IBM Software Development Kit for Multicore Acceleration
Version 3.1

Installation Guide

SC33-8323-04

IBM Software Development Kit for Multicore Acceleration
Version 3.1

Installation Guide

SC33-8323-04

Note
Before using this information and the product it supports, read the information in "Notices" on page 43.

Edition notice

This edition applies to version 3, release 1, modification 0 of the IBM Software Development Kit for Multicore Acceleration (product number 5724-S85) and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces SC33-8323-03.

Contents

About this publication

This book is the Installation Guide for the IBM® Software Development Kit (SDK) for Multicore Acceleration version 3.1.

The SDK is a complete package of tools that help you create applications for hardware platforms built on Cell Broadband Engine Architecture such as the IBM BladeCenter QS21 or the IBM BladeCenter QS22. The SDK is composed of runtime tools such as the Linux® kernel, development tools, software libraries and frameworks, performance tools, and example source files, all of which fully support the capabilities of the Cell Broadband Engine Architecture.

Packages containing code derived from GPL or LGPL open source software such as GCC are located on the BSC Web site: http://www.bsc.es/projects/deepcomputing/linuxoncell/.

A single integrated installation based on the open source tool *YUM* installs both IBM and BSC open source components. This book describes the details of installing the SDK for supported platforms.

Chapter 1. Introduction

New in this release

SDK 3.1 contains a number of significant enhancements over previous versions of the SDK and completely replaces those SDK versions.

These enhancements include:

- Support for the IBM BladeCenter QS22, which includes two enhanced double precision PowerXCell8i chips
- Support for Red Hat Enterprise Linux (RHEL) 5.2 including version 2.6.18 of the Cell/B.E.-enabled kernel
- New support for Fedora 9 including version 2.6.25 of the Cell/B.E.-enabled kernel
- Updated GCC toolchain including GCC compiler version 4.1.1, Binutils version 2.18.50, Newlib version 1.16.0 and GDB version 6.8.50. This toolchain provides enhanced performance and usability features such as:
  - Improved cross and remote debugging by removing the requirement that target libraries must be available on the host system
  - New GNU Profiler (gprof) support for SPU code
  - Improved overlay manager runtime performance
  - Automatic migration for large data area to effective address space accessible through the SW-managed cache
  - Newlib includes an integrated SPU timer library
  - GCC SPU compiler (spu-gcc) utilization of new auto-vectorization enhancements, including auto-vectorized doubly-nested looks and basic blocks
  - GFortran support for VMX intrinsics
- The SDK provides prototype GCC 4.3.2–based compilers as a technology preview
- The prototype OpenMP single-source XL C/C++ compiler has been withdrawn from the SDK and is planned to be graduated into an IBM product
- The Eclipse IDE is ported to the PTP (parallel tools platform) and supports the latest versions of Eclipse 3.3 and CDT 4.0. IDE usability is improved, including better code completion features for common code segments.
- The PDT and PDTR performance tools feature improved usability and performance such as the ability to add new groups of traced events without PDT code changes, per group summary reports, and integration into the Eclipse IDE
- Added Fortran bindings, PDT trace hooks, and example code to demonstrate the use of DaCS interfaces in both C and Fortran. DaCS performance on Cell/B.E. is improved and the SPU library size is reduced. DaCS for Hybrid (X86_64) support is upgraded from prototype to beta level and adds the ability to mix local and remote files when starting an accelerator process and improves the usability of the topology configuration file.
- ALF introduces a new light weight task model. The task model is a thin layer over LIBSPE2, provided for library developers who want greater control over SPE and shared ALF handles, for coexistence of multiple libraries in one application. ALF gets internal performance improvements such as optimization

for reuse of SPE contexts that reduce the performance gap between hand-tuned code and ALF library code. This release reintroduces the work block barrier provided as a prototype in SDK 2.1.

- Example code packages provide improved DMA examples, a new debugging DMA example, and a SW race checker to help you debug hard to discover DMA race conditions
- More CBEA optimizations for single and double precision real number functions in the BLAS library
- New CBEA-optimized LAPACK library for single and double precision real number functions such as matrix factorizations, singular value decompositions, and eigen values
- The Monte Carlo Random Number Generator library is now supported at product level and includes a PPE interface for all random number generator and transform functions. The Mersenne Twister and Sobol generators now support multiple random number streams for better performance. The Sobol generator now supports the generation of very large streams of random numbers. The Box-Mueller Sine and Inverse Cumulative Distribution Function (iCDF) transforms are added.
- The one- and two-dimensional FFT library is now supported at product level with improved performance, accuracy and function. Vectors larger than ten thousand (10,000) elements and single precision rectangular matrices are now supported.
- New prototype level library for single and double precision three-dimensional FFTs (LIBFFT3)
- The SIMD Math library includes performance and accuracy improvements
- The MASS/MASSV library is upgraded to include optimized double precision functions
- New prototype cryptography library
- The Security SDK package is now packaged with the SDK
- The IBM Full-System Simulator is withdrawn from the SDK. You can download it from the alphaWorks Web site: http://www.alphaworks.ibm.com/tech/cellsystemsim.
- The Sysroot image for the Full-System Simulator is upgraded to Fedora 9
- The cellsdk installation script no longer supports the –gui option. You can use an RPM-based GUI installer such as *gnome-packagekit* or *Yumex* to manually install the SDK.

Related products

You can use these related products together with the SDK components to provide additional capability.

Here is a list of related products and where to get them:

- XL C/C++ for Multicore Acceleration for Linux compilers available from IBM at http://www.ibm.com/software/awdtools/xlcpp/
- XL Fortran for Multicore Acceleration for Linux available from IBM at http://www.ibm.com/software/awdtools/fortran/
- Visual Performance Analyzer (VPA) available from alphaWorks® at http://www.ibm.com/software/awdtools/fortran/
- IBM Full-System Simulator for the Cell Broadband Engine™ Processor is available from alphaWorks at http://www.alphaworks.ibm.com/tech/cellsystemsim

Licenses

The source code and binaries that are part of the total SDK package are distributed with different licenses.

The packages on the BSC Web site are generally open source and are licensed under either:

- The General Public License (GPL)

  `http://www.gnu.org/copyleft/gpl.html`
- Lesser General Public License (LGPL)

  `http://www.gnu.org/licenses/licenses.html#LGPL`

If you are not familiar with these licenses, visit the Free Software Foundation (FSF) for more information.

The ISO images or physical media available from IBM have several licenses depending upon the package:

- The *RHEL5-Product* ISO or physical media has an International Program License Agreement (IPLA) number **L-PJSA-7F9L7T**.
- The *RHEL5-Devel* ISO and *Fedora-Devel* ISO have an International License Agreement for Non-Warranted Programs (ILAN) number **L-PJSA-7F9LPN**.
- The *RHEL5-Extras* ISO has an International License Agreement for Early Release of Programs (ILAER) number **L-PJSA-7F9LQK**.
- The *Fedora-Extras* ISO has an International License Agreement for Early Release of Programs (ILAER) number **L-PJSA-7F9LQ5**.

See http://www.ibm.com/software/sla/sladb.nsf for more information about IBM licenses.

Chapter 2. Supported operating environments

Hardware requirements

The SDK has specific hardware requirements. The following table shows the recommended minimum configuration for each hardware platform.

Table 1. Hardware prerequisites

System	Recommended minimum configuration
X86 or X86_64 architecture	2 GHz Intel® Pentium® 4 processor, or AMD Opteron ″F″ processor that supports the `RDTSCP` instruction.
PowerPC®	1.42 GHz 64-bit PPC. **32-bit PPC platforms are not supported.**
BladeCenter QS21	IBM BladeCenter® H chassis. Hardware firmware level QB-01.08.0-00
BladeCenter QS22	IBM BladeCenter H chassis (8852)

All systems must have:

- Hard disk space: 5 GB (minimum) to install the source package and the accompanying development tools
- 1 GB RAM (minimum) on the host system

Note: If you use the optional Full-System Simulator (available from alphaWorks) in addition to the SDK , the minimum amount of RAM installed must be twice the amount of simulated memory. For example, to simulate a system with 512 MB of RAM, the host system must have at least 1 GB of RAM installed.

Software requirements

The SDK requires Red Hat Enterprise Linux (RHEL) 5.2 or Fedora 9, which must be installed before you install the SDK.

Fedora 9 support

The support model for Fedora 9 is an ″as-is″ support model, free of charge, provided through an IBM developerWorks® Web forum online support. You can use the IBM developerWorks discussion forums to ask questions, share knowledge, ideas, and opinions about technologies and programming techniques with other developerWorks users. Use the forum content at your own risk. While IBM will attempt to provide a timely response to all postings, the use of this developerWorks forum does not guarantee a response to every question that is posted, nor do we validate the answers or the code that are offered.

Chapter 3. SDK components

This topic describes the components of the SDK and how they are packaged. Use this information to understand what gets installed and how to configure the installation for your own specific purposes.

SDK target platforms

The SDK can be installed on different target platforms. The native development platforms are 64-bit PowerPC and CBEA-compliant machines. Additionally, the development (build) platforms for cross compilation of Cell Broadband Engine (CBE) code can be X86 or X86_64 machines. There are two execution platforms, either CBEA-compliant hardware or the optional IBM Full-System Simulator. Note that you can also run the Full-System Simulator on Cell Broadband Engine Architecture (CBEA) hardware to assist with debugging. Executables built on any development platform should run on any execution platform using the same operating system.

The following table summarizes the development and execution platforms available for Red Hat Enterprise Linux (RHEL) 5.2:

Table 2. RHEL 5.2 platforms

Development platform	**CBEA execution platform (BladeCenter QS21 or BladeCenter QS22)**
X86	
X86_64	
PPC64	
Cell Broadband Engine Architecture hardware	✔

The following table summarizes the development and execution platforms available for Fedora 9:

Table 3. Fedora 9 platforms

Development platform	**CBEA execution platform (BladeCenter QS21 or BladeCenter QS22)**	**Full-System Simulator execution platform**
X86		✔
X86_64		✔
PPC64		✔
Cell Broadband Engine Architecture hardware	✔	✔

SDK directories

The SDK installs files into a number of different directories depending on the host platform and file type. This section describes the SDK standards for directories to help you understand where to find the parts of the SDK and how to best use the SDK development environment.

The root directory for the SDK is **/opt/cell**. Most of the SDK files are in this directory. There is one exception:

- **/usr** follows the Linux file system hierarchy standard (FHS), except for prototype-level code which is placed in the **/opt/cell/sdk/prototype/usr** directory. In some cases, subdirectories are used to store individual components under **include**, **lib** or **lib64**.

There are three main directories under **/opt/cell**:

- **sdk** contains the SDK files.
- **sysroot** contains a *fakeroot* used for cross compilation on X86 and X86_64 architecture systems. There are directories under the **/opt/cell/sysroot** directory that mirror either a native host system (such as **/usr**) or mirror the SDK **/opt/cell/sdk** directory.
- **toolchain** contains the GNU toolchain.

There are various subdirectories for parts of the SDK under the **/opt/cell/sdk** directory:

- **docs** contains the SDK documentation.
- **prototype** contains level 1 components. This is a separate directory to clearly distinguish those parts of the SDK that might change in a future release. Subdirectories of **/opt/cell/sdk/prototype** are similar to peer directories, for example there are **doc**, **src**, and **usr** directories below this directory.
- **src** contains source code such as examples.
- **usr** contains host-based tools.

RPMs

The SDK is distributed as a set of Red Hat Package Manager (RPM) files that can be installed on the target platform. The RPMs that can be installed depend on the host Linux operating system, the target hardware platform, and the options chosen by the user when installing the SDK. The SDK also depends on a number of RPMs provided by the base Linux operating system.

The SDK RPMs follow typical RPM naming conventions including version and revision, and standard name suffixes such as *devel* for development code and *debuginfo* for GDB debugging data. The SDK includes additional conventions that make is easier to identify what the RPM is used for. The following table details these conventions:

Table 4. RPM naming conventions

Convention	Explanation
-source suffix	The RPM contains source code, typically used for examples built using the SDK rather than rpmbuild which uses a source RPM (SRPM or src.rpm).
-debuginfo suffix	The RPM contains debugging information, to be used by debugging tools.
-devel suffix	The RPM contains files needed for development using this library, such as header files and static libraries.
-cross-devel suffix	The RPM contains development code for a cross-build environment (X86 or X86_64) rather than a native one.
-trace suffix	The RPM contains code that has been enabled for the IBM Performance and Debugging Tool (PDT).

Table 4. RPM naming conventions (continued)

Convention	Explanation
-hybrid suffix	The RPM contains code that is used in a hybrid runtime environment where the host is an X86_64 platform and the accelerator is a CBEA platform.
cell- prefix	The RPM is oriented for CBEA platforms and can be used to differentiate the RPM from a standard implementation.
ppu- prefix	The RPM contains PPU-only code.
spu- preifx	The RPM contains SPU-only code.

The SDK RPMs also use a number of different RPM targets. They are listed in the following table:

Table 5. SDK target platforms

Architecture/Platform	Explanation
PPC, PPC64	A CBEA application can be either 32 bit or 64 bit. Regular PowerPC platforms are treated as native for CBEA code only for development. Execution of this code still needs either a CBEA-compliant hardware platform or the Full-System Simulator.
i386, i686 (X86)	This is native code that executes on a 32 bit X86 platform.
X86_64	This native code only executes on a 64 bit X86 platform and is used for the hybrid programming model.
noarch	*noarch* is generally used to indicate an architecture-neutral RPM. For the SDK, *noarch* has the additional meaning that the RPM contains PPC or PPC64 target code that is to be installed on an X86 or X86_64 system for cross compilation. The noarch target is used so that the file installs without complaints from RPM or YUM.
src	The source code for some SDK components are available as SRPMs (src.rpm).

SDK component descriptions

The SDK is divided into components, each of which is at a particular level of development. Some components are prototype code and others have been fully tested and are warranted by IBM with the appropriate purchased license.

The following table details the component development levels:

Table 6. Component development levels

Development level	Description
1	Prototype-level code. There is no guarantee that the features and API will not change in a future release. IBM is particularly interested in customer feedback about this component.
2	Beta-level code that is stable.
3	Product-level code that is stable. However the function, which is typically example code, is provided on an *as-is* basis and might not be maintained or upgraded by IBM.
4	Product-level code that is stable and has been fully tested. This code is warranted on certain platforms and is fully supported by IBM through standard support channels.

Table 6. Component development levels (continued)

Development level	Description
5	GPL and LGPL open source code that is not directly supported by IBM but has been tested with the SDK.

The following table provides the list of SDK components with license, development level, and functional descriptions:

Table 7. SDK component list

Component	Level	License	Description
ALF for Cell/B.E.	4	IBM	The Accelerator Library and Framework (ALF) provides for ease of use in multi-core computing by simplifying the data distribution and work queue management for multiple tasks. The host is the PPU and the SPUs are the accelerators. The source and examples are available under a BSD license.
ALF for Hybrid	1	IBM	This version of ALF is directed toward a hybrid computing environment with an X86_64 host and CBEA hardware accelerators.
BLAS	4	IBM	The Basic Linear Algebra Subprograms for single and double precision linear algebra functions. The examples are available under a BSD license.
Cell Performance Counter	2	IBM	The cell-perf-counter (cpc) tool is used for setting up and using the hardware performance counters in the Cell/B.E. processor. You can use these counters to see how many times certain hardware events occur when analyzing the performance of software running on a Cell/B.E. system.
Crash SPU Commands	5	GPL	Crash extension with specific commands for analyzing the Cell Broadband Engine SPU run control state. This component is only available for the Fedora platform.
DaCS for Cell/B.E.	4	IBM	The Data Communication and Synchronization (DaCS) library contains functions for process management, data movement, data and process synchronization, topology features (such as the group concept), and error handling. DaCS is used only on CBEA hardware. The source and examples are available under a BSD license.
DaCS for Hybrid	2	IBM	The Data Communication and Synchronization (DaCS) library contains functions for process management, data movement, data and process synchronization, topology features (such as the group concept), and error handling. DaCS for Hybrid is used between an X86_64 host and CBEA hardware.
Documentation	4	IBM	Documentation consists of man pages, PDFs, and README files in individual directories. The PDFs for the SDK are installed into directories under the /opt/cell/sdk/docs directory.
Examples	3	IBM	This component contains example code including example libraries, demos, and a tutorial. The source code is available under a BSD license.

Table 7. SDK component list (continued)

Component	Level	License	Description
FDPR-Pro	4	IBM	The Feedback Directed Program Restructuring post-link program optimization tool allows you to instrument a program, run the instrumented version to collect its profile, and create a semantically-equivalent optimized version using that profile.
GNU Toolchain	5	GPL	The GNU Toolchain packages provide a full development tool chain (GCC compiler, assembler, linker, debugger, binary utilities, and runtime library) to generate and debug code for the Cell/B.E. PPE and SPE processor cores. The toolchain is provided both as native version running on Cell Broadband Engine Architecture and other PowerPC Linux systems, and as a cross toolchain hosted on X86 or X86_64 Linux systems. See the *SDK 3.1 Programmer's Guide* for more information about how to use the GNU Toolchain.
Hybrid Performance Tools	1	IBM	These tools are designed to help you use a number of the performance tools in a hybrid system with more than one processor architecture. In particular, the Cell Broadband Engine is used as an accelerator for a host system with a different architecture.
IDE	4	IBM	Eclipse-based integrated development environment for the SDK.
Kernel	5	GPL	The operating system kernel with Cell Broadband Engine Architecture support. The kernel is included in RHEL 5.2. For Fedora 9, the kernel is part of the SDK.
LAPACK	4	IBM	The Linear Algebra Package for Cell/B.E..
LIBFFT	4	IBM	This library provides a wide range of 1D and 2D Fast Fourier Transforms.
LIBFFT3D	1	IBM	This library provides 3D Fast Fourier Transforms.
LIBSPE1/LIBSPE2	5	LGPL	A low level library that defines the user space API to program for Cell Broadband Engine Architecture applications. LIBSPE2 is supplied with RHEL 5.2. For Fedora 9, LIBSPE2 and backwards compatibility support of LIBSPE1 is provided.
Monte Carlo library (libmc-rand)	4	IBM	A random number generator library suitable for simulation.
MASS Library	4	IBM	The Mathematical Acceleration Subsystem (MASS) consists of libraries of mathematical intrinsic functions, which are tuned specifically for optimum performance on the Cell/B.E. processor. Currently 32-bit, 64-bit PPU, and SPU libraries are supported.
netpbm	5	GPL	This graphics bitmap library is used by the Julia example. A cross development version is provided in the SDK for use on X86 and X86_64 platforms.
OProfile	5	GPL	OProfile is a tool for profiling user and kernel level code. It uses the hardware performance counters to sample the program counter every *N* events. OProfile is included with the RHEL 5.2 distribution. For users of Fedora 9, The SDK provides OProfile.

Table 7. SDK component list (continued)

Component	Level	License	Description
PDT	4	IBM	The Performance Debugging Tool (PDT) provides the ability to trace events of interest during application execution, and record data related to these events from the SPEs, the PPE, and the AMD Opteron processor.
PDTR	4	IBM	PDTR is a command line tool that reads and post-processes PDT traces.
SIMDMath	4	IBM	A math library that takes advantage of the Single Instruction, Multiple Data (SIMD) instructions in CBEA-compliant hardware.
SPU-Isolation	1	IBM	SPU-Isolation provides a build and runtime environment for signing and encrypting SPE applications. The SDK provides this component for the Fedora 9 platform. It has not been tested on RHEL 5.2.
SPU-Libcrypto	1	IBM	SPU-Libcrypto provides basic cryptographic routines optimized for execution on an SPE. These functions include AES, DES, Triple-DES, and RSA encryption and decryption routines; and MD5, SHA-1, and SHA-256 cryptographic hashing routines. The SDK provides this component for the Fedora 9 platform. It has not been tested on RHEL 5.2.
SPU-Timing Tool	2	IBM	The SPU static timing tool *spu_timing* annotates an SPU assembly file with scheduling, timing, and instruction issue estimates assuming a straight, linear execution of the program.
Sysroot Image	5	GPL/ LGPL	The system root image for the Full-System Simulator is a file that contains a disk image of Fedora 9 files, libraries, and binaries that can be used within the simulator. This component is provided only for the Fedora platform.

Notations:

1. For RHEL 5.2, the kernel and LIBSPE components are supplied by the operating system (distribution) and not the SDK.
2. The following components are only provided for Fedora 9:
   - Crash SPU commands
   - OProfile for SPU profiling (PPU-only profiling is provided for RHEL 5.2)
   - SPU-Isolation
   - SPU-Libcrypto
   - GCC 4.3 (shipped as part of the toolchain as prototype code
   - spu-tools package
3. The IBM Full-System Simulator is not provided with the SDK, and is available from alphaWorks.

YUM groups

YUM provides the ability to group RPMs together to facilitate installing a number of RPMs simultaneously. The SDK provides several YUM groups to ease finding related groups of components

The following groups are defined in the YUM metadata files located in /opt/cell/yum-repos:

- Cell Runtime Environment
- Cell Development Libraries
- Cell Development Tools
- Cell Performance Tools
- Cell Programming Examples

The *Cell Runtime Environment* group contains the RPMs that are only needed for runtime execution of Cell Broadband Engine Architecture applications. It does not contain any development libraries, tools or example code. When you use the `--runtime option` of the cellsdk script, only the *Cell Runtime Environment* group is included in the install, update or uninstall.

You can use the following YUM group commands to discover which RPMs are in a group and which groups are already installed:

- `grouplist`
- `groupinfo group1 [...]`
- `groupinstall group1 [group2] [...]`
- `groupupdate group1 [group2] [...]`
- `groupremove group1 [group2] [...]`

To display a list of RPMs and their YUM group, use the cellsdk *verify* task option. Because the output from this option is large, direct the output to a file by typing the following command:

```
/opt/cell/cellsdk verify > /tmp/cellsdk.verify
```

The output file will show, for each RPM, whether it is mandatory, default or optional; the YUM group name; and whether the RPM is installed on your system.

Note: The YUM group names are displayed without spaces. For example, *CellDevelopmentLibraries* is the YUM group "Cell Development Libraries".

To display a list of all the optional SDK RPMs, type the following command:

```
grep optional /tmp/cellsdk.verify
```

To display a list of all uninstalled SDK RPMs, type the following command:

```
grep "not installed" /tmp/cellsdk.verify
```

After installing the SDK you can install, update or uninstall components or individual RPMs in the SDK. For example, you can install the *alf-hybrid-devel* RPM by typing the following command as the user root:

```
# /opt/cell/cellsdk –iso <iso dir> mount
# yum install alf-hybrid-devel
```

YUM uses its repository information to ensure that you can only install the correct RPM on each platform. Some RPMs are only available with a target platform of X86_64 because they are needed for building X86_64 code using a host-based compiler such as GCC. The SDK contains several hybrid of these programming model libraries and performance tools.

Chapter 4. Installing, migrating, and removing the SDK

This topic describes how to install, migrate, or uninstall the SDK.

Packages removed in this release

This topic describes packages removed in this release of the SDK.

The packages listed in the following table are removed in SDK 3.1. To upgrade from SDK version 3.0 to version 3.1, first remove them with YUM or RPM. For example, type

```
yum remove spu-timer-devel
```

to remove the *spu-timer-devel* package.

Table 8. Removed packages

Package	Explanation
cell-spu-isolation-tool	This package relies on the openssl package, which varies by version between Fedora 9 and RHEL 5.2. To obtain a working isolation tool, follow the instructions in "Optional: Building the SPU-Isolation component" on page 34
cell-xlc-ssc-cmp, cell-xlc-ssc-help, cell-xlc-ssc-lib, cell-xlc-ssc-man, cell-xlc-ssc-omp, cell-xlc-ssc-rte, and cell-xlc-ssc-rte-lnk	The prototype OpenMP single-source XL C/C++ compiler has been withdrawn from the SDK and is being graduated into an IBM product.
spu-timer-devel, spu-timer-cross-devel, and spu-timer-devel-debuginfo	The spu-newlib package provides SPU timer functions.
systemsim-cell	The SDK does not provide the IBM Full-System Simulator. It is available from alphaWorks.

Compatability RPMs

The SDK provides the following compatibility RPMs:

- alf-compat-<version>.ppc64.rpm
- alf-compat-<version>.ppc.rpm
- dacs-compat-<version>.ppc64.rpm
- ppu-mass-compat-<version>.ppc64.rpm

These RPMs provide SDK library compatibility for software developed using previous SDK versions. To install them, follow the instructions in "Installing Required Development Libraries" on page 24.

Numactl

The numactl package is removed from SDK 3.1. If you have an older version of the SDK installed, uninstall that SDK so that the SDK-provided version of numactl is removed from your system. See "Uninstalling the SDK" on page 27.

Obtain the numactl package appropriate for your architecture from the distribution media or the distribution FTP site. Install the numactl RPMs before installing SDK 3.1. See "Installing Required Development Libraries" on page 24.

Default SDK installation

This topic describes the steps to perform a default installation of the SDK.

Before you begin

These steps assume you have already installed a supported Linux operating system and have satisfied the prerequisites listed in Chapter 2, "Supported operating environments," on page 5.

About this task

Follow these steps to install the SDK:

1. "Choosing a product set"
2. "Downloading the SDK files" on page 19
3. "Preparing for installation of the SDK" on page 20
4. "Installing the SDK Installer" on page 22
5. "Starting the SDK installation" on page 22
6. "Post installation configuration" on page 23

What to do next

There are several optional steps that you can follow after installing the SDK:

- "Optional: Setting up a YUM server for the SDK" on page 31
- "Optional: Installing additional SDK components" on page 32
- "Optional: Installing the ALF component" on page 32
- "Optional: Installing the IES Eclipse SDK for the Cell/B.E. IDE" on page 33
- "Optional: Building the SPU-Isolation component" on page 34
- "Optional: Building the SPU-Libcrypto examples" on page 34

Choosing a product set

A product set is a formal grouping of RPMs that compose the SDK. It is further defined as a YUM repository for a specific environment and operating system. Some product sets are packaged as ISO images to distribute the SDK. The YUM repository for each product set is installed and then enabled or disabled as part of installing the *cell-install* RPM.

Product sets are categorized as follows:

Table 9. Product set group descriptors

Descriptor	Options	Rationale
Distributor	IBM or BSC	All GPL or LGPL code is distributed by the Barcelona Supercomputing Center (BSC) or its mirrors and is in separate product sets from the IBM-owned code that is distributed using ISO images from either developerWorks or Passport Advantage®.

Table 9. Product set group descriptors (continued)

Descriptor	Options	Rationale
Operating system	RHEL 5.2 or Fedora 9	As noted in Chapter 3, "SDK components," on page 7, not all of the components are distributed for RHEL 5.2. The SDK requires different product sets for each supported operating system.
License	IPLA, ILAN or ILAER.	Different licenses apply to different components.

Product sets with *Open* in the name are not downloadable as an ISO image but are accessed directly by YUM from a repository on the BSC Web site. The product sets without *Open* in the name are distributed as ISO images that you can download from the developerWorks or Passport Advantage Web sites. For example, the ISO for the Devel-Fedora product set is named *CellSDK-Devel-Fedora.iso*.

The following table lists the components in each Fedora 9 product set:

Table 10. Fedora 9 product set component details

Component	Devel-Fedora	Extras-Fedora	Open-Fedora
ALF for Cell/B.E.	✓		
ALF for Hybrid		✓	
Basic Linear Algebra Subprograms (BLAS)	✓		
Cell Performance Counter		✓	
Crash SPU commands			✓
DaCS for Cell/B.E.	✓		
DaCS for Hybrid		✓	
Debugging tools			✓
Documentation	✓		
Examples	✓		
FDPR-Pro	✓		
GNU toolchain			✓
Hybrid performance tools		✓	
IDE	✓		
Kernel			✓
LAPACK	✓		
LIBFFT	✓		
LIBFFT3		✓	
LIBSPE/LIBSPE2			✓
MASS library	✓		
Monte Carlo library	✓		
netpbm			✓
OProfile			✓
PDT	✓		
PDTR	✓		

Table 10. Fedora 9 product set component details (continued)

Component	Devel-Fedora	Extras-Fedora	Open-Fedora
SIMDMath	✓		
SPU-Isolation		✓	
SPU-Libcrypto		✓	
SPU-Timing tool		✓	
Sysroot image			✓

The following table lists the components in each RHEL 5.2 product set:

Table 11. RHEL 5.2 product set component details

Component	Product-RHEL, Devel-RHEL	Extras-RHEL	Open-RHEL
ALF for Cell/B.E.	✓		
ALF for Hybrid		✓	
Basic Linear Algebra Subprograms (BLAS)	✓		
Cell Performance Counter		N/A	
Crash SPU commands			N/A
DaCS for Cell/B.E.	✓		
DaCS for Hybrid		✓	
Debugging tools			✓
Documentation	✓		
Examples	✓		
FDPR-Pro	✓		
GNU toolchain			✓
Hybrid performance tools		✓	
IDE	✓		
Kernel			Included in RHEL 5.2
LAPACK	✓		
LIBFFT	✓		
LIBFFT3		✓	
LIBSPE/LIBSPE2			Included in RHEL 5.2
MASS library	✓		
Monte Carlo library	✓		
netpbm			Included in RHEL 5.2
OProfile			Included in RHEL 5.2 (for PPU only)
PDT	✓		
PDTR	✓		
SIMDMath	✓		

Table 11. RHEL 5.2 product set component details (continued)

Component	Product-RHEL, Devel-RHEL	Extras-RHEL	Open-RHEL
SPU-Isolation		N/A	
SPU-Libcrypto		N/A	
SPU-Timing tool		✔	
Sysroot image			N/A

Downloading the SDK files

This topic describes how to download the SDK files needed for installation. You can skip this step if you have physical media for the SDK such as a CD.

The developerWorks Web site and the Passport Advantage Web site provide the IBM-licensed code and its documentation as ISO images. Passport Advantage is an IBM Web site that gives you information about software subscription and support, product upgrades and technical support under a single, common set of agreements, processes and tools.

To download the SDK perform the following steps:

1. Create a temporary directory for the images and the cell-install RPM by typing the following commands:

   ```
   # mkdir -p /tmp/cellsdkiso
   # cd /tmp/cellsdkiso
   ```

2. Download the cell-install RPM from developerWorks or Passport Advantage Web site and place it into the /tmp/cellsdkiso directory that you created in the previous step.
3. Download the ISO images into the same directory.

Here are the choices for ISO images for each supported Linux distribution:

Table 12. ISO images for Red Hat Enterprise Linux (RHEL) 5.2

The *Product* package contains all the mature technologies in SDK 3.1 plus access to IBM Support and is intended for production purposes.	*CellSDK-Product-RHEL_3.1.0.0.0.iso*	http://www.ibm.com/software/howtobuy/passportadvantage/
The *Developer* package is intended for evaluation of the SDK in a non-production environment and contains all the mature technologies in SDK 3.1.	*CellSDK-Devel-RHEL_3.1.0.0.0.iso*	http://www.ibm.com/developerworks/power/cell/downloads.html
The *Extras* package contains the "latest and greatest" technologies in SDK 3.1. These packages tend to be less mature or are technology preview code that might or might not become part of the generally available product in the future.	*CellSDK-Extra-RHEL_3.1.0.0.0.iso*	http://www.ibm.com/developerworks/power/cell/downloads.html

Table 13. ISO images for Fedora 9

Product set	ISO name	Location
The *Developer* package is intended for evaluation of the SDK in a non-production environment and contains all the mature technologies in SDK 3.1.	*CellSDK-Devel-Fedora_3.1.0.0.0.iso*	http://www.ibm.com/developerworks/power/cell/downloads.html
The *Extras* package contains the ″latest and greatest″ technologies in SDK 3.1. These packages tend to be less mature or are technology preview code that might or might not become part of the generally available product in the future.	*CellSDK-Extra-Fedora_3.1.0.0.0.iso*	http://www.ibm.com/developerworks/power/cell/downloads.html

You can verify the integrity of the files using the md5sum command. Checksums are provided on the download Web page.

Preparing for installation of the SDK

Prepare your system for installation by following these steps:

1. Verify that your BladeCenter QS21 or BladeCenter QS22 has the right firmware level. See “Hardware requirements” on page 5.
2. If necessary, install or upgrade to a supported operating system. See Chapter 2, “Supported operating environments,” on page 5, then follow the installation documentation provided by your Linux distribution.
3. The YUM updater daemon must not be running when installing the SDK. To see if it is running, type the following command:

   ```
   # /etc/init.d/yum-updatesd status
   ```

   If the command returns a result similar to:

   ```
   # /etc/init.d/yum-updatesd status
   yum-updatesd (pid 12260) is running...
   ```

   then type the command:

   ```
   # /etc/init.d/yum-updatesd stop
   ```

   You will see a result similar to:

   ```
   # /etc/init.d/yum-updatesd stop
   Stopping yum-updatesd:                                 [  OK  ]
   ```

   Later in the installation process you will restart the daemon.
4. If you previously added an exclude clause in the /etc/yum.conf file that includes the numactl, numactl-devel, blas, blas-debuginfo, blas-devel, oprofile or oprofile-debuginfo packages, temporarily remove the clauses to ensure that these packages are installed for the SDK.

5. If you plan to install the FDPR-Pro component, it requires the compat-libstdc++ RPM. For RHEL 5.2 only, this RPM should be installed first, otherwise the installation of the FDPR-Pro RPM will fail.
6. The SDK requires the packages *rsync*, *sed*, *tcl*, and *wget*. For PPC64 systems, the 64-bit *glibc-devel* package is also required.

   If you are installing on a PPC64 or BladeCenter system, type the following command as the root user to install these dependencies:

   ```
   # yum install glibc-devel.ppc64 rsync sed tcl wget
   ```

   On non-PPC64 systems, type the following command:

   ```
   # yum install rsync sed tcl wget
   ```
7. The DaCS for Hybrid daemon on the X86_64 platform requires the *expat* XML parsing library. If you want to use the DaCS for Hybrid package on X86_64, install expat by typing the following command as the root user:

   ```
   # yum install expat
   ```
8. If you have installed an older version of the SDK, you have two choices to install the current version. If you want to try an unsupported upgrade without uninstalling your current version, see "Optional: Migrating to SDK 3.1 from a previous release." Or, follow the recommend procedure by first removing the older version. See "Uninstalling the SDK" on page 27.

Optional: Migrating to SDK 3.1 from a previous release

How to upgrade the SDK from a previous version to version 3.1.

Upgrading the SDK using YUM is tested and should work. However, upgrading the SDK is not supported. If you choose to attempt to upgrade the SDK using YUM, you must be prepared to resolve any issues on your own. The recommended upgrade path is to completely remove any previous version of the SDK and install the new version. To remove the SDK, see "Uninstalling the SDK" on page 27.

To attempt to upgrade the SDK without removing the old version, follow this procedure while logged in as the user root:

- Unmount any SDK ISO images by typing the following command:

  ```
  # /opt/cell/cellsdk unmount
  ```
- Install the SDK 3.1 cell-install RPM by typing the following command:

  ```
  # rpm –Uvh cell-install-3.1.0-0.0.noarch.rpm
  ```
- To upgrade, type the following:

  ```
  # /opt/cell/cellsdk –iso <directory containing SDK 3.1 ISO images> install
  ```

If you experience YUM errors, type the following command, then try the install again:

```
# yum clean metadata
```

Optional: Upgrading from RHEL5 Developer to RHEL5 Product

You can upgrade the SDK from the *RHEL5 Developer* to the *RHEL5 Product* version simply by downloading the ISO image for *RHEL5 Product* from Passport Advantage and accepting the license.

To upgrade, type the following command:

```
# /opt/cell/cellsdk --iso /tmp/cellsdkiso install
```

and accept the IPLA license when it is displayed. The rest of the installation process is similar to a normal SDK installation. Because the content of the two ISOs is exactly the same, this second install will not install any additional RPMs.

Installing the SDK Installer

How to install the SDK Installer package.

Verify that your installation satisfies the necessary prerequisites. See "Hardware requirements" on page 5 and "Software requirements" on page 5.

To install the SDK, first install the SDK Installer which is provided by the cell-install RPM package.If you have physical media such as a CD, you can find this RPM in the root directory.

To install this RPM, type the following command:

```
# rpm -ivh cell-install-3.1.0.0.0.noarch.rpm
```

Note: If a message appears similar to:

```
warning: cell-install: Header V3 DSA signature: NOKEY, key ID 9ac02885
```

then the cellsdk RPM GPG key is not installed. The cellsdk script will install it automatically the first time you run the script.

Starting the SDK installation

This topic describes how to install the SDK. The cellsdk script is a *wrapper* around YUM. Install the SDK by following these steps:

1. Use the cellsdk script to install the SDK.
   - If you are installing from an ISO image, type:
     ```
     # /opt/cell/cellsdk --iso /tmp/cellsdkiso install
     ```
   - If you are installing from a local server (see "Optional: Setting up a YUM server for the SDK" on page 31) type:
     ```
     # /opt/cell/cellsdk install
     ```
   - If you are installing from a physical CD, load the CD and mount it on the `/media` directory. Now type:
     ```
     # /opt/cell/cellsdk install
     ```
2. Read the SDK licenses.

   There are several licenses that you must agree to. First are the GPL and LGPL licenses. Answer '**yes**' to the license question if you agree to the license terms. The second is either the International Program License Agreement (IPLA) or International License Agreement for Non-Warranted Programs (ILAN). Follow the on-screen menu to agree to the license. This IBM license is installed into the /opt/cell/license directory for later reference. If you downloaded the *Extras* ISO into the ISO directory, then you will also be presented with the International License Agreement for Early Release of Programs (ILAER). Again, follow the on-screen menu to agree to the license.
3. After you agree to both licenses, YUM will install the RPM files. Answer '**y**' to the package install question from YUM.

Post installation configuration

After you install the SDK packages, finish the installation and configure your system to use the SDK.

Preventing automatic updates from overwriting SDK components

If you are installing on an IBM BladeCenter QS21 or BladeCenter QS22, you must add an exclude clause to the /etc/yum.conf file in the [Main] section to prevent a YUM update from overwriting the SDK versions of these runtime RPMs.

For Red Hat Enterprise Linux (RHEL) 5.2 systems, add the following statement:

```
exclude=blas blas-devel lapack lapack-devel oprofile oprofile-devel
```

For Fedora 9 systems, add the following clause to the /etc/yum.conf file. The exclude statement must be all on one line; no line breaks.

```
exclude=blas blas-devel elfspe2 kernel lapack lapack-devel libspe2
libspe2-debuginfo libspe2-devel oprofile oprofile-devel
```

The installation script replaces distribution versions of the blas, lapack and elfspe2 (Fedora only), libspe2 and oprofile RPMs with the RPMs located in the /tmp/cellsdk/openSrc directory. In the future, the YUM update daemon might attempt to update SDK packages with a version not enhanced for the SDK. The exclude line will prevent this from occurring.

Note: If you exclude a package from regular updates, YUM will not automatically update it when a newer version becomes available. If a new version containing security updates or bug fixes is released, you must manually update the RPM.

Installing the libspe2 Ada bindings

On Fedora 9 PowerPC and IBM BladeCenter systems, the optional *libspe2-adabinding-devel* RPM must be installed with a specific command to ensure that the SDK version is installed. To install it, type the following command as the user root:

```
# yum --disablerepo=* --enablerepo=CellSDK-Open-Fedora-ppc64 install \
  libspe2-adabinding-devel
```

Next, add *libspe2-adabinding-devel* to the exclude line in the YUM configuration file, /etc/yum.conf. This prevents the YUM update process from overwriting the SDK version.

Installing the Linux Kernel (Fedora 9 only)

If you are installing Fedora 9 on BladeCenter hardware, the kernel must be manually installed. First, download the kernel from the Barcelona Supercomputing Center Web site. The kernel RPM URL is http://www.bsc.es/projects/deepcomputing/linuxoncell/cellsimulator/sdk3.1/kernel-2.6.25.14-108.20080910bsc.ppc64.rpm.

Your Fedora 9 system might have the *iwl4965-firmware* package installed. This firmware is unnecessary on a BladeCenter QS21 or BladeCenter QS22 server and must be removed before installing the BSC kernel. To remove the firmware package, type the following command as the user root:

```
# rpm -e iwl4965-firmware
```

Next, install the BSC kernel:

```
# rpm -ivh --force kernel-2.6.25.14-108.20080910bsc.ppc64.rpm
```

Finally, reboot to activate this kernel.

Installing Required Development Libraries

Developing applications for the Cell Broadband Engine Architecture requires additional development libraries. The following is a list of required and optional RPMs:

- numactl-<version>.ppc
- numactl-<version>.ppc64
- numactl-devel-<version>.ppc
- numactl-devel-<version>.ppc64
- netpbm-devel-<version>.ppc to run the Julia set examples
- Optional compatibility libraries to run code compiled against previous versions of the SDK:
  - alf-compat-<version>.ppc
  - alf-compat-<version>.ppc64
  - dacs-compat-<version>.ppc64
  - ppu-mass-compat-<version>.ppc64

To install the RPMs on Cell Broadband Engine Architecture or PPC64 systems, type the following command:

```
# yum install <RPM name>
```

To install the RPMs on X86 or X86_64 systems for cross compilation (installed in the /opt/cell/sysroot directory), type the following command:

```
# rpm -Uvh --force --nodeps --noscript --ignorearch \
  --root /opt/cell/sysroot <RPM name>
```

Installing the elfspe utility (RHEL 5.2 only)

For CBEA-compliant hardware such as the IBM BladeCenter QS21 or BladeCenter QS22, install the elfspe2 RPM from the RHEL 5.2 supplementary CD. For example, type the following command:

```
# rpm -ivh elfspe2-<version>.rpm
```

After you install the elfspe2 RPM, ensure that spufs (the SPU File System) is loaded correctly:

- Create the /spu directory by typing the following command:

  ```
  # mkdir -p /spu
  ```

- Add the following line to /etc/fstab if it does not already exist:

  ```
  spufs  /spu  spufs defaults  0  0
  ```

  spufs will now mount automatically at boot. To mount spufs immediately, type the following command:

  ```
  # mount /spu
  ```

Note: The version of SELinux included with RHEL 5.2 might prevent spufs from mounting at boot. If spufs did not mount at boot, type the following command:

```
# mount /spu
```

Disable SELinux if you want spufs automatically mounted at boot.

Installing the libspe2 package (RHEL 5.2 only)

For RHEL 5.2, you should install additional libraries. This step is not necessary for Fedora 9 because it is done automatically by the installation program.

The versions of the libspe2-devel packages for PPC and PPC64 are needed for application development. These RPMs can be found on the RHEL 5.2 supplementary CD.

To install on X86 or X86_64 architecture systems, change to the directory where you have the supplementary CD mounted and type the following commands:

```
# for r in `ls libspe2-2* libspe2-devel-2*`; do
   rpm -ivh --force --nodeps --noscripts --ignorearch \
   --root/opt/cell/sysroot $r
# done
```

To install on PowerPC architecture or CBEA-compliant systems, type the following commands:

```
# rpm -ivh libspe2-devel-<version>.ppc.rpm libspe2-devel-<version>.ppc64.rpm
```

Restarting automatic updates

Start the YUM updates daemon by typing the following command as root:

```
# /etc/init.d/yum-updatesd start
```

Installing the IBM Full-System Simulator (RHEL 5.2 only)

For RHEL 5.2 cross compilation only (X86 and X86_64), you should build and install the compiler *sysroot* RPMs as described in the information below. These RPMs cannot be provided by SDK and can only be built if you have a RHEL 5.2 license. For Fedora 9, these RPMs are supplied by the SDK and YUM installs them automatically. Follow these steps to build and install the sysroot on an X86 or X86_64 architecture RHEL 5.2 system:

1. Download the *ppu-sysroot.spec* file from the Barcelona Supercomputing Center Web site: http://www.bsc.es/projects/deepcomputing/linuxoncell/cellsimulator/sdk3.1/sources/toolchain/rhel5.2-ppu-sysroot/ppu-sysroot.spec. Place this file in the /usr/src/redhat/SPECS directory.
2. Copy the following RHEL 5.2 PowerPC binary packages (from your PPC architecture RHEL 5.2 distribution medium, such as a CD) into the /usr/src/redhat/SOURCES directory:

   ```
   glibc-<version>.ppc.rpm
   glibc-<version>.ppc64.rpm
   glibc-devel-<version>.ppc.rpm
   glibc-devel-<version>.ppc64.rpm
   glibc-headers-<version>.ppc.rpm
   kernel-headers-<version>.ppc.rpm
   gmp-<version>.ppc.rpm
   gmp-<version>.ppc64.rpm
   gmp-devel-<version>.ppc.rpm
   gmp-devel-<version>.ppc64.rpm
   ```
3. Edit the ppu-sysroot.spec file if necessary to match the version numbers of the RHEL 5.2 distribution RPMs. Verify or edit the version numbers in the following 3 lines:

```
%define glibc_version 2.5-24
%define kernheaders_version 2.6.18-92.el5
%define gmp_version 4.1.4-10.el5
```

4. Type the following command to build the RHEL 5.2 ppu-sysroot RPMs:

   ```
   # rpmbuild -ba --target=noarch \
     /usr/src/redhat/SPECS/ppu-sysroot.spec
   ```

   This will create the following sysroot RPMs:

   ```
   /usr/src/redhat/RPMS/noarch/ppu-sysroot-<version>.noarch.rpm
   /usr/src/redhat/RPMS/noarch/ppu-sysroot64-<version>.noarch.rpm
   ```

5. Type the following commands to install them:

   ```
   # cd /usr/src/redhat/RPMS/noarch/
   # rpm -ivh ppu-sysroot-<version>.noarch.rpm \
     ppu-sysroot64-<version>.noarch.rpm
   ```

   The installation places the target library files into the /opt/cell/sysroot directory.

   Note: If you want to install the sysroot on multiple RHEL 5.2 host systems, it is not necessary to build the RPMs on each system. Just copy the compiler sysroot RPMs to each system and install them.

Updating the IBM Full-System Simulator sysroot on BladeCenter QS22

If you have installed the IBM Full-System Simulator and the sysroot image on a BladeCenter QS22 system, update the sysroot image by typing the following command:

```
# /opt/cell/cellsdk_sync_simulator install
```

See the *Software Development Kit for Multicore Acceleration Programmer's Guide* for more information.

Adding SDK components

After the SDK is installed, you can install optional packages. See "Optional: Installing additional SDK components" on page 32.

The cellsdk script

The cellsdk script is used to install, update or uninstall the SDK. This script is a wrapper around the YUM tool.

cellsdk script options

When called without options or parameters, the cellsdk script displays an option list. The following is an example:

```
'cellsdk' is a script used for installing the IBM Cell/B.E. SDK
Usage: cellsdk [OPTIONS] [--iso ISO_DIR] TASK
   ISO_DIR is the directory where cellsdk ISOs have been downloaded.
      If not specified, network or cdrom install is assumed.
   TASK is one of install, update, uninstall, verify, mount, unmount, or
      updateDetails

The main tasks are:
   install:       ./cellsdk install            (starts yum)
   update:        ./cellsdk update             (starts yum)
   uninstall:     ./cellsdk uninstall          (starts yum)
```

```
   verify:        ./cellsdk verify              (lists RPMs installed)
   mount:         ./cellsdk --iso ISO_DIR mount  (mounts cellsdk iso images)
   unmount:       ./cellsdk unmount             (unmounts cellsdk iso images)
   updateDetails: ./cellsdk updateDetails      (lists cellsdk update options)

The cellsdk script uses the YUM-based tools according to the following options:
   [ -r | --runtime ] only install runtime, not development, files
   [ -o | --all] install all optional SDK rpms
   [ -a | --auto ] auto-install (silent, no prompts)
   [ -? | -h | --help ] display this help
   [ -V | --version ] display cellsdk version
   [ -q | --quiet ] quiet
   [ -v | --verbose ] verbose
   [ -vv | --very-verbose ] very verbose
```

cellsdk script verify

Pass the verify option to the cellsdk script to list the SDK RPMs installed and available to be installed on your system. You do not have to be the root user to run this command.

Updating the SDK with the cellsdk script

The SDK can be updated to a new version using the cellsdk script update option.

The most likely reason to update the SDK is to apply an IBM Fix Pack or to upgrade to a new version of the SDK. All fixes are cumulative. Fixes to the SDK, if available, are only for RHEL 5.2 product installations and are supplied on an ISO image with the name *CellSDK-Updates-RHEL*. Note that the version numbering for both the ISO image and the cell-install RPM uses the following 5 digit numbering scheme:

```
version.release.product.fixpack.interimfix
```

For example `3.1.0.2.0` is fix pack #2 for SDK version 3.1.0.

Download the new cell-install RPM and the CellSDK-Updates-RHEL ISO image from Passport Advantage and save them to a directory on your machine. You can either manually install the new cell-install rpm, or it will be done for you as part of the update process.

Update the SDK by typing the following command as the root user:

```
# /opt/cell/cellsdk --iso /tmp/cellsdkiso update
```

In this example, the /tmp/cellsdkiso directory contains the downloaded update ISO image and the new cell-install RPM.

You must accept the SDK licenses each time you apply an update.

Installed RPMs are updated during the update process, and new mandatory or default RPMs are installed. Optional RPMs that are not installed at the time you apply the update must be later installed using the YUM tool..

After applying a Fix Pack it can be backed out. See “Uninstalling an SDK fix package” on page 29.

Uninstalling the SDK

The following topics describe how to uninstall the SDK.

Uninstalling SDK version 2.1

If you previously installed version 2.1 of the SDK from the IBM alphaWorks Web site, save any files you need from the /opt/ibm/cell-sdk directory.

About this task

Uninstall the SDK by typing the following commands as the user root:

1. `# /opt/ibm/cell-sdk/prototype/cellsdk uninstall`
2. `# rm -rf /opt/ibm/systemsim-cell`
3. `# rm -rf /opt/ibm/cell-sdk`
4. `# rm -rf /opt/cell`
5. `# rm -rf /opt/ibmcmp`
6. `# umount /mnt/cellsdk`
7. `# rmdir /mnt/cellsdk`

Uninstalling SDK version 3.0

If you installed version 3.0 of the SDK, first save any files you need from the /opt/cell directory and the /opt/ibm/systemsim directory.

About this task

Uninstall the SDK by following these steps:

1. Uninstall the SDK using the uninstall option of the cellsdk script.
   a. Type the following commands:
      ```
      # /opt/cell/cellsdk uninstall
      ```
      Answer **'y'** when asked by YUM to uninstall the packages.
      After YUM has uninstalled all of the SDK RPMs, there are a series of questions about how much cleanup you want to do for other directories used by the SDK. To perform a full uninstall, answer **'y'** to all questions.
2. Clean up the YUM cache.
   a. Type the following commands:
      ```
      # yum clean all
      # rm -rf /var/cache/yum/CellSDK*
      ```
3. Clean up the YUM configuration.
   a. Remove the SDK exclude clause added to the /etc/yum.conf file. See "Preventing automatic updates from overwriting SDK components" on page 23.
4. Uninstall the Cell/B.E. IDE.
   a. Start Eclipse
   b. Click **Help -> Software Updates -> Manage Configuration**
   c. Click **Cell IDE feature**
   d. Click the right mouse button
   e. From the popup menu, click **disable**
   f. Click **uninstall**

Uninstalling SDK version 3.1

How to uninstall SDK version 3.1.

Before you begin

First save any files you need from the /opt/cell directory and the /opt/ibm/systemsim directory.

About this task

Uninstall the SDK by following these steps:

1. Uninstall the SDK using the uninstall option of the cellsdk script.
   a. Type the following commands:
      ```
      # /opt/cell/cellsdk uninstall
      ```
      Answer **'y'** when asked by YUM to uninstall the packages.
      After YUM has uninstalled all of the SDK RPMs, there are a series of questions about how much cleanup you want to do for other directories used by the SDK. To perform a full uninstall, answer **'y'** to all questions.
2. Clean up the YUM cache.
   a. Type the following commands:
      ```
      # yum clean all
      # rm -rf /var/cache/yum/CellSDK*
      ```
3. Clean up the YUM configuration.
   a. Remove the SDK exclude clause added to the /etc/yum.conf file. See "Preventing automatic updates from overwriting SDK components" on page 23.

Uninstalling an SDK fix package

This topic describes how to back out a fix pack that was applied using the cellsdk update option.

For more information about updating the SDK, see "Updating the SDK with the cellsdk script" on page 27.

Type:

```
# /opt/cell/cellsdk updateDetail
```

to see a list of RPMs and which previous RPM should replace each one. Follow the instructions released with the fix pack for directions on how to remove it.

Note: You must mount the original installation ISO so that the cellsdk script will discover the replacement RPMs

Troubleshooting: SDK installation

This topic describes what to do if things go wrong when installing the SDK.

YUM continues to install the SDK packages even if some of the RPMs were not completely downloaded from the BSC Web site. The failure messages from YUM do not clearly state this failure. To verify if all files were installed correctly, type the command:

```
# /opt/cell/cellsdk verify
```

In the list output by this command, verify that all default RPMs were installed. If they were not, retype the `./cellsdk install` command. YUM will attempt to

download any required RPMs that were not downloaded during a past attempt to install the SDK. YUM resumes the download process from the previous failure point.

Sometimes YUM operates incorrectly. It writes files in /var/cache/yum, and sometimes these no longer reflect the correct state of the command. If this happens, type the command:

```
# yum clean metadata
```

This will remove the incorrect status files.

There are other options to YUM that are useful to use if things go wrong. If the previous command did not restore correct operation of YUM, try typing the command:

```
# yum clean all
```

This will remove additional state files, and might cause the YUM installation process to succeed on the next invocation.

If the preceding commands do not restore correct operation of the YUM installation process, manually remove the cached state files. To do this, type the following command:

```
# rm -rf /var/cache/yum/CellSDK*
```

Note: When completely removing the SDK from a diskless CBEA system, you might see the following error: `rm: cannot remove `/opt/cell/.nfs00000000006002e700000358': Device or resource busy`. This message does not mean that removal has failed. In environments where the cellsdk process is running from the mounted /opt/cell directory, the NFS process renames the directory that is being deleted from its original name (to the form `nfs<string>`) instead of removing it. The temporary file should go away then the NFS process completes or when the machine is rebooted.

To see more information about YUM, set the options:

```
debuglevel=10
errorlevel=10
```

in the /etc/yum.conf file. As an alternative, add the string **-d 10 -e 10** to any YUM command.

You can specify the flags **-v** for verbose output, or **-vv** for very verbose output when typing the cellsdk script. YUM writes a log to /var/log/yum.log. The cellsdk install script writes a log to /var/log/cellsdk.log. Looking at these files might provide helpful information about what went wrong.

If you do not use an *excludes* line in the yum.conf file, the distribution version of packages provided by the SDK might be installed instead of the SDK version if YUM sees that a non-SDK package has a higher version number. If you later want to reinstall the SDK version of the RPMs, run the `cellsdk install` command again. See “Preventing automatic updates from overwriting SDK components” on page 23 for more information.

Chapter 5. Optional installation and configuration steps

Optional: Setting up a YUM server for the SDK

This optional topic is for advanced users who want to set up a local YUM server. A YUM server allows multiple users to access SDK files without having to download them from the Barcelona Supercomputing Center Web site, its mirrors, or use the ISO images. A YUM server is useful if your network has a firewall that prevents direct access to the Internet.

About this task

Follow these steps to set up a local YUM server.

1. Install an HTTP server and optionally enable FTP access to a directory for downloading the RPMs.
2. Create a directory for the SDK files on the server. For example,

   ```
   [root@myserver]# mkdir /var/www/html/sdk31
   [root@myserver]# cd /var/www/html/sdk31
   ```

   Create the sdk31 directory below the directory (in this example `/var/www/html/`) that your web server uses to serve files. In the following instructions, it is assumed that the directory created by the previous step is sdk31. Substitute the actual directory name created by the preceding command in subsequent examples.
3. Copy all the files from the source material from the ISO images to the sdk31 directory.
4. Or, download the files from the BSC Web site directly to the sdk31.
   a. You can download the files from the BSC Web site using the wget tool:

      ```
      # cd /var/www/html/sdk31
      ```

   b. For RHEL 5.2, type:

      ```
      # export myDistro="RHEL"
      ```

   c. For Fedora 9, type:

      ```
      # export myDistro="Fedora"
      ```

   d. If you only want the files for a specific architecture, you can further qualify the bsc directory to get only the CBEA, PPC64, X86_64 or X86 architecture packages.
   e. Type the following command with no line breaks: Downloading might take a long time.

      ```
      # wget --mirror --no-host-directories --no-parent --cut-dirs=4 --tries=3
       http://www.bsc.es/projects/deepcomputing/linuxoncell/cellsimulator/
      sdk3.1/CellSDK-Open-${myDistro}
      ```
5. Create updated SDK YUM *repo* files which you have edited to point to the internal server by setting the `baseurl` paths. For example, the /etc/yum.repos.d/cellsdk-Fedora.repo file might contain the following (remove any line breaks in the `baseurl` line)

   ```
   [CellSDK-Devel-Fedora-x86]
   baseurl=http://myserver.com/sdk31
      file:///opt/cell/yum-repos/CellSDK-Devel-Fedora/x86
   ```

```
[CellSDK-Open-Fedora-x86]
baseurl=http://myserver.com/sdk31
   file:///opt/cell/yum-repos/CellSDK-Open-Fedora/x86
```

Note: Different protocols can be used to retrieve the files from the server including FTP, HTTP or a local file directory on your own system.

6. Decide how to distribute these new repo files to your users. A simple option is to instruct them to install the cell-install RPM and then overwrite the repo files in the /etc/yum.repos.d directory with the new versions.

What to do next

You can keep a local copy of the RPMs on your system and use the localinstall or localupdate YUM options. The advantage of this approach is that YUM manages the dependencies and uses the configured repositories to resolve dependencies. The following is an example using the localinstall option:

```
# yum localinstall /tmp/sdk31/spu-gcc-fortran-4.1.1-166.i686.rpm
```

Optional: Installing additional SDK components

You can install additional components using the `yum install` command.

To use YUM to install additional SDK packages, the SDK ISO images must be mounted. Images are unmounted at the end of the installation process and during the reboot process. To mount or remount an image, type the following command:

```
#  /opt/cell/cellsdk --iso /tmp/cellsdkiso mount
```

The image will remain mounted until you unmount it or reboot.

To display a list of optional or uninstalled SDK RPMs, type the following command:

```
# /opt/cell/cellsdk verify > /tmp/cellsdk.verify
# grep optional /tmp/cellsdk.verify
# grep "not installed" /tmp/cellsdk.verify
```

Optional: Installing the ALF component

If you are interested in developing applications using the ALF programming model but in a hybrid host-accelerator environment, install the optional ALF for Hybrid component. This component has both runtime and development RPMs. The runtime RPMs are needed on an X86_64 machine for the host and a BladeCenter QS21 or BladeCenter QS22 for the accelerator.

To develop applications on an X86_64 system, install the development RPMs. Because ALF for Hybrid depends on ALF for Cell/B.E., YUM will install these dependencies if they are not already installed.

On an the X86_64 system type the following command:

```
# yum install alf-hybrid alf-hybrid-devel alf-hybrid-cross-devel \
   alf-hybrid-examples-source
```

On a BladeCenter QS21 or BladeCenter QS22, type the following command:

```
# yum install alf-hybrid
```

You might also want to install the ALF man pages that are provided with the *alf-manpages* RPM.

Note: ALF for Hybrid depends on DaCS for Hybrid. See "Optional: Installing additional SDK components" on page 32.

Optional: Installing the IES Eclipse SDK for the Cell/B.E. IDE

IES is an IBM package of open source Eclipse projects from eclipse.org. If you have installed the SDK and want to install the optional IES Eclipse Cell/B.E. IDE component, follow these steps:

About this task

1. The SDK provides an IBM 32-bit JRE version 1.5 SR8. To install the PPC version, type the following command as the user root:

   ```
   # yum install ibm-java2-ppc-jre
   ```

   To install the X86 version, type:

   ```
   # yum install ibm-java2-i386-jre
   ```

2. Configure your environment to use the new Java™ JRE.
   a. Determine the default version of Java being used. Type the command:

      ```
      $ java -version
      ```

      If this returns the IBM 32-bit JRE version 1.5 SR8, skip to step 3.
   b. Eclipse uses the Java virtual machine that is referenced by the *PATH* environment variable. Update the *PATH* variable to reference the installed IBM Java. For example:

      ```
      PATH=$HOME/bin:/opt/java/jdk1.5.0_08/jre/bin:$PATH
      ```

      To permanently change your *PATH* variable, add that line to your $HOME/.bash_profile file.
   c. You can skip step 2b and start Eclipse directly by passing the -vm flag to specify the IBM JVM. For example, type:

      ```
      $ eclipse -vm /opt/java/jdk1.5.0_08/jre/bin
      ```

3. Install IES Eclipse bundled with CDT 4.0, PTP Remotetools 2.0, and the latest *cellide* plug-ins into the /opt/cell/ide/eclipse directory. Type the following command as the user root:

   ```
   # yum install cellide
   ```

4. Optional: Install the ALF IDE template, Performance Debugging Tool (PDT) and Cell/B.E. SPU Timing packages.
   a. To use the ALF IDE Wizard, install the ALF IDE template package:

      ```
      # yum install alf-ide-template
      ```
   b. To enable the PDT feature in the IDE, install the PDT package:

      ```
      # yum install pdt
      ```
   c. To use the SPU Timing feature, install the Cell/B.E. SPU Timing package:

      ```
      # yum install cell-spu-timing
      ```

What to do next

For issues with the IES Eclipse IDE, see Appendix B, "Known limitations," on page 41. For more information about the IDE, see the Eclipse IDE help topic. To access the IDE help, in Eclipse click: **Help → Help Contents → IDE for Cell Broadband Engine SDK**.

Optional: Building the SPU-Isolation component

After you have installed the optional SPU-Isolation component, finish the installation by building the spu-isolated-app tool and the example code. All steps in this topic require root user authority.

To install the SPU-Isolation component, type the following commands:

```
# /opt/cell/cellsdk verify > /tmp/cellsdk.verify
# grep isolation /tmp/cellsdk.verify | awk '{print $3'} | xargs yum install
```

The following packages must be installed on your system to build the spu-isolated-app tool:

- elfutils-libelf
- elfutils-libelf-devel
- openssl
- openssl-devel
- zlib

If your build system is a PowerPC or Cell Broadband Engine Architecture system, type the following command:

```
# yum install elfutils-libelf.ppc elfutils-libelf-devel.ppc openssl.ppc
  openssl-devel.ppc zlib.ppc
```

If your build system is a cross-compilation environment for the target (execution) system, copy the following PPC architecture RPMs from your distribution media or the distribution FTP server to a temporary directory: elfutils-libelf, elfutils-libelf-devel, openssl, openssl-devel, and zlib. Next, change to the temporary directory where you placed these files and type the following command:

```
# rpm –ivh –force –ignorearch –ignoreos –nodeps –noscripts
   –root /opt/cell/sysroot elfutils-libel<version>.ppc.rpm \
   elfutils-libelf-devel-<version>.ppc.rpm \
   openssl-<version>.ppc.rpm openssl-devel-<version>.ppc.rpm \
   zlib-<version>.ppc.rpm
```

To build the spu-isolated-app tool and example code, follow these steps:

1. Type the following commands to build and install the *spu-isolated-app* tool:

   ```
   # cd /opt/cell/sdk/prototype/usr/src/spu-isolated-app/
   # make
   ```

2. Next, build the examples by typing the following commands:

   ```
   # cd /opt/cell/sdk/prototype/src/examples/isolation/
   # tar xf isolation_examples_source.tar
   # cd examples/isolation/
   # make
   ```

Optional: Building the SPU-Libcrypto examples

After you have installed the optional SPU-Libcrypto-Examples component, finish the installation by building the example code.

To install the SPU-Libcrypto component, type the following commands as the root user:

```
# /opt/cell/cellsdk verify > /tmp/cellsdk.verify
# grep crypto /tmp/cellsdk.verify | awk '{print $3'} | xargs yum install
```

The following packages must be installed on your system:

- openssl
- openssl-devel
- zlib

If your build system is a PowerPC or Cell Broadband Engine Architecture system, install these packages directly using YUM by typing the following command:

```
# yum install openssl.ppc openssl-devel.ppc zlib.ppc
```

If your build system is a cross-compilation environment for the target (execution) system, follow these steps to install the required packages into the sysroot directory:

1. At a shell prompt, change to an empty directory on the build system.
2. Download the PPC RPMs for the target system from the Fedora 9 installation media, or download them from http://download.fedora.redhat.com/pub/fedora/linux/releases/9/Fedora/ppc/os/Packages
3. Install the packages by typing the following command (all on one line)

   ```
   # rpm –ivh –force –ignorearch –ignoreos –nodeps –noscripts \
     –root /opt/cell/sysroot openssl-<version>.ppc.rpm \
     openssl-devel-<version>.ppc.rpm zlib-<version>.ppc.rpm
   ```

Build the SPU-Libcrypto example code by typing the following commands as the user root:

```
# cd /opt/cell/sdk/prototype/src/examples/crypto/
# tar xf libcrypto-example-source.tar
# cd examples/crypto/
# make
```

Chapter 6. Getting support

The SDK is supported through the CBEA architecture forum on the developerWorks Web site at http://www.ibm.com/developerworks/power/cell/.

Commercial support from IBM is available if you purchased the SDK from Passport Advantage.

The XL C/C++ compilers are supported through the XL compiler Web site. See http://www.ibm.com/software/awdtools/xlcpp/support/.

The XL Fortran compiler is supported through the XL compiler Web site. See http://www.ibm.com/software/awdtools/fortran/support/.

This version of the SDK supersedes all versions of the SDK that were available from alphaWorks.

If you have a problem with the IBM BladeCenter QS21 or BladeCenter QS22 that you think is caused by running the Barcelona Supercomputing Center kernel on Fedora 9, report a bug to the public `cbe-oss-dev@ozlabs.org` mailing list. Archives and subscription information for this list are available from https://ozlabs.org/mailman/listinfo/cbe-oss-dev/. Since Fedora 9 is not a supported IBM product, IBM provides no guaranteed reply or target dates for fixes for this configuration. Commercial support is available for Red Hat Enterprise Linux (RHEL) 5.2.

Appendix A. Accessibility features

Accessibility features help users who have a physical disability, such as restricted mobility or limited vision, to use information technology products successfully.

The following list includes the major accessibility features:

- Keyboard-only operation
- Interfaces that are commonly used by screen readers
- Keys that are tactilely discernible and do not activate just by touching them
- Industry-standard devices for ports and connectors
- The attachment of alternative input and output devices

IBM and accessibility

See the IBM Accessibility Center at http://www.ibm.com/able/ for more information about the commitment that IBM has to accessibility.

Appendix B. Known limitations

These are the known limitations and restrictions in this version of the SDK.

Applications linking to SDK shared example libraries

For applications that link with the SDK example shared libraries (such as the libraries whose source is found in /opt/cell/sdk/src/lib) these libraries are not located in a standard library search path. To locate the libraries, set the *LD_LIBRARY_PATH* environment variable or specify an *rpath* when linking the application. For example:

```
LDFLAGS += -R/opt/cellsdk/sdk/usr/lib     # for 32-bit apps
LDFLAGS += -R/opt/cellsdk/sdk/usr/lib64   # for 64-bit apps
```

Or, use the *SDKEXRPATH* variable that make.footer defines:

```
LDFLAGS += -R${SDKEXRPATH}
```

IES Eclipse IDE limitations

32 bit IES Eclipse fails to load on 64 bit Fedora 9

- This occurs because of the lack of specific 32 bit libraries included in the Fedora 9 X86_64 version. Here are the solution steps:
  1. Become the user root.
  2. Install the cellide package.
  3. Install the 32 bit libcairo package by typing `yum install libcairo.i386`.
  4. Install the 32 bit libXtst package by typing `yum install libXtst.i386`.

32 bit IES Eclipse fails on X86 RHEL 5.2

- This is caused by an interaction with the xulrunner library. To work around this problem, remove the original xulrunner package by typing `yum erase xulrunner` as the user root.

Notices

This information was developed for products and services offered in the U.S.A.

The manufacturer may not offer the products, services, or features discussed in this document in other countries. Consult the manufacturer's representative for information on the products and services currently available in your area. Any reference to the manufacturer's product, program, or service is not intended to state or imply that only that product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any intellectual property right of the manufacturer may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any product, program, or service.

The manufacturer may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the manufacturer.

For license inquiries regarding double-byte (DBCS) information, contact the Intellectual Property Department in your country or send inquiries, in writing, to the manufacturer.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: THIS INFORMATION IS PROVIDED "AS IS " WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. The manufacturer may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to Web sites not owned by the manufacturer are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this product and use of those Web sites is at your own risk.

The manufacturer may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact the manufacturer.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, IBM License Agreement for Machine Code, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning products not produced by this manufacturer was obtained from the suppliers of those products, their published announcements or other publicly available sources. This manufacturer has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to products not produced by this manufacturer. Questions on the capabilities of products not produced by this manufacturer should be addressed to the suppliers of those products.

All statements regarding the manufacturer's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

The manufacturer's prices shown are the manufacturer's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to the manufacturer, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. The manufacturer, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

CODE LICENSE AND DISCLAIMER INFORMATION:

The manufacturer grants you a nonexclusive copyright license to use all programming code examples from which you can generate similar function tailored to your own specific needs.

SUBJECT TO ANY STATUTORY WARRANTIES WHICH CANNOT BE EXCLUDED, THE MANUFACTURER, ITS PROGRAM DEVELOPERS AND SUPPLIERS, MAKE NO WARRANTIES OR CONDITIONS EITHER EXPRESS OR

IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OR CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT, REGARDING THE PROGRAM OR TECHNICAL SUPPORT, IF ANY.

UNDER NO CIRCUMSTANCES IS THE MANUFACTURER, ITS PROGRAM DEVELOPERS OR SUPPLIERS LIABLE FOR ANY OF THE FOLLOWING, EVEN IF INFORMED OF THEIR POSSIBILITY:

1. LOSS OF, OR DAMAGE TO, DATA;
2. SPECIAL, INCIDENTAL, OR INDIRECT DAMAGES, OR FOR ANY ECONOMIC CONSEQUENTIAL DAMAGES; OR
3. LOST PROFITS, BUSINESS, REVENUE, GOODWILL, OR ANTICIPATED SAVINGS.

SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF DIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, SO SOME OR ALL OF THE ABOVE LIMITATIONS OR EXCLUSIONS MAY NOT APPLY TO YOU.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

If you are viewing this information in softcopy, the photographs and color illustrations may not appear.

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

alphaWorks
BladeCenter
developerWorks
IBM
Passport Advantage
POWER™
PowerPC
PowerPC Architecture™
System p™

Cell Broadband Engine and Cell/B.E. are trademarks of Sony Computer Entertainment, Inc., in the United States, other countries, or both and is used under license therefrom

Intel, MMX, and Pentium are trademarks of Intel Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

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

Terms and conditions

Permissions for the use of these publications is granted subject to the following terms and conditions.

Personal Use: You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative works of these publications, or any portion thereof, without the express consent of the manufacturer.

Commercial Use: You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of the manufacturer.

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any data, software or other intellectual property contained therein.

The manufacturer reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by the manufacturer, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

THE MANUFACTURER MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THESE PUBLICATIONS ARE PROVIDED ″AS-IS″ AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Related documentation

This topic helps you find related information.

Document location

Links to documentation for the SDK are provided on the IBM developerWorks Web site located at:

`http://www.ibm.com/developerworks/power/cell/`

Click the **Docs** tab.

The following documents are available, organized by category:

Architecture

- *Cell Broadband Engine Architecture*
- *Cell Broadband Engine Registers*
- *SPU Instruction Set Architecture*

Standards

- *C/C++ Language Extensions for Cell Broadband Engine Architecture*
- *Cell Broadband Engine Linux Reference Implementation Application Binary Interface Specification*
- *SIMD Math Library Specification for Cell Broadband Engine Architecture*
- *SPU Application Binary Interface Specification*
- *SPU Assembly Language Specification*

Programming

- *Cell Broadband Engine Programmer's Guide*
- *Cell Broadband Engine Programming Handbook*
- *Cell Broadband Engine Programming Tutorial*

Library

- *Accelerated Library Framework for Cell Broadband Engine Programmer's Guide and API Reference*
- *Basic Linear Algebra Subprograms Programmer's Guide and API Reference*
- *Data Communication and Synchronization for Cell Broadband Engine Programmer's Guide and API Reference*
- *Example Library API Reference*
- *Fast Fourier Transform Library Programmer's Guide and API Reference*
- *LAPACK (Linear Algebra Package) Programmer's Guide and API Reference*
- *Mathematical Acceleration Subsystem (MASS)*
- *Monte Carlo Library Programmer's Guide and API Reference*
- *SDK 3.0 SIMD Math Library API Reference*
- *SPE Runtime Management Library*
- *SPE Runtime Management Library Version 1 to Version 2 Migration Guide*
- *SPU Runtime Extensions Library Programmer's Guide and API Reference*

- *Three dimensional FFT Prototype Library Programmer's Guide and API Reference*

Installation

- *SDK for Multicore Acceleration Version 3.1 Installation Guide*

Tools

- *Getting Started - XL C/C++ for Multicore Acceleration for Linux*
- *Compiler Reference - XL C/C++ for Multicore Acceleration for Linux*
- *Language Reference - XL C/C++ for Multicore Acceleration for Linux*
- *Programming Guide - XL C/C++ for Multicore Acceleration for Linux*
- *Installation Guide - XL C/C++ for Multicore Acceleration for Linux*
- *Getting Started - XL Fortran for Multicore Acceleration for Linux*
- *Compiler Reference - XL Fortran for Multicore Acceleration for Linux*
- *Language Reference - XL Fortran for Multicore Acceleration for Linux*
- *Optimization and Programming Guide - XL Fortran for Multicore Acceleration for Linux*
- *Installation Guide - XL Fortran for Multicore Acceleration for Linux*
- *Performance Analysis with the IBM Full-System Simulator*
- *IBM Full-System Simulator User's Guide*
- *IBM Visual Performance Analyzer User's Guide*

IBM PowerPC Base

- *IBM PowerPC Architecture Book*
  - *Book I: PowerPC User Instruction Set Architecture*
  - *Book II: PowerPC Virtual Environment Architecture*
  - *Book III: PowerPC Operating Environment Architecture*
- *IBM PowerPC Microprocessor Family: Vector/SIMD Multimedia Extension Technology Programming Environments Manual*

Glossary

This glossary provides definitions for terms included in the *SDK Installation Guide*.

ALF. Accelerated Library Framework. This an API that provides a set of services to help programmers solving data parallel problems on a hybrid system. ALF supports the multiple-program-multiple-data (MPMD) programming style where multiple programs can be scheduled to run on multiple accelerator elements at the same time. ALF offers programmers an interface to partition data across a set of parallel processes without requiring architecturally-dependent code.

Barcelona Supercomputing Center. Spanish National Supercomputing Center, supporting IBM BladeCenter servers and Linux on Cell/B.E.

BE. Broadband Engine.

BOOTP. Bootstrap Protocol. A UDP network protocol used by a network client to obtain its IP address automatically. Replaced in many networks by DHCP.

Broadband Engine. See *CBEA*.

CBEA. Cell Broadband Engine Architecture. A new architecture that extends the 64-bit PowerPC Architecture. The CBEA and the Cell Broadband Engine are the result of a collaboration between Sony, Toshiba, and IBM, known as STI, formally started in early 2001.

Cell BE processor. The Cell BE processor is a multi-core broadband processor based on IBM's Power Architecture.

Cell Broadband Engine processor. See *Cell BE*.

DaCS. The Data Communication and Synchronization (DaCS) library provides functions that focus on process management, data movement, data synchronization, process synchronization, and error handling for processes within a hybrid system.

DHCP. Dynamic Host Configuration Protocol. Similar to BOOTP, DHCP is a protocol for assigning IP addresses to client devices on a network.

FDPR-Pro. Feedback Directed Program Restructuring. A feedback-based post-link optimization tool.

Fedora. Fedora is an operating system built from open source and free software. Fedora is free for anyone to use, modify, or distribute. For more information about Fedora and the Fedora Project, see the following Web site: http://fedoraproject.org/.

firmware. A set of instructions contained in ROM usually used to enable peripheral devices at boot.

GNU. GNU is Not Unix. A project to develop free Unix-like operating systems such as Linux.

GPL. GNU General Public License. Guarantees freedom to share, change and distribute free software.

GUI. Graphical User Interface. User interface for interacting with a computer which employs graphical images and widgets in addition to text to represent the information and actions available to the user. Usually the actions are performed through direct manipulation of the graphical elements.

host. A general purpose processing element in a hybrid system. A host can have multiple accelerators attached to it. This is often referred to as the master node in a cluster collective.

HTTP. Hypertext Transfer Protocol. A method used to transfer or convey information on the World Wide Web.

Hybrid. A module comprised of two Cell BE cards connected via an AMD Opteron processor.

hypervisor. A control (or virtualization) layer between hardware and the operating system. It allocates resources, reserves resources, and protects resources among (for example) sets of SPEs that may be running under different operating systems. The Cell Broadband Engine has three operating modes: user, supervisor and hypervisor. The hypervisor performs a meta-supervisor role that allows multiple independent supervisors' software to run on the same hardware platform. For example, the hypervisor allows both a real-time operating system and a traditional operating system to run on a single PPE. The PPE can then operate a subset of the SPEs in the Cell Broadband Engine with the realtime operating system, while the other SPEs run under the traditional operating system.

IDE. Integrated Development Environment. Integrates the Cell/B.E. GNU tool chain, compilers, the Full-System Simulator, and other development components to provide a comprehensive, Eclipse-based development platform that simplifies Cell/B.E. development.

initrd. A command file read at boot

ISO image. Commonly a disk image which can be burnt to CD. Technically it is a disk image of and ISO 9660 file system.

kernel. The core of an operating which provides services for other parts of the operating system and

provides multitasking. In Linux or UNIX® operating system, the kernel can easily be rebuilt to incorporate enhancements which then become operating-system wide.

LGPL. Lesser General Public License. Similar to the *GPL*, but does less to protect the user's freedom.

Makefile. A descriptive file used by the `make`command in which the user specifies: (a) target program or library, (b) rules about how the target is to be built, (c) dependencies which, if updated, require that the target be rebuilt.

netboot. Command to boot a device from another on the same network. Requires a TFTP server.

NUMA. Non-uniform memory access. In a multiprocessing system such as the Cell/B.E., memory is configured so that it can be shared locally, thus giving performance benefits.

Oprofile. A tool for profiling user and kernel level code. It uses the hardware performance counters to sample the program counter every N events.

PDF. Portable document format.

PPC. See *Power PC®*.

PPC-64. 64 bit implementation of the *PowerPC Architecture*.

proxy. Allows many network devices to connect to the internet using a single IP address. Usually a single server, often acting as a firewall, connects to the internet behind which other network devices connect using the IP address of that server.

RPM. Originally an acronym for Red Hat Package Manager, and RPM file is a packaging format for one or more files used by many Linux systems when installing software programs.

SDK. Software development toolkit for Multicore Acceleration. A complete package of tools for application development.

SIMD. Single Instruction Multiple Data. Processing in which a single instruction operates on multiple data elements that make up a vector data-type. Also known as vector processing. This style of programming implements data-level parallelism.

SMP. Symmetric Multiprocessing. This is a multiprocessor computer architecture where two or more identical processors are connected to a single shared main memory.

Tcl. Tool Command Language. An interpreted script language used to develop GUIs, application prototypes, Common Gateway Interface (CGI) scripts, and other scripts. Used as the command language for the Full System Simulator.

TFTP. Trivial File Transfer Protocol. Similar to, but simpler than the Transfer Protocol (FTP) but less capable. Uses UDP as its transport mechanism.

X86. Generic name for Intel-based processors.

yaboot. Linux utility which is a boot loader for PowerPC-based hardware.

YUM. Yellow Dog Updater, Modified. A package manager for RPM-compatible Linux systems.

Index

Printed in USA

SC33-8323-04