# Tornado Interface for MAJIC

User's Manual

EPI Embedded Performance, Inc.

P/N 0380-0200-10 Rev 04

October, 2002

Acknowledgments:

MAJIC, JEENI, EDB and EPI are trademarks of Embedded Performance, Inc.
Tornado, Tornado2 and VxWorks are trademarks of WindRiver Systems.
MIPS, R4000, R3000 are trademarks of MIPS Technologies, Inc.
Windows is a trademark of Microsoft Corporation.
All other trademarks are trademarks of their respective companies.

# *Table of Contents*

# Preface

## Contents of this Manual

Specifically, the chapters of this manual are organized as follows:

| | |
|---|---|
| Preface: | This Introduction. |
| Chapter 1: | This chapter discusses basic concepts of the Tornado environment , the distribution package contents and installation instructions. |
| Chapter 2: | This chapter discusses the setup, configuration, and Loading VxWorks. |
| Chapter 3: | This chapter discusses the Tornado Target Server operation with MAJIC. |
| Chapter 4: | This chapter discusses the two debugging mode (task and system). |
| Chapter 5: | This chapter provides some discussion on advanced topics such as session logging and trouble shooting. |
| Chapter 6: | Provides addresses and phone numbers for technical support. |

## Notational Conventions

The following conventions are used in the syntax descriptions of this manual.

| | |
|---|---|
| `Courier Font` | this fixed width font is used for characters that must be entered exactly as shown. |
| `[button text]` | A fixed width font enclosed in `[]` is used to signify a screen button. |
| Ariel Font | this font is used to show screen elements like dialog names, or prompts. |
| *`Italic`* | is used to indicate a specific category of input that will be described in detail in the text. |

| | |
|---|---|
| `<key>` | Angle brackets indicate that the item enclosed within the brackets is the name of a special key on the host's keyboard. Some examples are `<enter>`, `<backspace>`, and `<esc>`. In some cases several keys must be held down simultaneously, which is indicated with a dash between them. For example, `<ctrl-C>`. |
| [ ] | Square brackets are used to enclose an optional operand or group of operands. The brackets are not to be entered in the command. |
| { } | Curly braces are used for grouping purposes. These are not to be entered in the command. They either enclose a list of alternatives, one of which must be chosen, or they enclose a group of operands that are to be taken together in the context of a list of alternatives or a subsequent repetition. |
| ... | An ellipses (three dots in succession) is used to indicate that the preceding operand, or group of operands if enclosed by [ ] or { }, may optionally be repeated one or more times. |
| \| | A vertical bar is used to indicate that the operand, or group of operands if enclosed by [ ] or { }, on either side of the bar may be entered, but not both. |
| .. | Two dots in succession indicate the inclusion of sequential items between given start and stop points. For example, **a..z** refers to the entire alphabet including **a** and **z**. |

# 1 Introduction & Installation

Before beginning with this manual, we recommend starting your introduction by reading the MAJIC User's Manual and learning how to startup and use your MAJIC unit with EPI's low-level debugger MONICE. This will ensure that your unit is operating correctly and is configured for your hardware environment.

Note that although this document contains references to different kinds of host systems, it does not imply the availability of the Tornado Interface for MAJIC on any particular host operating system. Please contact EPI's sales department for the latest supported host information.

## Basic Concepts

The MAJIC Wind River Interface integrates the MAJIC JTAG Emulator into the Wind River Tornado tool chain. The host computer is linked to the MAJIC via Ethernet, which provides fast, reliable communication. For Windows, the interface is provided as a dynamically linked library (DLL), which is contained in a single file `epiwrbe.dll`. This interface is called up by the Wind River Target Server, which allows the other Wind River tools access to the target system through the MAJIC.

The Wind River Tornado Interface supports two debugging modes:

System mode:

Where the debugger sees the entire system as a single program to be debugged. In this mode, the MAJIC can be used to download a new VxWorks image into system RAM and test it. This mode can also be used to test standalone executables (such as diagnostics) that do not run under VxWorks. A characteristic feature of system mode debugging is that when a breakpoint is reached, the processor halts (placed into debug mode).

System mode is closest in concept to that of a hardware emulator, since debugging focuses on the state of the processor, but there is some degree of task awareness in this mode. For example, it is possible to examine the state of the currently running tasks.

| | |
|---|---|
| Task mode: | Where a single task is debugged while the operating system and other tasks continue to run. This type of debugging is implemented at the software level by a task running on the target. For Tornado, this task is called the “WDB agent”. |

The MAJIC Interface supports both debugging modes. As a JTAG emulator, the debugging services required in system mode are handled directly by the MAJIC. The CPU provides JTAG access to the processor's internal registers, the processor memory bus, and special breakpoint hardware. This hardware breakpoint support means that, in System mode, breakpoints may be placed at locations in ROM.

In Task mode, the MAJIC operates as a background communication link between the host computer (where the Wind River tools are running) and the WDB task running under VxWorks on the target. In this mode, the MAJIC can be used to download object files which are dynamically linked and run. This mode gives a greater degree of task awareness. For example, breakpoints may be specified to be active within a given task context.

## Distribution Package

After installing your distribution CD review the contents of the directory `<`*`your installation dir`*`>\wrbe.` You should find the following files:

| | |
|---|---|
| `readme.txt` | The latest release notes, tips, etc. |
| `epiwrbe.dll` | The Tornado Backend DLL. |
| `majiccom.c` | C source file for the VxWorks modification. |
| `examples\` | A directory containing some example files to help get you started. The files are listed below:<br><br>Sample 1: `samp1.c`<br>Sample 2: `samp2a.c, samp2b.c`<br>Sample 3: `samp3.cpp, classx.h, classy.h classx.cpp, classy.cpp` |

## Installing the Driver

The `epiwrbe.dll` file is the host-side executable that provides the interface to the MAJIC. Before starting up the driver, it must be placed in a directory that is known to the Wind River Target Server (hereafter referred to as the back-end directory). The exact path depends on the Tornado installation, but it is typically something like:

```
\tornado\host\x86-win32\lib\backend
```

To install the driver:

1. Copy the `epiwrbe.dll` file from the distribution package `wrbe` directory into the back-end directory. Also copy the `keyfile.bin` file from the distribution package `bin` directory into the back-end directory

During the setup/installation of your MAJIC unit you were introduced to our MONICE debugger and the required initialization files. `epiwrbe.dll` operates much in the same way as MONICE, and uses the same configuration files. These files (`startice.cmd` and possibly a separate target board initialization file *`board`*`.cmd`) need to be copied either to your back-end directory or to your Tornado project directory. If you have multiple projects that need different `startice.cmd` and/or *`board`*`.cmd` files, they should go in each project directory. Otherwise they should go in the back-end directory.

1. Copy the appropriate `startice.cmd` and *`board`*`.cmd` files to your back-end or project directory.

Later in this manual, you'll be instructed how to add commands to `startice.cmd` to load your VxWorks image. If you have multiple projects, this means that the `startice.cmd` file will be project-specific and should go in the project directory.

# 2 Configuration/Loading of VxWorks

Task mode debugging requires a WDB agent running on the target system. With the MAJIC, this agent communicates with the host through the JTAG connection. The VxWorks image must be modified to add special communications drivers for this mode.

## Customizing VxWorks for MAJIC

Below is an outline of the steps (using Tornado II) necessary to create a VxWorks project, modify it to be MAJIC aware, and build it. It is assumed that you already have a VxWorks BSP (Board Support Package) available for your target board, or

### Creating a Project

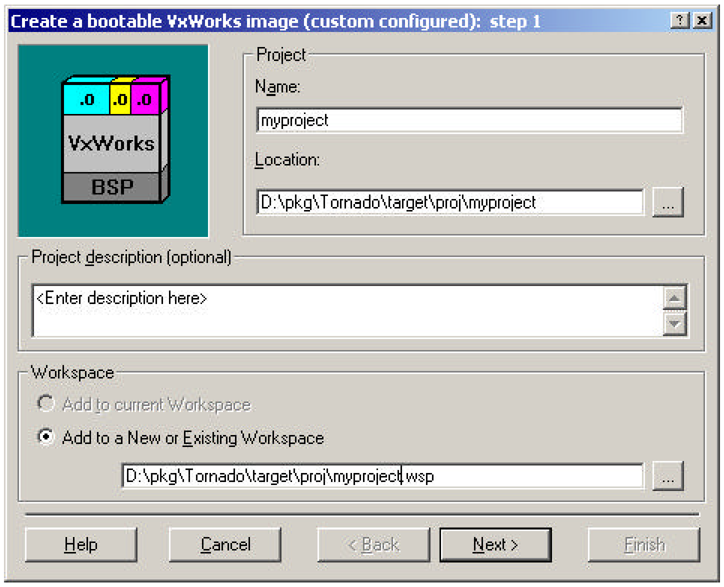
First we must create a Tornado Project to begin our modification work with.

have already customized one for your own hardware.

1. Start up Tornado and select the File/New Project Menu option. This brings up up the Create Project in New/Existing Workspace dialog.

2. Make sure the New tab is selected and then select the Create a bootable VxWorks image (Custom configured) item from the list.

3. This brings up a new dialog titled Create a bootable VxWorks image (Custom configured) step 1. Modify the project name, file and directory location information to suit your needs, and then click the `[Next >]` button.

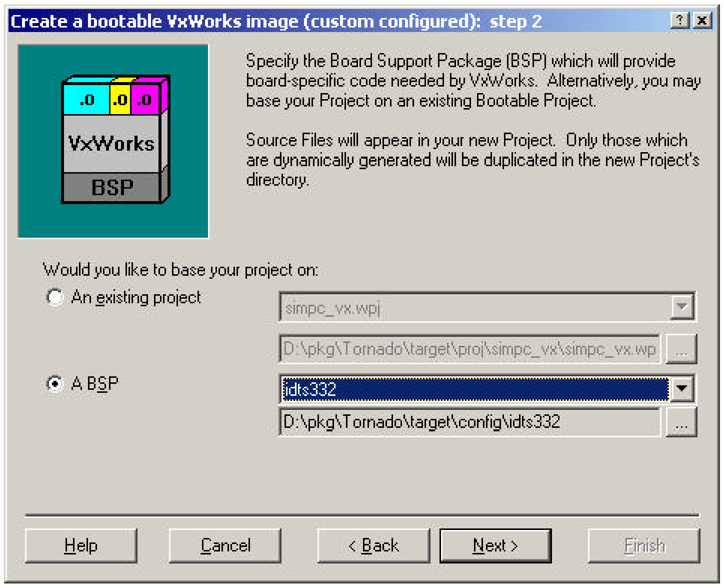

Create a bootable VxWorks image (custom configured): step 1

Project
Name: myproject
Location: D:\pkg\Tornado\target\proj\myproject

Project description (optional)
<Enter description here>

Workspace
Add to current Workspace
Add to a New or Existing Workspace
D:\pkg\Tornado\target\proj\myproject.wsp

4. This brings up step 2 where you select the board support package to base your VxWorks configuration on:


Create a bootable VxWorks image (custom configured): step 2

Specify the Board Support Package (BSP) which will provide board-specific code needed by VxWorks. Alternatively, you may base your Project on an existing Bootable Project.

Source Files will appear in your new Project. Only those which are dynamically generated will be duplicated in the new Project's directory.

Would you like to base your project on:
An existing project: simpc_vx.wpj, D:\pkg\Tornado\target\proj\simpc_vx\simpc_vx.wp
A BSP: idts332, D:\pkg\Tornado\target\config\idts332

Select the "A BSP" option and then select the specific package you wish to base your project on and click the `[Next >]` button.

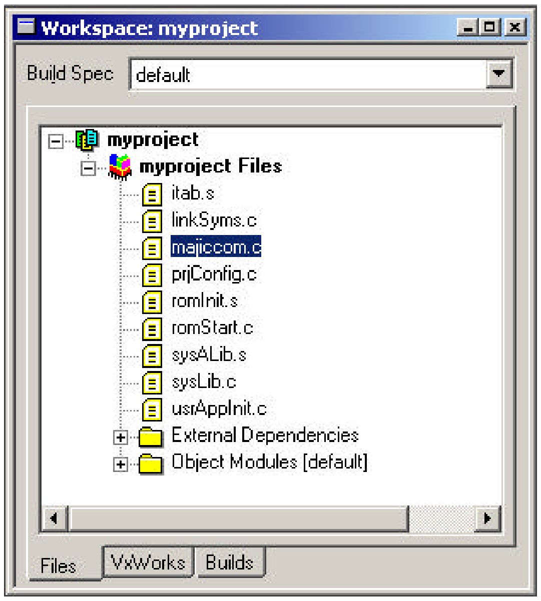
5. This brings up the step 3 dialog which is a simple confirmation page. Click the `[finish]` button and your project and workspace will be created.

## Customizing the Project

Now you are ready to modify the project to work with your MAJIC unit. The process is quite simple. First we need to add one source file to the project and then configure the WDB agent connection method. The steps to do this are shown below.

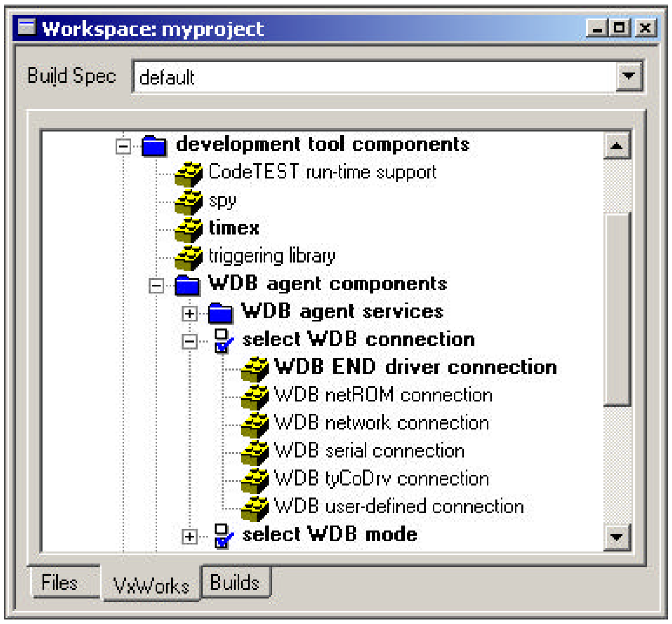
1. First, copy the file `majiccom.c` from the EPI distribution to your new project directory. The distribution directory is `<`*`your installation dir`*`>\wrbe`. Your project directory is something like: `\tornado\target\proj\<`*`your project name`*`>`

2. Next, add the new file your Tornado Project. Select the menu item Project > Add/Include > File. Browse to your project directory, select the `majiccom.c` file and click the `[ok]` button. Your workspace window should now look something like:



myproject
  myproject Files
    itab.s
    linkSyms.c
    majiccom.c
    prjConfig.c
    romInit.s
    romStart.c
    sysALib.s
    sysLib.c
    usrAppInit.c
    External Dependencies
    Object Modules [default]

3. Now click the VxWorks tab in your workspace window. Find the development tools components item and open it by clicking the `[+]` box. Under the open tree branch, find the WDB agent components item and open it. Lastly, find the “select WDB connection” and open it. You should then see something like:



4. Note which connection method is selected (currently bold), right-click on it and select Exclude 'WDB …' from the menu. This brings up the Exclude Component(s) dialog. Click the dialog's `[OK]` button. You may get a follow-on dialog about your project having configuration errors. Ignore this message as the next step will resolve the errors.

5. Right click on the WDB user-defined connection item and select the menu item Include 'WDB user-defined connection' This brings up the Include Component(s) dialog. Click the `[OK]` button.

6. Lastly, we need to build the project. Select the Build/Build All menu item. On some PC hosts, you may get a fatal “File creation error“ error when building your project. If so, edit the file named `makefile`, created by the project manager in your project directory. Search for the redirection symbol (>), and change the forward slashes in the following pathname to backslashes. This will probably occur in two places in the `makefile`. After fixing this problem, you may get some build time warnings, but the build should complete resulting a new VxWorks image in the `default` sub-directory of your project directory.

# Loading VxWorks

There are two different ways to operate MAJIC in conjunction with your VxWorks kernel. They are described below along with the setup changes that you need to make.

- ♦ MAJIC Operation as a Bootloader.

## MAJIC Operation as a Bootloader

If you would like MAJIC to load your VxWorks image into RAM on the target, the sequence of operations is as follows:

- ♦ Running with a pre-loaded VxWorks image.
- ♦ Previously, you copied the file `startice.cmd` into the Tornado back-end directory or your project directory. Edit this file adding a line at the end like the following, but accounting for project name/directory differences:

  ```
  L \tornado\target\proj\<proj_name>\default\vxworks
  ```

  This line tells the MAJIC back-end (`epiwrbe.dll)` to load the referenced VxWorks image at startup. Note that the load command supports both COFF and ELF file formats.
- ♦ If your target uses an ARM or Intel XScale processor, and you are not using the `-Esystem` option to start up in system mode, you must also add the following line:

  ```
  EO vector_catch = 0
  ```

  This line tells the MAJIC not to intercept any of the ARM exception vectors, since the VxWorks image will handle all exceptions.

## Running with VxWorks in Flash Memory

If the VxWorks image is already present in flash memory on the target board, or if an independent method is used to download the image (such an ftp server) follow these steps:

- ♦ You are now ready to start the Tornado Target Server.
- ♦ Previously, you copied the file `startice.cmd` into the Tornado back-end directory. Edit this file and add a line at the end like the following, but accounting for project name/directory differences:

  ```
  LS \tornado\target\proj\<proj_name>\default\vxworks
  ```

  This line tells the MAJIC back-end (`epiwribe.dll`) to load the symbols from the referenced VxWorks image at startup (but not the code), and set the pc to the entry point. Note that the load command supports both COFF and ELF file formats.

- ♦ If your target uses an ARM or Intel XScale processor, and you are not using the `-Esystem` option to start up in system mode, you must also add the following line:

  ```
  EO vector_catch = 0
  ```

  This line tells the MAJIC not to intercept any of the ARM exception vectors, since the VxWorks image will handle all exceptions.

## Advanced Downloading

The load command can also be used to download additional program images and control the sections downloaded. For reference, the full load command syntax is:

- ♦ You are now ready to start the Tornado Target Server.

```
L[S] [[-options]... [filename]]...
```

The LS command is short-cut that tells the Load command to only read in the symbol information and process the programs (*filename*) entry point. Each of the *filename* arguments and their respective *options* are remembered for future Load commands (with no arguments) until explicitly changed in a subsequent Load command with arguments. So if there is no *filename* explicitly specified, the current program is reloaded. In this case the symbol table is not normally reloaded, but you can use the **-o** option to force a reload. If new files to be loaded are specified, the existing global symbol table is purged and symbols are loaded from the new files by default.

*options* These options control the loading of executable files:

`o` Next parameter specifies section types to load:

| | |
|---|---|
| `t` | text |
| `d` | data |
| `b` | bss |
| `l` | rdata |
| `s` | symbols |

`n` Negate: "`-no [t|d|b|l|s]...`" means load all but the specified sections.

*filename* specifies an executable file to be loaded according to *options*.

# 3 Target Server Operation

The link between the host debugger and the MAJIC interface is provided by the Tornado Target Server. EPI's target server back-end is somewhat unique in that it launches a separate command shell running a subset of EPI's MON debugger command language. This shell is referred to as the EPIWRBE Console Window. Normally, this window hides itself as an item or icon on the taskbar and or desktop.

### Via the Tornado GUI

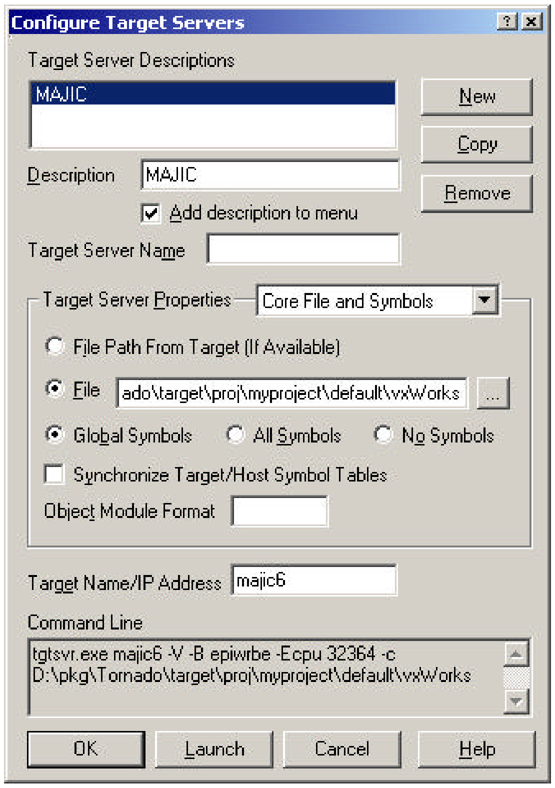
To connect the Tornado environment to the MAJIC, select the menu item Tools > Target Server > Configure. In the window that comes up, perform the following selections:

## Starting the Target Server

- Select epiwrbe as the back-end.
- Fill in the Description box with a title of your choosing.
- In the Target Name/IP Address box enter the IP address or hostname of your MAJIC. If you are not sure about your MAJIC's IP address or have not set one up yet, please refer to the MAJIC User's Manual for details.
- With the Target Server Properties drop down box set to Back End, fill in the Other Options box with
  `-Ecpu <`*`your CPU name`*`> -ETver <`*`Tornado version`*`>`.
  *`Tornado version`* is one of the following decimal values specifying the version of Tornado you are using:
  `200` – Version 2.0.x
  `210` – Version 2.1.0
  `211` – Version 2.1.1

- Change the Target Server Properties drop down box to Core File or Symbols, select the File option then click the browse [...] button to browse to and select your VxWorks project image.

When these steps are all done your configuration dialog should something like:



Description: MAJIC
[✓] Add description to menu
Target Server Properties: Core File and Symbols
[✓ File]: ado\target\proj\myproject\default\vxWorks
[✓ Global Symbols]
Target Name/IP Address: majic6

```
tgtsvr.exe majic6 -V -B epiwrbe -Ecpu 32364 -c
D:\pkg\Tornado\target\proj\myproject\default\vxWorks
```

Note that the next section covers other options supported by our back-end (`epiwrbe.dll`) that can be specified in this dialog.

You are now ready to launch the target server (via the `[Launch]` button). After

## Via a DOS or Unix Command Shell

The target server can also be run from a DOS command line. Assuming the tornado executables are on your current path:

launching, please proceed to the section *EPIWRBE Console Window* on Page 15

```
shell> tgtsvr -B epiwrbe -Ec <cpu_name> -ETv <version>
-c VxWorks_path_name hostname/device
```

## Target Server and epiwrbe.dll Options

Below are some of the useful options supported by the target server running in a MAJIC environment. Note that options beginning with `-E` are unique to our MAJIC back-end environment and can be entered in Tornado via the other options edit field in the Back End Target Server Properties dialog.

`-B epiwrbe`

(required) tells the target server that the interface is to be `epiwrbe.dll` (a MAJIC box).

`-c` *`VxWorks`*

Tells the target server where to find the VxWorks image that is to be used. Note that this does not cause the image to be downloaded. In fact, this image may have been programmed into system ROM or may have been downloaded previously. Either way, the target server needs to be able to read the image symbol table.

`-Ec[pu]` *`cpu_name`*

Identifies the CPU in use on the target board. Please see your MAJIC User's Manual for details.

`-ETv[er]` { 200 | 210 | 211 }

Specifys the version of Tornado you are using:

`200` – Version 2.0.x
`210` – Version 2.1.0
`211` – Version 2.1.*`cpu_name`*

Identifies the CPU in use on the target board. Please see your MAJIC User's Manual for details.

`-El[ittle]`

Identifies the CPU to the back-end as Little Endian based. The default is Big Endian.

`-Elog` *`log_file`*

Tells the back-end to create a log file of back-end operations. *`log_file`* specifies the filename and location. If *`log_file`* is specified as `console`, log messages are displayed in the EPIWRBE Console Window. Two `-Elog` options can be specified to enable simultaneous logging to a file and the console window. Logging can also be enabled via an environment variable and described in the chapter *Advanced Topics* on page 22.

`-Es[ystem]`

Tells the back-end not to auto-start the VxWorks image. Instead after MAJIC initialization, the PC is left at the reset address or the loaded VxWorks image entry point. This allows you to use the Tornado debugger to debug the VxWorks kernel in system mode.

`-EX[pt]` *`MajicXpt_address`*

This option provides a back-door method of providing the target server the needed address for communication. To get the address of the *`MajicXpt_address`*, open a shell window and use the command:

```
objdumpmips --syms vxworks | grep MajicComBuf
```

*`hostname/device`*

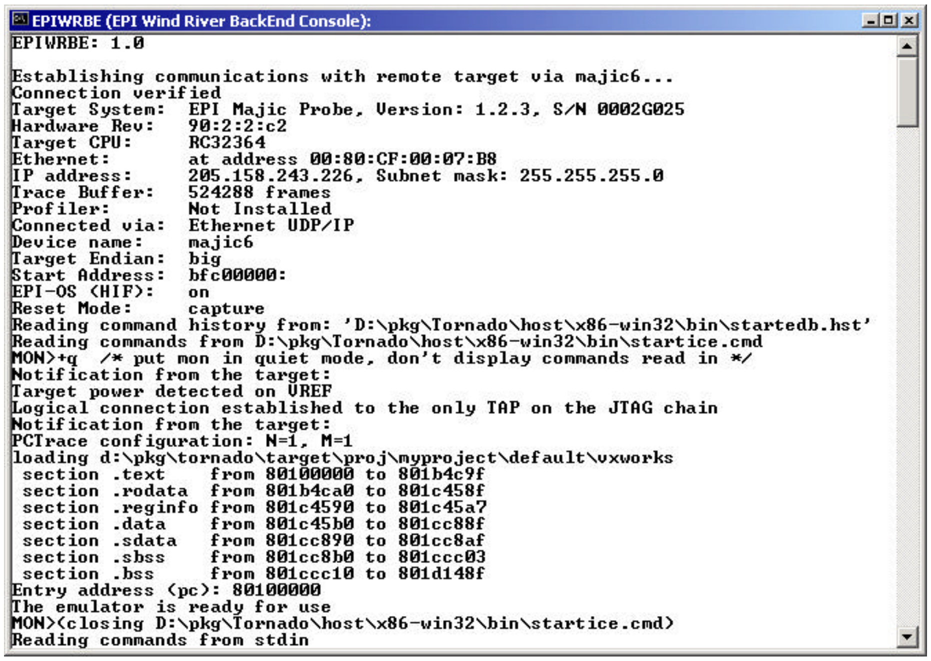
Is what specifies the communication device (COM1 for example) or IP address/hostname used to communicate with the MAJIC unit.

When the Target Server is run from the command line, the Logon Banner message from the MAJIC is displayed as the target server makes contact with the MAJIC. This block of 5 (or so) lines is coming directly from the `epiwrbe.dll` interface module. The other messages are coming from the Wind River Target Server.

## EPIWRBE Console Window

The EPIWRBE Console window is an advanced feature of EPI's Tornado back-end. It provides much of the functionality available in EPI's MONICE debugger enabling you access to MAJIC's advanced debug features that the Tornado interface is lacking. It also serves to provide visual feedback on the setup/status of your Tornado to MAJIC connection.

Below is a sample Console window. This window comes up at target server initialization time and automatically hides itself down on the task bar. Note that the Tornado target server itself also creates a log window and hides itself on the task/tray bar.



```
EPIWRBE: 1.0

Establishing communications with remote target via majic6...
Connection verified
Target System:   EPI Majic Probe, Version: 1.2.3, S/N 0002G025
Hardware Rev:    90:2:2:c2
Target CPU:      RC32364
Ethernet:        at address 00:80:CF:00:07:B8
IP address:      205.158.243.226, Subnet mask: 255.255.255.0
Trace Buffer:    524288 frames
Profiler:        Not Installed
Connected via:   Ethernet UDP/IP
Device name:     majic6
Target Endian:   big
Start Address:   bfc00000:
EPI-OS (HIF):    on
Reset Mode:      capture
Reading command history from: 'D:\pkg\Tornado\host\x86-win32\bin\startedb.hst'
Reading commands from D:\pkg\Tornado\host\x86-win32\bin\startice.cmd
MON>+q  /* put mon in quiet mode, don't display commands read in */
Notification from the target:
Target power detected on VREF
Logical connection established to the only TAP on the JTAG chain
Notification from the target:
PCTrace configuration: N=1, M=1
loading d:\pkg\tornado\target\proj\myproject\default\vxworks
 section .text    from 80100000 to 801b4c9f
 section .rodata  from 801b4ca0 to 801c458f
 section .reginfo from 801c4590 to 801c45a7
 section .data    from 801c45b0 to 801cc88f
 section .sdata   from 801cc890 to 801cc8af
 section .sbss    from 801cc8b0 to 801ccc03
 section .bss     from 801ccc10 to 801d148f
Entry address (pc): 80100000
The emulator is ready for use
MON>(closing D:\pkg\Tornado\host\x86-win32\bin\startice.cmd)
Reading commands from stdin
```

The MAJIC User's Manual contains a complete description of MON's command language. Users using a MAJIC unit supporting trace should spend some time learning the trace related commands of MONICE described in the MAJIC User's Manual.

Note that this window shuts down automatically when the target server is shutdown.

## Exiting the Target Server

When the debugging session is complete, the target server may be shut down by double-clicking its icon on the tray bar, then clicking on Shutdown button. Upon exiting the target server, the MAJIC will stop the target processor (leaving it in debug mode). Note that shutting down the target server also shuts down the EPIWRBE Console Window. Please do not shutdown the EPIWRBE Console Window via either it's title bar [x] button, or close menu item.

# 4 Debugging Modes

Earlier we described the differences between system and task debug modes (refer back to *Basic Concepts* on Page 1 for details). Below, we go into more details on setting up to use the different modes.

## System Mode Debugging

With an emulator (such as MAJIC), System mode debugging does not depend on software resources on the target. This means that the emulator may be used for debugging a new version of the operating system itself, or it may be used for debugging standalone software modules that do not use a standard operating system. For example, a processor board may have some simple diagnostics that reside in ROM and do no more than read and write to a few registers. MAJIC makes it easy to

### Getting Started

The steps below outline the steps to getting System Mode started.

step through and debug such standalone code.

1. Add the `-Esystem` option to the Other Options field of your Tornado Target Server Dialog (See Page 12).

2. Start the target server. This starts the EPIWRBE Console Window.

3. Start the Tornado debugger. Menu Tools > Debugger…

4. Launch the debugger command window. Menu Debug > Debug Windows > Debug Command Line

5. At the debugger command line, type the command: `attach system.` This bring up a disassembly window starting at the entry point to the VxWorks image

You can now proceed to use Tornado debug commands to step, set breakpoints, etc.

# Task Mode Debugging

Task mode debugging requires that first modify and build the VxWorks as described in section *Customizing VxWorks for MAJIC* on Page 5. After doing so you can

## Task Mode Debugging Examples

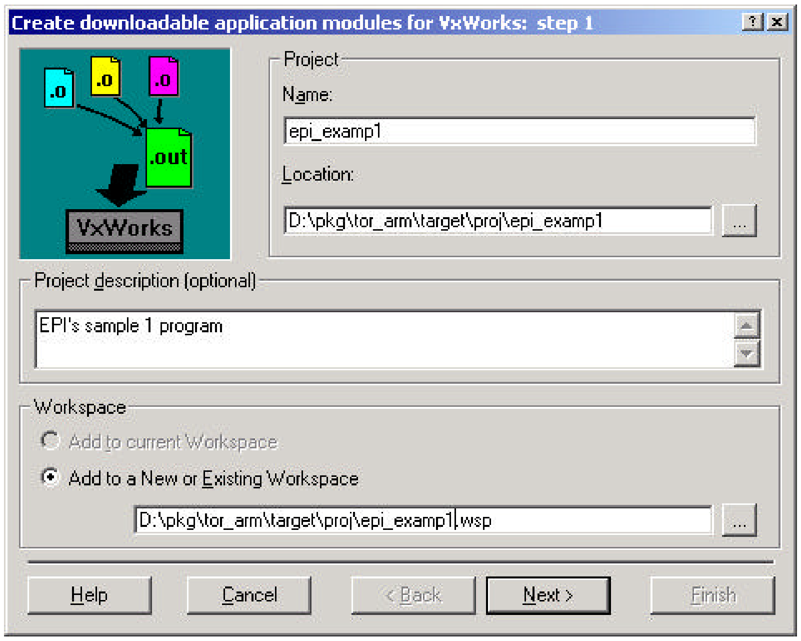
To build each example in Tornado II:

proceed to build some sample applications and debug them.

1. Select New Project... from the File menu. This displays a dialog box labeled Create Project in New/Existing Workspace.

2. Select the option: Create a downloadable application module for VxWorks.

3. In the next dialog box, enter the Project Name, Location, Workspace name and (optionally) Project description. See example below


Create downloadable application modules for VxWorks: step 1

Project
Name: epi_examp1
Location: D:\pkg\tor_arm\target\proj\epi_examp1

Project description (optional)
EPI's sample 1 program

Workspace
( ) Add to current Workspace
(●) Add to a New or Existing Workspace
D:\pkg\tor_arm\target\proj\epi_examp1.wsp

4. Click the `[Next >]` button and in the next dialog In the next dialog box, click the radio button for A toolchain, and select the correct toolchain for your target system. Click this dialogs `[Next >]` button and the following dialogs `[Finish]` button and your project is created.

5. Now that the project has been created, copy the appropriate sample file(s) to your new project directory and use the menu item Project select Add/Include > File to insert them into your Tornado project.

6. Build the project via the menu item Build > Build.

## Sample Program Files

The directory under your <`epitools`>`\wrbe\examples` directory contains three samples program describe below.

Sample 1 is a very simple example Tornado program. Build it for your target system, download it into the target and run the function `Samp1()`. It spawns a task named `runThis()`. There is only one source file: `Samp1.c`.

Sample 2 allows you to build the object files for your target system and download them separately. There are two source files in this sample program (`Samp2a.c, Samp2b.c)`. Run the `Samp2()` function (which is in `Samp2b.`c). This will be dynamically linked to some functions in `Samp2a.c.`

Sample 3 is C++ based and contains the files `ClassX.cpp, ClassY.cpp, Samp3.cpp` and the class header files `ClassX.h`, `ClassY.h`. The entry point for sample3 is `Samp3()` in `Samp3.c`.

# 5 Advanced Topics

## Troubleshooting

If you are having start-up issues with `epiwrbe.dll,` consider the following:

- Does your MAJIC unit function correctly with MONICE and the same `startice.cmd` file.
- Make sure that the VxWorks image is built with the proper address range for your target board.
- If you are loading your VxWorks image via the Load (`l`) command, make sure your target board has enough memory installed to load theVxWorks image.
- If your target board has a memory controller needing initialization at power-on, make sure either the on-board boot PROM properly initializes it, or when EPIWRBE starts, that the proper initialization script is run to initialize your board.
- Try running the memory test feature of MAJIC.  See the MONICE documentation in your MAJIC User's Manual for details.

## Logging

If you encounter problems with the target server you can try using various logging

### Console Window Logging

EPI's MONICE debugger contains a rich set of output logging commands.  For example, to log all the session output to a file,  simply add the following command to your `startice.cmd` file in the Tornado backend directory.

methods to help explore the problem:

```
MON> FW O c:\logging\epiwrbe.log
```

A complete description of the `FW` (file Write) command can be found in the MON

## Target Server Interface Logging

This form of logging writes both service call details of all information passing between the Tornado target server and EPI's backend `epiwrbe.dll,` plus all the session data appearing in the EPIWRBE Console Window.

reference section of your MAJIC User's Manual

To enable logging for the `epiwrbe.dll` interface, you can define an environment symbol `MAJICLOG`: in your `autoexec.bat` (or equivalent) startup file. For example:

```
set MAJICLOG=c:\logging\majic.log
```

Alternatively, you can set the target server/`epiwrbe.dll`'s `-Elog <`*`filename`*`>` option in the Other Options field of the target server configuration dialog (see example dialog on Page 12).

Note: Typically, this kind of logging information is not as useful to an end user as it is to an EPI support representative. The instructions are provided here for reference in case we need you to capture a log file for us.

# 6 Technical Support

For additional information, questions, or comments, you can contact EPI via the addresses and phone numbers listed below: E-Mail tends to be the best method for support issues.

When contacting EPI please provide your serial number and as much information as possible on the nature of the problem. Your serial number can be found in the epitools installation directory as a file named `XXXXXX.sn.`

| | |
|---|---|
| E-Mail: | `support@epitools.com` |
| | `sales@epitools.com` (for sales related issues) |
| Web FAQs: | `www.epitools.com` |
| Telephone: | `(408) 957-0350` |
| FAX: | `(408) 957-0307` |