

# A close-up on SDK 3.0, Part 1: Rebuilding code from src.rpm

Step-by-step instructions to Install and rebuild SDK code from a src.rpm package

Skill Level: Introductory

[Brian Horton \(brianh@linux.vnet.ibm.com\)](mailto:brianh@linux.vnet.ibm.com)  
Cell/B.E. Software Developer  
IBM

26 Aug 2008

The Cell/B.E. SDK 3.0 includes several src.rpm packages that contain the source code for some of the SDK libraries. This article describes the steps needed to install the src.rpm, unpack the source into a directory where it can be viewed and changed, and rebuild a new rpm.

## Introduction

Several of the libraries in the Cell Broadband Engine™ (Cell/B.E.) SDK 3.0 include their source code in a src.rpm file. This means that you can modify the source and rebuild the code to create new rpm packages with the modified code. This feature is useful to change functionality in the library code. You can also select different build-time options that can affect the size or performance of the library code.

This article addresses:

1. Setting up the directories
2. Installing and unpacking the src.rpm
3. Selecting options for the rpmbuild command

4. Working with the code
5. Rebuilding an installable rpm

## Setting up the directories

Before you can install and use the src.rpm, you need to create the needed directories. By default, the path for root for src.rpm files is /usr/src/redhat. There are two options:

- Do everything as root.
- Do everything as a non-root user (this option is the safer one).

The default path is only available to the root user. But both root and non-root can set up and use a different location by defining a path in an ~/.rpmmacros file. For example:

```
TOPDIR=/home/brianh/rpm
echo %_topdir $TOPDIR > ~/.rpmmacros
```

Note that you must use a full path here because ~ or \$HOME does not work in the .rpmmacros file. The TOPDIR environment variable is not required by rpmbuild, but it is used in this article's example.

Next, create the directories that rpmbuild needs. For example:

```
mkdir $TOPDIR
cd $TOPDIR
mkdir BUILD RPMS SOURCE SPECS SRPMS
mkdir RPMS/{ppc,ppc64,noarch,i386,i686,x86_64}
```

## Installing and unpacking the src.rpm

After you create the directories, you can install the src.rpm file and unpacks the source code. The src.rpm files that are shipped with the Cell/B.E. SDK 3.0 are on the CellSDK-Devel iso image in the src directory. Invoke rpm to install the src.rpm file, which causes the source and spec file to be copied into the appropriate directories under \$TOPDIR. For this example, use simdmath-3.0-5.src.rpm file that comes with the SDK 3.0.

### Listing 1. Installing the src.rpm

```
$ rpm -iv /mnt/src/simdmath-3.0-5.src.rpm
simdmath-3.0-5
$ find . -type f
./SPECS/simdmath.spec
./SOURCES/simdmath.tgz
```

## Selecting options for the rpmbuild command

Use the rpmbuild tool to rebuild the rpm file. Listing 2 shows the rpmbuild options that are relevant for this example:

### Listing 2. rpmbuild options

```
--target=<arch>
    arch is one of ppc, ppc64, noarch, i686, x86_64
-bp
    unpack the tar.gz file in the SOURCE directory into the
BUILD directory
-bc
    same as -bp AND build (make) the code
-bb
    same as -bc AND build the binary rpm package.
-ba
    same as -bb AND build the new src.rpm file as well.
```

## Working with the code

After you install the src.rpm, the spec file and the tar.gz file are available. rpmbuild uses the spec file for instructions on how to build the code.

Set rpmbuild to untar the tar.gz file:

```
cd $TOPDIR/SPECS
rpmbuild -bp simdmath.spec
```

This instruction untars the tar.gz file into the BUILD directory. If the src.rpm has any patch files, rpmbuild also applies those patches to the source code.

The source code is now available to be modified and rebuilt. The code is located in the BUILD directory. Any changes made and rebuilt there are available only in that directory. You need to copy the libraries for applications to use them.

## Rebuilding an installable rpm

Care must be taken at this point if you want to keep this set of source code. The

rpmbuild command uses this BUILD directory when it rebuilds the rpm, including untarring the SOURCE tar.gz file. This overwrites any changes you made.

Before the rpmbuild command is run again, recreate the SOURCE tar.gz file from the files in the BUILD directory. One option to keep these changes across rpmbuild calls is to rename the directory under BUILD. That directory name is not part of the tar.gz file, but it is specified in the spec file.

After you make the changes, the following instructions rebuild the installable rpm. Begin by recreating the tar.gz file:

```
cd $TOPDIR/BUILD/simdpath-3.0-5
tar -zcf ../../SOURCE/simdmath.tgz *
```

Then use the rpmbuild command to rebuild an installable rpm package file:

```
cd $TOPDIR/SPECS
rpmbuild -bb --target=ppc64 simdmath.spec
rpmbuild -bb --target=ppc simdmath.spec
```

Note that some of the .src.rpm packages in the SDK 3.0 build might have a bug that causes an error if this is run on a Cell/B.E. system. You might need to change the following:

```
< %define __strip /opt/cell/toolchain/bin/ppu-strip
< %define __objdump /opt/cell/toolchain/bin/ppu-objdump
---
> %define __strip ppu-strip
> %define __objdump ppu-objdump
```

When these steps are complete, the rpm files are in the ppc and ppc64 directories under the \$TOPDIR/RPMS directory. The rpm files can then be installed on the target computer.

## Conclusion

Sometimes the simplest part of using development tools (like setting them up properly) can be the most confusing, especially for those relatively new to development. Future article in the series address such topics as building code with makefiles that use the SDK's make.footer and rebuilding Cell/B.E. SDK examples in a local sandbox.

# Resources

## Learn

- Use an [RSS feed](#) to request notification for the upcoming articles in this series. (Find out more about [RSS feeds of developerWorks content](#).)
- Check out the Fedora Project for an excellent [RPM Guide](#) that covers everything about the subject, from why you need package management to dependencies to automation to popt to language support.
- Find every Cell/B.E. document imaginable—guides, handbooks, tutorials, specifications—on the [Cell Resource Center's Docs page](#).
- Need roundups of fast info on how to use major Cell/B.E.-related technologies?
  - See the [Programming with ALF series](#) (developerWorks, October 2007-July 2008) for a roundup of short guides and longer articles to help you understand and use the Accelerated Library Framework with the SDK.
  - See the [Programming with DaCS series](#) (developerWorks, October 2007-August 2008) for a roundup of short guides and longer articles to help you understand and use Data Communication and Synchronization services with the SDK.
  - See the [Programming with BLAS series](#) (developerWorks, November 2007-July 2008) for a roundup of short guides and longer articles to help you understand and use the Basic Linear Algebra Subroutines library with the SDK.
- Learn more about Cell/B.E. programming from the developerWorks series:
  - ["Programming high-performance applications on the Cell/B.E. processor"](#)
  - ["PS3 fab-to-lab"](#)
  - ["The little broadband engine that could"](#)
- Refer to the [Cell Broadband Engine documentation](#) section of the IBM Semiconductor Solutions Technical Library for a wealth of downloadable manuals, specifications, and more.
- Sign up for the [developerWorks newsletter](#) and get the latest developer news and Cell/B.E. happenings delivered to your inbox each week. Check *Power Architecture*® when you sign up to receive Cell/B.E. news in your newsletter.

## Get products and technologies

- Get your copy of the [IBM SDK for Multicore Acceleration 3.0](#) or browse through the [library of Cell/B.E. documentation](#).

- Find all Cell/B.E.-related articles, discussion forums, downloads, and more at the IBM developerWorks [Cell Broadband Engine resource center](#): your definitive resource for all things Cell/B.E.
- Contact IBM about [custom Cell/B.E.-based or custom-processor based solutions](#).

## Discuss

- Check out the [Cell Broadband Engine Architecture forum](#) to get your technical questions about the processor answered. Juicy problems and answers from the forums are rounded up periodically and highlighted in the "[Forum watch](#)" [blog series](#).
- Go to the [Cell Broadband Engine/Power Architecture blog](#) for [news](#), downloads, instructional resources, and event notifications for Cell/B.E. and other Power Architecture-related technologies. You can find the popular "[Forum watch](#)" blog series (Q&A roundup), the "FixIt" technology updates, and the [Infobomb](#) quick-read technology introductions.

## About the author

Brian Horton

Brian Horton has been with the Cell/B.E. SDK development team since early 2007, working on issues with the samples and build environment, as well as with the DaCS library.

## Trademarks

IBM, developerWorks, and Power architecture are trademarks of IBM Corporation in the United States, other countries, or both.

Cell Broadband Engine is a trademark of Sony Computer Entertainment Inc.

Other company, product, or service names may be trademarks or service marks of others.