

# A P P N O T E S <sup>SM</sup>

## Using A MAJIC® Probe With A Linux PC

Revision: 2.5

Last Modified: November 15, 2006

©Copyright Mentor Graphics Corporation 2006. All rights reserved.

This document contains information that is proprietary to Mentor Graphics Corporation. The original recipient of this document may duplicate this document in whole or in part for internal business purposes only, provided that this entire notice appears in all copies. In duplicating any part of this document, the recipient agrees to make every reasonable effort to prevent the unauthorized use and distribution of the proprietary information.

Trademarks that appear in Mentor Graphics product publications that are not owned by Mentor Graphics are trademarks of their respective owners.

### Introduction

This application note explains the steps necessary to configure and use a MAJIC® Intelligent Debug Probe with a PC running Linux. While the *MAJIC Probe User's Manual* provides complete information on the configuration process and the files that are involved, this application note is an addendum that provides specific information on the following:

- Installing the EPI Development Tools (EDT) software package on your Linux PC using the Linux Installer,
- Installing the EDT software package manually from a command shell,
- Installing a Service Pack Update,
- Connecting to MAJIC probe with the MONICE command line debugger,
- Connecting to MAJIC probe with the GNU GDB debugger,
- Performing maintenance updates on the MAJIC probe.

**NOTE:** This is not intended to be a Linux tutorial; basic familiarity with the Linux environment is assumed. Nor is it a tutorial on GDB itself; it explains how to connect GDB to the MAJIC probe, but assumes the reader is familiar with the GDB debug environment.

## Additional Documentation

Additional documentation for the EDT software package and MAJIC probes is installed as part of the EDT software package. On a Windows PC, use the EDT Launchpad in the Windows system tray to open the EDT Documentation Index. On a Linux PC, open `./manuals/edtX_docs_index.html` in your EDT software installation. The following documents will prove particularly useful through the course of this application note:

<i>MAJIC Probe User's Manual</i>	Complete information on configuration and operation of the MAJIC probe, and the MON command language.
<i>MDI for MAJIC User's Guide</i>	Details on the MDI interface to the MAJIC probe, and the MDI configuration file.
<i>gdb-readme.txt</i>	Additional technical details on setting up and using GDB and mdi-server with a MAJIC probe.

## Getting Support

For additional assistance configuring the MAJIC® probe for use in a Linux environment, please visit <http://www.mentor.com/supportnet/>.

## EPI Development Tools Installation

The EPI Development Tools (EDT) package is distributed on a CD included with the MAJIC probe. The EDT package for Linux runs on any 32-bit x86 Linux environment that has glibc 2.1 or later. Updates are generally distributed on SupportNet. Before installing the EDT package, it is a good idea to check the SupportNet site to make sure you have the current release.

A graphical-based installer, `einstall`, is provided to simplify the installation process. This installer requires that the host be running the X Window System and have the GIMP toolkit library (`gtk+`) installed. If your Linux environment does not support the X Window System, or `einstall` is not able to run correctly due to one or more missing shared libraries, please follow the instructions the `einstall_cmd` command line version in the [Installing Manually](#) section of this application note.

### NOTE:

- For additional installer information, please refer to the file `readme.txt`, located in the same directory as `einstall`.
- The default installation directory is `/opt/edtX`, where X is an “a” to denote ARM/XScale or “m” to denote MIPS. Normally, the directory `/opt` requires root access to create or update files and directories. So you can either login as root or change the install directory to any directory you have full access permissions for.

## Using the Linux Installer

An installation key is required in order to install the EDT software package. Normally the key is sent by email when the MAJIC probe is shipped. If you do not have an installation key for the current software release, use the **Get Key** button in the main `einstall` window to register the MAJIC probe and request the installation key, which will be sent to the email address you provide. After you receive the installation key, rerun `einstall`, copy the key from the email into the main `einstall` window, and proceed with the installation.

## Installing from CD

If you are installing from a CD, place your CD into your CD-ROM drive. If your installation directory requires root access, then you will need to log in as root first. For some Linux hosts, a desktop file browser will open and you can double-click `einstall` in the CD Linux directory to launch the installer.

Otherwise, you can run the script from a terminal. For example,

```
/mnt/cdrom/linux/einstall
```

Once you launch the installer, simply follow the directions on the dialog boxes.

**NOTE:**

If your Linux host does not auto-mount the CD-ROM drive, please become the root user via the following command:

```
su -
```

and mount it with a command similar to:

```
mount /dev/cdrom
```

Before removing the CD, make sure to *unmount* it.

**Installing Updates Downloaded from SupportNet**

1. Download the appropriate EDT software update file from SupportNet via the following link:

<http://www.mentor.com/embedded/customer/downloads/>

2. Un-tar and uncompress the downloaded file as follows:

```
cd download_path // directory in which you saved the downloaded file
tar -xzf edtXXXXX.tgz // name of the downloaded file
```

This creates these five files:

```
edtXXXX_tgz.fc // actual file name is based on release number
einstall
einstall_cmd
eula.txt
readme.txt
```

3. Open a file browser to click on the `einstall` executable, or run it from the command shell as shown in the [Installing Manually](#) section of this application note. Once you launch the installer, simply follow the directions on the dialog boxes.

```
/download_path/einstall
```

4. Use the **Get Key** button to request a new key (the key from prior releases will not work on new releases).
5. Once you receive the new key, rerun `einstall`, paste in the key, and proceed with the installation.
6. When installing a new software release, don't forget to update the MAJIC firmware with the included update. See the [MAJIC Firmware Updates](#) section of this application note.

**Installing Manually**

`Einstall` will not run successfully on all Linux installations due to missing shared libraries. For this case, you can either install the required packages (on RedHat, `gtk+`, `XFree86-libs`, `libstdc++`, `glibc`, `glib` are required), or manually install the tools by following these steps.

1. If you do not have an installation key (An installation key is sent by email when a MAJIC probe is shipped), or you are installing a new software release, then paste the following registration template into a new email, fill in the “?” fields and send it to [epi\\_register@mentor.com](mailto:epi_register@mentor.com). Please set the email’s subject heading to the following:

Register: S/N, company\_name

S/N is the serial number of your MAJIC probe and *company\_name* is the name of your company.

```

--- Manual MAJIC Registration (Linux) ---
Site ID:      ?
Company:      ?
First_Name:   ?
Last Name:    ?
EMail:        ?
Address:      ?
State:        ?
Zip/Prov:     ?
Country:      ?
Software Release: EDT 2.3b
Architecture Used: ?
MAJIC Serial #(s): ?
Host OS:      Linux ?
Target OS:    ?
Comments:
-----

```

2. Please allow one business day for a reply email to be sent which includes an installation key.
3. Upon receiving your installation key, you can complete the installation with the following commands. In double quotes, replace *key* with the installation key. The installation directory defaults to */opt/edta* or */opt/edtm* based on the key, unless you pass in an install directory parameter during invocation.

```

cd /mnt/cdrom/linux // to install from CD (mount the CD first)
cd /____/_____ // to install from SupportNet download
einstall_cmd "key" [install_directory]

```

### Updating PATH

After installing the EDT package, it is recommended to add the */opt/edtX/bin* directory to your search path. This can be done by editing the proper configuration file, which is usually *.bash\_profile*, found in your */home/username* directory for Red Hat users. Alternatively, the PATH can be set for a single console session by running the commands below in that console window.

```

PATH=$PATH:/opt/edtX/bin /* using edta or edtm as appropriate */
export PATH

```

**NOTE:** When adding a path to the PATH variable remember to use a colon (:) to separate each additional path.

## Installing a Service Pack Update

1. If you have not already done so, install the EDT software package as instructed above. Make a note of the root directory where you choose to install the tools.
2. Change directory to the root directory of your EDT software installation and extract the service pack files from there. This is the directory which includes *.bin*, *.ice*, *.targets*, and several other sub-directories. The service pack file will contain updates for some of the installed files, and may also install some new files.

Service Packs are supplied as a password-protected ZIP file (.zip). Extract the EDT files from the ZIP file into the EDT installation directory, substituting the path to, and the name of, the service pack ZIP file. You will be prompted to enter the password, which you were given along with the URL to download the Service Pack.

```
unzip -o /path/edtXXXX.zip
```

3. If the service pack includes a new version of MAJIC firmware, make sure to update the firmware in your MAJIC probe by following the steps in the *MAJIC Firmware Updates* section of this application note.

## EDT Directory Hierarchy

After successful installation of the EDT package, you will find the following subdirectories beneath the installation directory (*/opt/edtX*):

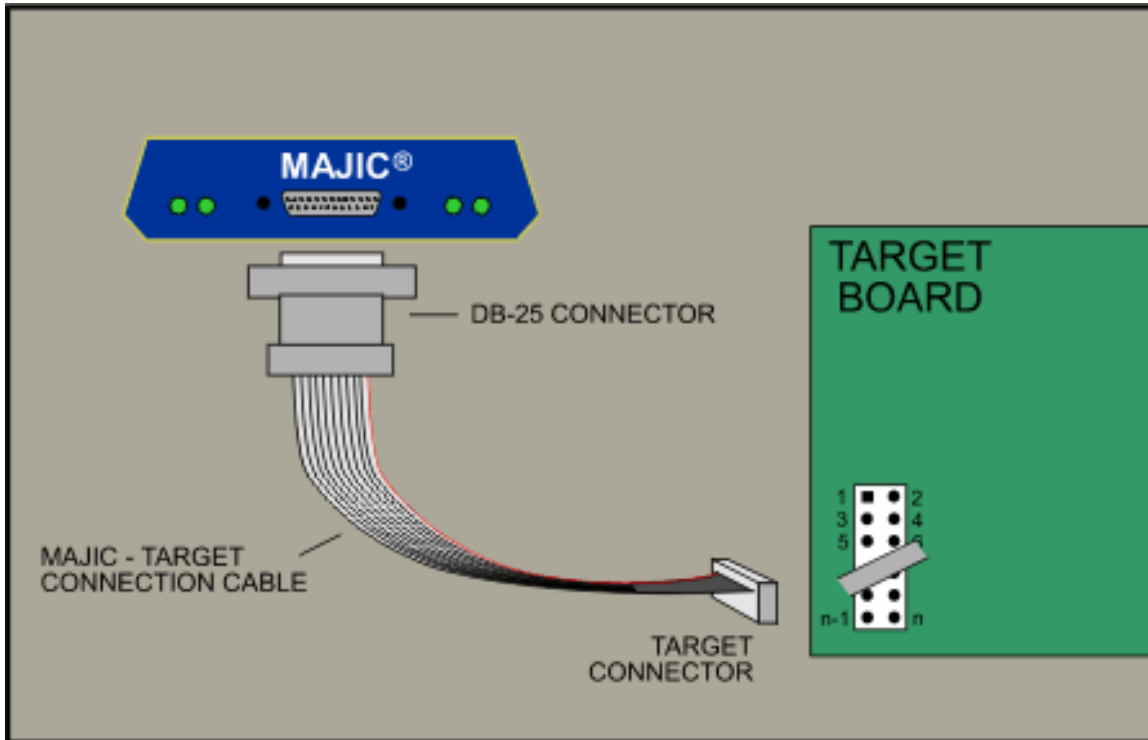
<i>.bin</i>	Executable programs and associated files.
<i>.ice</i>	MAJIC firmware and MAJIC-PLUS <sup>®</sup> trace control PLD files. New users can ignore this for now, but if you are installing an update, you should check the <i>MAJIC_RelNotes.html</i> file for the current MAJIC firmware version number and revision information. Instructions for updating the MAJIC firmware and MAJIC-PLUS trace control PLD are provided at the end of this application note.
<i>.manuals</i>	Installed documentation. The index file <i>edta_doc_index.html</i> (for ARM and XScale), or <i>edtm_doc_index.html</i> (for MIPS), provides easy access to all the installed documentation.
<i>.mdi</i>	MDI back-end files for the MAJIC probe.
<i>.samples</i>	Sample programs. Source code is in <i>.samples</i> , and pre-built executables are in the <i>be/</i> (big endian) and <i>le/</i> (little endian) sub-directories. Lower level subdirectories are provided for different link addresses. Although EPIFlash is really part of the EDT software package, as opposed to a sample program, pre-built versions for big and little endian mode at different RAM addresses are also provided beneath the <i>.samples</i> directory.
<i>.targets</i>	Preconfigured start-up files for many common reference platforms, plus a template for building board initialization files for your own board.

## Hardware Connections

Chapter 2 of the *MAJIC Probe User's Manual* provides full instructions for setting up the MAJIC hardware, but the key points are summarized in the following sections as well.

### JTAG Port

The MAJIC probe is connected to the JTAG debug connector on your target board using a MAJIC Cable Kit. The details of each cable kit depend on the MAJIC model, and the type of debug connector provided on your board. Each cable kit includes an application note with specific information on how to install it. The following image is a typical example.



### NOTE:

1. **Be Careful!** Make sure you have the cable installed on the right connector, and in the right orientation. Remember that some boards may have more than one JTAG connector for different purposes—you want the one that is connected to the processor's debug interface.
2. There are many different “standard” JTAG connectors, and using the wrong one may damage the MAJIC probe, target board, or both. If the cable does not fit the target board's debug connector, OR the pin-out of your debug connector does not match the *MAJIC Interface Specifications* application note, please contact technical support to see if there is a more suitable cable kit for your board.
3. Check the documentation for the board to see if there are any jumpers or switches that must be set in order to enable and use the JTAG debug interface.

### Communication Ports

The MAJIC probe may be connected to the host computer with a high speed RS-232C serial interface or through an Ethernet network connection. For serial connection, simply connect the serial cable provided from the RS-232 port on the rear panel of the MAJIC probe to the serial port on your computer. Make a note of which serial port you connected to, as the debugger will need to know this.

In order to use the MAJIC probe with an Ethernet connection, it must have an IP address. As described in Chapter 2 of the *MAJIC Probe User's Manual*, there are three ways for the MAJIC probe to obtain its IP address. The easiest way is to program a static IP address into the MAJIC probe using the serial port, so that it always comes up knowing its address. To program the static IP address, perform the follow steps:

1. Assign (or obtain from your network administrator) a static IP address for the probe, and a host name (this is optional). The network administrator may want the Ethernet (MAC) address of the MAJIC probe, which is available on the serial number tag on the bottom, and may also want to review Appendix A of the *MAJIC Probe User's Manual*.
2. Connect the serial cable provided from the RS-232 port on the rear panel of the MAJIC probe to the serial port on your computer. Leave the JTAG cable disconnected for now.
3. Open a terminal window, and start MONICE to connect to the MAJIC probe as shown in the following example, replacing the port device name as appropriate. Several informational messages should scroll by when MONICE starts.

```
monice -d /dev/ttyS0 -7
```

**NOTE:** Remember that almost everything in Linux is case sensitive, so */dev/ttyS0* is not the same as */dev/ttys0*.

4. Set the *tv\_ip\_address* (*tia*) configuration option using the Enter Option (*eo*) command, filling in the blanks with the IP address for your MAJIC® probe.

```
MON> eo tia = ____ . ____ . ____ . ____
```

After a few seconds, MONICE will report that the IP address has been programmed.

5. Quit MONICE with the *Q* command, then cycle power on the MAJIC box or press its reset button.
6. Ping the MAJIC probe's IP address as shown below, filling in the blanks with the IP address you just programmed, to make sure the computer can communicate with the MAJIC probe.

```
ping ____ . ____ . ____ . ____
```

7. Remove the serial cable.

**NOTE:**

- To remove a static IP address, follow the procedure above, and set the **tia** = **0.0.0.0**.
- If your MAJIC probe and computer are located on separate subnets, you may also need to set the *tv\_ip\_netmask* (**tin**) and *tv\_ip\_gateway* (**tig**) options in step #4. See Appendix A of the *MAJIC Probe User's Manual* for additional information.

## The Debug Environment

Chapter 3 of the *MAJIC Probe User's Manual* explains the configuration process and the files that are involved in detail. The following sections of this application note outline the basic steps necessary to configure the debug environment on Linux, first using MONICE, and then with GDB.

### 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.

1. Create a new, empty directory for staging the configuration files, then **cd** into that directory. For example:

```
mkdir majic
cd majic
```

2. The EDT software package includes pre-validated start-up files for many common reference platforms, in the *./targets* directory of your EDT installation. If you are using a standard reference platform, you can simply copy the corresponding files into the MAJIC configuration directory you created above. If your board is similar to a standard reference platform, you may need to adjust it to account for the hardware differences by editing it with your preferred text editor.

```
cp /opt/edtX/targets/_____/*.cmd . /* using edta or edtm as appropriate */
```

If your board is your own design, you will need to create suitable start-up files. Copy the template files from the *./targets/\_template* directory, then fill in the details as explained in the comment blocks and *readme.txt* file.

### Starting MONICE

There are two reasons to run MONICE first, even if you intend to use GDB normally. First, to establish a baseline operation with your EDT installation and MAJIC configuration files. Second, to create the *epimdi.cfg* file required when using GDB with the MAJIC probe.

To start MONICE, **cd** into the MAJIC configuration directory that you created and populated above, then run **monice** with suitable *-d*, *-v*, and *-l* switches. For full information on the MONICE invocation line, please refer to Appendix C of the *MAJIC Probe User's Manual*.

#### Examples:

```
monice -vh // Show the supported CPU version (-v) switches
monice -vARM926EJS -d /dev/ttyS0 -7 -1 // RS232 (-d ____), 115k (-7), little endian (-1)
monice -v5Kc -d 205.158.243.236:e // Ethernet (-d ____:e), big endian (no -1)
```

**NOTE:** A space is required between the *-d* switch and the port name or IP address, but a space is prohibited in the *-v* switch. The little endian (*-l*) switch is a lower case L, not a number one.

When MONICE starts, it displays a number of messages with version and configuration information, then it displays a **MON>** prompt. Assuming there were no errors in your start-up command files, at this point you should be able to access and view both registers and memory on your target. You can even download, step through, and run a program. Chapter 4 of the *MAJIC Probe User's Manual* provides numerous examples of using the MON command language, and Chapter 5 details the syntax of each command and parameter type. A hierarchical help system is available with the *H* command.

## MDI Configuration File

The MDI shared library is responsible for managing the details of the MAJIC configuration. It uses the same *startice.cmd* and (optional) board initialization command file as MONICE. However, CPU information and communication parameters are set via the *epimdi.cfg* configuration file instead of using command line switches. The easiest way to create *epimdi.cfg* is to start MONICE as described above, and then ask MONICE to create the file for you. The following command creates an *epimdi.cfg* file in your current working directory that matches the configuration in use by MONICE.

```
MON> fw mdi epimdi.cfg
```

**NOTE:** The *epimdi.cfg* file created in this way supports just one specific MAJIC probe and target. If you have more than one MAJIC probe and/or target system, you can edit the *epimdi.cfg* file to define multiple “devices” and “controllers”, and then pick the desired configuration for each debug session. Please refer to the *MDI for MAJIC User’s Manual* for details on creating *epimdi.cfg* files that support multiple MAJIC® probes and/or targets, and the *MDI-Server Options* section of this application note for information on choosing the active configuration.

## Running SDE-MIPS 6.x GDB

If you are using version 6 or later of the SDE-MIPS GNU toolkit distributes by MIPS Technologies, your version of GDB includes direct support for the MDI API and you don’t have to run mdi-server as described in the next session. Instead, use the *target mdi* command to connect directly to the MDI shared library (*mdi.so*), instead of the *target remote* command shown in the *Running GDB* section. See Chapter 14 of the *MIPS SDE 6.x Programmer’s Guide* for details.

### NOTE:

- Since the GDB remote protocol is not involved, the next section (including *Download Performance*) does not apply and you can skip to the [Running GDB](#) section.
- As of version 6.02 of the MIPS SDE, there is a bug in GDB that prevents hardware instruction breakpoints from working. You can use the MON bsh command to set hardware instruction breakpoints, as well as data breakpoints with options not available in GDB. See section [MON Commands](#).

## 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 (via the MDI shared library). Therefore, the mdi-server program must be running before GDB can connect.

Once you have your MAJIC 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. The following is the minimum command for launching mdi-server and linking in the requisite MDI shared library:

```
mdi-server -l /opt/edtX/bin/mdi.so // using edta or edtm as appropriate
```

**NOTE:** The load (-l) switch is a lower case L, not a number one.

## MDI-Server Options

In some cases, additional start-up options beyond the minimum invocation line above may be desired or even required. Running mdi-server with the *-h* switch shows the MDI-Server version number and usage information. Some of the commonly used options are explained below, and additional information is available in the *./manuals/gdb-readme.txt* file.

If you have created an *epimdi.cfg* file with multiple “DevName” definitions (see the *MDI User’s Guide* for details), then you may include a *-d* switch to identify which you intend to use. If your *epimdi.cfg* file has multiple “DevName” definitions but *-d* is omitted, then mdi-server will display a list of configurations available and prompt you to choose one each time GDB connects, which may cause some GDB versions to timeout.

There are several options relating to how GDB represents the number of registers, their order, and size. ARM and XScale users can generally ignore this issue, but there are significant differences between various MIPS-targeted GDB configurations, so additional mdi-server switches may be necessary to accommodate different GDB builds.

- f* Specifies whether a MIPS-targeted GDB is configured to expect floating point registers to be passed as single or double precision in the remote protocol.
- n* Specifies how many target registers GDB knows about.
- r* Specifies the register size passed in the remote protocol.

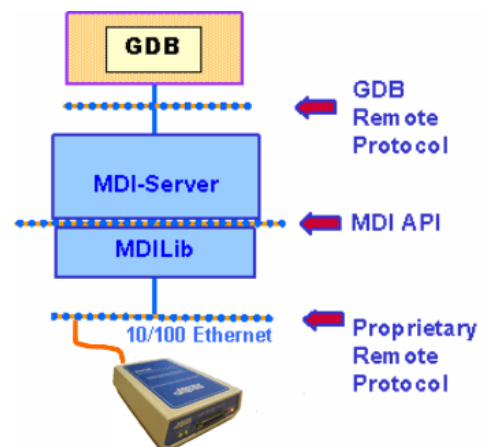
You may be able to determine the correct settings for these options from the GDB source code and build options. Otherwise, you will need to take several steps to figure out these options should be set for your GDB build:

1. Start mdi-server the first time with just *-n 38* added to the invocation line shown above. This specifies the minimum possible set of registers, sent as four bytes each, so GDB will not report a “packet too long” error. If GDB reports “Couldn’t establish connection to remote target” with no apparent reason, add the *-r 8* option.
2. After connecting, check how many registers GDB supports by entering the *info all-reg* command. GDB will display all the registers it knows about. If the last register displayed is *pc*, then *-n 38* is correct. If it is *fp*, use *-n 72* if your target has a floating point unit, else use *-n 38*. If the last register is *PRid*, use *-n 90*, and if it is *Config1*, then the default (*-n 107*) is correct.
3. If your GDB supports more than 72 registers it is important to get the *-f* option right even though your processor most likely does not have a floating point coprocessor. Now that you have the correct settings for the *-r* and *-n* options, try adding *-f 8*. If it is not correct, GDB will report the “Remote packet too long” error and refuse to connect.

If you cannot connect successfully using this procedure, please enable remote debugging with the command *set debug remote 1* before attempting the connection with *target remote*. This will display the packets being sent and received. Include this output when you contact Technical Support.

### The GDB Debugger

GDB is a standard source-level debugger used under the Linux and OS-X Operating Systems. GDB normally communicates with remote debug agents via its own “remote protocol”. The MAJIC probe supports third party debuggers through an open Application Programming Interface (API) called the Meta Debug Interface (MDI). To allow GDB to connect to MAJIC, the target side of the GDB remote protocol is implemented by a program called mdi-server, which works in conjunction with a shared library called MDILib. mdi-server translates GDB “remote protocol” debug service requests into MDI API calls to the functions exported by the MDILib, which then provides the connection to the MAJIC probe.



With this connection method you don't need to build a special GDB for the MAJIC probe, 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 then reads the *epimdi.cfg* file created above, initializes the MAJIC probe and target, and then begins servicing debug requests from GDB.

### Building GDB

If you do not already have a cross-GDB built and installed you can create one by downloading the latest source for GDB available from <http://www.gnu.org/software/gdb/gdb.htm> and building it using the appropriate *-target=xxx* setting. For example the version of GDB used to validate these instructions was configured as follows:

```
---target=arm-elf --prefix=/opt
```

### Running GDB

Except for SDE-MIPS 6.x, you must start the mdi-server program as described previously before starting GDB so that it will be ready to complete the interface between GDB and the MAJIC probe. Once mdi-server is ready, open a separate terminal window and start the GDB debugger version that is intended for cross development of your target processor (i.e. not the default Linux GDB).

**NOTE:** Not all GDB's are created equal. The commands and variables supported by a particular GDB depend on its version and on how it was configured when it was built. The commands and variables mentioned here are valid in GDB 4.18 (target xscale-elf) and 5.0 (target mipsisa32-elf).

From the (*gdb*) prompt enter the following commands to load your file, 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 yourfile /* Open your program file */
(gdb) target mdi /* Open MDI connection (SDE-MIPS 6.x only) */
(gdb) target remote localhost:2345 /* Open MDI connection via mdi-server */
(gdb) load /* Download your program image */
```

**NOTE:**

- 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 be 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* start-up command file. Setting the internal GDB variable named *remotetimeout* prevents most GDB builds from timing out prematurely.
- *GDB's args* variable can not be used to pass command line arguments via *argc* and *argv* to your program. Instead, use the MON Load command's *-c* option, for example:

```
(gdb) mon l -c pgmname arg1 arg2 arg3
```

Using GDB's monitor command to pass MON commands to the MDILib. See the *MON Commands* section of this application note for further information.

Now you are ready to set breakpoints and run or single step through your program. If you are new to GDB you may want to visit [http://sources.redhat.com/gdb/current/onlinedocs/gdb\\_toc.html](http://sources.redhat.com/gdb/current/onlinedocs/gdb_toc.html) for information on using GDB.

**MON Commands**

The GDB *monitor* command can be used to pass *MON* commands through to the MDILib, as shown in the following examples. This allows access to MAJIC 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 Probe User's Manual* provides a table that shows 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 */
```

**NOTE:** If you want MON commands to have access to symbols in your program, either add an *LS* (Load Symbols) command to the *startice.cmd* or just enter it directly with the *mon* command. If you download your program with the MON Load (*L*) command, then the *LS* command is superfluous.

### Download Performance

By default, GDB generally downloads program code and data in small chunks (a few hundred bytes) that are not necessarily a multiple of four bytes in length. This causes program download times to be slower than necessary, especially with ARM targets. There are two GDB internal variables (whose names depend on the GDB version) that affect this. To improve GDB download performance, you should set the download write size to a binary value like 4096 or 8192, and the memory write packet size to a larger value to allow for packet overhead (+100 bytes is plenty). For example, to download 4KB at a time:

```
(gdb) set remote memory-write-packet-size fixed           /* Using GDB 5.x */
(gdb) set remote memory-write-packet-size 4200
(gdb) set download-write-size 4096

(gdb) set remotewritesize fixed                           /* Using GDB 4.x */
(gdb) set remotewritesize 4200
(gdb) set download-write-size 4096
```

An even better way to get the fastest possible program download speed is to tell MDI to do the download via MON's *Load (L)* command. The first example below downloads the whole program, the second example downloads everything except the .bss section, which is even faster, but requires that your boot code zero-fill the whole .bss section.

```
(gdb) mon 1 progfile           /* Download your program file via MDI */
(gdb) mon 1 -no b progfile /* Download your program file via MDI, excluding bss section */
```

Since this avoids the overhead of the GDB remote protocol, which is not very efficient even with the tweaks mentioned above, using the MON *Load* command is the recommended method for downloading large programs.

## MAJIC Firmware Updates

The MAJIC probe has internal firmware that can be updated by the customer. This is referred to as a “Firmware Update” and should be done whenever moving to a new EDT software release or service pack update. It is a good idea to check the SupportNet site periodically to check for updates and revision information. Service Pack releases are most often associated with new device additions to currently supported families, and occur between major EDT releases. Check with [epi\\_sales@mentor.com](mailto:epi_sales@mentor.com) if you are interested in a device that is not yet shown on the SupportNet site.

If you are updating to a new EDT Software release, install the new software as described earlier in this application note. The new MAJIC firmware is located in the `./ice/majic` directory of your new installation and revision information is provided in `MAJIC_RelNotes.html`. If you are performing a special firmware update from an EDT Service Pack, then the firmware update files will usually be in a separate sub-directory such as `./ice/majic.vXXX`.

### Firmware update

The MONICE command line debugger is used to perform the firmware update. If you perform the update via the serial port, the update may take several minutes. Performing the update via Ethernet is should take less than 15 seconds.

First `cd` into the directory containing the desired firmware update, then use MONICE to update the firmware with a command such as follows, substituting the communication port name or IP address as appropriate:

```
monice -d /dev/ttyS0 -7 fwupdate.cmd // RS232 (-d ____), 115k (-7)
monice -d 205.158.243.236:e fwupdate.cmd // Ethernet (-d ____:e)
```

### NOTE:

- **It is important** to start MONICE in the directory containing the desired `fwupdate.cmd` and `majic.abs` files.
- MONICE is located in the `./bin` directory of your EDT installation, which should be in your search path. If not, then you can simply pre-pend the appropriate path to the MONICE program name.
- The CPU version displayed by MONICE is irrelevant, because the MAJIC probe will not connect to the target during the update process.

When the update process begins, you should see basic connection information pass by on the screen, and then the following update information:

```
Reading commands from C:\Pkg\EPITools\edtm23b\ice:majic\startice.cmd
MON> /*-----*/
MON> /* startice.cmd: startup command file for F/W update. */
MON> /*-----*/
MON> //
MON> eo Ice_Power_Sense = Off /* leave disconnected during F/W update */
MON> // <eof>
MON> (closing C:\Pkg\EPITools\edtm20b\ice:majic\startice.cmd)
Reading commands from C:\Pkg\EPITools\edtm20b\ice:majic\fwupdate.cmd
MON> +Q
    loading memory image file...

Validating download and programming EEPROMs
Flash update completed successfully
Please cycle emulator power so new firmware can take effect
```

**NOTE:** Check to make sure that the Status LED on the front panel has turned off, which is the indication that the firmware update was successful. Even if error messages are displayed by MONICE, an extinguished status light means that the firmware update was completed.

**DO NOT** cycle power while the status LED is RED! Once the status LED has gone out, cycle power on the MAJIC probe so that the new firmware can take affect. This is very important, as the new firmware will not be run until you reboot the MAJIC probe. After rebooting the MAJIC probe, connect to your target with MONICE again to verify the new version number. Rerun MONICE as above, with only the connection (-d) switch. For example:

```
monice -d /dev/ttyS0 -7 // RS232 (-d ___), 115k (-7)
monice -d 205.158.243.236:e // Ethernet (-d ___:e)
```

When you connect to the MAJIC probe via MONICE the system information will be displayed automatically, but if it scrolled off the screen, you can display it again with the *di* command. Look for the line that displays the firmware version number:

```
Target System: EPI Majic Probe, Version: 4.0.0, S/N 0208G030...
```