Application Note
0380-0252-10 Rev 1.2

Embedded Performance, Inc.
606 Valley Way, Milpitas, CA 95035
Telephone: (408) 957-0350
FAX: (408) 957-0307
e-mail: support@epitools.com
web: www.epitools.com

Using a MAJIC® Probe with MontaVista™ Software

0380-0252-10 Rev 1.2

January 25, 2005

Introduction

A MAJIC® Intelligent Debug Probe performs debug services by using the target processor's JTAG interface to access registers within its *Debug Support Unit* (DSU). Using the MAJIC® probe to debug your MontaVista Linux kernel will eliminate the need to rely on a debug agent running on the target. This allows you to debug the kernel at the very early stages of development and have full control of the processor.

This application note explains the steps necessary to configure and use MontaVista software tools with a MAJIC® probe. It will provide specific information on:

- Connecting a MAJIC® probe using GDB to debug the MontaVista Linux Kernel,
- Setting up the debug environment for simultaneous kernel level debug and application level debug,
- Setting up the debug environment for loadable kernel module debug.

Notes:

- This is not intended to be a MontaVista tutorial. Nor is it a tutorial on GDB itself; it explains how to connect GDB to the MAJIC® probe, but assumes that the reader is familiar with the GDB debug environment. Please contact MontaVista technical support or visit their web site http://www.mvista.com for additional assistance.
- It is presumed that you already have the EDT and MontaVista 3.1 software installed, have the MontaVista Linux Support Package (LSP) installed for your target board, and access to the MontaVista™ *Cross-Development Quick Start* document for host and target board configuration. The EDT package is normally distributed on a CD included with the MAJIC® probe. Before installing the EDT package, it's a good idea to check the EPI website to make sure you have the current release.

Additional Documentation

Additional documentation for the EDT software package and MAJIC® probe can be found in the `./manuals` directory of the EDT installation, and the `./manuals` directory on the EDT software package CD. The `edta_doc_index.html` (for ARM and XScale), or `edtm_doc_index.html` (for MIPS), provide easy access to all the on-line documentation. The following documents will prove particularly useful through the course of this application note:

MAJIC® User's Manual	Complete information on configuration and operation of the MAJIC® probe, and the MON command language.
Using a MAJIC® Probe With A Linux PC	Application note that outlines installation of EDT software, and using the MAJIC® probe on a Linux hosted environment.
MDI for MAJIC® User's Guide	Details on the MDI interface to the MAJIC® probe, and the MDI configuration file.
gdb-readme.txt	Additional technical details on setting up and using GDB and MDI-Server with a MAJIC® probe.
MontaVista™ Linux® User's Guide	Complete information on configuration and installation of the MontaVista software tool suite.
Cross-Development Quick Start Guide	MontaVista's Linux Support Package (LSP) documentation that provides detailed information on software installation, host and target board configuration.

Getting Support

Please do not hesitate to contact our technical support group if you have any questions or need assistance in configuring the MAJIC® probe for your system. We recognize that these issues are complex, and are committed to making sure our tools work well for you.

The Debug Environment

Chapter 3 of the *MAJIC® User's Manual* explains the configuration process and the files that are involved in detail. The following section outline the basic steps necessary to configure the MontaVista Linux kernel for use with the MAJIC® probe, and the GDB debug environment.

MAJIC® Configuration Files

Every target system is different, so the MAJIC® probe needs information about the target system design in order to operate correctly. The primary file, called `startice.cmd`, configures the JTAG interface and certain capabilities of the MAJIC® probe. In some cases `startice.cmd` will call in a board initialization file to initialize the target hardware, and declare the memory map of the target board.

As part of your EDT software installation, we provide you with a `./targets` directory that contains validated configuration files for many common reference platforms. These can be used as is if you have one of these platforms, and they also serve as a template for creating custom initialization files for you own target hardware.

Configuring the MontaVista™ Linux Kernel

To use the MAJIC® probe for kernel debugging, you will need to configure the MontaVista kernel by disabling `KGDB` and turning debug information on.

1. The MontaVista software tool suite provides a configuration tool which allows you to customize your Linux kernel. To configure your kernel run the following command in `/<your_home>/montavista/devrocket/workspace/<kernel_project_name>`.

   ```
   $ make menuconfig
   ```

2. Select the `Kernel hacking` menu.
3. Enable the `Include debugging information in kernel binary` option. Then enable the `Kernel debugging` option and any of its sub-option you have interest in. Finally, make sure to disable the `KGDB support` option.
4. Build your kernel following the instructions outlined in the *Build the Kernel Without DevRocket* section of the *MontaVista™ Linux® User's Guide.* Make sure to run the following command in the Linux shell before executing **`make clean`**.

   ```
   $ export PATH=/opt/montavista/pro/host/bin:/opt/montavista/pro/devkit
   /<arch>/<processor>/bin:$PATH
   ```

   Note: Enter the above command in a single line.
5. Please refer to the MontaVista *Cross-Development Quick Start* document for further host and target board configuration.

Starting MDI-Server

GDB communicates to the MAJIC® probe through a separate program named MDI-Server. GDB sends "remote protocol" debug requests to MDI-Server which then sends corresponding debug requests to the MAJIC® probe. However, the MDI-Server program must be running before GDB can connect. With this connection method you don't need to build a MAJIC-specific GDB, but can use an off-the-shelf GDB (or any of the GUI front-ends available for GDB) and connect to MDI-Server with the "target remote" command.

MDI-Server, by way of the MDI shared library, is responsible for managing the details of the MAJIC® probe configuration. MDI-Server reads in JTAG settings and target board configuration from the configuration files. CPU information and communication parameters are set via the `epimdi.cfg` configuration file instead of using command line switches. Please refer to the *Using a MAJIC® Probe With A Linux PC* application note for details on creating `epimdi.cfg`.

Once you have your MAJIC probe configuration files and `epimdi.cfg` file prepared, you are ready to start MDI-Server. You should always start MDI-Server from the directory containing the `epimdi.cfg` file. Open a Linux shell and execute the following minimum command for launching MDI-Server:

```
$ ./mdi-server -l ./mdi.so
```

When MDI-Server is running it simply waits for an instance of GDB to connect on port 2345. In some cases, additional start-up options beyond the minimum invocation line above may be desired or even required. Please refer to the *Using a MAJIC® Probe With A Linux PC* application note for more details.

Loading a MontaVista™ Kernel Image

For those unfamiliar with `minicom`, it is a serial communication application analogous to Windows' `HyperTerminal`. It provides a means to interact with the target board's boot loader and the MontaVista Linux kernel. Open a second Linux shell, become super user, and execute the following command.

```
# minicom
```

The `minicom` application will need to be configured correctly to successfully connect to your target board. Please refer to the MontaVista *Cross-Development Quick Start* document for setup instructions. This same document will also outline in detail how to load the embedded Linux kernel. However, before you load the kernel, you should start an instance of GDB.

Starting GDB

Before starting GDB, you must start the MDI-Server program as described above so that it will be ready to complete the interface between GDB and the MAJIC® probe. Once MDI-Server is ready, open a third Linux shell and start the GDB that is included in your target board's MontaVista LSP. For example,

```
$ /opt/montavista/pro/devkit/arm/xscale_be/bin/xscale_be-gdb
```

From the `(gdb)` prompt enter the following commands to load your file and connect to the target and begin debug. Note that these commands can be placed in a GDB startup script so that they are run automatically at execution.

```
(gdb) set heur 0                          /* MIPS only—see note below */
(gdb) set remoteti 10                     /* Remote timeout—see note below */
(gdb) file /path/to/your/vmlinux_file     /* Read in debug information */
(gdb) target remote localhost:2345        /* Open MDI connection to MAJIC */
```

Notes:

- GDB has an internal variable named `heuristic-fence-post`. It controls how far back from the current PC GDB will scan memory looking for a function start. This variable should default to 0, but it may not in all GDB builds. If it is not 0, GDB may try to access invalid memory locations below the PC. This is primarily a concern for MIPS targets, where the initial PC value when connecting is usually 0xbfc00000 (the reset vector), and addresses in the 0xbfbxxxxx range are invalid.
- When GDB first connects, it can take a few seconds for MDI-Server to open the MDI connection to the MAJIC® probe and process the `startice.cmd` startup command file. Setting the internal GDB variable named `remotetimeout` prevents most GDB builds from timing out prematurely.
- The `vmlinux` file is the binary that contains the kernel image along with symbolic debug information.

At this point, you should see a successful MDI connection in the MDI-Server Linux shell. The output should be something similar to the following:

```
Accepted gdb MDI connection.
Capabilities:
        TraceOutput
        TraceCtrl
End of Capabilities List
Devices:
Index   Name
[0]     my_ixp425 via my_205.158.243.200
Selecting device [0] of 1
Notification from the target:
Target power detected on VREF
Auto JTAG detection process detected 1 TAP
JTAG connection established

JTAG ID Register:  19274013
Done.
```

Because the MAJIC® probe has reset and stopped the processor upon JTAG connection, you need to run the following command.

```
(gdb) continue
```

Now in the minicom Linux shell, you should see a boot loader prompt. You are now ready to download your MontaVista Linux kernel onto your target board. Because the command(s) to do this differs from target to target, you should refer to the MontaVista *Cross-Development Quick Start* document for your target board for details.

Note: Depending on the reset management of the target design, sometimes an explicit reset is necessary before the GDB `continue` command can be issued. To reset the target board, execute the following MON command. Please refer to the *MON Commands* section below for more information.

```
(gdb) mon rt                    /* MON command to reset target */
```

Once the kernel uncompresses, you can stop GDB (control-C) to set breakpoints and run, or single step through your kernel.

MON Commands

The GDB `monitor` command can be used to pass MON commands through to the MDILib, as shown in the examples below. This allows access to the MAJIC® probe features that are not available in GDB, such as trace display, and access to all CPU registers (and user defined registers) rather than the limited set that GDB knows about. Appendix C of the *MAJIC® User's Manual* provides a table showing which MON commands are supported by MDILib, and Chapter 5 documents the MON command language in detail.

```
(gdb) mon di                    /* Pass  di  command to MDILib and display response */
(gdb) mon h                     /* Pass  h  command to MDILib to display help on MON commands */
(gdb) mon h h                   /* Display help on using MON help */
(gdb) mon h d                   /* Display help on MON Display commands */
(gdb) mon h dw                  /* Display help on MON Display Word command */
(gdb) mon fr c cmdfile          /* Read and execute a MON command script file */
```

Application Level Debug

Using a MAJIC® probe along with MontaVista Software provides more debug capability by giving you a way to have simultaneous kernel level debug and application level debug and not just application level debug alone. The following section describes the necessary communication setup process using GDB and gdbserver.

For your application to be available on the target board, you will need to copy your application into the directory of the NFS mounted root file system on the host. If you want to have source level debug, make sure to copy the source files there also. Open a fourth Linux shell and execute something similar to the following:

```
$ cd /opt/montavista/pro/devkit/mips/fp_le/target/root
$ cp /path/to/application .
```

When this is done, launch a new instance of GDB from your MontaVista Linux Support Package installation. Make sure that you launch GDB from your `/path/to/application` directory. In the shell where you launched `minicom`, run `gdbserver`. Be aware that you first have to release control of the processor by executing `continue` in the first GDB window.

```
# gdbserver host:socket_number your_application_executable
```

Then in the new instance of GDB, type the following commands:

```
(gdb) target remote IP_of_target:socket_number
(gdb) symbol-file your_application_executable
```

Now you are setup to debug your application, like setting a breakpoint and running to it, or single stepping through the application. If the execution of your application causes a kernel breakpoint to be hit, then you can witness simultaneous kernel level debug and application level debug in the two instances of GDB. Note that when a kernel breakpoint is hit in the GDB/MDI/MAJIC environment, the CPU is stopped, so the GDB/gdbserver environment is unusable until execution later resumes.

Loadable Kernel Module Debug

Another capability you have with the MAJIC® debug probe and the MontaVista Software tool set, is the ability to debug loadable kernel modules (LKM). The following section describes the necessary debug configuration.

Three things are required to debug the kernel module. First, the loadable kernel module must be built with debug information. Second, the module needs to be saved in the NFS mounted root file system on the host. And third, you will need to know where in the target system's memory the kernel will save the module's text area.

To save the file, run something similar to this in a Linux shell:

```
$ cd /opt/montavista/pro/devkit/mips/fp_le/target/root
$ cp /path/to/<module_name>.o .
```

Now run this similar command from the Linux kernel running on your target board

```
# insmod -m hello-1.o > map
```

to launch the kernel module and store the memory address of where the module's text area is in the `map` file. You should be able to do this from the minicom Linux shell, otherwise, refer to the *Debug Environment* section above. We can see this memory location by

```
# grep .text map
.text           00000050  c68c0060  2**2
c68c0060 T __insmod_hello-1_S.text_L80
c68c0060 t .text
```

Note: Be aware that you first have to release control of the processor by executing `continue` in the already-running first instance of GDB.

Then in GDB, read in the debug symbols of the kernel module.

```
(gdb) add-symbol-file /home/ln/mv_demo_apps/klm_demo/hello-1.o
      0xc68c0060
add symbol table from file "/home/ln/mv_demo_apps/klm_demo/hello-1.o" at
.text_addr = 0xc68c0060
(y or n) y

Reading symbols from /home/ln/mv_demo_apps/klm_demo/hello-1.o...done.
```

The GDB `add-symbol-file` command needs a memory address as a parameter, make sure to use the .text location from the `map` file. At this point, you can set breakpoints and run, or single-step through your loadable kernel module.