Building GDAL 1.4.2 for Linux on Fedora Core 5

V. 1.0

Daniele Romagnoli

Dott. Ing. Simone Giannecchini

Ing. Alessio Fabiani

1. Introduction

This simple guide will provide you some instructions about how to build GDAL 1.4.2 with support for the following formats:

- Kakadu (v. 5.2.6)
- MrSID (v 6.0.7)

2. Preliminary steps

In order to add support for the listed formats, you need to achieve some preliminary steps for each format.

2.1 - KAKADU build

Go in the main kakadu folder.

Enter in `coresys/make` and modify the `Makefile-Linux-x86-gcc` file as follow:

Enable the static build by setting `KDU_GLIBS = -static -static-libgcc`

Then run make: `make -f Makefile-Linux-x86-gcc`

This will generate libs in `kakadu/lib/Linux-x86-gcc`.

From the kakadu folder, run

```
cp lib/Linux-x86-gcc/* /usr/local/lib
```

After this, enter in `apps/make` and modify the `Makefile-Linux-x86-gcc` file as follow:

Enable the static build by setting `KDU_GLIBS = -static -static-libgcc`

Set LIB_SRC as follow: `LIB_SRC=$(LIB_DIR)/libkdu.a`

Then, run make: `make -f Makefile-Linux-x86-gcc`

Finally, run `ldconfig`

2.2 - MrSID

As a first requirement, you need the LizardTech Decoding Software Development Kit (DSDK).

You can download it free of charge from this site:

http://developer.lizardtech.com (You need to be registered in order to download it).

After logged in, select “Download” -> “Software Development Kits” -> “Download SDK's”.

Select the proper version of SDK to be download (select the **GeoExpress SDK for Linux (x86) - gcc 3.4**)

3. Configuring GDAL

Firstly, you need to download GDAL 1.4.2 from OSGeo SVN at this location:
https://svn.osgeo.org/gdal/tags/1.4.2/gdal

From the location where you want to download GDAL, run:

```
svn co http://svn.osgeo.org/gdal/tags/1.4.2/gdal
```

Then, you need to apply the patch available at this location:

https://imageio-ext.dev.java.net/svn/imageio-ext/trunk/patches/1.4.2GDAL.patch

This patch contains several changes for:

- Kakadu support: Multithreading added; More Kakadu options supported. Makefile modified
- Java bindings: improved data access (read Dataset at once instead of read RasterBands at once; PixelSpace, LineSpace and BandSpace parameters now are allowed)

To apply the patch, enter in the GDAL main folder and run:

```
patch -p0 <PATH_TO_DOWNLOADED_1.4.2GDAL.patch
```

3.1 – FORMATS Settings

To customize your GDAL building settings, you need to launch the `./configure` command from your GDAL home. Depending on the required formats, you need to add some options to this command.

3.1.1 - KAKADU SETTINGS

Add `--with-kakadu=KAKADU_FOLDER --without-libtool` option to the `./configure` command.

Where `KAKADU_FOLDER` is the location where your KAKADU library has been placed.

Checks the `GDAL/frmts/jp2kakadu/GNUmakefile` file is properly set.

(Probably, you should add `$(KAKDIR)/apps/make/kdu_stripe_decompressor.o` to the `APPOBJ` setting (at the line stating “`# The following are just for Kakadu 5.1 or later. APPOBJ+=`))

3.1.2 - MRSID SETTINGS

Add `--with-mrsid=MRSID_FOLDER` option to the `./configure` command.

Where `MRSID_FOLDER` is the location where you previously downloaded GeoDSDK.

NOTE: If during build process (Chapter 4) a similar error occurs:

/......./include/base/**lti_sceneBuffer.h**:356:
error: extra qualification 'LizardTech::LTISceneBuffer::' on member

Fix the issue in the **header** (`MRSID_FOLDER/include/base/`**`lti_sceneBuffer.h`**), simply remove the extra qualification from the `inWindow` declaration. Line 356 should look like this:

```
bool inWindow(lt_uint32 x, lt_uint32 y) const;
```

Then repeat build process as suggested in chapter 4.

4. Building GDAL

Now, you are ready to build GDAL. Supposing you properly configured as explained in section 3.1, run:

```
make clean

make

make install
```

When the build is terminated, copy the generated libs in `/usr/lib` and run `ldconfig`.

Then you need to generate JAVA bindings.

5. Generating Java Bindings

5.1 - Requirements

Be sure you have properly downloaded SWIG, the Simplified Wrapper and Interface Generator which allow to produce JAVA bindings for GDAL. You can obtain it by simply running:

```
yum update swig
```

You also need ANT which will be used in order to build a JAR containing all JAVA classes generated by SWIG. You can download it from: http://ant.apache.org/

When you downloaded it (as an instance, on `/usr/local/apache-ant-1.7.0`), you may create a symbolic link as follow:

```
ln -s /usr/local/apache-ant-1.7.0/bin/ant /usr/bin/ant
```

5.2 - Running SWIG

Now, you are ready to generate java bindings. From the Command Line, enter in your `GDAL\SWIG\JAVA` and run

```
make clean

make generate

make build
```

This command will automatically generate wrappers and bindings. Then, copy the generated libs in `/usr/lib` and run `ldconfig`.