microtronix

Linux Development Kit

Getting Started Guide

Version 1.5

System Requirements

The Linux Development Kit requires the following before beginning the installation:

1. Workstation (PC) with:
    -two serial COM ports available
    -Windows™ NT™ v4 or Windows 2000™

2. Altera Excalibur™ Development Kit featuring the Nios™ Embedded Processor (Version 1.1.1), with the GNUPro Tools and Cygwin™ installed.

3. QUARTUS II™ Programmable Logic Development Tools
    -This is only required if new devices are to be supported on the Excalibur Board that will require the Nios processor core to be rebuilt.

This guide assumes that the reader has basic knowledge of the Linux operating system and its conventions.

If you are new to the Linux operating system, it is recommended that one of many beginner or intermediate guides be purchased to give the necessary background information.

Introduction to the Linux Development Kit

The Linux Development Kit (LDK) includes an embedded version of Linux intended for processors without a Memory Management Unit (also referred to as MMU-less processors). The version of Linux distributed with the LDK is known as uClinux™, which Microtronix was responsible for porting to the Nios™ soft-core embedded processor. Also included are selected hardware components that have been developed to work with the Altera™ Excalibur™ Development Kit featuring the Nios processor. All of the tools and libraries necessary to supplement the Altera Software Development Kit (SDK) are also included to provide a complete development environment for building, loading, and debugging the Linux kernel. A group of user applications and examples completes the set.

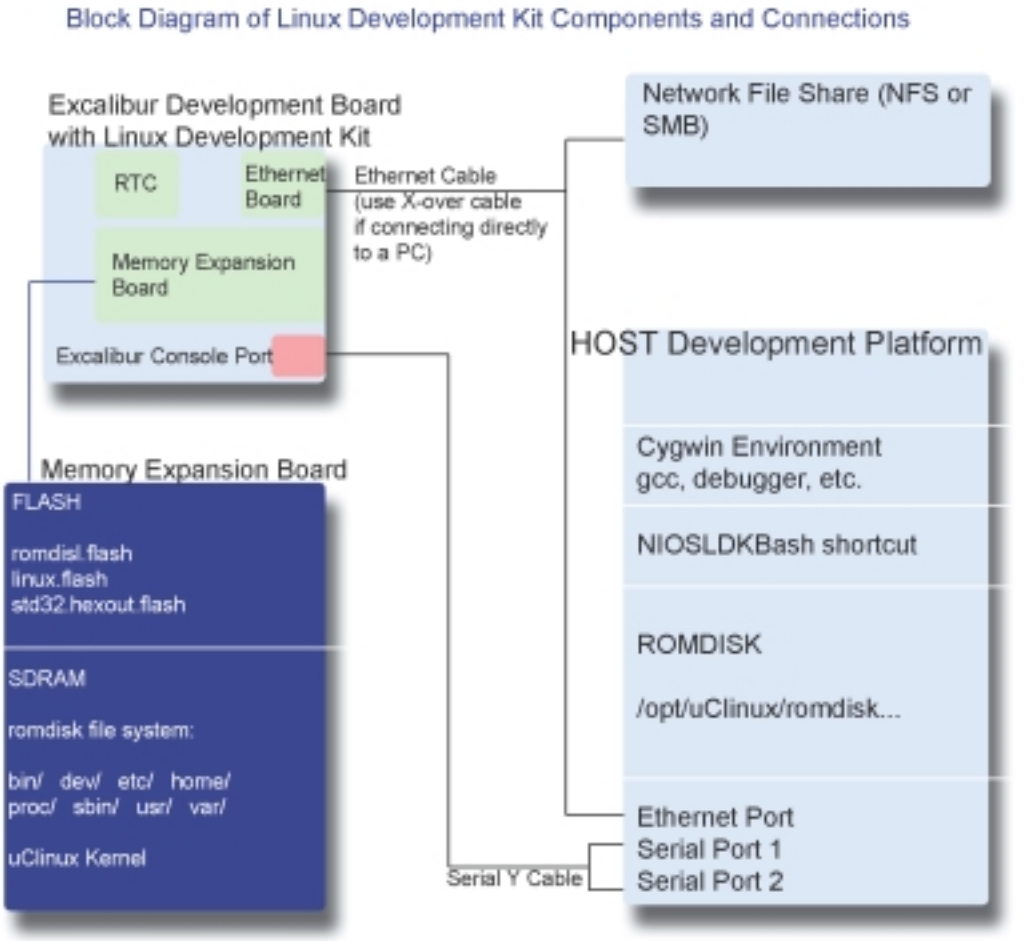
uClinux Overview

uClinux is an open source version of Linux modified to run on embedded processors lacking, or having limited, memory management capabilities -- such as the Altera Nios soft-core processor. Its memory usage is flexible enough to provide a small footprint with limited features, or it can be built as a feature-rich model.

The default configuration included in the LDK is medium size, providing TCP/IP networking, a DHCP client, a web server, telnet access, and both NFS and SMB clients for accessing Linux and Windows NT servers. These clients allow the development workstation to host the executable application images.

NOTE: Please see the README.TXT file located on the Software Support Package CDROM supplied with the Linux Development Kit for up to date information, including application notes.

Linux Development Kit Overview



The LDK is based on a Windows NT or 2000 user environment using the Cygwin tool chain. It is possible to set up a development environment in native LInux, however this environment is unsupported.

Linux Development Kit Overview

Working With Linux

If you have worked with a desktop version of Linux, you will be familiar with many of the conventions and utilities that you will find are a part of the Linux Development Kit environment. It is outside of the scope of this user guide to provide the background necessary to begin doing embedded development with Linux. Therefore, we recommend the following resources:

For more information on the uClinux kernel, visit:
www.uclinux.org

For more information on embedded Linux in general, visit:
www.linuxdevices.com
www.alllinuxdevices.com

For written resources on getting started with Linux, visit:
www.oreilly.com

For a free subscription to the Embedded Linux Journal (if you qualify), visit:
www.embeddedlinuxjournal.com

For updates to the Linux Development Kit documentation, visit:
www.microtronix.com

Linux Development Kit Overview

The Linux Filesystem

There are standard directories in any Linux filesystem that hold certain files. Not all of these directories need to be present (certainly not in your embedded system), and there may be others. Listed are the well known directories and a brief description of their purpose.

/bin: user commands are in this directory. This directory is part of the PATH statement that allows you (or your script files) to execute commands in this directory from any location.

/sbin: location of statically linked binaries (generally 'non-user' system tools).

/opt: this is the standard directory space for applications.

/mnt: this directory refers to temporarily mounted file systems. On a desktop system this might refer to a floppy or CDROM drive. In the LDK it is used for SMB and NFS mount points, described later on in this guide under the section entitled "Running Applications From A Server".

/lib: Runtime libraries are located in this directory.

/dev: the references in this directory represent physical devices, such as the serial port.

usr/src/linux: the Linux kernel sources are located in this directory.

/etc: applications that use configuration files will usually place the files here.

Linux Development Kit Overview

File System Structure and the LDK

It is important to remember that there are actually two file systems you will need to keep track of. One is the standard structure that is in your Cygwin environment. When you install the LDK for example, a subdirectory under /opt will be created called uClinux. This subdirectory will house all of the specific LDK files on your host machine.

The target file system that is expanded from flash into SDRAM when uClinux is initialized is created by using the ROMDISK.

More on the ROMDISK

The ROMDISK is a pre-built file system (located in the Cygwin subdirectory /opt/uClinux/romdisk) that is loaded from the flash memory and extracted into DRAM during the uClinux initialization process. The LDK comes with a sample ROMDISK loaded in the flash memory on the Memory Expansion board. The file system, once loaded out of flash into DRAM, has readonly access. However, you can build a custom ROMDISK by modifying the files and directories in the /opt/uClinux/romdisk structure, building a flash image, and loading it onto the Memory Expansion Board. For more information on building a custom ROMDISK, please refer to the “Rebuilding the ROMDISK” section in this guide.

Linux Development Kit Overview

Directories and Files in the Default ROMDISK

To view the default ROMDISK, simply launch the Cygwin shortcut (see your Cygwin documentation for details) and go to the /opt/uClinux/romdisk subdirectory. Everything in and below this directory will be collected into the flash image that you can load onto the LDK Memory Expansion Board.

Directories in the Default ROMDISK

`/bin  /dev  /etc  /home /mnt  /proc /sbin /usr  /var`

These directories serve essentially the same purpose as in any standard Linux distribution. However, because this is an embedded system, there are some differences.

First, there are no user profiles established (this is why it doesn't matter which login name you use when you login into the uClinux environment). Every time a login is made, it is at the system adminstrator level (or "root" as it is referred to in Linux.

Secondly, although this is a "default" file system, when you start to work on specific applications for your board you will customize this file structure to reflect the most efficient use of space. In the prebuilt system we've provided, included everything by default, however there will probably be a lot of applications, files, and directories that wouldn't be part of a "real" embedded environment.

Navigating the Default ROMDISK

Listed here are some important files and their locations:

bin/
All of the utility applications ported to the Nios Processor are included here.

etc/
Any configuration files used in your LDK are located here. Make sure that you understand any changes being made before attempting to modify or delete any files in this directory. For example, in the /config subdirectory you will find the configuration file for the Boa Web Server (boa.conf). Usually by referring to the text of a configuration file you will find which application the file is related to, and instructions on the proper method of modifying it.

sbin/
Applications used by the system (such as the Boa Web Server) are here.

dev/
All devices accessed by Linux are represented by a file. In this directory you will find references to specific devices that are available to your uClinux environment. Before attempting to modify or delete any files, it is important that you understand what is being changed!

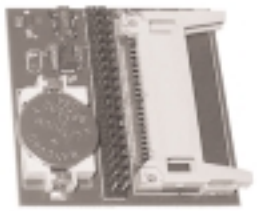
home/httpd
When uClinux initializes, the Boa Web Server starts up automatically. If you have established Ethernet connectivity to the board (see later in this guide for details), you can view pages from the Boa Web Server by opening up a web browser on your host machine and entering the IP address of the Ethernet Connectivity Board in the address field of the browser:

`http://[`*`IP address of Ethernet Board`*`]`

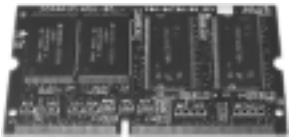
NOTE: For instructions on setting up the IP address for your LDK (if a DHCP server is not available) see the section "Running Applications From A Server" under the heading "Setting Up an IP Address. To find out what IP address your Ethernet Board is using, type the **`ifconfig`** *command at the Nios shell prompt.*

Getting Started - Hardware Components

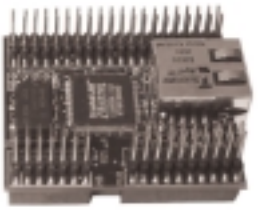
The Linux Development Kit contains a Memory Expansion Board, an Operating System Support Board, and an Ethernet Connectivity Board. Also included with the Kit is a Software Support Package CD, two ethernet patch cables (one straight-through, one crossover cable), and a serial Y cable. In the event that you have not received any of these components, please contact the Altera Applications Support Hotline at the number listed on the back of this Getting Started Guide.



Operating System Support Board



Memory Expansion Board



Ethernet Connectivity Board

IMPORTANT NOTICE

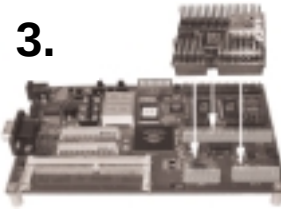
This documentation assumes that you are working with the default flash image that ships with the Excalibur Board. If you have programmed a user flash image onto your Excalibur Board it is mandatory that you reset the board to the default factory image by jumpering JP2 and resetting the board.

Getting Started - Hardware Installation

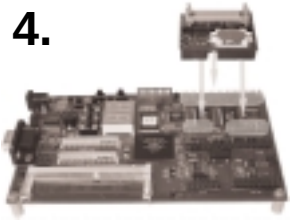
1. Power down the Excalibur Board, after verifying that the board is configured with the default factory flash image (see notice on previous page).

2. Plug the Memory Expansion Board into the SODIMM socket J2 by placing the card in the slot at the angle demonstrated in the figure to the right (A). When the card is inserted at this angle, press down lightly to lock it in place (B). You should hear a small click once the board has been secured.

3. Plug the Ethernet Connectivity Board onto the 3.3V Expansion Prototype Connector by lining up the sockets and pressing the board vertically downward. This connector consists of headers JP8, JP9, and JP10.



4. Plug the Operating System Support Board onto the 5V Expansion Prototype Connector by lining up the sockets and pressing the board vertically downward. This connector consists of headers JP11, JP12, and JP13.



5. If the Ethernet Connectivity Board is to be connected to a hub then the Ethernet Patch Cable should be used. If the Ethernet Board is to be attached directly to another ethernet card, the Ethernet Cross-Over Patch Cable should be used.

6. The Serial Y-Cable is used to connect two UARTs built into the core to two seperate serial COM ports on a PC. The male end should be connected to the serial port connector J3 on the Excalibur Board. The female end, labelled COM1, is the main console serial port, and it must be connected to the PC's COM1 port before any communication with the board can occur. The other female end (labelled COM2) is meant to be used for debugging with GDB. You only need to connect this port if you plan to perform debugging tasks.

7. Power on the Excalibur Board.

Getting Started - Software Installation

Installing the Linux Development Kit

1. Ensure that the "Software Support Package" CD is placed in an available CD-ROM drive. You should have Adobe Acrobat Reader installed to view all documentation files.

2. Start up a Cygwin /bash window and change to the directory corresponding to the CD-ROM drive containing the "Software Support Package" CD (for example, d:/). From there, the Nios LDK Installation script can be run:

```
[bash]..src/: cd d:/
[bash]..d/: ./Install.sh
```

Where "d:" is the letter designation of the CD-ROM drive. Install.sh is a script file that will extract the LDK from its archive on the "Software Support Package" CD and copy a shortcut onto your desktop (Note: this shortcut points to a path that assumes Cygwin is installed to the default C:\Cygwin\. If your installation of Cygwin does not reside in this default directory, the properties of the desktop shortcut will need to be changed).

3. The Nios LDK environment is now set up. When developing using the LDK you should run bash with this new environment using the desktop shortcut, **NiosLDKBash**.

Getting Started - Software Installation

In order to support the devices now plugged into the Excalibur Board, the Nios core needs to be updated with the one supplied in the LDK. The following steps must be taken to load this new core into the flash memory located on the Excalibur board.

1. Ensure that the serial Y-Cable is connected between the Excalibur port and COM1 of the PC.

2. Invoke the LDK bash window by double-clicking the **NiosLDKBash** icon on the desktop.

3. Make sure GERMSmon is running on the board and load the new core into flash by entering the command:

```
[LDKbash]..uClinux/: nios-run std32.hexout.flash
```

NOTE: ***Follow the on-screen instructions to complete the load.***

4. After **nios-run** indicates that the download has completed successfully, restart the Board by performing a power cycle.

The prebuilt images of uClinux and its companion ROMDISK are pre-loaded into the Memory Expansion Board's flash memory. The new GERMSmon will look for an executable image in this flash when it is restarted.

nios-run (which has remained in terminal mode) will now display the messages from the uClinux startup.

5. In order to log into uClinux, at the login prompt any user name can be used. However, the password that must be entered is **uClinux**

When uClinux is initialized it is set up to look for a DHCP Server by default. However, if a DHCP Server is not available, an error message will be displayed. To manually set the IP address and other variables, you must use the **`ifattach`** *command. A description of how to use this command is under the "Setting an IP Address" heading in the "Running Applications From a Server" section later on in this guide. If you need to find out the IP address of the board, use the* **`ifconfig`** *command at the Nios shell prompt.*

Creating New Applications

Creating a new application is a relatively straightforward task, as most of the build process has been automated. The steps are as follows:

1. Create a directory under the **root** of the application tree. The default is **/opt/uClinux/niosuserland**. Example:

```
[LDKbash]..//: cd /opt/uClinux/niosuserland
[LDKbash]../niosuserland/: mkdir myNewApp
```

2. Edit the Makefile in the application root and add your new application. To do this, locate the **SUBDIRS** variable near the bottom of the make file and add your application directory name there, for example, add **$(myNewApp)**. Then scroll up to where the application directory names are defined (such as **myNewApp:=myNewApp**). Here you can also comment out the applications you don't want to build. For example, **myOldApp:=myOldApp** would result in **$(myOldApp)** evaluating to **“”**, thus it wouldn't get built.

3. Peruse the **niosuserland** tree and find a Makefile that most closely matches your requirements. For a simple application, consider the makefile in **hello/Makefile**. A more complex example can be found in **route/Makefile**. This makefile builds a library and an executable using that library. Edit the makefile to reflect your application source files. Note that the default target for executables is **/opt/uClinux/romdisk/bin**.

If your needs differ (for instance, if you wish to place your executable onto an SMB server), edit the **FTARGET** variable in your makefile to reflect the proper path, for example: **/opt/uClinux/niosuserland/bin** as in **hello/Makefile.** Refer to the “Running Applications From a Server” section for more details.

4. Build the application according to the steps defined in this guide under the section “Rebuilding the niosuserland Applications.”

Points to Keep in Mind

1. The top-level Makefile and Rules.mak do most of the work for you, allowing your makefiles to be relatively simple.

2. Try to reuse an existing makefile whenever possible.

Rebuilding the niosuserland Applications

The source code for the included applications reside in the directory **/opt/uClinux/niosuserland/**. In this directory, there is a top level makefile that builds all of the applications which reside in the various subdirectories.

To force a rebuild of all applications, use the following command:

```
[LDKbash]..niosuserland/: make clean
```

This is only necessary if the uC-libc library or the Linux header files have changed.

To build a new application or rebuild an application whose source files have changed, use the following command:

```
[LDKbash]..niosuserland/: make
```

The **make** command copies the resulting executables to:

```
/opt/uClinux/romdisk/bin/
/opt/uClinux/romdisk/sbin/
```

This is so that the applications will be included in the ROMDISK image when it is rebuilt (refer to the section “Rebuilding the ROMDISK” for information about this procedure).

Executables may also be copied to a subdirectory outside of the ROMDISK to be run from the target via SMB or NFS. Refer to the section “Running Applications From a Server” for details. The default directory is:

```
/opt/uClinux/niosuserland/bin/
```

The individual application makefiles dictate to which directory executables are copied. A ROMDISK rebuild and reload is required if the **/romdisk/** directory changes.

Rebuilding the ROMDISK

The Nios LDK comes with a pre-built romfs image (ROMDISK) programmed into the flash memory of the Memory Expansion Board. The loadable image is located at:

```
/opt/uClinux/romdisk.flash
```

The ROMDISK acts as the root filesystem, and includes various device files (nodes) and application executables. The ROMDISK is an image made from an existing directory tree which resides under:

```
/opt/uClinux/romdisk/
```

Therefore, any new files that are to be included in the root filesystem should first be placed under this directory. The following command should then be run to rebuild the image from that tree:

```
[LDKbash]../uClinux/: ./mkflashromfs
```

The mkflashromfs script will create the new romdisk.flash, which contains the necessary commands for erasing the flash reserved for this purpose, and the romdisk image itself. The image can then be loaded onto the Excalibur Board's flash memory. The procedure for accomplishing this is:

```
[LDKbash]../uClinux/: nios-run romdisk.flash
```

Note: If there is already a Linux image running on the board, you must press and hold the SW4 button and press the RESET button (SW2) before using this command.

This will program the image into flash. **nios-run** will remain in terminal mode to allow testing of the new romfs. If the Linux kernel is already loaded into flash, it should start up automatically; otherwise press the CLEAR (SW3) button to restart the kernel.

Running Applications From a Server

The default kernel configuration shipped with the LDK includes both SMB and NFS support. This will allow you to mount a remote SMB share or NFS mount point. If you remove either of these and rebuild the kernel you will not be able to mount the respective remote system. Because of the convenience of being able to load and execute an application from a remote system where the applications are being built it is recommended that SMB and NFS be left in the kernel until application debugging is complete.

SMB

For SMB you will need to share the drive or folder that you want to access from the Linux Development Kit. For information on doing this follow the Windows Operating System instructions.

NFS

For NFS you will need to install the NFS server software. You will also need to specify which file system to export using NFS. For information on doing this, follow the NFS HOWTO document, that can be found at **http://www.linuxdoc.org/HOWTO/NFS-HOWTO**.

Once you have the remote file system mounted the applications can be built and tested without having to rebuild the romdisk. This provides a significant time benefit.

Setting Up an IP Address

Before mounting a remote file server, the ethernet controller needs to be activated and an IP address assigned. At the uClinux console, type the **ifattach** command:

```
ifattach  --addr <local_IP_address> --mask <subnet_mask>
          --net <network_id> --gw <gateway_address>
          --if <device>
```

Example:

```
# ifattach --addr 10.1.2.9 --mask 255.255.255.0
      --net 10.1.2.0 --if eth0
```

NOTE: In order to setup a loopback interface (i.e. 127.0.0.1) and route it is necessary to run the **ifattach** command with no arguments at the command prompt.

Running Applications From a Server

Setting Up an IP Address continued...

To make this more permanent, edit the **rc** file in the **/etc** directory of the ROMDISK. See the section entitled “Rebuilding the ROMDISK” for procedures on rebuilding and reloading the ROMDISK in flash.

Mounting an SMB Remote File System From uClinux

To SMB mount a remote shared drive or folder you will need to know the server name, share name, server IP address and the password (if one was assigned). You will also need a mount point which was either created on the rom file system or in the ram disk. For example:

To mount the following system:

Server Name:	**hobbit**
Share Name:	**frodo**
Password:	**theonering**
IP Address:	**10.1.1.50**

at mount point: **/mnt/golem**

You would type (on one line):

```
# smbmount //hobbit/frodo /mnt/golem
                -I 10.1.1.50 -P theonering
```

Mounting an NFS Remote File System from uClinux

To NFS mount a remote NFS server mount point you will need to know the exported file system and the NFS server IP address. For example:

To mount the following system:

Server Name:	**gandalf**
Exported File System:	**/opt**
IP Address:	**10.1.1.52**

at mount point: **/mnt/gandalf/opt**

You would type:

```
# mount -t nfs 10.1.1.52:/opt /mnt/gandalf/opt
```

Rebuilding the Linux Kernel

The Nios LDK comes with a pre-built Linux kernel, named **linux.srec**, located in the **/opt/uClinux/linux** directory. This is a compiled, ready to run kernel that can be loaded onto the Excalibur Board.

However, if changes are to be made to the Linux kernel code, this srec file must be rebuilt and reloaded onto the board. Rebuilding the kernel can be accomplished by using the following commands:

```
[LDKbash]..linux/: make clean
[LDKbash]..linux/: make
```

Where:

make clean deletes all executable and object files.This ensures that existing object files are not reused.

make will actually compile the source and build the final srec file. This is the only step required if changes have been made to existing source files and the dependencies and configuration have not been modified.

Other make options are:

```
make mrproper
make config
make dep
```

make mrproper deletes all executable, object, configuration, and dependency files. This ensures a fresh start for building the kernel.

make config will allow the desired kernel options to be selected and used. Note that at this time, loadable kernel modules are not supported. This creates the configuration files used to control the build process.

make dep will create the dependencies. This step is mandatory if a source or header file has been added to the kernel tree or if include directories have been added or removed. This step also needs to be run if a **make mrproper** has just been run, since **make mrproper** deletes dependencies.

Rebuilding the Linux Kernel

After building a new kernel, the **linux/linux.srec** file is suitable for downloading into RAM memory using **nios-run**, for example:

```
[LDKbash]../linux/: nios-run linux.srec
```

To make a more permanent image of the kernel that will start up automatically when power is applied or the CLEAR button (SW3) is pressed, the image can be loaded into flash. The procedure for accomplishing this is:

1. Prepare a loadable file in the uClinux directory:

```
[LDKbash]../linux/: cd ..
[LDKbash]../uClinux/: ./mkflashlinux
```

This creates **linux.flash**, which contains the necessary commands for erasing the flash reserved for this purpose, and the kernel load itself.

2. Program the flash:

Note: If there is already an existing Linux image running, you must press and hold the SW4 button, then press and release the reset button (SW2) before using this command.

```
[LDKbash]../uClinux/: nios-run linux.flash
```

This will put the image into flash and begin execution. **nios-run** will remain in terminal mode, and the Linux startup messages will be displayed.

Annual Support Contracts

Microtronix offers many support options depending on your needs.

Standard Annual Support Contract

Unlimited Email Support (next business day response)
Six Phone Support Incidents (next business day response)
GNUPro Support
Software Patches and Upgrades available
Future version upgrades special pricing

Premier Annual Support Contract

Unlimited Email Support (Four hour response, 9am - 5pm EST)
Ten Phone Support Incidents (Four hour response, 9am - 5pm EST)
GNUPro Support
Software Patches and Upgrades available
Future version upgrades special pricing

Customized Support

Microtronix offers customized project support and engineering services. Please contact sales@microtronix.com for more information.

Microtronix Website

Be sure to visit our website for up to date information, including other add on modules for your Linux Development Kit.

If you require assistance installing your LDK, please email Altera Technical Support:

support@altera.com

If you would like to purchase an annual support contract, or if you are interested in any of the following services:

Device Driver Development
Hardware Design, Layout, and Manufacturing
Linux Kernel Development
Programmable Logic Design and/or Prototyping
IP Integration
Embedded Systems Services

Please email ldk@microtronix.com or visit our website (www.microtronix.com) for more information.

Microtronix Datacom Ltd.
120 Bessemer Road
London, ON, CA
N6E1R2

www.microtronix.com

951-6000-01-B