XAPP981(v1.0) February 23, 2007

Using the BDI-2000 Interface to Debug a Linux Kernel on the ML403 Embedded Development Platform

Author: Ed Meinelt, Lester Sanders

Summary

This application note describes how to debug a Linux Kernel using the BDI-2000 JTAG Debug Interface for GNU Debugger. An example uses a reference system for the On-Chip Peripheral Bus Inter IC (OPB IIC) core using the IBM PowerPC™ 405 Processor (PPC405) based embedded system in the ML403 Embedded Development Platform. The configuration and building of the Linux kernel for BDI-2000 use is discussed. Software and hardware setup procedures are given. A step by step flow for debugging the Linux kernel is provided.

Included Systems

This application note includes one reference system:

www.xilinx.com/bvdocs/appnotes/xapp981.zip

The project name used in xapp981.zip is ml403_ppc_bdi.

Required Hardware/Tools

Users must have the following tools, cables, peripherals, and licenses available and installed:

- Xilinx EDK 8.2.02i
- ISE 8.2.03i
- Xilinx Download Cable (Platform Cable USB or Parallel Cable IV)
- Monta Vista Linux v2.4 Development Kit
- BDI-2000 Software
- BDI-2000 Hardware

Introduction

This application note uses a system built on the ML403 development board. Figure 1 is a block diagram of the system used in this flow.

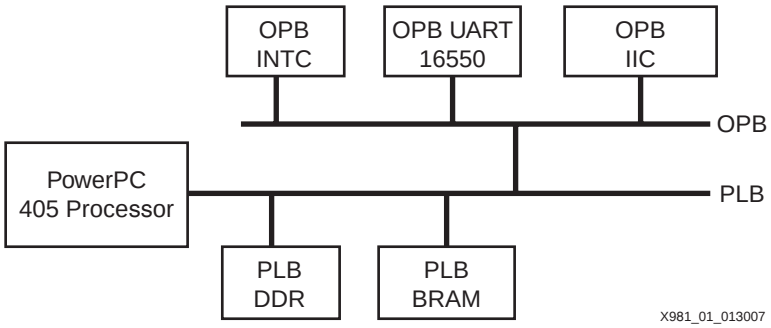


Figure 1: **OPB IIC Reference System Block Diagram**

The system uses the embedded PowerPC (PPC) as the microprocessor and the OPB IIC core.

Configuring and Building the Kernel

This section provides the steps to configure and build the Linux kernel which are specific to using the BDI-2000.

Figure 2 shows the menu which is provided after running the **`make xconfig`** command on a Linux machine. After configuring the kernel for the BSP general functions, select **Kernel Hacking**.

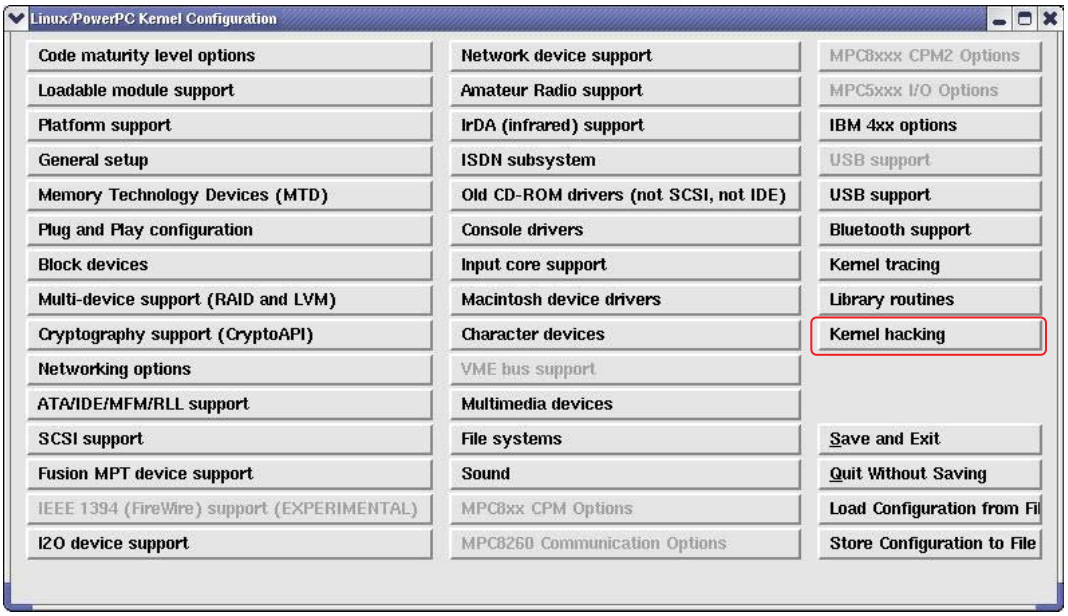


Figure 2: **make xconfig Menu**

Select **Kernel hacking** → **Include BDI-2000 user context switcher** as shown in Figure 3. This sets up pointers to allow BDI to locate the page tables. The `head4xx.S` and `pgtable.c` files contain the page table information needed to use software breakpoints.

The **Include BDI-2000 user context switcher** option prevents the kernel from modifying the debug registers in `ppc4xx_setup.c`.

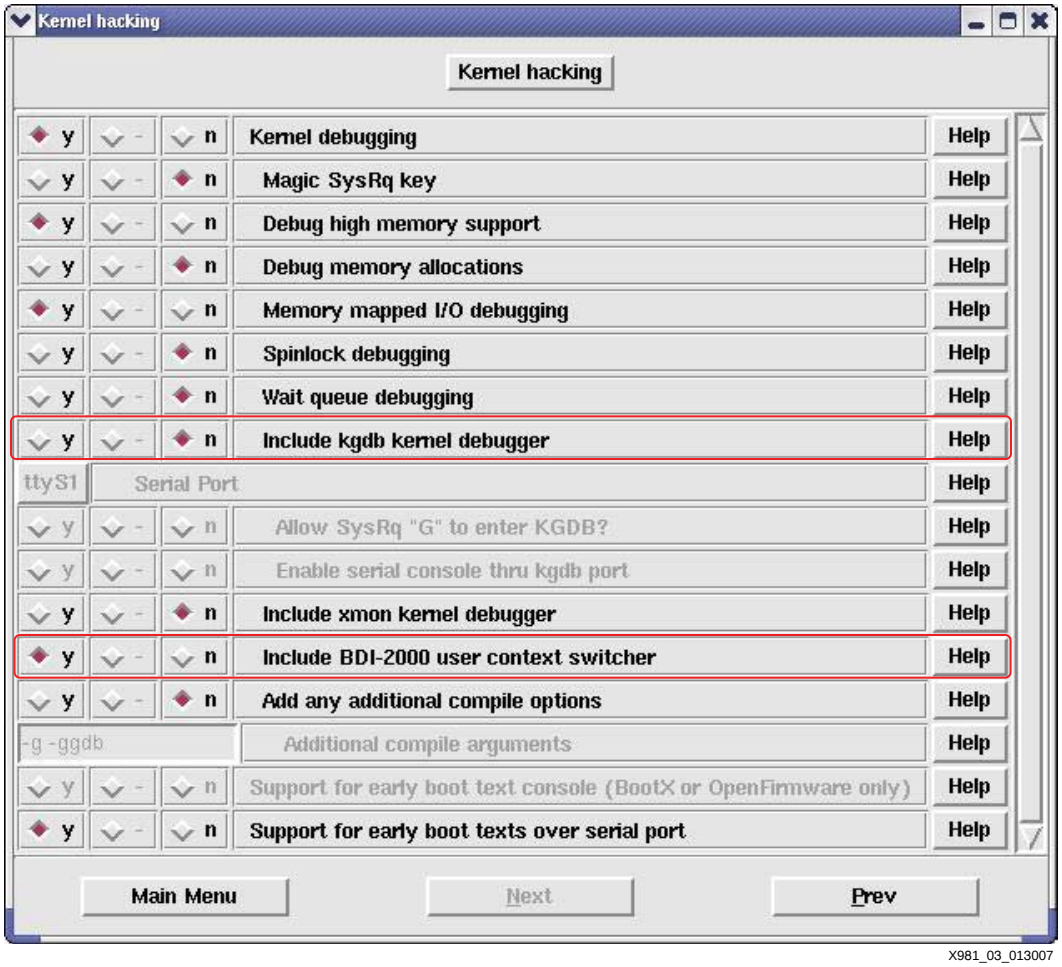


Figure 3: **Kernel Hacking**

Edit to the Makefile to use the **-g** option on CFLAGs as shown in Figure 4.

Note: The makefile depends on tab characters. Do not replace them with spaces.

Run **`make clean dep zImage.initrd`**.

The location of `system.map` and `vmlinux` is in the `linux kernel` root directory. The compressed kernel, `zImage.initrd.elf`, is 2 MB to 3 MB and the uncompressed kernel vmlinux is 25 MB.

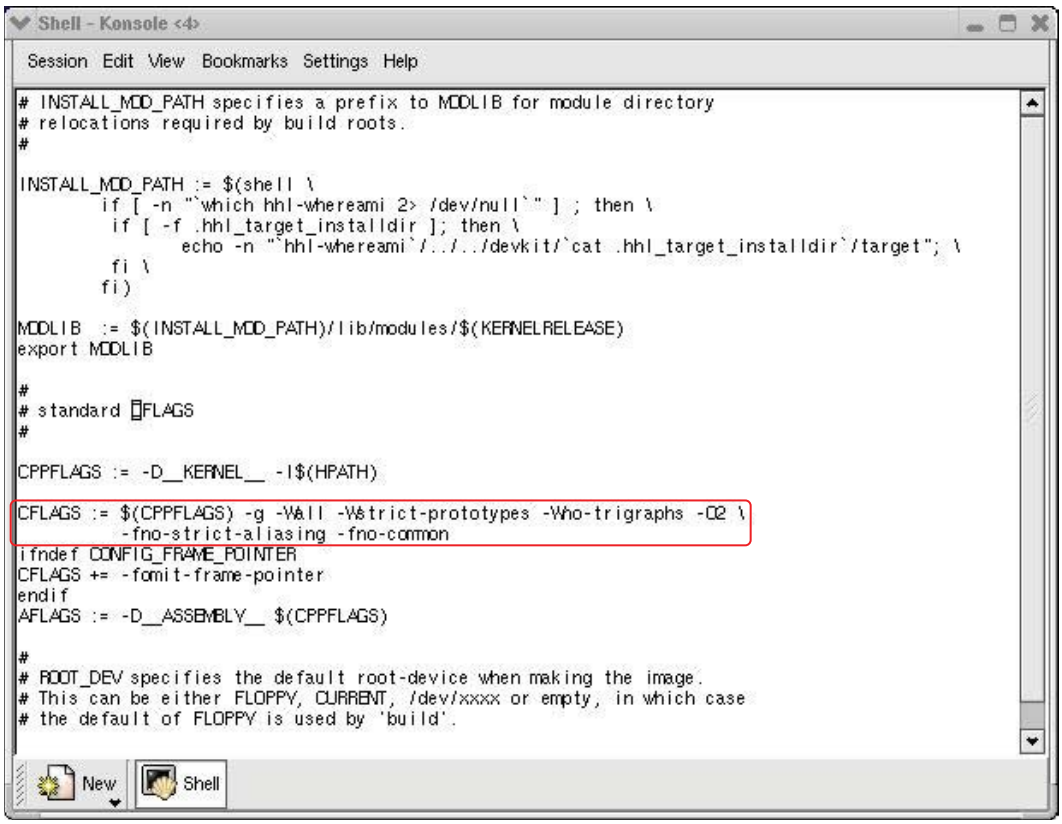


Figure 4: **Makefile Edits**

Software Setup

Software is available for the BDI-2000 from Abatron. The installation of this software is documented in the JTAG debug interface for *GNU Debugger User Manual* (see the References section).

The `ml403_bdi.cfg` file (included in design files) is used in the software setup.The `ml403_bdi.cfg` has the following settings.

IP 149.199.109.4

FILE H:\designs\ml403_ppc_bdi\bdi\zImage.initrd.elf

FORMAT ELF

STARTUP RESET

WM32 0x000000f0 0x00000000

MMU XLAT

The IP 149.199.109.4 is the IP address of the TFTP server. The MMU XLAT is needed to debug Linux after virtual addressing is enabled. Because the Linux bootloader registers initialization, the INIT statements in `ml403_bdi.cfg` must be commented to avoid a possible conflict.

Figure 5 shows the BDI setup GUI invoked by running b20pp4gd.exe at the command prompt. When in configuration mode, a red LED on the BDI-2000 flashes. Click on both **Connect** and **Transmit**. The LED stops flashing after setup. Another BDI setup method is to run the command:

```
bdisetup -c -i149.199.109.220 -h149.199.109.4 -fml403_bdi.cfg
```

After BDI setup, disconnect the serial cable from the BDI and connect it to the ML403 for Teraterm.

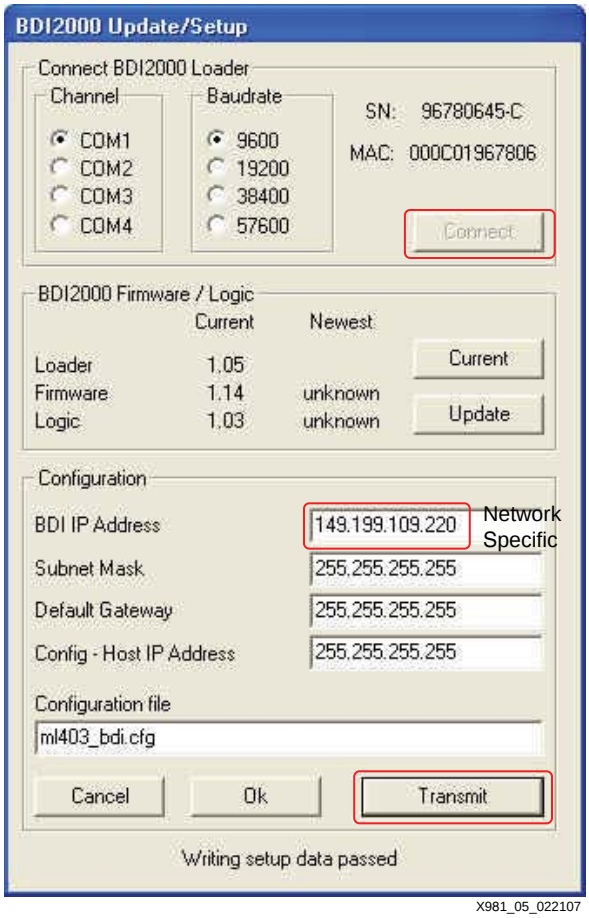


Figure 5: **BDI Setup**

Invoke a terminal emulation window as TeraTerm or HyperTerminal as shown in Figure 6. Set Baud rate to **9600**.

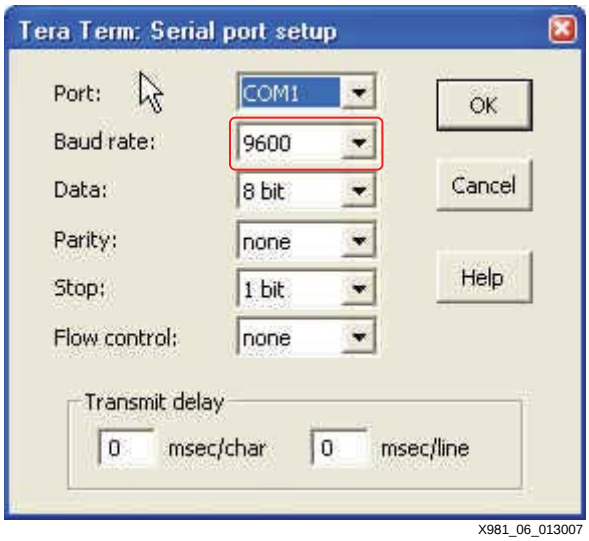


Figure 6: **TeraTerm Settings**

Hardware Setup

The hardware setup involves connecting the BDI-2000 and ML403 to the Ethernet and PC. Depending on the flow, at various phases of the process, hardware connections may require changes.

Figure 7 shows a hardware initial setup for configuring the BDI firmware and downloading the bitstream into the FPGA.

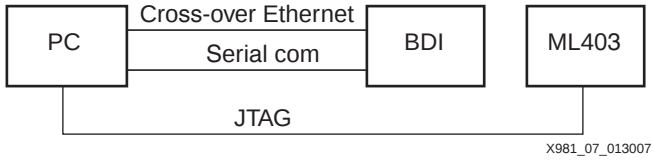


Figure 7: **Initial Setup**

After these two functions have been completed, connect the ML403 JTAG port to the Target B port on the BDI using16-pin ribbon connector.

Figure 8 shows the connections of the BDI - ML403 after the initial setup. The JTAG connection between the BDI-2000 and the ML403 consists of two ribbon cables.

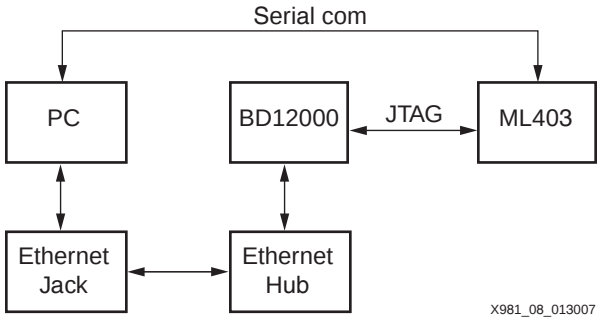


Figure 8: **BDI - 403 Connections**

Using the BDI-2000

To use the BDI-2000:

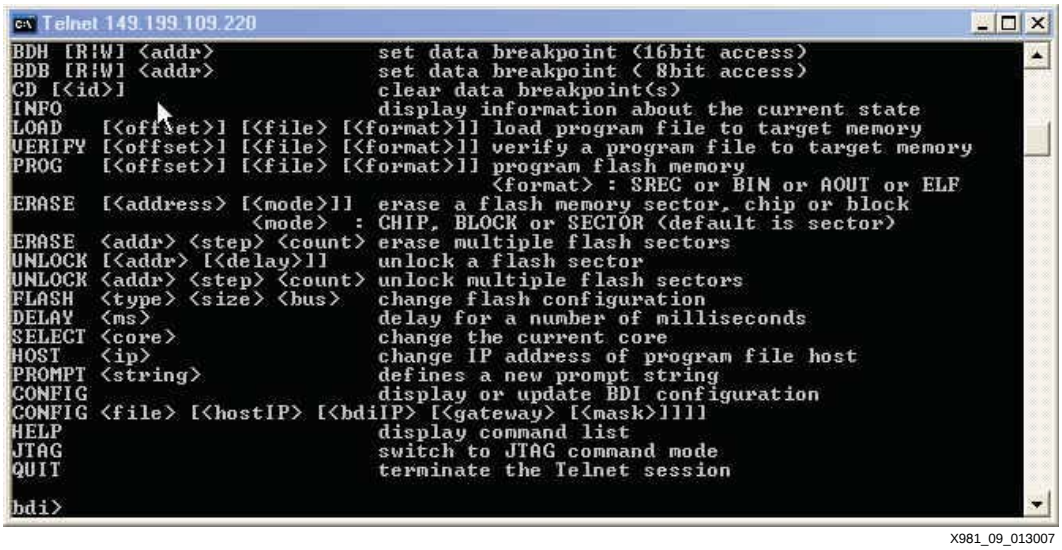
1. Using XPS, download the bitstream in the ml403_ppc_bdi project into the FPGA on the ML403 board.

From the command prompt, run **`tftpsrv dH:\designs\ml403_ppc_bdi\bdi\.`**

2. From the command prompt, run **`b20pp4gd`** to invoke the BDI-2000 setup software. Enter the BDI IP Address and `ml403_bdi.cfg` as the Configuration file, click **Connect**, then click **Transmit**.
3. At the command prompt, telnet to the BDI using the command

   `telnet 149.199.109.220`

Figure 9 shows the Telnet window with the BDI> prompt displayed along with Help information on BDI commands.



X981_09_013007

Figure 9: **Telnet BDI Help Window**

Using the BDI instructions, set a breakpoint at the start_kernel routine using the `bi` instruction, which is located just after the MMU is turned on. Address translation errors will occur if gdb is enabled before the MMU is enabled. The System.map provides the location of start_kernel.

To get the address of start_kernel, run

`grep start_kernel System.map.`

The first part of the kernel code cannot be debugged.

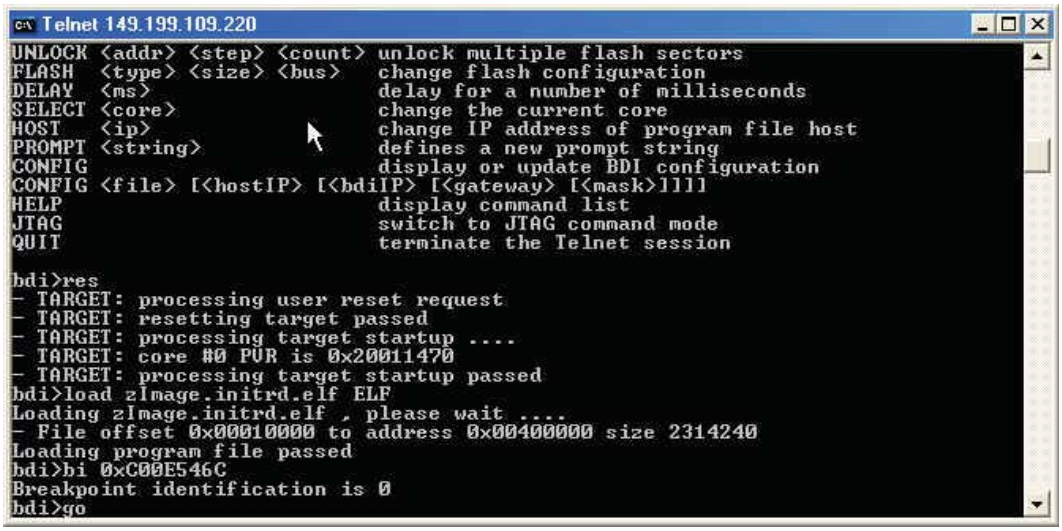
If the target responds with `Target must be in debug mode for this action`, type **`halt`** and re-issue the bi command

`bi 0xC015a46c <start_kernel>`

At the BDI telnet session, enter **`load zImage.initrd.elf ELF.`**

Note: Do not use an absolute path to `zImage.init.elf`.

Figure 10 shows the BDI telnet session.



X981_10_013007

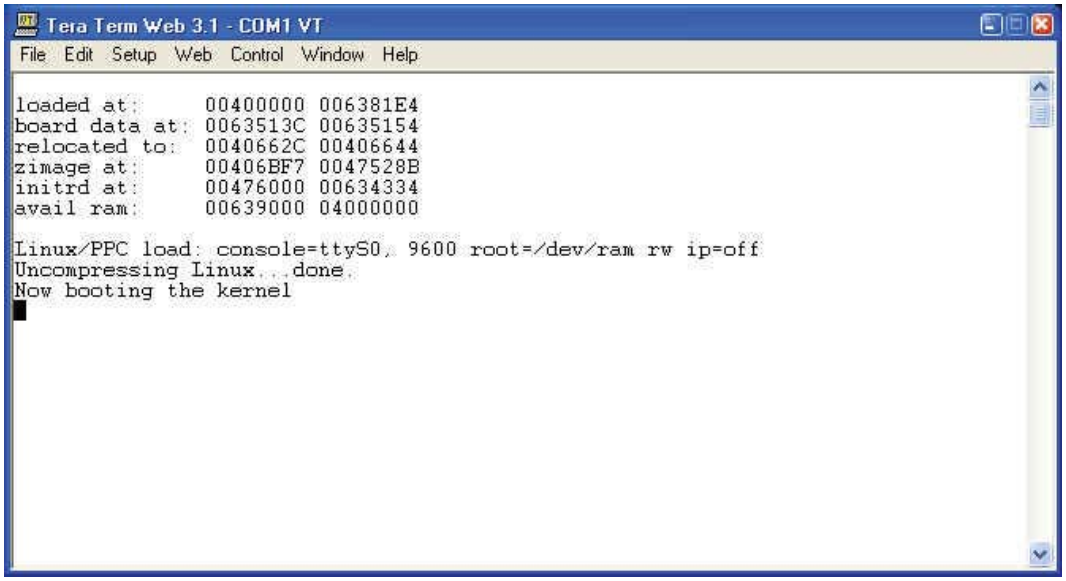
Figure 10: **BDI Telnet Session**

Run the instructions below:

```
go
ci
```

The `ci` instruction clears breakpoints.

Figure 11 shows the Teraterm output in which the Linux boot process stops at **`Now booting the kernel`**.



X981_11_013007

Figure 11: **TeraTerm Output**

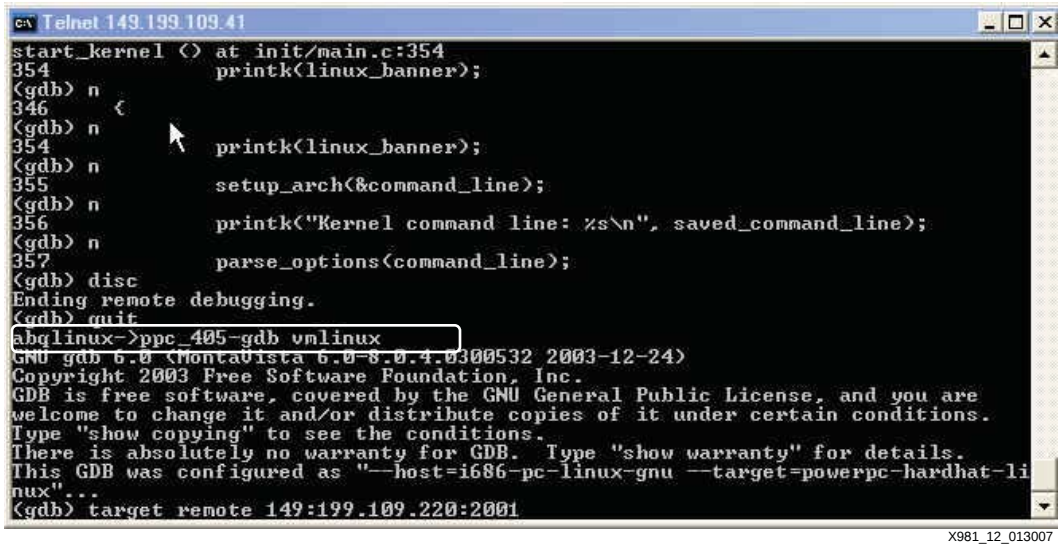
4. From a computer with the Monta Vista Linux software, enter:

   `ppc_405-gdb vmlinux`

   `target remote 149.199.109.220:2001`

The 149.199.109.220:2001 is the BDI IP address connected to port 2001.

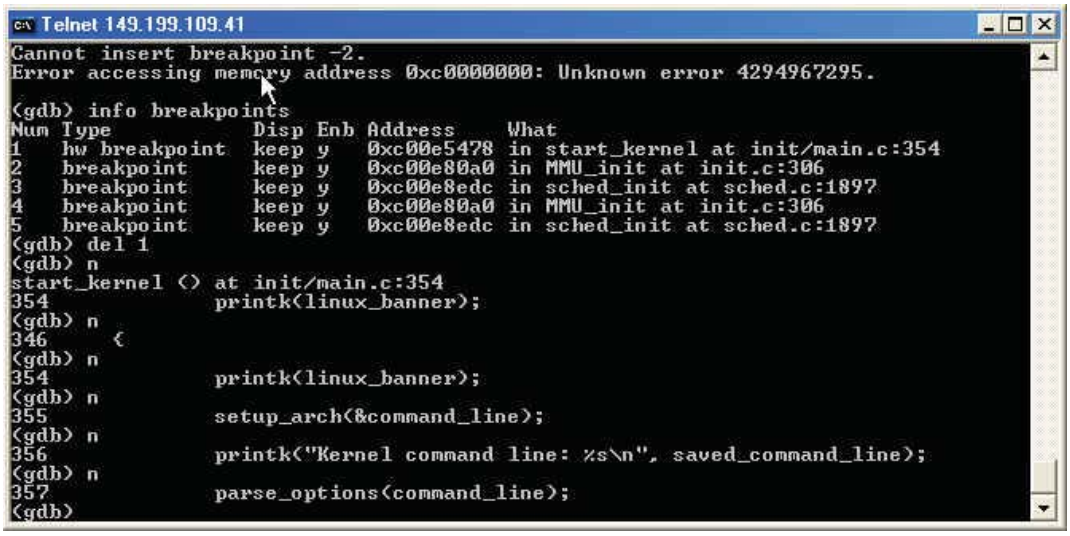
Figure 12 shows the Telnet window for the gdb session.



X981_12_013007

Figure 12: **Starting gdb Session**

Figure 13 shows the Telnet window with a debug session with gdb.



X981_13_013007

Figure 13: **Debugging with gdb**

Some example GDB instructions are given below. These remove a breakpoint and then create two new breakpoints.

```
del 1
break MMU_init
break sched_init
n
```

Conclusion

This application note describes how to use the BDI--2000 to debug Linux kernel problems. This is done for a Monta Vista Linux kernel running on the Xilinx ML403 Evaluation Platform.

References

JTAG Debug Interface for GNU Debugger - PowerPC 4xx User Manual v1.14 for BDI-2000

Using the BDI2000 to Debug a Linux Kernel Ultimate Solutions #02-001a, T. Michael Turney

Hardware Assisted Debug with Embedded Linux Ultimate Solutions #02-002

Debugging Linux with the BDI-2000 and bdiGDB #04-002 Ultimate Solutions, Fahd Abidi

DS434 OPB IIC Bus Interface (v1.02a)

XAPP765 Getting Started with EDK and MontaVista Linux

The I2C Bus Specification Version 2.1 January 2000 Philips Semiconductors

Building Embedded Linux Systems, O'Reilly

Revision History

This table below shows the revision history for this document.

Date	Version	Revision
2/23/07	1.0	Initial Xilinx release.