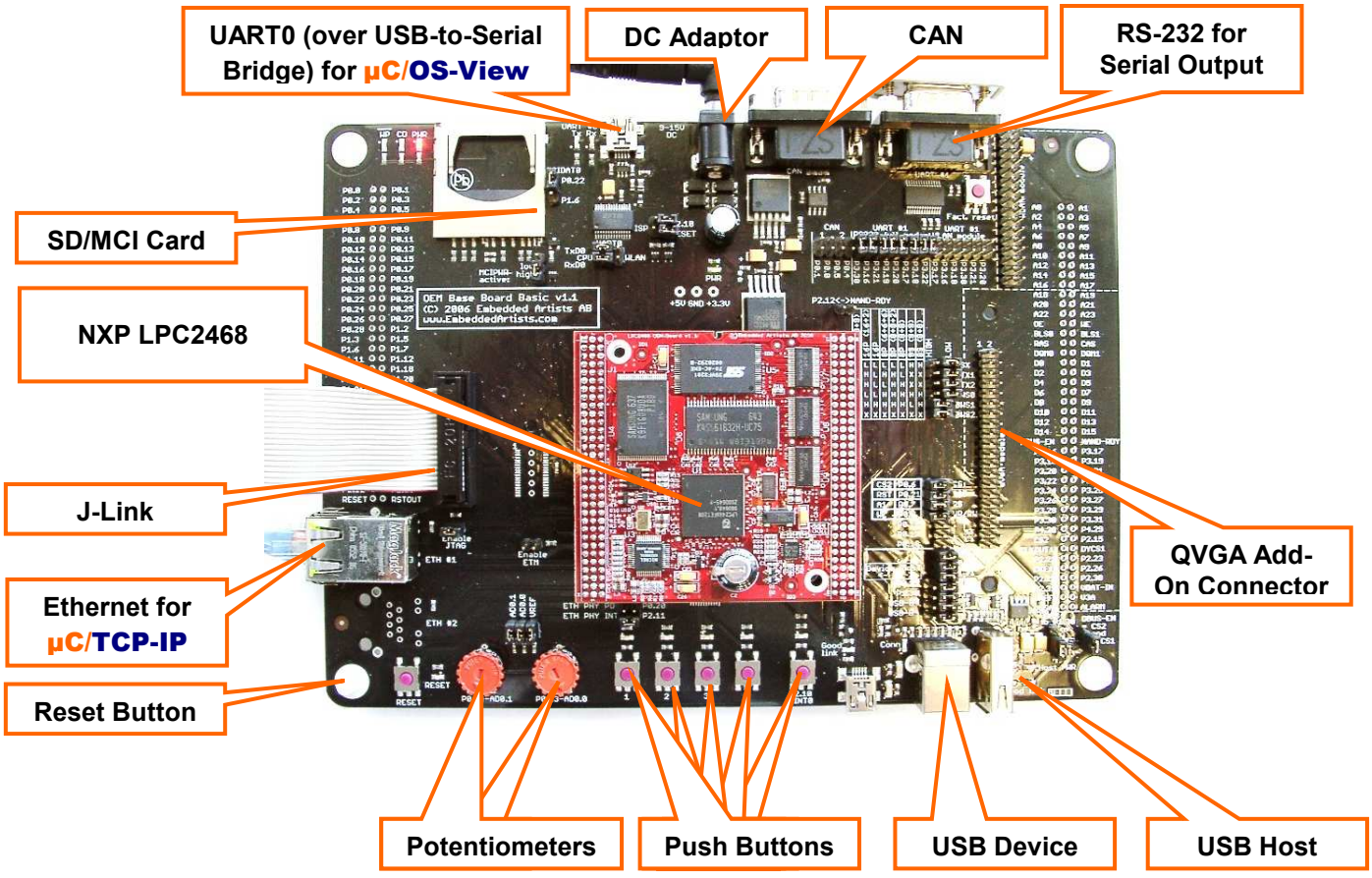

Figure 1-1. Embedded Artists LPC2468 OEM Evaluation Board

The board can be used with Embedded Artist's QVGA add-on module, as shown in Figure 1-2, containing a 320 x 240 TFT LCD. This figure shows the precompiled **μC/OS-II**, **μC/TCP-IP**, and **μC/GUI** example included with this appnote.

Figure 1-2. Embedded Artists QVGA Add-On Module

If this appnote was downloaded in a packaged executable zip file, then it should have been found in the directory */Micrium/Appnotes/AN9xxx-MULT/AN9468-NXP-LPC2468* and the files referred to herein are located in the directory structure displayed in Section 2.02; these files are desribed in Section 3.

2. Getting Started

The following sections step through the prerequisites for using the demonstration application described in this document, *AN-9468*. First, the setup of the hardware will be outlined. Secondly, the steps to build the projects and load the application onto the boad through the JTAG will be described. Lastly, instructions will be provided for using the example application.

2.01 Setting up the Hardware

The application can potentially have two serial outputs. Firstly, an application task outputs data about the current state of **μC/OS-II** and **μC/TCP-IP**. This task is configured to use UART1 via the port labelled "RS-232 for Serial Output" in Figure 1-1. Secondly, **μC/OS-View** can be used. This is configured to use UART0, which outputs data via a USB port. As described in the next paragraph, you will need to install a driver on your PC and remove the ISP jumpers to use this output.

The board can be powered either by an external DC adapter or through the USB port used for the USB-to-serial bridge (labelled "UART0 (over USB-to-Serial Bridge" in Figure 1-1). When first connecting the USB cable between the evaluation board and your computer, you will be prompted to install the FTDI driver (see the board manual for details). Once you have this driver installed, your computer can use this particular USB port on the board port as if it were a COM port. Included in this is the ability to program the LPC2468 through the UART0 (ISP). If you want to use UART0 (or use power from this USB port), but do not want to use ISP, you will need to disconnect the jumpers shown in Figure 1-3.

As mentioned in the previous paragraph, the LPC2468 on the evaluation board can be programmed through the USB-to-serial bridge. In our tests, the board was always programmed and debugged through an external debugger, either a J-Link (with IAR EWARM) or a ULINK or ULINK2 (with Keil μVision). Additional details about using the ISP interface are available in the evaluation board user manual.

If you want to be able to ping the board, connect an Ethernet cable between the board's port and an available port on your network. The IP address of the project is configured to be 10.10.1.129; see Section 2.05 for information about modifying this.

The pre-compiled binaries included with this application using **μC/GUI**. Though the projects do not require a board with the QVGA add-on module—the board can be pinged in any configuration—this project is most enlightening when used with the QVGA add-on.

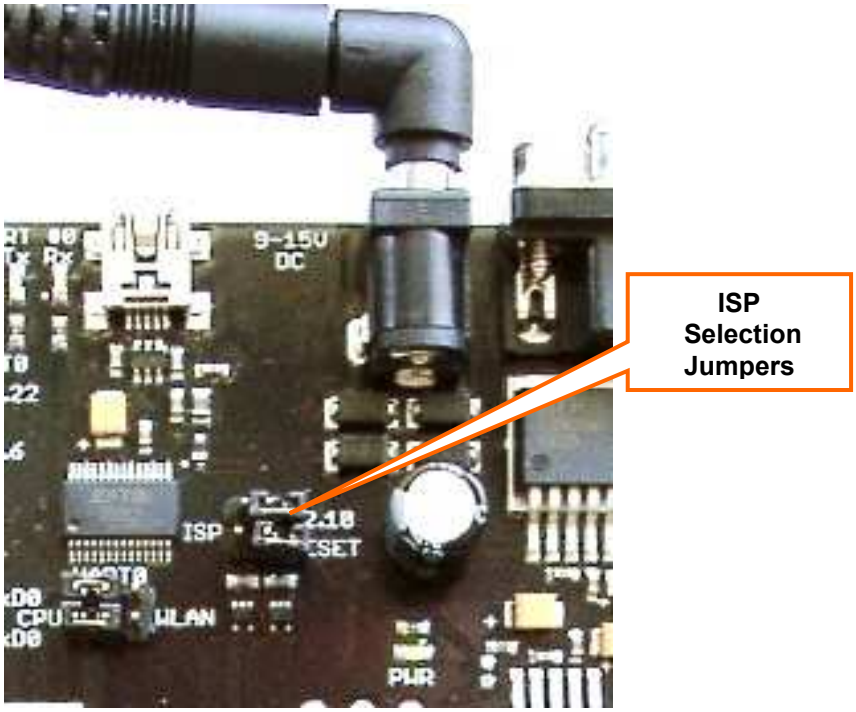

Figure 2-1. ISP Selection Jumpers

2.02 Opening and Viewing the Project

If this file were downloaded as part of an executable zip file (which should have been named *Micrium-NXP-uCOS-II-TCPIP-GUI-LPC2468-EA.exe*), then the files referred to herein are located in the directory structure shown in Figure 2-3.

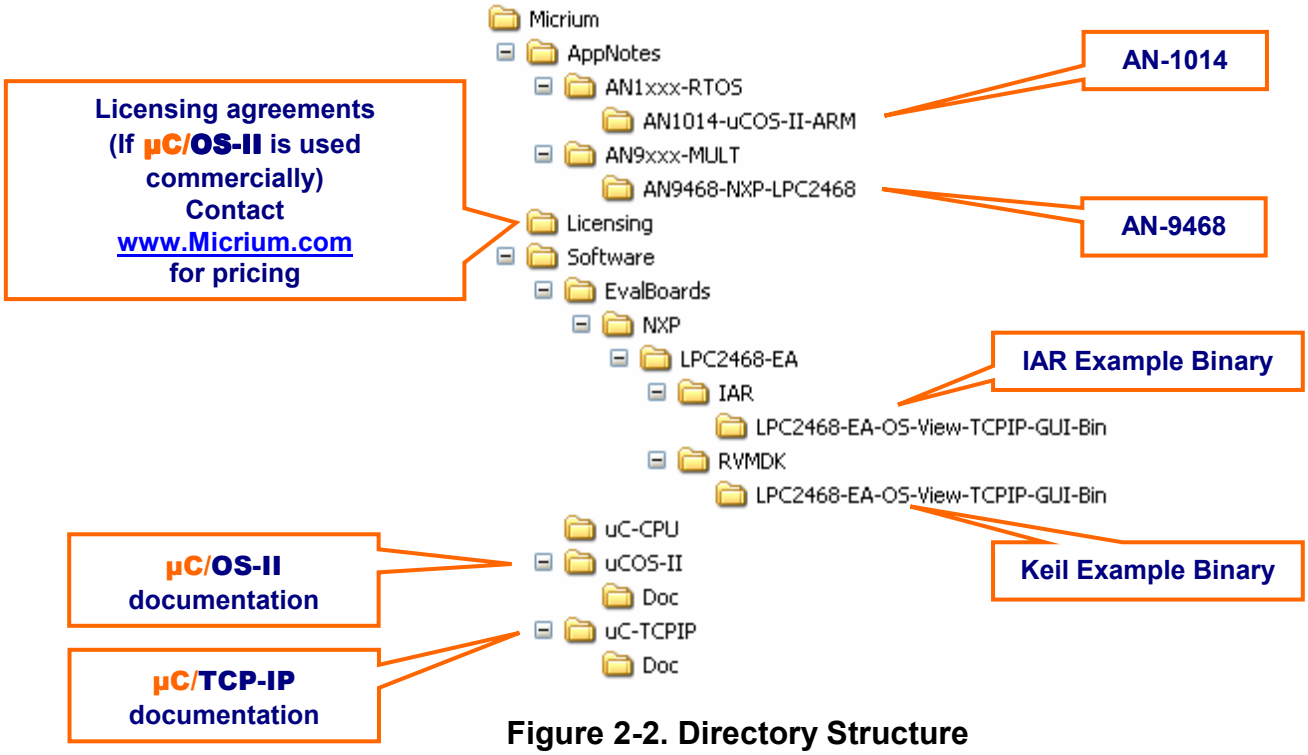

Figure 2-2. Directory Structure

2.03 Using the IAR Project

An IAR project file named ***LPC2468-EA-OS-View-TCPIP-GUI-Bin.ewp*** is located in the directory (marked "IAR Example Binary" in Figure 2-3)

/Micrium/Software/EvalBoards/NXP/LPC2468-EA/IAR/OS-View-TCPIP-GUI-Bin

To view this example project, start an instance of IAR EWARM, and open the workspace file *LPC2468-EA-OS-View-TCPIP-GUI-Bin.eww*. To do this, select the "Open" menu command under the "File" menu, select the "Workspace…" submenu command and select the workspace file after navigating to the project directory. (In addition, the workspace should be openable by double-clicking on the file itself in a Windows Explorer window.)

The project tree should contain only one file, the compiled binary ***LPC2468-EA-OS-View-TCPIP-GUI-Bin.d79***.

Once the connections described in Section 2.01 are made between your PC and the Embedded Artists LPC2468 OEM Evaluation Board, the code can be loaded onto the board. To load the code through the J-TAG debugger onto the connected evaluation board, select the "Debug" menu item from the "Project" menu. The project is setup to use a J-Link debugger; if you wish to use a different debugger, please select the appropriate DLL in the project options dialog box (select "Debugger" in the listbox).

2.04 Using the Keil μVision3 Project

A Keil μVision3 (RV-MDK) project file named ***LPC2468-EA-OS-View-TCPIP-GUI-Bin.uV2*** is located in the directory (marked "Keil Example Binary" in Figure 2-7)

/Micrium/Software/EvalBoards/NXP/LPC2468-EA/IAR/OS-View-TCPIP-GUI-Bin

To view this example project, start an instance of Keil μVision3, and open the project file ***LPC2468-EA-OS-View-TCPIP-GUI-Bin.uV2***. To do this, select the "Open Project.." menu command under the "Project" menu and select the project file after navigating to the project directory. (In addition, the project should be openable by double-clicking on the file itself in a Windows explorer window.)

The project tree should contain only one file, the compiled binary ***LPC2468-EA-OS-View-TCPIP-GUI-Bin.axf***.

Once the connections described in Section 2.01 are made between your PC and the Embedded Artists LPC2468 OEM development board, the code can be loaded onto the board. To load the code through a ULINK or ULINK2 onto the connected evaluation board, select the "Download" menu item from the "Flash" menu. Finally, the LPC2468 can either be debugged (by choosing the "Start/Stop Debug Session" from the "Debug" menu) or allowed to run (by resetting the board).

If you receive the error (or one similar) shown in Figure 2-3, then it is likely that the debug options have become corrupted (or have been completely reset). Restore the settings by doing as follows (assuming you are using a ULINK or ULINK2):

Figure 2-3. μVision3 Error: Incorrect Debug Settings

1. Right-click on the target name, "Flash" in the Project Workspace and choose "Options for the target 'Flash'". Choose the "Debug" tab in the dialog box that appears, as shown in Figure 2-4. Choose the proper debugger (probably "ULINK ARM Debugger").

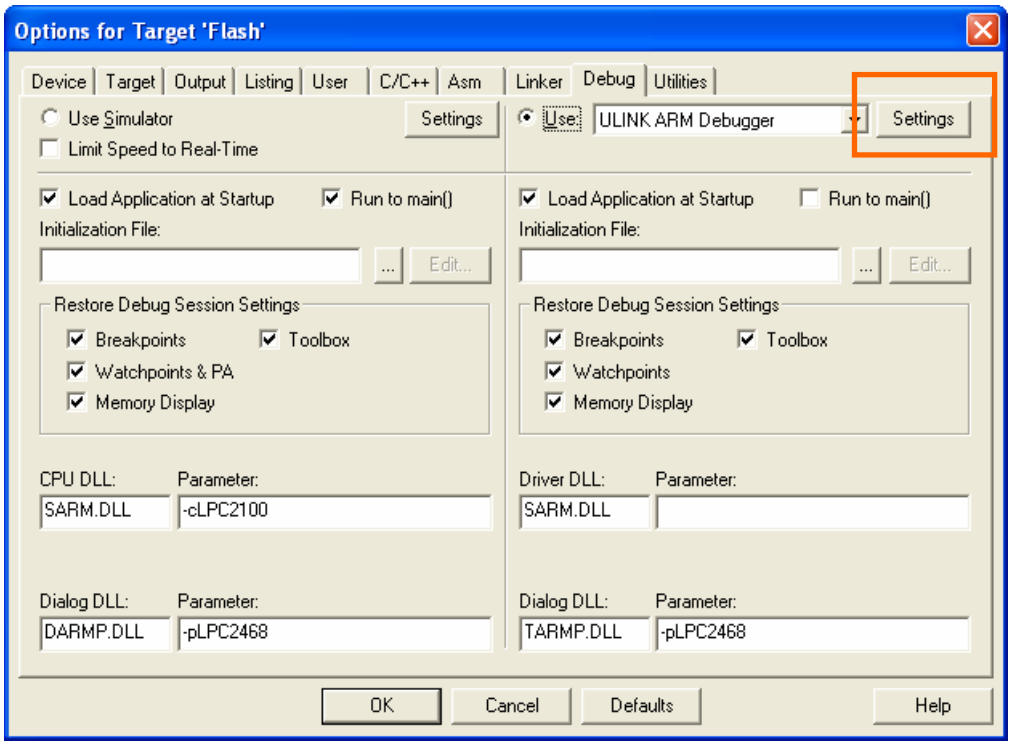

Figure 2-4. μVision3 Debug Options

2. Press the “Settings” button on this tabpane. A new dialog, as shown in Figure 2-5, should appear. Match the settings as appears in Figure 2-5. It is possible that a JTAG clock speed of 100kHz may be too fast for either your target or your ULINK; consequently, if you have problems at 100 kHz, please decrease this and try again.

3. Press OK in the settings dialog and switch to the “Utilities” tab. Select the “Use Target Driver for Flash Programming” radio button and select “ULINK ARM Debugger” in the drop-down menu. Press the “Settings” button. A dialog should appear, as shown in Figure 2-6. Add the appropriate flashloader (as has already been done in Figure 2-6) by selecting the “Add” button and choosing “LPC2000 IAP2 512kB Flash” in the list. This flashloader will then appear in the listbox in the “Flash Download Setup” dialog.

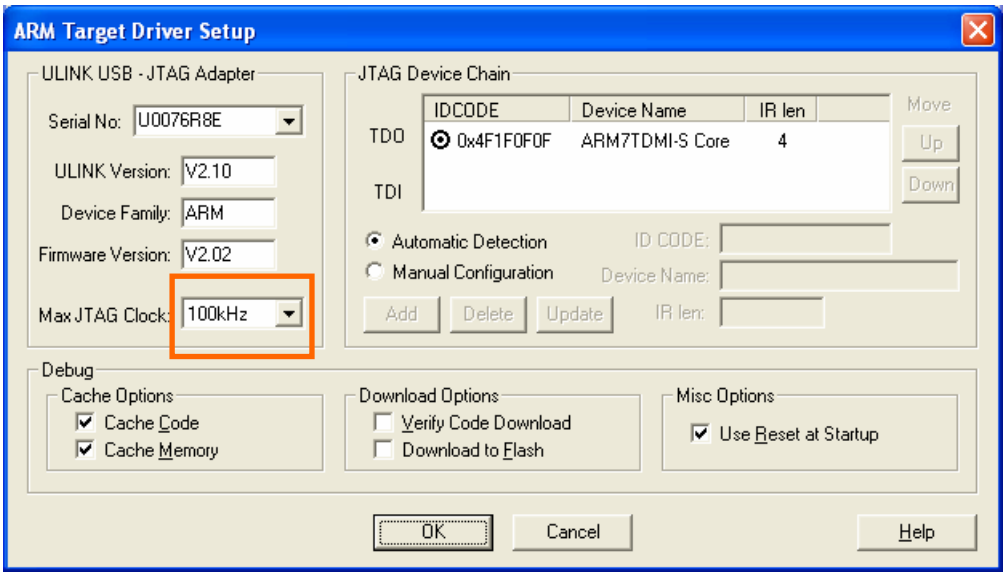

Figure 2-5. μVision3 ULINK Debugger Settings

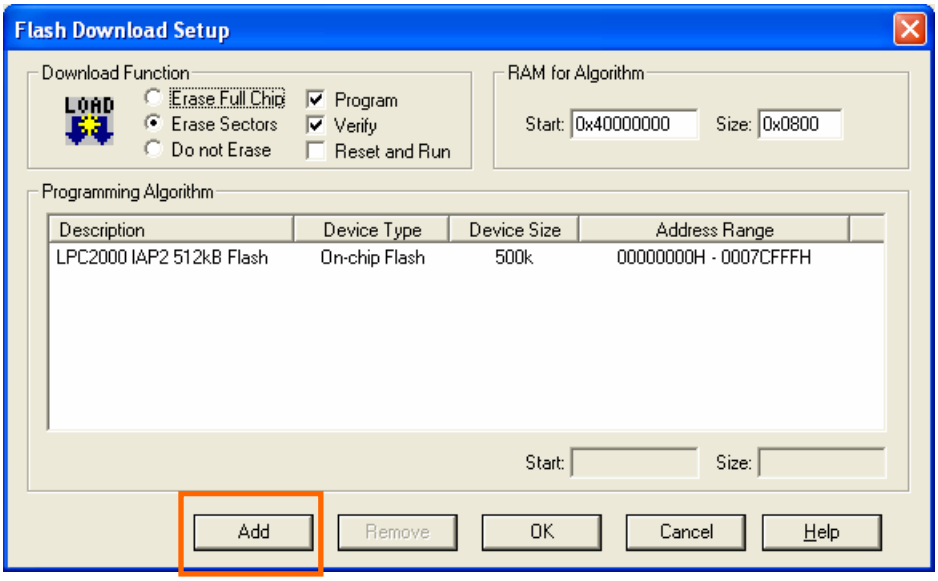

Figure 2-6. μVision3 Flash Download Settings

2.05 Example Application

The example application contains application tasks which respond to the push buttons, output data over the serial port, and toggle the LEDs. In addition, **μC/TCP-IP** will respond to TCP-IP packets over an Ethernet connection, allowing the board to be pinged and **μC/GUI** will display a demonstration on a connected LCD on the QVGA add-on.

2.05.01 Application Tasks

When the application is started, the LEDs will begin turning on, one-by-one. The LEDs will then be all turned off. If any of the left four push buttons (but not the one attached to P2.10) have been pressed since this last occurred, then the corresponding LEDs (the LEDs directly above those push buttons) will blink rapidly 20 times.

The right-hand potentiometer controls the delay between the lighting of the LEDs in the initial phase; turning the potentiometer counter-clockwise will decrease the delay. Similarly, the left-hand potentiometer controls the rate at which the LEDs blink which correspond to the pressed push buttons; turning the potentiometer counter-clockwise will decrease the delay.

The right-most push button can be used to control the serial output. After setting up a Windows terminal utility (such as Hyperterminal, as covered in Section 2.05.04), an output similar to that which is shown in Figure 2-7 will appear. Successive presses of the push button will advance the output through several sets of information. If the code is loaded onto the board after the HyperTerminal connection has been setup, then a greeting (the first output line in Figure 2-7) will be shown. After one second, a line specifying the **μC/OS-II** version and tick rate will appear and be updated 10 times per second. The push buttons will advance the output to a different piece of information, which will be updated 10 times per second.

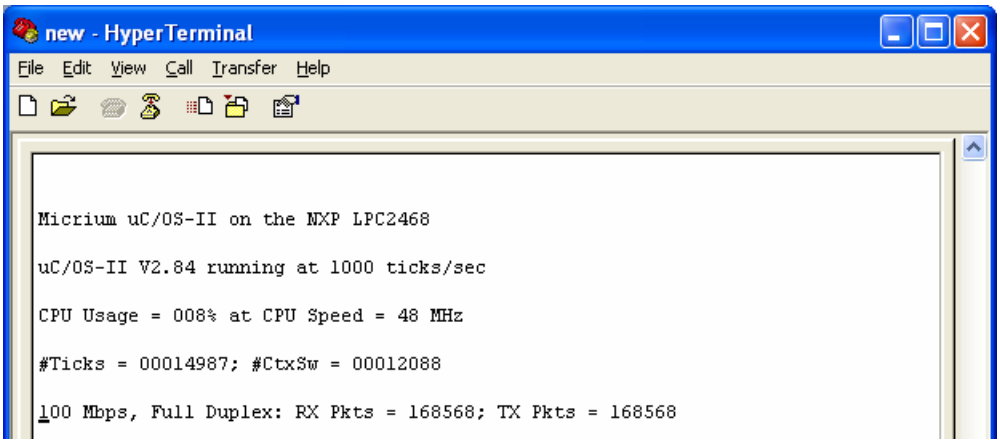

Figure 2-7. Serial Port Output (over UART1)

2.05.02 Pinging the Board

The project is currently configured with an IP address of 10.10.1.129. You should be able to open a command window and ping the board, as shown in Figure 2-8.

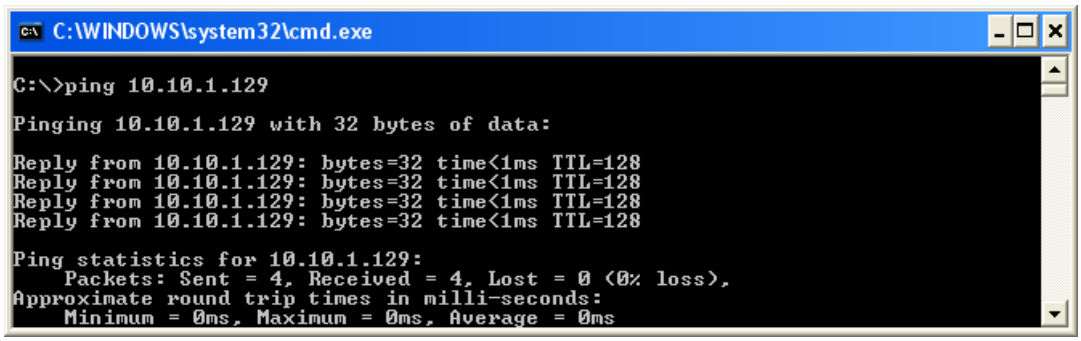

Figure 2-8. Pinging the LPC2468-EA.

2.05.04 Setting up Hyperterminal

To communicate with the board through RS-232, connect a serial cable between the evaluation board and your PC and open a HyperTerminal window (often located in the start menu, on the “Communications” submenu of the “Accessories”). As shown in Figure 2-9, enter a name for the connection and press the “OK” button; In the “Connect To” window, choose the appropriate COM port and press “OK”. In the COM properties window which appears, match the settings shown in Figure 2-10.

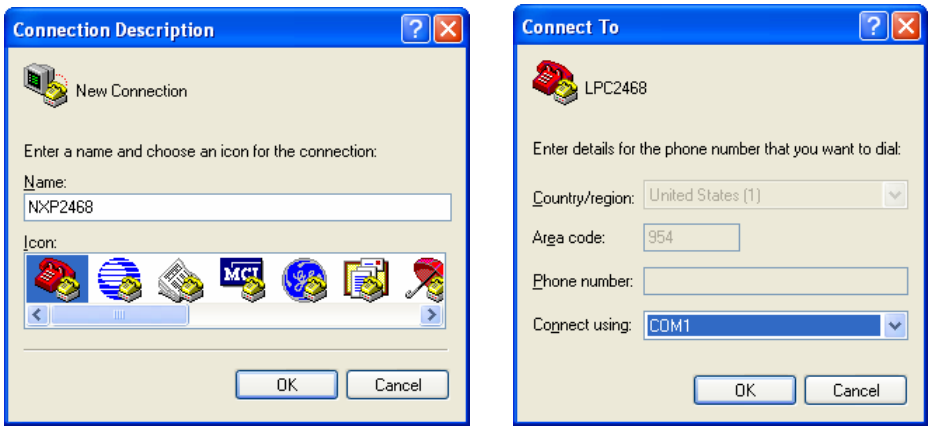

Figure 2-9. Connection Specification

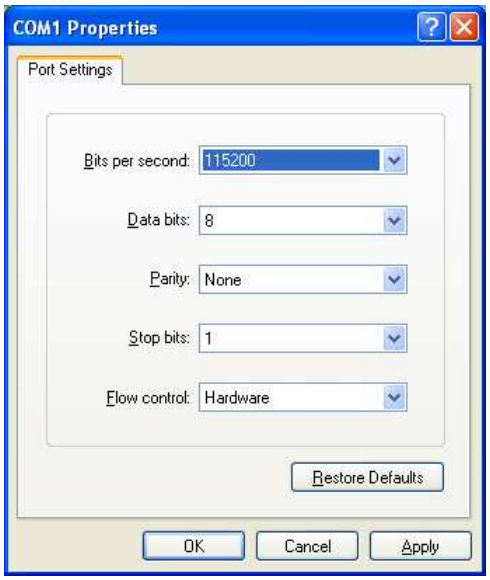

Figure 2-10. COM Properties

2.05.02 Additional Information

Including the **μC/OS-II** system tasks, the example application includes eight tasks, as listed in Table 1-1. The board hardware used in the application is diagrammed in Figure 2-11.

Task Name	Priority	Function
`AppTaskStart()` "Start Task"	1	Starts **μC/OS-View** and **μC/TCP-IP**; reads ADCs, blinks LEDs.
`AppTaskKbd()` "Keyboard"	2	Reads status of push buttons; passes IDs of pressed buttons to `AppTaskStart()` in a queue.
`AppTaskSer()` "RS-232 Output"	4	Output information to UART1.
`AppTaskGui()` "GUI Task"	7	Executes the GUI demo.
"Net IF Rx Task"	5	The task in which **μC/TCP-IP** receives data from a NIC or EMAC driver.
"Net Timer Task"	6	The task used by **μC/TCP-IP** to manage its timers.
"uC/OS-II Idle"	31	Executes when no other task is exeucting
"uC/OS-II Stat"	30	Collect stack usage statistics
"uC/OS-II Tmr"	8	Manages **μC/OS-II** timers

Table 2-1. Example Application Tasks

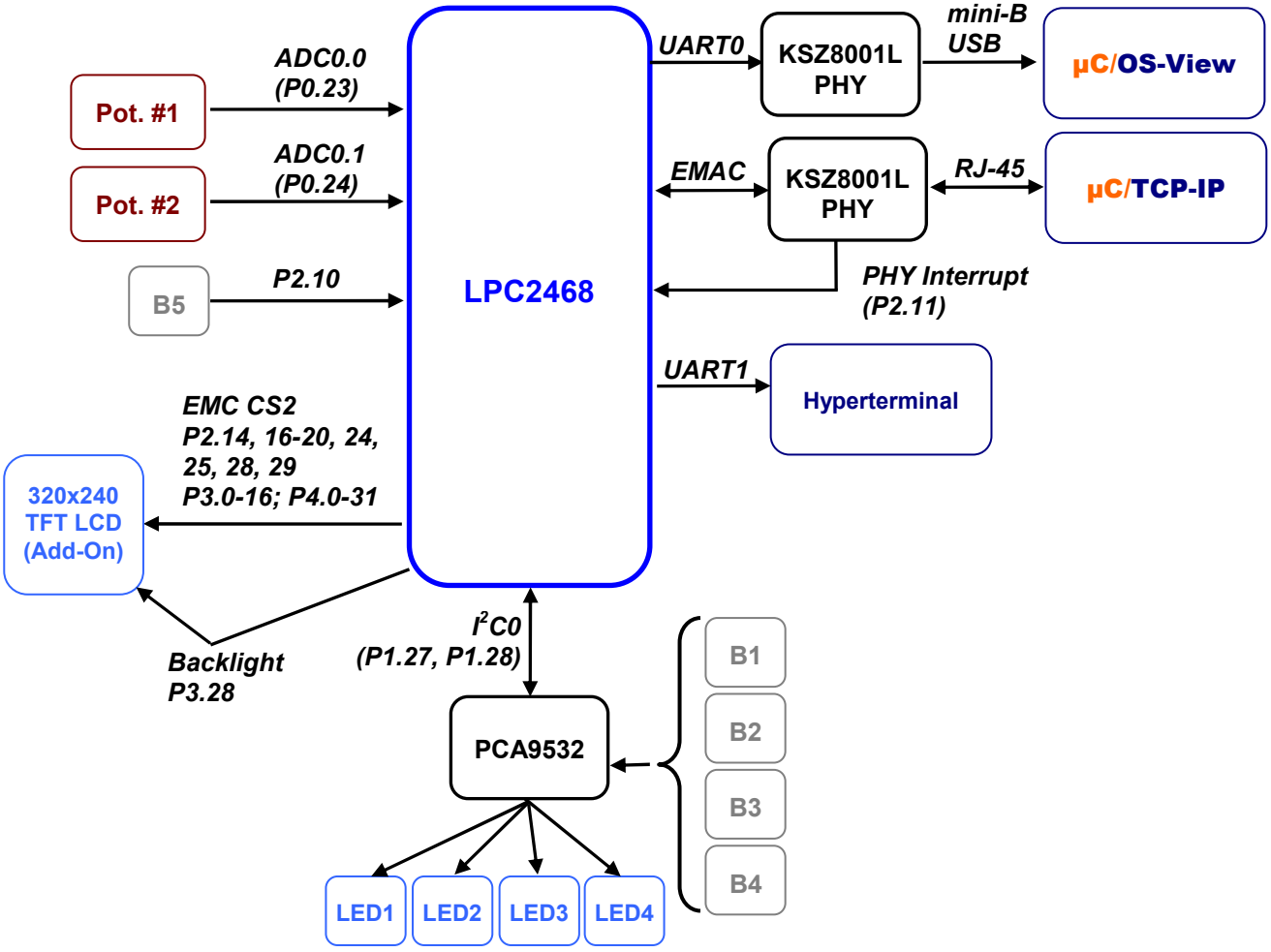

Figure 2-16. Example Application Hardware Use

If **μC/OS-View** is enabled, then information about the state of **μC/OS-II** will be available through the **μC/OS-View** viewer Windows application. For more information about enabling and using **μC/OS-View**, see Section 4.

Memory Range	Size	Segment(s)
0x00000000-0x0000003F	64 B	Exception vectors
0x00000040-0x0007FFFF	512 kB	Code
0x40000000-0x4000FFFF	64 kB	Stacks, data

Table 2-2. Memory Setup

3. Directories and Files

Application Notes

\Micrium\AppNotes\AN1xxx-RTOS\AN1014-uCOS-II-ARM

This directory contains *AN-1014.pdf*, the application note describing the ARM port for **μC/OS-II**, and *AN-1014-PPT.pdf*, a supplement to *AN-1014.pdf*.

\Micrium\AppNotes\AN9xxx-MULT\AN9468 -NXP-LPC2468

This directory contains this application note, *AN-9468.pdf*.

Licensing Information

\Micrium\Licensing

Licensing agreements are located in this directory. Any source code accompanying this appnote is provided for evaluation purposes only. If you choose to use **μC/OS-II** in a commercial product, you must contact Micriμm regarding the necessary licensing.

μC/OS-II Files

\Micrium\Software\uCOS-II\Doc

This directory contains documentation for **μC/OS-II**.

μC/TCP-IP Files

\Micrium\Software\uC-TCPIP\Doc

This directory contains documentation for **μC/TCPIP**.

Application Code

\Micrium\Software\EvalBoards\NXP\LPC2468-EA\IAR\OS-View-TCPIP-GUI-Bin

This directory contains the compiled executable for IAR EWARM, *LPC2468-EA-OS-View-TCPIP-GUI-Bin.d79*.

\Micrium\Software\EvalBoards\NXP\LPC2468-EA\RVMDK\OS-View-TCPIP-GUI-Bin

This directory contains the compiled executable for Keil μVision3 (RVMDK), *LPC2468-EA-OS-View-TCPIP-GUI-Bin.axf*.

4. μC/OS-View

μC/OS-View, a module that allows you to view useful statistics gathered from **μC/OS-II**, can be readily added to the example application. After licensing **μC/OS-View**'s source files from Micrium and obtaining the module's Windows application, you can begin to use **μC/OS-View** after completing a few simple operations. First, you will need to use a serial cable to connect the board's RS232 port (which is marked "UART #0") to an available serial port on your PC.

After making these preparations, build and run your application and start **μC/OS-View**'s Windows application. Through the Setup dialog box, specify the COM port on your computer to which the board is connected and a baud rate of 115200. When you have completed these initializations, the Windows application will begin receiving packets from the board, eventually resulting in a graph resembling that which is shown in Figure 6-1.

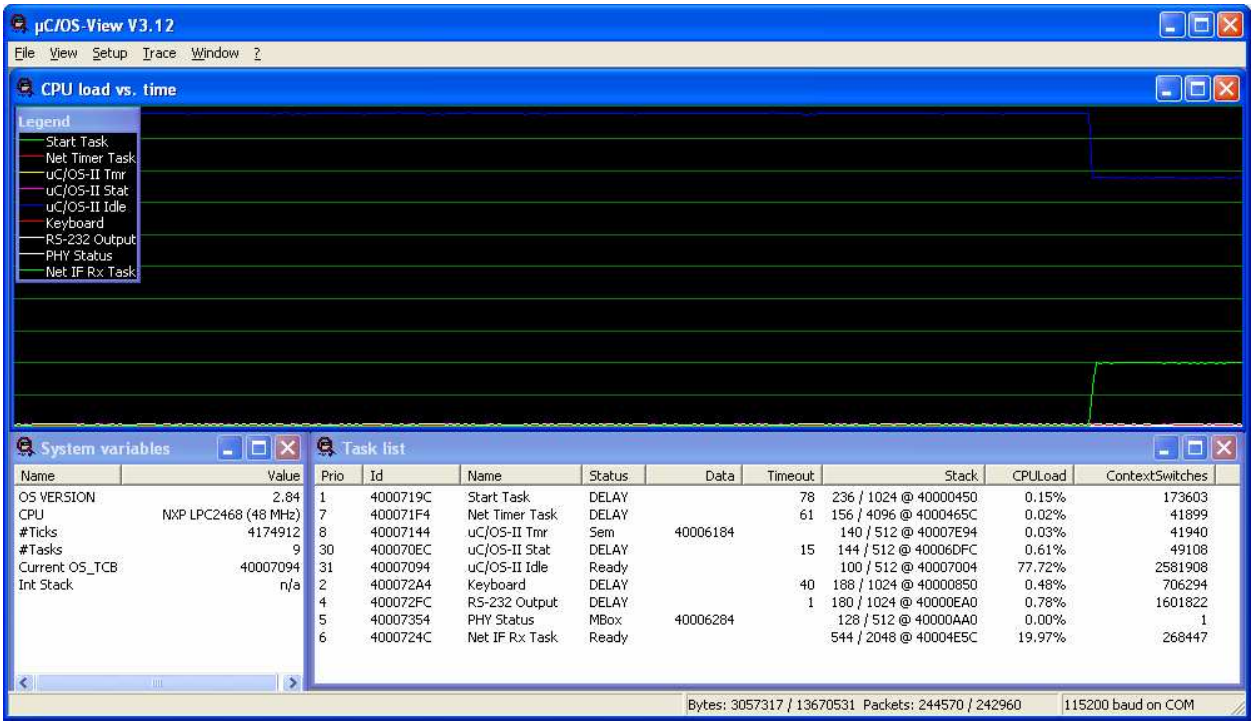

Figure 6-1, μC/OS-View Windows "Viewer"

μC/OS-View is a combination of a Microsoft Windows application program and code that resides in your target system (in this case, the Embedded Artists LPC2468 OEM evaluation board). The Windows application connects to your system via an RS-232C serial port. The status of the tasks which are managed by **μC/OS-II** can be viewed with the Windows application.

μC/OS-View allows you to view the following information from a **μC/OS-II** based product:

- The address of the TCB of each task (up to 63 tasks);

- The name of each task (up to 63 tasks);
- The status (e.g., ready, delayed, waiting on event) of each task;
- The number of ticks remaining for a timeout or if a task is delayed;
- The amount of stack space used and left for each task;
- The percentage of CPU time each task relative to all the tasks;
- The number of times each task has been 'switched-in'; and
- The execution profile of each task.

μC/OS-View also allows you to send commands to your target and allow your target to reply back and display information in a 'terminal window'.

μC/OS-View is licensed on a per-developer basis. In other words, you are allowed to install **μC/OS-View** on multiple PCs as long as the PC is used by the same developer. If multiple developers are using **μC/OS-View** then each needs to obtain his own copy. Contact Micriμm for pricing information.

Licensing

μC/OS-II is provided in source form for **FREE** evaluation, for educational use or for peaceful research. If you plan on using **μC/OS-II** in a commercial product you need to contact Micriμm to properly license its use in your product. We provide **ALL** the source code with this application note for your convenience and to help you experience **μC/OS-II**. The fact that the source is provided does **NOT** mean that you can use it without paying a licensing fee. Please help us continue to provide the Embedded community with the finest software available. Your honesty is greatly appreciated.

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

IAR Systems
Century Plaza
1065 E. Hillsdale Blvd
Foster City, CA 94404
USA
+1 650 287 4250
+1 650 287 4253 (FAX)
e-mail: Info@IAR.com
WEB : www.IAR.com

CMP Books, Inc.
1601 W. 23rd St., Suite 200
Lawrence, KS 66046-9950
USA
+1 785 841 1631
+1 785 841 2624 (FAX)
e-mail: rushorders@cmpbooks.com
WEB : http://www.cmpbooks.com

Micriμm
949 Crestview Circle
Weston, FL 33327
USA
+1 954 217 2036
+1 954 217 2037 (FAX)
e-mail: Jean.Labrosse@Micrium.com
WEB : www.Micrium.com

NXP
1110 Ringwood Court
San Jose, CA 95131
USA
+1 408 474 8142
WEB : www.nxp.com