Cell/B.E. SDK: Code sample directory

Keep track of where to find code examples for the SDK

Anita Bateman
VanDung To

July 15, 2008

In this article, you'll find tables indicating the locations of code samples that illustrate how to use the IBM SDK for Multicore Acceleration. This article will be updated with new code samples.

About the SDK for Multicore Acceleration

To enable you to take full advantage of the Cell Broadband Engine® (Cell/B.E.) architecture and the PowerXCell 8i processor, IBM has developed a software development kit designed to accelerate production-ready, multi-core programming.

The IBM Software Development Kit (SDK) for Multicore Acceleration provides the libraries, tools, and resources that businesses need to develop and tune applications for Cell/B.E. technology. You can easily:

- Port and optimize applications and algorithms quickly.
- Increase ease-of-programming and developer productivity.
- Obtain a reliable development tool kit with warranty and support.
- Plug in third-party ISV libraries to integrate and build your software ecosystem.

Cell/B.E. demos

These full demonstration programs are designed and optimized to show the capabilities of the Cell/B.E.

Table 1. Cell/B.E. demos

Example	Description	RPM Default installed directory	SDK
FFT16M	Tuned Fast Fourier Transform. Performs a 4-way SIMD single-precision complex FFT on an array of size 16,777,216 elements.	cell-demos-source /opt/cell/sdk/src/demos/FFT16M	3.0
Julia Set	Quaternion Julia Set Ray-tracing Sample.	cell-demos-source /opt/cell/sdk/src/demos/julia_set	3.0
Matrix Multiply	Parallel matrix multiplication workload for CBE.		3.0

Cell/B.E. examples

These small examples show different features of the SDK and the Cell/B.E. architecture.

Table 2. Cell/B.E. examples

Example	Description	RPM Default installed directory	SDK
cache-cp	Demonstrates a standalone *spulet* that copies one file to another. The file contents are mapped into the effective address space, and a software managed cache is used to stage the data into LS. The example demonstrates the use of software-managed cache.	cell-examples-source /opt/cell/sdk/src/examples/cache/cache-cp	3.0
cache: sort	Demonstrates examples of quicksort and heapsort that use a software managed data cache. The example demonstrates the use of software-managed cache.	cell-examples-source /opt/cell/sdk/src/examples/cache/sort	3.0
ch_prof	Demonstrates an example of a statistical profiling utility that can be used to determine channel read and write activity of an SPU program. This example also contains an example application on how to use the ch_prof utility.	cell-examples-source /opt/cell/sdk/src/examples/ch_prof	3.0
DMA: simple	Demonstrates simple DMA calls within one SPE,	cell-examples-source /opt/cell/sdk/src/examples/DMA/simple	3.0
DMA: complex	Demonstrates DMA calls for multiple SPEs using single-buffered DMA, double-buffered DMA, single-buffered DMA List, and double-buffered DMA list.	cell-examples-source /opt/cell/sdk/src/examples/DMA/complex	3.0
overlay: overview	Demonstrates an example implementation for SPU code overlay,	cell-examples-source /opt/cell/sdk/src/examples/overlay/overview	3.0
overlay: simple	Demonstrates a simple example of using SPU code overlay.	cell-examples-source /opt/cell/sdk/src/examples/overlay/simple	3.0
overlay: large matrix	Demonstrates an example of using SPU code overlay for large-matrix math.	cell-examples-source /opt/cell/sdk/src/examples/overlay/large_matrix	3.0
ppe_address_space	Demonstrates an implementation of quick-sort on a random list of floats using compiler PPE address space support. The example illustrates the use of the automatic software caching feature supported by the SPU compiler. The "__ea" qualifier is used within the source file to indicate to the SPU compiler that a memory reference is in the effective address space, rather than in local store.	cell-examples-source /opt/cell/sdk/src/examples/ppe_address_space	3.0
spe_interrupt	Demonstrates the use of an assembly first level interrupt	cell-examples-source	3.0

	handle (FLIH) that calls ABI compliant, registered, C-sourced second level interrupt handlers (SLIH) as a function of the interrupt events.	/opt/cell/sdk/src/examples/ spu_interrupt	
spe_interrupt_fast	Demonstrates the use of the fast interrupt handler.	cell-examples-source /opt/cell/sdk/src/examples/ spu_interrupt	3.0
spu_let	Demonstrates examples of standalone SPU programs called *spulets*. These are C-language programs that have been compiled to run on an SPU and can invoke C-library functions, such as printf(3), open(2), and so on. A spulet can be executed directly from the Linux® command prompt. The following entries are spulet examples.	/opt/cell/sdk/src/examples/spulet	3.0
spu_let: hello	Demonstrates a spulet version of *hello world*.	cell-examples-source /opt/cell/sdk/src/examples/spulet/ hello	3.0
spu_let: spe-sum	Demonstrates a simple spulet that computes a checksum for a file. The file contents are mmap'ed into the effective address space, and DMAs are used to stage data into the LS. The example illustrates how an SPE-based filter might be constructed.	cell-examples-source /opt/cell/sdk/src/examples/spulet/ spe-sum	3.0
spu_let: spe-audio	Demonstrates audio filtering on the SPU. Two examples are included. cpaudio copies one channel of a stereo audio file to the other channel. normalize normalizes the volume of a mono audio file. Both examples take raw, unsigned, halfword audio files as input, and they output the same format.	cell-examples-source /opt/cell/sdk/src/examples/spulet/ spe-audio	3.0
spu_let: spe-cp	Demonstrates how a simple spulet copies one file to another. The file contents are mmap'ed into the effective address space, and multi-buffered DMAs are used to stage data into and then out from LS.	cell-examples-source /opt/cell/sdk/src/examples/spulet/ spe-cp	3.0
sync	Demonstrates and tests low-level atomic and thread-type synchronization functions from the sample library libsync.	cell-examples-source /opt/cell/sdk/src/examples/sync	3.0

Cell/B.E. libraries

These example libraries show different sets of functions to the programmer. The examples are optimized for running on the PPE and SPE. Find detailed information about each of the routines in each of the libraries in the 3.0 document SDK_Example_Library_API_v3.0.pdf.

Table 3. Cell/B.E. libraries

Example	Description	RPM Default installed directory	SDK

fft	Contains a collection of fast fourier transform routines. Includes the following two routines.	cell-libs-source /opt/cell/sdk/src/lib/fft	3.0
fft_1d_r2	Performs a single precision, complex Fast Fourier Transform using Discrete Fourier Transform with radix-2 decimation in time. This function is built only for use on the SPE.	cell-libs-source /opt/cell/sdk/src/lib/fft	3.0
fft_2d	Transforms 4 rows of complex 2D data from the time domain to the frequency domain or vice versa. This function is available on both the PPE and the SPE.	cell-libs-source	3.0
gmath	Contains the following game math functions: cos8, cos8_v, cos14, cos14_v, cos18, cos18_v, pack_normal16, pack_normal16_v, pack_rgba8_v, sin8, sin8_v, sin14, sin14_v, sin18, sin18_v, spec9, spec9_v, unpack_normal16, unpack_normal16_v, unpack_rgba8, unpack_rgba8_v, pack_color8, pack_rgba8, set_spec_exponent9, specThresholds, unpack_color8.	cell-libs-source /opt/cell/sdk/src/lib/gmath	3.0
image	Contains a series of convolution and histogram functions.	cell-libs-source /opt/cell/sdk/src/lib/image	3.0
large_matrix	Contains several large matrix functions for the SPE. The matrix must fit the SPE local store.	cell-libs-source /opt/cell/sdk/src/lib/large_matrix	3.0
matrix	Contains matrix routines for the PPE and SPE.	cell-libs-source /opt/cell/sdk/src/lib/matrix	3.0
misc	Contains miscellaneous functions for PPE and SPE, including: calloc_align, clamp, clamp_0_to_1, clamp_0_to_1_v, clamp_minus1_to_1, clamp_minus1_to_1_v, clamp_v, free_align, load_vec_unaligned, malloc_align, max_float_v, max_int_v, max_vec_float3, max_vec_float4, max_vec_int3, max_vec_int4, min_float_v, min_int_v, min_vec_float3, min_vec_float4, min_vec_int3, min_vec_int4, rand_0_to_1, rand_0_to_1_v, rand_minus1_to_1, rand_minus1_to_1_v, rand_v, realloc_align, srand_v, store_vec_unaligned, copy_from_ls, copy_from_ls_aligned, copy_to_ls, copy_to_ls_aligned, ls_sync.	cell-libs-source /opt/cell/sdk/src/lib/misc	3.0
mpm	Contains several functions that perform math on multi-precision large numbers on the SPE.	cell-libs-source /opt/cell/sdk/src/lib/mpm	3.0
sync	Contains several interfaces that are patterned after POSIX threads	cell-libs-source /opt/cell/sdk/src/lib/sync	3.0

	mutex and that condition variables for the Cell.		
vector	Contains several vector math functions.	cell-libs-source /opt/cell/sdk/src/lib/vector	3.0

Cell/B.E. tutorials

These examples are contained within the SDK Programming Tutorial.

Table 4. Cell/B.E. tutorials

Example	Description	RPM Default installed directory	SDK
simple	Demonstrates a simple program that creates 8 identical SPE threads that take turns printing messages to stdout and then terminate.	cell-tutorial-source	3.0
euler	Demonstrates how a simplified scalar function can be ported and accelerated for parallel execution on Cell.	cell-tutorial-source	3.0

ALF Cell/B.E. examples

These examples show how the Accelerated Library Framework works on Cell/B.E.

Table 5. ALF Cell/B.E. examples

Example	Description	RPM Default installed directory	SDK
BlackScholes_ALF	Demonstrates how to perform a Black Scholes pricing model. It demonstrates a closed form solution for Black Scholes using ALF.	alf-examples-source /opt/cell/sdk/src/alf/ BlackScholes_ALF/	3.0
FFT16M_ALF	Demonstrates an ALF version of the FFT16M workload in the cell-examples-source RPM.	alf-examples-source /opt/cell/sdk/src/alf/FFT16M_ALF	3.0
hello_world	Demonstrates a *hello_world* program that shows a simple ALF program. It contains an SPU computational kernel that issues a printf of the hello_world program.	alf-examples-source /opt/cell/sdk/src/alf/hello_world	3.0
inout_buffer	Demonstrates the usage of overlapped I/O buffers. The first example implements C=A +B, where A, B, and C are matrices. The second example implements A=A+B, where matrix A is overwritten by the result.	alf-examples-source /opt/cell/sdk/src/alf/inout_buffer	3.0
inverse_matrix_ovl	Demonstrates the use of overlay with ALF by calculating the inverse of a block diagonal matrix.	alf-examples-source /opt/cell/sdk/src/alf/ inverse_matrix_ovl	3.0
matrix_add	Demonstrates how to program with ALF API. By adding two 1024x512 single precision floating point matrices, the sample code	alf-examples-source /opt/cell/sdk/src/alf/matrix_add	3.0

	demonstrates how a simplified scalar function can be ported and accelerated for parallel execution on ALF. This simple example illustrates different features in ALF, including data partitioning on the host, data partitioning on the accelerator, overlapped in/out buffer, and data set.		
matrix_transpose	Demonstrates how to program with the ALF API. By transposing a 1024x512 single precision floating point matrix, the sample code shows how a simplified scalar function can be ported and accelerated for parallel execution on ALF. The sample contains a scalar, non-ALF version of the code, a data partition on host version, a data partition on accelerator version, and a tuned version with SIMD.	alf-examples-source /opt/cell/sdk/src/alf/ matrix_transpose	3.0
PI	Demonstrates computing PI by Buffon's Needle method using ALF. It shows how to use the ALF context buffer for global parameters and how to collect computing results on each task instance. This program also shows a progress bar during the computation by using the sync point feature in ALF.	alf-examples-source /opt/cell/sdk/src/alf/PI	3.0
pipe_line	Demonstrates the task dependency feature in ALF. It shows how task dependency is used in a two-stage pipeline application. The application is a simple simulation. An object P is placed in the middle of a flat surface with a bounding rectangular box. On each simulation step, the object moves a random distance in a random direction. It moves back to the initial position when it hits the side walls of the bounding box. The problem is to calculate the number of hits to the four walls in a given time period.	alf-examples-source /opt/cell/sdk/src/pipe_line	3.0
task_context	Contains a collection of small samples that demonstrate the use of task context in ALF. Four included samples follow.	alf-examples-source /opt/cell/sdk/src/task_context	3.0
task_context: dot_prod	Computes the dot product of two large vectors. It shows how to use the bundled work block distribution together with the task context to handle situations where the work block cannot hold the partitioned data because of a local memory size limit.	/opt/cell/sdk/src/task_context/ dot_prod	3.0
task_context: dot_prod_multi	Computes the dot product of two large vectors. It shows how to use	/opt/cell/sdk/src/task_context/ dot_prod_multi	3.0

	the multi-use workblocks together with work block parameter and context buffers.		
task_context: min_max	Shows how to use the task context to keep the partial computing results for each task instance and then to combine these partial results into the final result.	/opt/cell/sdk/src/task_context/min_max	3.0
task_context: table_lookup	Shows how the task context buffer is used as a large lookup table to convert the 16-bit input data into 8-bit output data.	/opt/cell/sdk/src/task_context/table_lookup	3.0

ALF hybrid examples

These examples show how the Accelerated Library Framework works on the hybrid system.

Table 6. ALF hybrid examples

Example	Description	RPM Default installed directory	SDK
BlackScholes_ALF	Demonstrates how to perform a Black Scholes pricing model. It demonstrates a closed form solution for Black Scholes using ALF.	alf-hybrid-examples-source /opt/cell/sdk/prototype/src/alf/BlackScholes_ALF/	3.0
FFT16M_ALF	Demonstrates an ALF version of the FFT16M workload in the cell-examples-source RPM.	alf-hybrid-examples-source /opt/cell/sdk/prototype/src/alf/FFT16M_ALF	3.0
hello_world	Demonstrates a *hello_world* program that shows a simple ALF program. It contains an SPU computational kernel that issues a printf of the hello_world program.	alf-hybrid-examples-source /opt/cell/sdk/prototype/src/alf/hello_world	3.0
inout_buffer	Demonstrates the usage of overlapped I/O buffers. The first example implements C=A+B, where A, B, and C are matrices. The second example implements A=A+B, where matrix A is overwritten by the result.	alf-hybrid-examples-source /opt/cell/sdk/prototype/src/alf/inout_buffer	3.0
inverse_matrix_ovl	Demonstrates the use of overlay with ALF by calculating the inverse of a block diagonal matrix.	alf-hybrid-examples-source /opt/cell/sdk/prototype/src/alf/inverse_matrix_ovl	3.0
matrix_add	Demonstrates how to program with ALF API. By adding two 1024x512 single precision floating point matrices, the sample code demonstrates how a simplified scalar function can be ported and accelerated for parallel execution on ALF. This simple example illustrates different features in ALF, including data partitioning on the host, data partitioning on the accelerator, overlapped in/out buffer, and data set.	alf-hybrid-examples-source /opt/cell/sdk/prototype/src/alf/matrix_add	3.0
matrix_transpose	Demonstrates how to program with the ALF API. By transposing a 1024x512 single precision	alf-hybrid-examples-source /opt/cell/sdk/prototype/src/alf/matrix_transpose	3.0

	floating point matrix, the sample code shows how a simplified scalar function can be ported and accelerated for parallel execution on ALF. The sample contains a scalar, non-ALF version of the code, a data partition on host version, a data partition on accelerator version, and a tuned version with SIMD.		
PI	Demonstrates computing PI by Buffon's Needle method using ALF. It shows how to use the ALF context buffer for global parameters and how to collect computing results on every task instance. This program also shows a progress bar during the computation by using the sync point feature in ALF.	alf-hybrid-examples-source /opt/cell/sdk/prototype/src/alf/PI	3.0
pipe_line	Demonstrates the task dependency feature in ALF. It shows how task dependency is used in a two-stage pipeline application. The application is a simple simulation. An object P is placed in the middle of a flat surface with a bounding rectangular box. On each simulation step, the object moves a random distance in a random direction. It moves back to the initial position when it hits the side walls of the bounding box. The problem is to calculate the number of hits to the four walls in a given time period.	alf-hybrid-examples-source /opt/cell/sdk/prototype/src/pipe_line	3.0
task_context	Contains a collection of small samples that demonstrate the use of task context in ALF. (The same samples under task_context in Table 5.)	alf-hybrid-examples-source /opt/cell/sdk/prototype/src/task_context	3.0

LAPACK examples

These examples show how to use the LAPACK library.

Table 7. LAPACK examples

Example	Description	RPM Default installed directory	SDK
dsteqr_F	Computes all eigenvalues and optionally eigenvectors of a symmetric tridiagonal matrix using the implicit QL or QR method.	lapack-examples-source /opt/cell/sdk/src/lapack-examples/dsteqr_F	3.0.0.3
inverse_matrix	Computes inverse of a matrix on Cell.	lapack-examples-source /opt/cell/sdk/src/lapack-examples/inverse_matrix	3.0.0.3

BLAS examples

These examples show how to use the BLAS library.

Table 8. BLAS examples

Example	Description	RPM Default installed directory	SDK
blas_simple	Demonstrates the use of the BLAS library at the PPU level, which has the standard BLAS interface.	blas-devel /opt/cell/sdk/src/blas-examples/ blas_simple	3.0
blas_thread	Demonstrates the use of the BLAS library with a multi-threaded PPU application.	blas-devel /opt/cell/sdk/src/blas-examples/ blas_thread	3.0
spulet	Demonstrates the use of the SPU BLAS library containing SPU BLAS kernel routines.	blas-devel /opt/cell/sdk/src/blas-examples/ spulet	3.0

LIB FFT examples

These examples show how to use the FFT library.

Table 9. LIB FFT examples

Example	Description	RPM Default installed directory	SDK
FFT1D	Demonstrates the use of the prototype libfft code. In particular, this example demonstrates how the library can be made to perform with high efficiency on carefully crafted 1D FFTs.	libfft-examples-source /opt/cell/sdk/prototype/src/libfft/ FFT1D	3.0
FFT1D.spu_only	Demonstrates the use of the prototype libfft code. In particular, this example demonstrates how the library can be made to perform with high efficiency on carefully crafted 1D FFTs on an SPU.	libfft-examples-source /opt/cell/sdk/prototype/src/libfft/ FFT1D.spu_only	3.0
FFT2D	Demonstrates the use of the prototype libfft code. In particular, this example demonstrates how the library can be made to perform with high efficiency on carefully crafted 2D FFTs.	libfft-examples-source /opt/cell/sdk/prototype/src/libfft/ FFT2D	3.0
FFT2D.stream	Demonstrates the use of the prototype libfft code. In particular, this example demonstrates how the library can be made to perform with high efficiency on carefully crafted 2D FFTs.	libfft-examples-source /opt/cell/sdk/prototype/src/libfft/ FFT2D.stream	3.0