



# ***EFCAPI Installation Guide for Linux***

Software Drivers, Documentation, and API  
(Application Programming Interface) for  
EFC, GRAVITY and GRAVITY II Cards

Rev B  
October 31, 2005

---

#### **Contact Information**

<b>Telephone</b>	1 (866) 478-4491
<b>Company Fax</b>	(505) 883-1375
<b>Email</b>	grt@greatrivertech.com
<b>Web Address</b>	www.greatrivertech.com
<b>Location</b>	6121 Indian School Rd NE Ste 141 Albuquerque, NM 87110

---

## Table of contents

---

<b>TABLE OF CONTENTS .....</b>	<b>2</b>
<b>OVERVIEW.....</b>	<b>3</b>
<b>FILES ON THE SUPPLIED SETUP CD .....</b>	<b>4</b>
<b>INSTALLATION STEPS.....</b>	<b>5</b>
INSTALL THE API LIBRARY .....	5
INSTALL THE DEVICE DRIVER.....	6
<i>Using a pre-built device driver module .....</i>	<i>6</i>
1. Determine your kernel version .....	6
2. Check for pre-built kernel module on the setup CD. ....	6
3. Copy pre-built driver module to library directory .....	6
<i>Building the device driver to your kernel .....</i>	<i>7</i>
1. Copy device driver tar file .....	7
2. Extract source files from the tar file .....	7
3. Set environmental variables.....	8
4. Set up path for Linux source files .....	9
5. Build device driver module .....	9
6. Copy driver module to library directory .....	10
LOAD THE DEVICE DRIVER.....	10
<b>INSTALLED FILES.....</b>	<b>12</b>
<b>UPGRADING TO A NEW LIBRARY ONLY .....</b>	<b>13</b>
<b>INSTALLING FEDORA CORE 3, KERNEL 2.6 SOURCE.....</b>	<b>14</b>

---

## Overview

---

This document will outline the steps necessary to install the Great River Technology EFCAPI software for Linux operating systems.

This software is for use on Linux kernel versions 2.2.X, 2.4.X and 2.6.X. Due to the numerous flavors of Linux and hardware variations, GRT cannot test and support on all Linux distributions and hardware. However, most configurations should work with minimal porting effort.

Hardware supported is GRT\_EFCPCI-001 Fibre Channel card, the GRT-FCAV-PMC card or any GRAVITY or GRAVITY II (64 bit) series data and video cards.

This document describes the software installation only. Refer to the API Users Manual for detailed information on how to use the API for custom programming.

## Files on the Supplied Setup CD

The following files are on the setup CD.

Directory	Filename	Description
\Library	EFCAPI_X_Y_Z.tar	API Library tar file, where X_Y_Z is the library version number
\Driver\Source	PlxLinuxGRT.tar	Device driver source code tar file, including GRT-specific code for device driver build (files Driver.c, DriverDefs.h and ModuleVersion.c)
\Driver\Modules	Pci9054_VVV_X_Y_Z.ko Pci9656_VVV_X_Y_Z.ko Pci9054_VVV_X_Y_Z.o Pci9656_VVV_X_Y_Z.o	Pre-built device driver modules for some Linux kernels, where VVV is the vendor ID and X_Y_Z is the Linux kernel version. For vendor ID: FC1 = Fedora Core 1 FC2 = Fedora Core 2 FC3 = Fedora Core 3 S80 = SuSE 8.0 S81 = SuSE 8.1
\Docs	EFCAPI_InstallGuide_Linux.pdf	This document
\Docs	SDK_HL2DAPI_UserMan.pdf	Hotlink II Data functions API Users guide
\Docs	SDK_HL2VAPI_UserMan.pdf	Hotlink II Video functions API Users guide
\Docs	SDK_FCAVAPI_UserMan.pdf	FCAV functions API Users guide

---

## Installation Steps

---

The steps to install are

1. Install your hardware.
2. Install the API library.
3. Install the device driver.
4. Load the device driver.

### *Install the API Library*

The API library is on the setup CD in the  
    \Library  
directory.

The API library files are in the tar file  
    EFCAPI\_X\_Y\_Z.tar  
Where X\_Y\_Z is the API library version number.

Installation:

The installation package consists of a tar zip compressed archive (EFCAPI\_X\_Y\_Z.tar where 'X'=version, 'Y'=revision, 'Z'=release), which must be UN-zipped as 'root' user. This operation will install all system files required of the package, and produce '/grtefc' directory tree and a product information file 'Readme.txt' in that directory.

Installation is accomplished as follows:

Login as superuser and go to the top-level directory

```
cd /
```

Copy the archive to the top-level directory with the cp command:

```
cp <cdpath>/Library/EFCAPI_X_Y_Z.tar /
```

where <cdpath> is the full path to your CDROM drive (for example: /mnt/cdrom).

Untar the archive as follows:

```
tar -xvpPf EFCAPI_X_Y_Z.tar
```

Create the shared library links and cache for run time linking of the newly installed libraries by typing:

```
ldconfig
```

The sample program source and makefiles, and read-me will now be available in a new directory called /root/grtefc. This directory can be relocated if desired. Refer to the read-me for package update information and general library usage instructions.

## ***Install the Device Driver***

Depending on your Linux vendor and kernel version, you may be able to use one of the pre-built device driver modules on the Linux API setup CD. GRT supplies some pre-built device driver files for specific kernels. If you are running one of these kernel versions you may use the supplied pre-built device driver module, saving you a considerable amount of work. Of course, you may choose to build the device driver module even if the pre-built device driver file is supplied. Do not attempt to use a pre-built device driver module that does not match your kernel version. Use one of the following options (A or B) to install the device driver for your card(s).

### Using a pre-built device driver module

1. Determine your kernel version

Use the following command:

```
uname -r
```

to get your kernel version.

2. Check for pre-built kernel module on the setup CD.

Pre-built device driver modules are in:

```
/Driver/Modules
```

These pre-built device driver files are named as follows:

```
Pci9054_VVV_X_Y_Z.ko
```

```
Pci9656_VVV_X_Y_Z.ko
```

```
Pci9054_VVV_X_Y_Z.o
```

```
Pci9656_VVV_X_Y_Z.o
```

Where the Pci9054\*.\* files are for the 32-bit cards and the Pci9656\*.\* files are for the 64-bit cards. In these file names, VVV is the vendor ID and X\_Y\_Z is the Linux kernel version. The \*.ko files are for kernel 2.6.X and the \*.o files are for kernels prior to 2.6.X. The following vendor IDs are used:

```
FC1 = Fedora Core 1
```

```
FC2 = Fedora Core 2
```

```
FC3 = Fedora Core 3
```

```
S80 = SuSE 8.0
```

```
S81 = SuSE 8.1
```

For example, the kernel modules for Fedora Core 3, kernel 2.6.9, are Pci9054\_FC3\_2\_6\_9.ko (32-bit card) and Pci9656\_FC3\_2\_6\_9.ko (64-bit card).

If you do not find a pre-built device driver module for your kernel version, proceed to the section below titled "Building the device driver to your kernel". Do not attempt to use a pre-built device driver module that does not match your kernel version.

3. Copy pre-built driver module to library directory

Do not attempt to use a pre-built device driver module that does not match your kernel version.

If the directory  
    /usr/lib/efc  
does not exist, create it.

If you found a pre-built module for your kernel on the CD, copy this file to:  
    /usr/lib/efc/Pci9054.\*  
for 32-bit cards, or  
    /usr/lib/efc/Pci9656.\*  
for 64-bit cards, where the “\*” is “ko” for kernel 2.6.X and “o” for kernels previous to 2.6.X.

If you were able to use a pre-built device driver module and have completed this step, you do not need to build the device driver to your kernel so skip the section titled “Building the device driver to your kernel” below and proceed to the section titled “Load the Device Driver Module”.

## Building the device driver to your kernel

If you were able to use a pre-built device driver module, you do not need to build the device driver to your kernel so skip this section and proceed to the section titled “Load the Device Driver Module”.

The *kernel sources package* must be installed in order to build the drivers. Without the kernel sources the kernel header files are not be available for building the device driver module. For kernel versions 2.4.X, the kernel sources can be installed during Linux installation. Typically this package must be manually selected during the installation process. Please refer to your Linux installation documentation. Starting with Kernel 2.6, the Linux kernel build system was introduced. In order to build the drivers, the Linux build system requires that the Linux kernel source be installed and the current running kernel must be configured and built. If this is not done, the driver builds will fail. The Fedora Core 3 distribution no longer includes the kernel source package, so the kernel source RPM file must be downloaded and the RPM installed. If you are running Fedora Core 3 with kernel 2.6, see the instructions for installing the kernel source in the section titled “Installing Fedora Core 3, Kernel 2.6 Source” in this document before proceeding with the GRT EFC API installation.

### 1. Copy device driver tar file

The device driver source file  
    PlxLinuxGRT.tar  
is on the setup CD in the directory:  
    **Driver\Source**

Copy the file *PlxLinuxGRT.tar* to the working directory  
    /usr/src

### 2. Extract source files from the tar file

Use a TAR file extractor to extract all the files from the tar file  
-OR-  
Open a terminal window and change the current directory to  
    /usr/src  
then type the following command  
    tar -xvf PlxLinuxGRT.tar

to extract the files.

This will create a folder called *PlxLinux* with all PLX device driver source supplied files and folders inside:

<PlxLinux/linux/include>:	PLX SDK include files
<PlxLinux/linux/api>:	PLX Host API Library source
<PlxLinux/linux/driver>:	PLX device driver (Linux installable module)
<PlxLinux/linux/makefiles>:	Shared makefile components used to build PLX applications, libraries, and drivers
<PlxLinux/bin>:	Scripts to load, unload, and assist in debug of the driver/module

### 3. Set environmental variables

In order to build the device driver, the shell environment variables **PLX\_SDK\_DIR** and **KERNEL\_VER** must be set. These may be set from the command line or commands added to the shell initialization script.

#### **PLX\_SDK\_DIR**

The shell environment variable **PLX\_SDK\_DIR** must be set. This must be set to the root location of where the *PlxLinux* folder is created. This can be set with the *declare* command. This command may be executed from the command line or placed in the shell initialization script (e.g. “.*bashrc*”) so that it is automatically set when a new terminal session is started.

The command to set this is:

```
declare -x PLX_SDK_DIR=/usr/src/PlxLinux
```

This command may be placed in the shell initialization script (e.g. “.*bashrc*”). Here is an example entry for the .*bashrc* file:

```
# PLX SDK environmental variables
echo "SETTING PLX PATH"
declare -x PLX_SDK_DIR=/usr/src/PlxLinux
declare -x KERNEL_VER=2.4
# PLX SDK command paths
export PATH=$PATH:/usr/src/PlxLinux/linux/driver
export PATH=$PATH:/usr/src/PlxLinux/bin
```

Once these changes are made to the .*bashrc* file, you must exit your terminal and restart another one or reboot to incorporate the changes. You may wait to reboot after modifying the .*bashrc* file for the **KERNEL\_VER** variable (described below).

#### **KERNEL\_VER**

The Linux device driver code has been tested with Linux kernels 2.2, 2.4, and 2.6. By using special coding techniques, drivers may be built for each of these kernels, while still retaining a common code base. The build scripts will attempt to automatically select the correct kernel version based on the existence of certain files in the system. The desired kernel version can be overridden, however, by setting the **KERNEL\_VER** environment variable. If the **KERNEL\_VER** variable is not set in the environment, it will be set in the **builddriver** script file.

The value of **KERNEL\_VER** determines the kernel version to build the driver for. Note that once **KERNEL\_VER** is set, all subsequent driver builds will be for the newly selected kernel. To build for the alternate kernel, the variable must be modified and the driver re-built.

The **KERNEL\_VER** variable can be set with the *declare* command. This command may be executed from the command line or placed in the shell initialization script (e.g. “.bashrc”) so that it is automatically set when a new terminal session is started.

Method 1: Set the environment variable **KERNEL\_VER** to the desired kernel version, 2.2, 2.4, or 2.6. This can be set from the command line or set in the shell initialization script (e.g. “.bashrc”) so that it is automatically set when a new terminal session is started.

An example of the command to set this variable for Linux kernel 2.4 is:

```
declare -x KERNEL_VER=2.4
```

This command may be placed in the shell initialization script (e.g. “.bashrc”). Some example entries for the .bashrc file follow:

For kernel v2.2, use:

```
declare -x KERNEL_VER=2.2
```

For kernel v2.4, use:

```
declare -x KERNEL_VER=2.4
```

For kernel v2.6, use:

```
declare -x KERNEL_VER=2.6
```

#### 4. Set up path for Linux source files

If you are on kernel 2.4, rename the kernel source directory named /usr/src/linux-2.4.xx-xxxxx to /usr/src/linux-2.4 using a command like:

```
mv /usr/src/linux-2.4.xx-xxxxx /usr/src/linux-2.4
```

Where xx-xxxxx is correct for the directory name found on your system.

If you are on kernel 2.6, create a symbolic link to the kernel source directory with the following command:

```
ln -s /usr/src/redhat/BUILD/kernel-2.6.X/linux-2.6.X /usr/src/linux-2.6
```

Where X is the third number in your kernel version. For example, for kernel 2.6.9-1.667, the command would be:

```
ln -s /usr/src/redhat/BUILD/kernel-2.6.9/linux-2.6.9 /usr/src/linux-2.6
```

#### 5. Build device driver module

Go to the directory

```
\usr\src\PLXLinux\linux\driver
```

and type

```
./buildalldrivers cleanall
```

This removes all the .o (and .ko) files so that a new full build will occur.

To build the driver files for Gravity 32 bit cards, go to directory

```
\usr\src\PLXLinux\linux\driver
```

and type

```
./bulddriver 9054
```

This builds the Pci9054.o (or Pci9054.ko) driver used for Gravity 32 bit cards.

To build the driver files for Gravity 32 bit cards go to directory

`\usr\src\PLXLinux\linux\driver`

and type

**`./builddriver 9656`**

This builds the `Pci9656.o` (or `Pci9656.ko`) driver, used with 64 bit GravityII cards .

## 6. Copy driver module to library directory

If the directory

`/usr/lib/efc`

does not exist, create it now.

### **For Gravity 32 bit cards**

If the `Pci9054.o` file (or `Pc9054.ko` for kernel 2.6) is built ok then copy this file from

`/usr/src/PLXLinux/linux/driver/Pci9054/Driver`

to

`/usr/lib/efc`

For the 32 bit GravityII cards, this is the loadable module that is custom built to your kernel.

### **For Gravity 64 bit cards**

If the `Pci9656.o` file is built ok then copy this file from

`/usr/src/PLXLinux/linux/driver/Pci9656/Driver`

to

`/usr/lib/efc`

For the 64 bit GravityII cards, this is the loadable module that is custom built to your kernel.

## ***Load the Device Driver***

Now you have either copied in the pre-built device driver file or built it, so you can load the module and see if it connects to our board.

Loading and unloading of the driver is accomplished by running the appropriate script as superuser (root user). The script `'/usr/sbin/efcload'` or `'/usr/sbin/efcload2'` will create the required device special files and load the driver into the kernel, and `'/usr/sbin/efcunload'` or `'/usr/sbin/efcunload2'` does the reverse. Calls to these scripts can also be made at startup or login time in the runtime command (rc) section of the `'/etc'` tree, or user startup files, respectively. To gain a better understanding of what the scripts do, it is suggested that the user view them.

To load the 32-bit Gravity card driver, type

**`./efcload`**

to load the `Pci9054.o` file (or `Pci9054.ko` file for kernel 2.6). This script will first look for `Pci9054.ko` and then `Pci9054.o`, so make sure you have installed the `Pci9054.ko` file if you are on Linux kernel 2.6. Note that if you get the error message:

`insmod: error inserting '/usr/lib/efc/Pci9054.ko': -1 No such device`

this means that the hardware is not installed or you are attempting to load the driver for the wrong card type (32-bit vs. 64-bit).

To load the 64-bit GravityII card driver module, type

**`./efcload2`**

to load the `Pci9656.o` file (or `Pci9656.ko` file for kernel). This script will first look for `Pci9054.ko` and then `Pci9054.o`, so make sure you have installed the `Pci9054.ko` file if you are on Linux kernel 2.6 (Linux 2.4 only uses the `*.o` file). Note that if you get the error message:

insmod: error inserting '/usr/lib/efc/Pci9054.ko': -1 No such device  
this means that the hardware is not installed or you are attempting to load the driver for the wrong card type (32-bit vs. 64-bit).

Should you need to unload the driver module (not normally necessary), the commands are as follows.

To unload the 32bit driver module, type

**./efcunload**

To unload the 64-bit Gravity card driver module, type

**./efcunload2**

Once the loadable kernel module is loaded, you can write code using calls to the libefc.so library functions and run your applications. All projects that use this API must link to this API library during compilation. The Makefile found in the 'sample' directory demonstrates how this is done.

**EXAMPLE of a successful module load for the Pci9054.o file using efcload:**

```
*****  
* NOTE: You must be superuser, logged in as root, *  
* or have sufficient rights to install *  
* modules or this script will not work. *  
*****
```

```
Installing module (Pci9054)....
```

```
insmod version 2.4.19
```

```
Using /usr/lib/efc/Pci9054.o
```

```
Symbol version prefix ''
```

```
Getting Module major number..... Ok (MajorID = 254)
```

```
Creating device node path..... /dev/plx already exists
```

```
Creating device nodes..... Ok
```

```
Module load complete.
```

```
[rootlinux root]#
```

## Installed Files

---

Upon installation the following files will be installed on the system.

Header files:

/usr/include/efclib.h      Library header file

Library files:

/usr/lib/libefc.a          Static library file  
/usr/lib/libefc.so.x.xx.0   Shared library file  
/usr/lib/libefc.so.x       Shared library link  
/usr/lib/libefc.so         Shared library link

Driver module load/unload scripts:

/usr/sbin/efcload          Loads the driver for 32 bit Gravity cards  
/usr/sbin/efcunload        Unloads the driver for 32 bit Gravity cards  
/usr/sbin/efcload2         Loads the driver for Gravity 64 bit cards  
/usr/sbin/efcunload2       Unloads the driver for Gravity 64 bit cards

Sample project files:

~/grtefc/efctest.c          Sample code for FCAV  
~/grtefc/Makefile\_EFCTest    Makefile for efctest  
~/grtefc/efcdata.c          Sample code for Gravity HL2D (Data cards)  
~/grtefc/Makefile\_Data       Makefile for efcdata  
~/grtefc/efchl2v.c          Sample code for Gravity HL2V (Hotlink Video)  
~/grtefc/Makefile\_HL2V       Makefile for efchl2v  
~/grtefc/efcfcav.c          Sample code for Gravity FCAV (Fiber Channel Audio Video)  
~/grtefc/Makefile\_fcav       Makefile for efcfcav  
~/grtefc/HL2Vrate1.c        Sample code for Gravity HL2V (Hotlink Video)  
~/grtefc/Makefile\_HL2Vrate1   Makefile for HLK2Vrate1  
(note: '~' is the root user's home directory)

Documentation:

~devel/grtefc/EFCAPI\_LinuxInstall.pdf      This file  
~devel/grtefc/GRAVITY\_FCAV\_UserMan.pdf     User manual for the FCAV card  
~devel/grtefc/GRAVITY\_HL2D\_UserMan.pdf     User manual for the HL2D card  
~devel/grtefc/GRAVITY\_HL2V\_UserMan.pdf     User manual for the HL2V card  
~devel/grtefc/SDK\_FCAVAPI\_UserMan.pdf      User manual for the FCAV API  
~devel/grtefc/SDK\_HL2DAPI\_UserMan.pdf      User manual for the HL2D API  
~devel/grtefc/SDK\_HL2VAPI\_UserMan.pdf      User manual for the HL2V API

Links (kernel 2.6 only):

/usr/src/linux-2.6 -> /usr/src/redhat/BUILD/kernel-2.6.X/linux-2.6.X

---

## Upgrading to a New Library Only

---

Note that these steps are for upgrading the library files only (\*.so and \*.a) and do not cover upgrading of the kernel module(s). The include file used to build applications is also upgraded. Any applications built using the EFCAPI library will need to be rebuilt following this upgrade procedure.

The library upgrade is supplied as a compressed TAR file in the format

EFCAPI\_X\_Y\_Z\_LIBS.tar (where X\_Y\_Z is the version number)

This TAR file contains new library files and an updated header file

libefc.a	Static library file
libefc.so.x.xx.0	Shared library file
libefc.so.x	Shared library link
libefc.so	Shared library link
efclib.h	Library header file

1. Log in as superuser (root user)
2. First the old libraries and links must be removed.  
Got to the /usr/lib directory and delete **libefc.so.x.xx.0** first.  
Then remove **libefc.a**, **libefc.so** and **libefc.so.x**.  
In /usr/include, remove the file **efclib.h**
3. Untar the file by placing it in the top-level directory  
  
Go to the top-level directory  
  
**cd /**  
  
Copy the tar file from the CD to the top-level directory.  
  
UN-tar the archive as follows:  
  
**tar -xvpPf EFCAPI\_X\_Y\_Z\_LIBS.tar**
4. Create the shared library links and cache for run time linking of the newly installed libraries by typing:  
  
**ldconfig**
5. Rebuild all applications that use the EFCAPI library.

---

## Installing Fedora Core 3, Kernel 2.6 Source

---

These instructions are for installing the kernel source for Fedora 3, kernel 2.6.X, and should only be performed by personnel with experience in Linux system administration and configuration. The kernel source must be installed before building the GRT drivers.

### 1. Determine your kernel version

Perform the following command to determine your current kernel:

```
uname -r
```

This will produce output something like this:

```
2.6.9-1.667smp
```

This output shows the current running kernel, 2.6.9-1.667smp in this example. Note that in this example the "smp" at the end of the kernel version indicates that this is the SMP version of the kernel. Do not continue with this procedure until you determine your kernel version, and you have determined that you are running Fedora Core 3 and your kernel is 2.6 or greater. If you have Fedora Core 1 or 2, or kernel 2.4.X or lower, you should have the kernel source package in your Linux distribution. In this case, follow the instructions include in your Linux distribution for installing the package.

### 2. Download the kernel source RPM file

Download the kernel source RPM file for your kernel version. You can get this from one of the common RPM repositories or the following locations:

```
http://download.fedora.redhat.com/pub/fedora/linux/core/3/SRPMS/
```

```
http://prdownloads.sourceforge.net/linux-ntf
```

In the example above, the kernel source file is kernel-2.6.9-1.667.src.rpm. Note that in this example the "smp" specifier is not in the file name. Download the kernel source file to any available location such as /tmp. Make sure you get the correct kernel source for your kernel.

### 3. Install the source RPM

Install the source RPM using the following command:

```
rpm -ivh <kernel file name>
```

where <kernel file name> is the name of the kernel source file (in this case the file name is kernel-2.6.9-1.667.src.rpm). After performing this step, the RPM source files (many) will be in /usr/src/redhat/SOURCES and there will be a spec file in /usr/src/redhat/SPECS. Do not proceed to the next step unless you get a clean RPM install with no errors.

### 4. Build the kernel source package

Change to the /usr/src/redhat/SPECS directory and execute the following command:

```
rpmbuild -bp --target=<arch> kernel-2.6.spec
```

Where <arch> is the architecture (such as i686). Note that rpmbuild can be used to build both binary and source software packages. In this case we are using the "-bp" option that executes the "%prep" stage from the spec file. This unpacks the sources and applies any patches. Do not proceed to the next step unless you get a clean build with no errors.

5. Copy the kernel configuration file

Change to the `/usr/src/redhat/BUILD` directory, then change to the directory for your kernel version (i.e. `./kernel-2.6.9/linux-2.6.9`). In the "configs" directory, locate the config file for your configuration.

This will be something like:

```
./configs/kernel-2.6.9-i686-smp.config
```

Copy this file to:

```
./config
```

The command for this is:

```
cp <config file> ./config
```

Where `<config file>` is your config file. In this example the file is

```
./configs/kernel-2.6.9-i686-smp.config
```

So the command is:

```
cp ./configs/kernel-2.6.9-i686-smp.config ./config
```

If asked if you want to overwrite the file, answer "yes".

6. Check the kernel configuration file

Perform the command:

```
make oldconfig
```

This command reads in your existing configuration information and then prompts you for a value for any kernel configuration variables that were not provided in the existing kernel configuration file. You may be asked questions about new kernel options. Since our intention here is to install the source for the current kernel and NOT modify the kernel, answer "no" to all the questions and make notes of the new options. The "make oldconfig" command is essentially checking for consistency between your current kernel and the source you just installed. If this command reports no changes you are ready to continue on to installing GRT API for Linux. If there are any significant new changes, you may not have gotten the correct RPM file for your kernel and may need to get the correct file and repeat this procedure.

END OF DOCUMENT