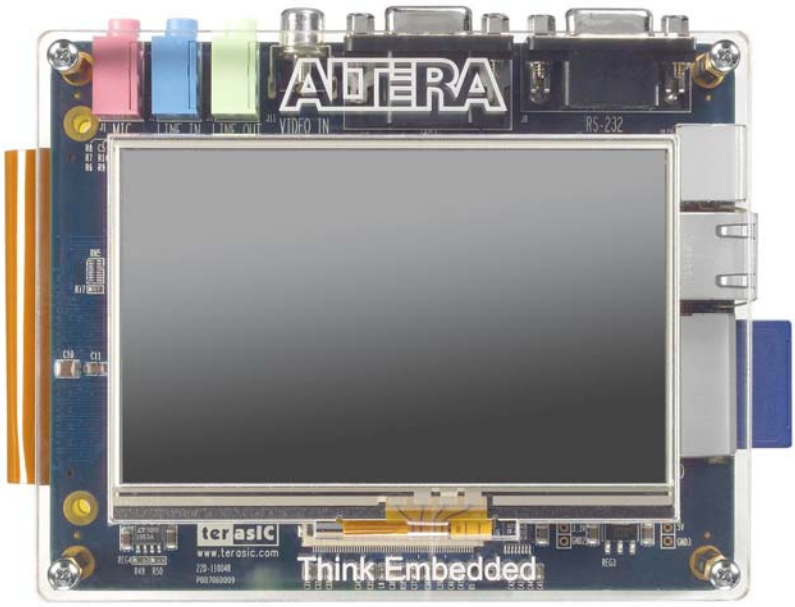
Getting Started with TargetOS

on Altera's

Nios II Embedded Evaluation Kit (NEEK)

Cyclone III Edition



1 Introduction

This document covers how to get started with Blunk Microsystems' TargetOS embedded operating system on Altera's NEEK Nios II evaluation kit. It covers the following steps:

- Installing the Altera Tools: Quartus II Programmer and Nios II GNU toolchain.
- Installing the CrossStep IDE, TargetOS Basic RTOS, and NEEK demo package.
- Setting up a new NEEK: cables and power.
- Installing an FPGA bitstream image and target monitor (with debug-over-Ethernet support) into the NEEK's NOR flash.
- Building Blunk's kbench kernel benchmark sample application and running it on the NEEK.

2 Installing the Altera Tools

If you have a NEEK board, you may already have the Altera tools installed. If not, you can get them from Altera's website. The free Web Edition will do. You will need the Quartus II Programmer and the Nios II GNU toolchain. For the latter, be sure your download includes the Nios II Embedded Design Suite (EDS). Version 13.1 is the last that supports the Cyclone III FPGA used by the NEEK. Install the Altera tools first, using their default location, as the CrossStep installer looks for your Nios II GNU toolchain and automatically configures itself to use it.

3 IDE/RTOS installation

This topic is covered lightly here because a CrossStep wizard guides you through this during its installation. To start, download CrossStep from www.crossstep.com and run its installer. Pick 'Altera Nios2 Embedded Evaluation Kit, Cyclone III Edition' when asked to choose a target. When asked to pick an initial sample application, pick 'installer' to be in sync with the explanations below.

4 Board Setup

The NEEK has no jumpers or switches to configure and just a few cables you need to attach. Start with the red on/off switch (SW1) in the OFF (up) position.

- Connect an Ethernet cable between the network your PC is on and the NEEK's RJ-45 jack, located on the right-hand side of the NEEK's middle board.
- Connect a USB cable between your PC and the NEEK's USB connector (J3), located on the right-hand side of the rear board. This serves as a USB-Blaster JTAG connection.
- Connect the supplied 12v DC power supply to the barrel connector on the upper right-hand side of NEEK's rear board.
- Connect a UART cable between your PC and the DB-9 connector on top of NEEK's middle board, on the right-hand side.
- Open a terminal program on this channel using 38400 baud rate, 1 stop bit, no parity, and 8-bit character length.
- Turn the board on by pressing the red power switch SW1.

If the "Found new hardware" wizard appears, use it to install the drivers. Otherwise, open the Windows Device Manager and find "USB-Blaster" (under "Other devices"). Right-click on it and select "Update Driver Software…" In the resulting dialog box, select "Browse my computer for driver software" and navigate to `C:\altera\13.1\quartus\drivers\usb-blaster`. Be sure the "include subfolders" box is checked. Press the "next" button in the wizard to install the drivers.

We will install a new Nios II system shortly, but if your NEEK already contains a Nios II design, you can test that the USB-Blaster driver is installed properly and that the hardware connection is good by starting a Nios II Command Shell (using the "Nios II EDS" submenu in the Altera section of your PC's Start Menu) and issuing the following command:

```
nios2-gdb-server -r -C
```

Although it may take a few seconds, you should see messages indicating that the USB Blaster has reset and paused the target processor.

Before you can do anything useful with the NEEK, you need to install a couple files into its NOR flash. That is the topic of the next section.

5 Preparing NEEK's NOR Flash

The NOR flash holds 1) a bitstream image that instantiates a Nios II system in the FPGA and 2) a Nios II boot application. You can use Blunk's 'installer' application to download and install these files into the NOR flash via TFTP, but before you can run an app, a Nios II system must be loaded into the NEEK's FPGA. This requires the Altera Quartus II Programmer.

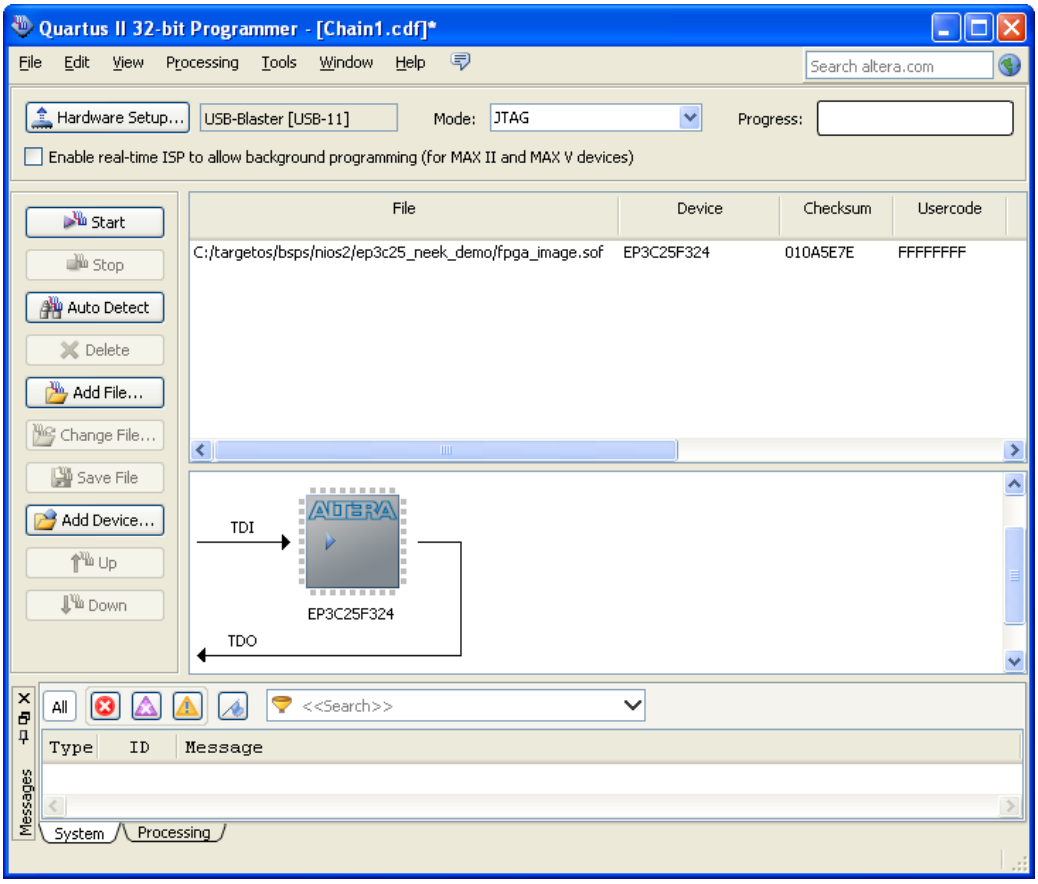
The first step is to start it. If the Quartus II Programmer is not a separate option in the Altera section of your PC's start menu, you must start Quartus II. If prompted to open a project, close that dialog since a project is not needed to use the Programmer. Select *Programmer* from the Quartus II *Tools* menu and perform the following steps:

1. Use the *Add File* button to browse to and select 'fpga_image.sof', the Nios II system this demo was written for, located in the following subdirectory of your TargetOS installation:

   ```
   bsps\nios2\ep3c25_neek_demo\
   ```

2. If not already selected, press the *Hardware Setup* button in the upper left corner of the programmer, and in the resulting dialog, select *USB-Blaster* as the selected hardware.

3. Click the programmer's *Start* button to send the Nios2 IP to the FPGA. The result should resemble the picture below:



Now that the demo Nios II system is running in the (volatile) FPGA, you can use the installer application to install the 'fpga_image' image and a TargetOS boot application into NOR flash, so they are present each reset. This requires a debug connection.

Until Blunk's debug-over-Ethernet target monitor is installed as the NEEK's boot application, you must use Altera's USB-Blaster JTAG interface to run programs on the NEEK. You can start that by entering the following command in a Nios II command shell:

```
nios2-gdb-server -r -C --tcpport 4040
```

The above command times out if you don't establish a debugger connection quickly enough. To keep the USB-Blaster running as long as you want and for as many debug connections as you want, it is convenient to create a file (i.e. "blunk") that contains the following:

```
while true; do nios2-gdb-server -r -C --tcpport 4040;done
```

And to execute it in a Nios II command shell by entering:

```
./blunk
```

Next you need to build and download the installer application. If you specified the installer application during the CrossStep installation, it will already be loaded. Otherwise, use CrossStep's Project/Open dialog to browse to and select this file:

```
apps\installer\installer.gbp
```

If you open ‘main.c’ for this application, you will see the following lines at the top of the file:

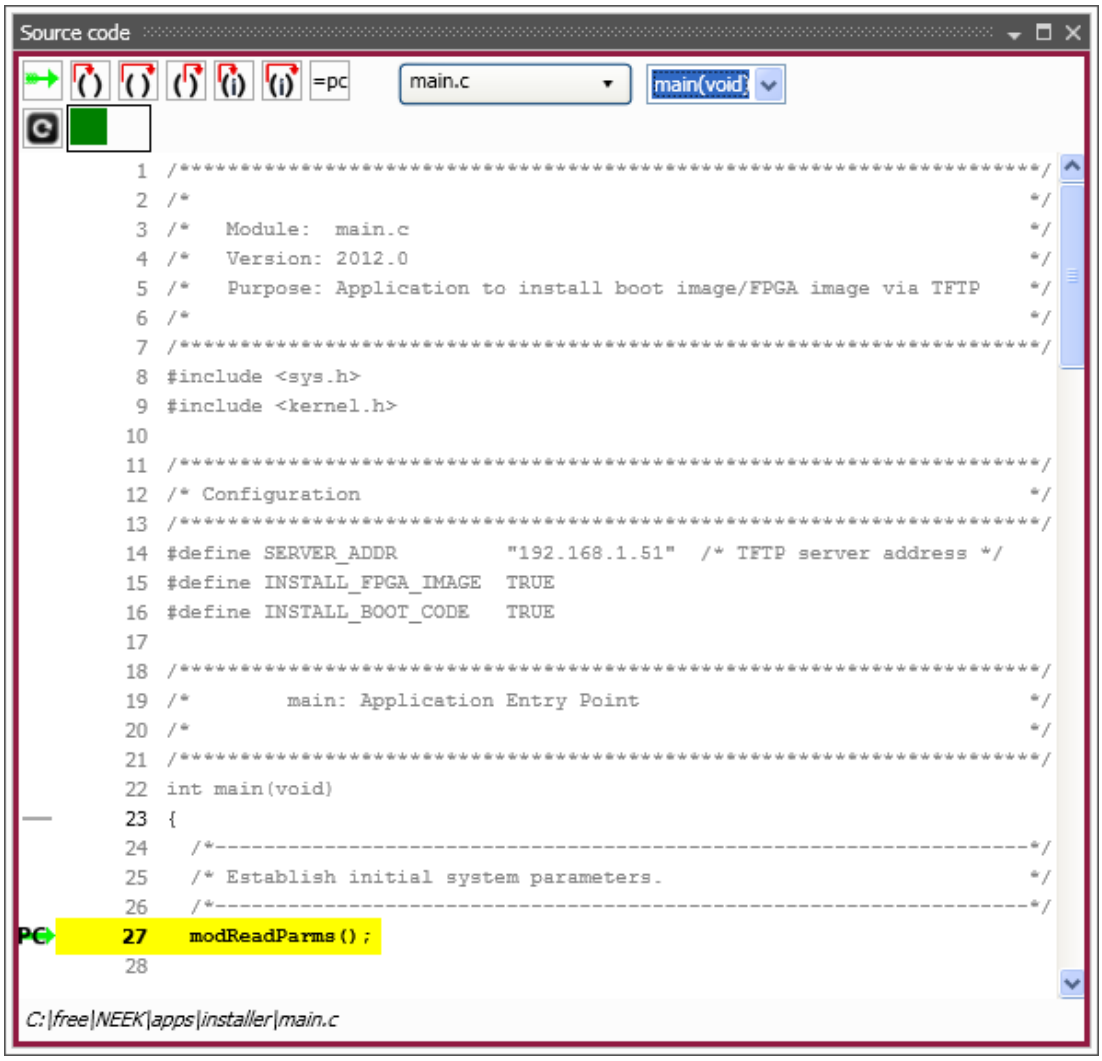
```
/*********************************************************************/
/* Configuration                                                     */
/*********************************************************************/
#define SERVER_ADDR         "192.168.1.51"  /* TFTP server address */
```

You need to edit this IP address, changing it to the IP address of your PC. (An easy way to open ‘main.c’ is to click on the ‘Source File’ tab of CrossStep’s Project Manager and then double-click the line for ‘main.c’).

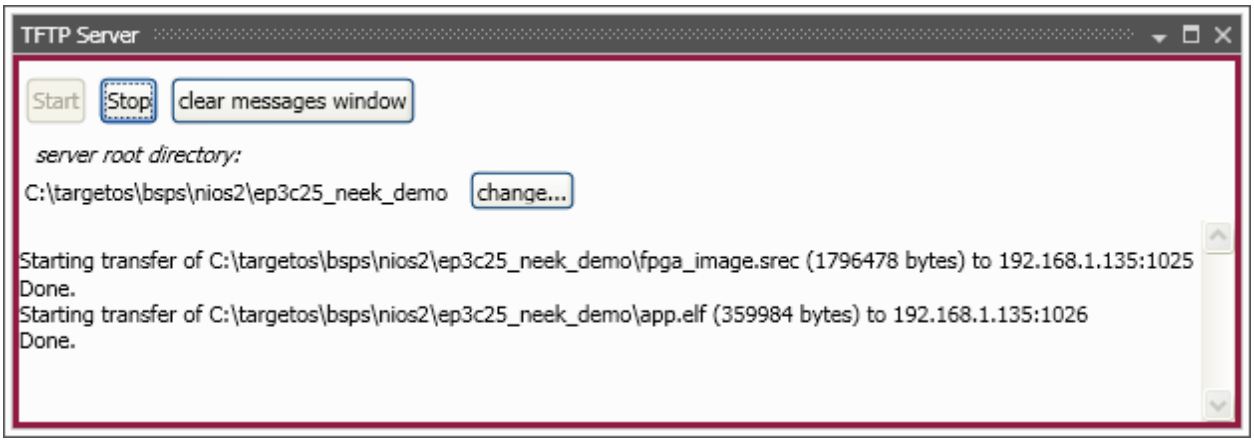
Build the application by clicking the button with the hammer icon  in CrossStep’s upper-left corner. When the build finishes, “app.elf successfully built” should appear in the Output window. To launch the source code debugger, click on the button with the “no bug” icon  (third from the left in the row of buttons near the top).

CrossStep will ask if you want to add a Debug Interface to your target. Answer yes, and select “USB Blaster” in the wizard which follows. All other settings in the wizard should be left as is (use the defaults). When you exit the wizard, CrossStep will ensure your project is up-to-date, download it to the board, and start a debug session by running the board initialization code and stopping at the beginning of main().

What you see in the Source code window of the debugger should resemble the picture below.



Start a TFTP server on your PC, serving the demo BSP directory, the same directory containing the FPGA image file you used with the Quartus II Programmer. If you use CrossStep's TFTP server (accessed via the Tools menu), it should look like the following:



Click on CrossStep's green run key  to run the installer. You should see the app acquire an IP address and then install the FPGA image and boot application. Your terminal connected to the NEEK UART channel should show the following:

```
Starting installer

Waiting for network interface initialization
TSE100: link=up speed=100M duplex=full
TSE100: DHCP IPv4 address 192.168.1.135

Opening FPGA image file
Filling buffer with 0xFF
Downloading FPGA image file
................................................
Loaded 718569 bytes
Clearing flash memory
.......
Programming FPGA image to flash
Programming 718569 bytes
............................................................
Finished!
Opening source file: /dev/tftp/192.168.1.51/app.elf
Filling buffer with 0xFF
Downloading S-record/ELF file
.........
Loaded 310932 bytes
Clearing flash memory
........
Programming boot image to flash
............................................................
Finished!
Installation has completed successfully
```

Click on  to close the debugger. The installed FPGA image file and boot application will now automatically load from flash after each power-on reset. You no longer need the Quartus II Programmer or USB-Blaster and can close both programs. If you toggle the NEEK's power switch, your terminal should show the boot application start, which looks like the lines below.

```
TargetOS, Release 2016.0. Built at 5/7/2016 6:52:58 PM.
Copyright 2016, Blunk Microsystems. All rights reserved.
--------------------------------------------------------
Starting TargetMon
Core: 100MHz fast, 4KB icache, 4KB dcache, hwmul, hwdiv
FPGA: CYCLONEIII, SysID 0x4990EBD8 at 18:19:40 05/03/11
Memory: 32MB SDRAM, 16MB Flash, 64B LCD, 1MB SSRAM
Hostname: 3c25neek
IPv4 Default gateway: 192.168.1.1
TSE100 IP addr/mask: DHCP
TSE100 MAC address: 00:00:00:00:2E:E2
Starting FTP server for IPv4
Starting Telnet server for IPv4
Starting TFTP server for IPv4
Using: 0x00000000 - 0x00100000
--------------------------------------------------------
Press 'M' in 3 seconds to modify these settings: .Starting..
TSE100: link=up speed=100M duplex=full
```

```
TSE100: DHCP IPv4 address 192.168.1.108
```

The boot application contains TargetMon, Blunk's debug-over-Ethernet target monitor, which we will use next. To prepare for the next section, leave the TargetMon application running.

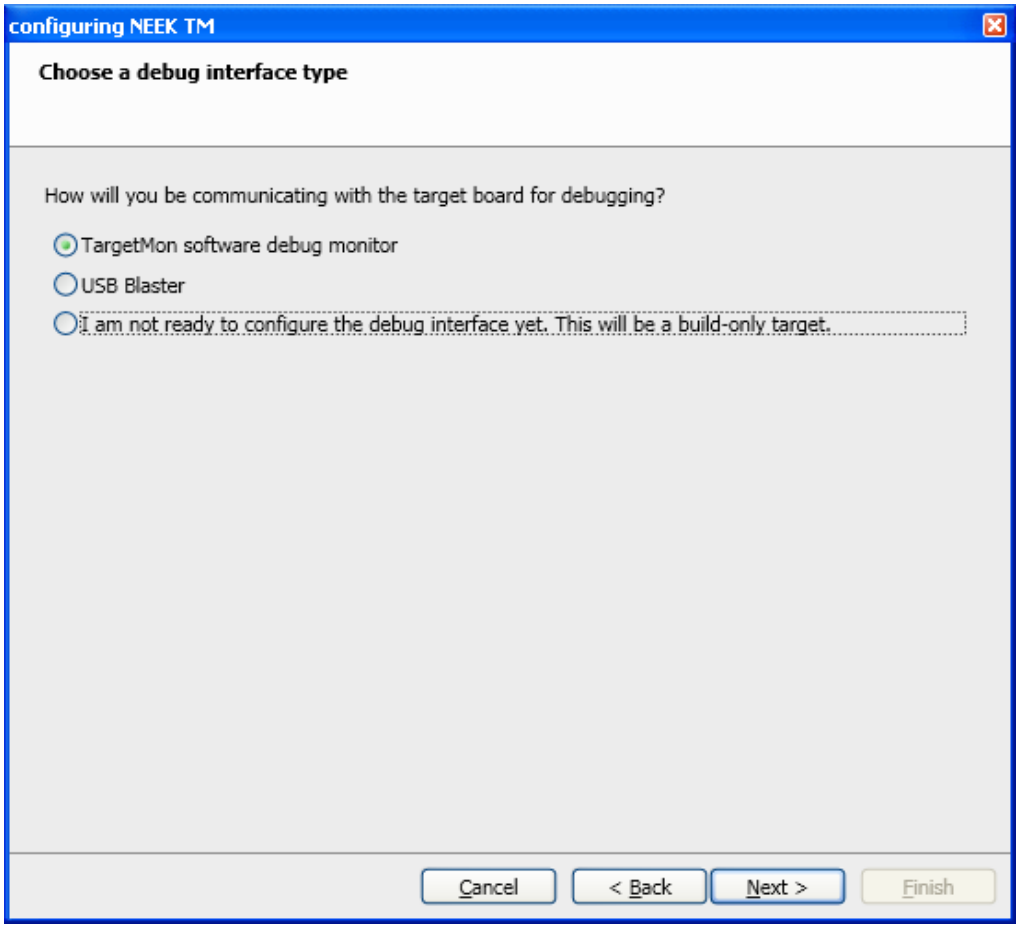
6 Running kbench

Now you are ready to build and run the kbench sample application, which measures TargetOS kernel performance, and to use a debug-over-Ethernet connection. Use CrossStep's Project/Open dialog to browse to and select this file:

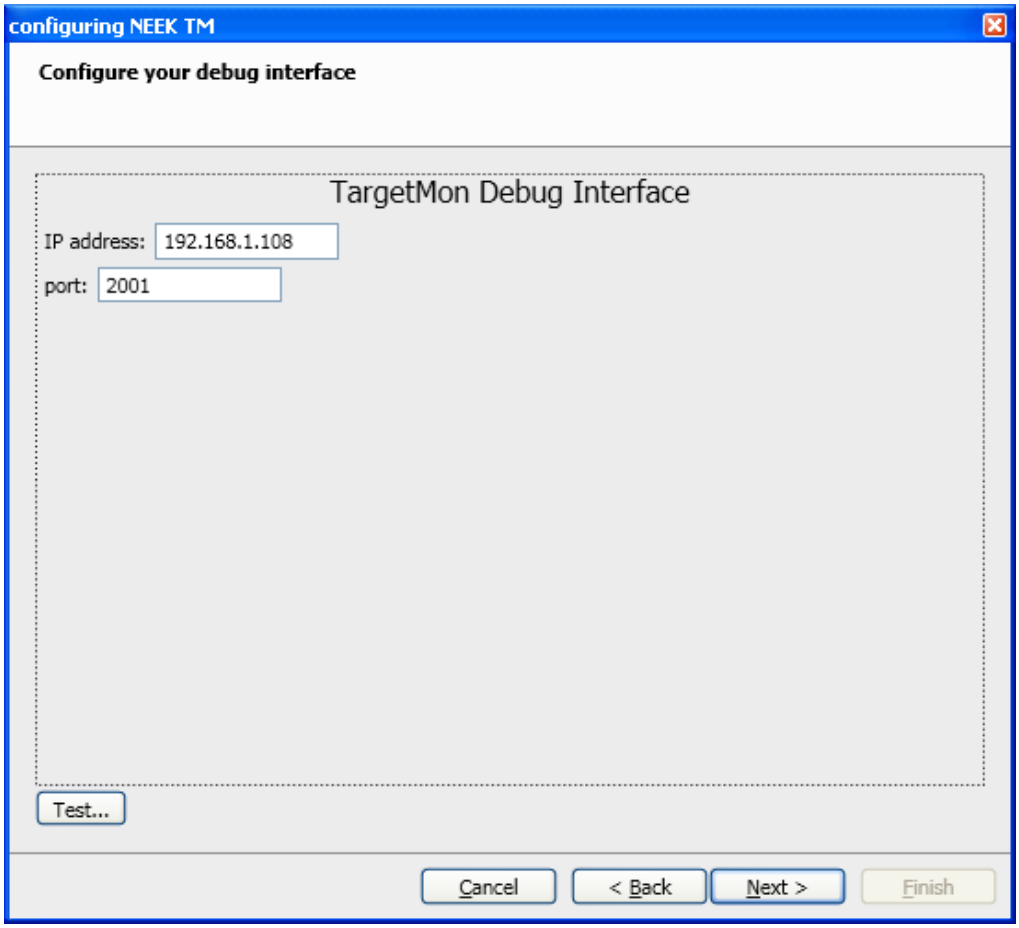
```
apps\kbench\kbench.gbp
```

You already have one CrossStep target configuration, for connecting via USB Blaster. Select "New Target..." in the Option menu to create another target configuration for debugging via Ethernet. Enter a suitable name in the first dialog. We used "NEEK TM". In the next dialog, if not already the default, browse to select the 'bsp.gbl' file in the demo BSP.

Click "Next" until you get to the dialog shown below, then choose 'TargetMon', as shown.



Enter your NEEK's IP address in the next panel, getting the address from the UART terminal which showed the boot application's start-up. The IP address is on the last line ("192.168.1.108" in our example). Leave the port number as 2001.



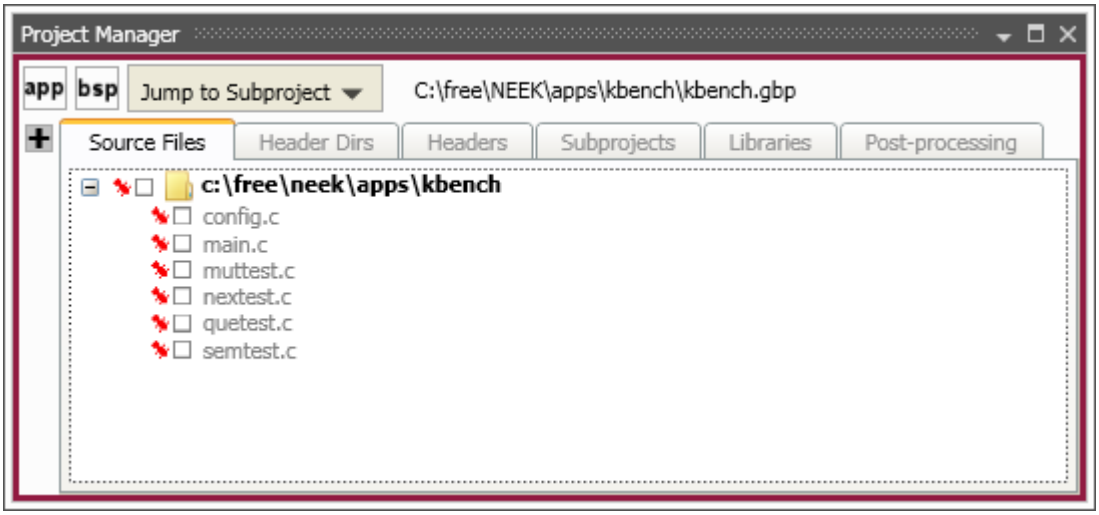
Optionally test the debug connection by clicking the "Test..." button. Click "Next" until the last dialog and then click "Finish" to close the target modification wizard.

Build the kbench application by clicking the button with the hammer icon. Or you can use the keyboard shortcut control-B. The Stop button next to the Build button becomes active whenever a build is in progress. You can click that icon if you need to stop a build for any reason.

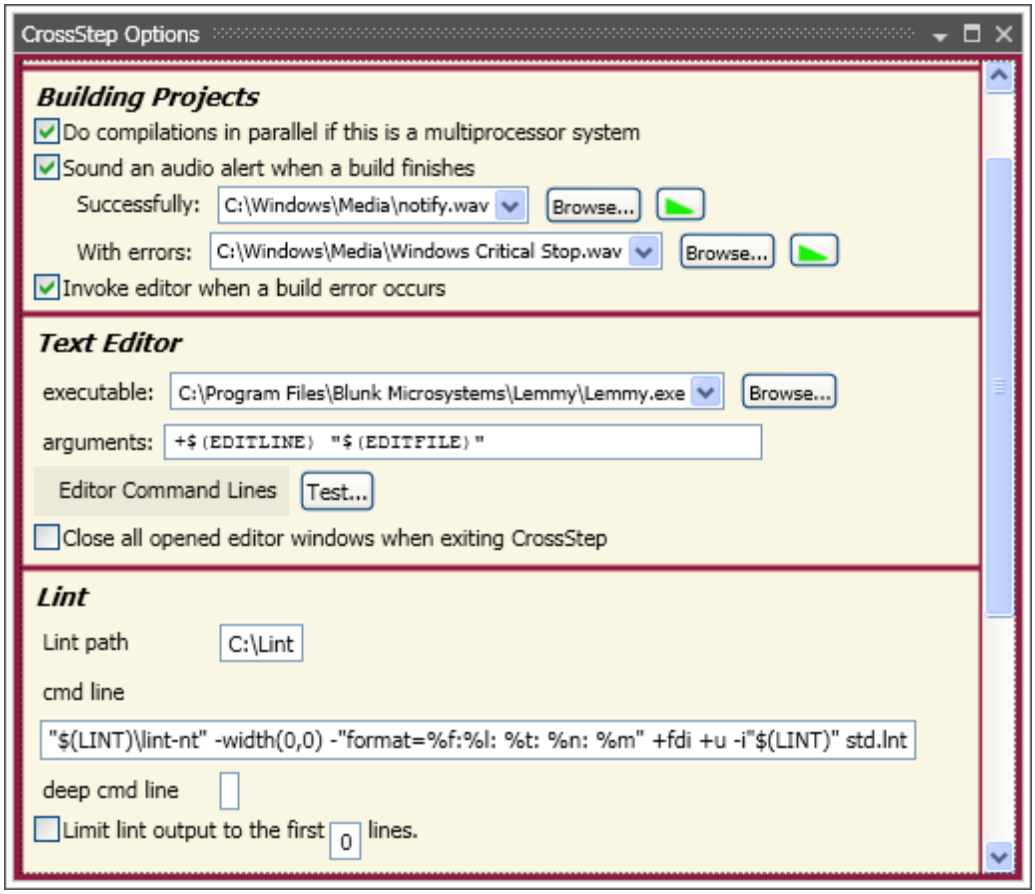
After the build completes, or instead of clicking the build button if you want to give a combined "build and launch debugger" command, click on the "debug" button or use keyboard shortcut control-D. This ensures the build is up-to-date, launches the source code debugger, downloads your application and (optionally) runs to its main() function.

You can use the debugger's step-in, step-over, step-out, etc. commands and breakpoint facility to peruse the sample application. The user interface is fairly intuitive.

Clicking the **Source Files** tab in the Project Manager window displays the application's source files. Double-click on any source file line to launch an editor on the selected file. Files listed in the **Headers** tab can be opened similarly.



In the Options menu, select 'General Options' to specify the editor CrossStep uses to open source files:



Enter the path to your preferred editor on the “executable” line. $(EDITLINE) and $(EDITFILE) on the “arguments” line are replaced with the desired line number and file name, respectively, when CrossStep launches your editor to - for example - open a source file at the line of a compile error. You must use these macros in the expected command line format for your editor. Click on “Editor Command Lines” to select among prefilled argument strings for several popular editors.

When finished with the debugger, click on the “End debug session” button  to close it. If you make changes to the code and save them, you can use the “debug” button  to rebuild the project, download it, and launch the debugger again.

7 Sample Applications

The demo package has other applications you can experiment with. The following sample applications are provided, which demonstrate various TargetOS middleware components. Each application contains a “readme.txt” with more details on the application.

bsearch - Loops over all installed file system volumes, measuring the number of binary searches performed in a set time interval.

clients – Cycles through this list of network clients: TCP echo, TCP discard, TCP chargen, UDP echo, and UDP chargen. The server IP address is specified via the TargetOS boot menu.

file_test - Loops over all installed TargetOS file systems, performing file system regression tests that primarily focus on the POSIX API.

graphics – Demonstrates TargetGraphics, Blunk’s embedded graphics drawing library. Only supported on targets with a frame buffer based display.

kbench - Measures kernel operations, including context switches in various conditions. The worst case value after multiple measurements is printed to stdout.

server - Runs the following servers: TCP echo, TCP discard, TCP chargen, UDP echo, and UDP discard. The server IP address is specified via the TargetOS boot menu.

shell - Launches the TargetOS command line interface. This is a shell that can be extended with user commands and accessed via UART connections and Telnet.

ui_demo – Demonstrates TargetOS’s support for touch panels, button, and joy sticks. Only supported on targets with a frame buffer based display and UI elements.

webserv - Demonstrates TargetWeb, Blunk’s embedded web server, both sending data to the target using browser forms and getting data from the target using custom pages that contain target data. Includes sample HTML pages.

Blunk Microsystems, LLC
Tel: 408/323-1758
Web: www.blunkmicro.com
Email: support@blunkmicro.com