Brown Deer Technology

Release Notes for the COPRTHR® SDK version 1.6

The CO-PRocessing THReads® SDK provides libraries and tools for application developers targeting multi-core and many-core parallel co-processing computer platforms.

---

1 Comments on the "Freewill" release of the COPRTHR SDK

The COPRTHR SDK version 1.6 (Freewill) is released just over 4 years after the original design of the key components such the STDCL® API for compute offload accelerators (once this was synonymous with GPUs - times have surely changed). During this time various APIs for heterogeneous computing including "industry standards" have been promoted by vendors. Nevertheless, the community has failed to reach a consensus around a single approach to the fundamental task of developing portable code for heterogeneous platforms with accelerators. The relatively recent hardware entry of Intel Xeon Phi has compounded the confusion and lack of a clear path toward portable performant code.

The COPRTHR SDK continues to provide unique capabilities for programming accelerators. Examples include the STDCL API and the clcc compiler tools that support a standard compilation model. STDCL provides an intuitive API that is superior in many ways to the many alternatives that exist today. Code written years ago in STDCL remains as portable and stable today as code written using one of the many "industry standards" promoted by various vendors. If anything, applications written using STDCL have "aged" better than many popular alternatives.

With some APIs focused exclusively on just-in-time (JIT) compilation models that reflect the exception and not the norm for many application developers, the COPRTHR clcc tools allow developers to use a conventional workflow employing pre-compiled libraries and executables on fixed hardware platforms.

The COPRTHR SDK provides innovative software technology like CLRPC for networked devices and CLETE for expresssion template based automatic acceleration. With the Freewill release we add a new low-level API based on a logical extension of pthreads for co-processors in an effort to address new requriements for co-processor architectures with fully divergent cores and other capabilities not found with traditional GPUs.

We continue to push forward with the COPRTHR SDK because we believe it provides a valuble tool for code development, and we will continue to explore software innovation in the area of accelerators and co-processors.

DAR

---

2 New In Version 1.6 ("Freewill") Release

- *New capabilities in the COPRTHR compiler* `clcc`

  Support for a standard compilation model targeting heterogeneous systems of processors and co-processors is enhanced with several new capabilities

  - Strong binding of kernel symbols targeting a new STDCL dynamic linker
  - Pre-compiled kernels linkable with shared libraries or executables
  - Cross-compiler support for Android and MIC
  - Pre-compiled kernel support extended to FreeBSD

- *New support for Intel MIC architecture*

  - STDCL and OpenCL APIs for offload of native execution
  - CLRPC server support for MIC accelerators enabling networked topologies

- *New low-level COPRTHR API including "pthreads for co-processors"*

  - Simple interface to co-processors used to support higher level APIs like STDCL and OpenCL

- Includes logical extension of Pthreads for co-processors

  - Support for architectures with divergent cores and capabilities not accessible with APIs like OpenCL

- *Updated support for Epiphany/Parallella platforms*

  - Updated support for production Parallella platforms using the Epiphany SDK version 5

---

3 Support and Requirements

Full SDK support is provided for Linux, Ubuntu, and FreeBSD operating systems. Additionally, the STDCL API is supported for Windows 7.

Supported hardware includes AMD and Nvidia GPUs, Intel and AMD x86 multicore CPUs, Intel MIC accelerators, ARM multicore CPUs, and Epiphany multicore processors. The COPRTHR SDK leverages vendor OpenCL implementations for portability and also provides OpenCL implementions for multicore processors to provide truly cross-vendor/cross-device support for heterogeneous computing platforms.

Package dependencies (O=Optional, R=Required)

1. libelf-0.8.13 www.mr511.de/software/libelf-0.8.13.tar.gz [Required]
2. libconfig-1.4.8 or later www.hyperrealm.com/libconfig/libconfig-1.4.9.tar.gz [Required]
3. libevent-2.0.18 or later github.com/downloads/libevent/libevent/libevent-2.0.21-stable.tar.gz [Required]
4. GCC 4.6 or later [Required]

Additionally, vendor support is needed for certain specific OpenCL devices

5. AMD APP developer.amd.com/sdks/AMDAPPSDK/downloads [Optional]
6. Nvidia CUDA 4 developer.nvidia.com/cuda-toolkit-40 [Optional]
7. Intel OCL 1.5 software.intel.com/en-us/articles/vcsource-tools-opencl-sdk [Optional]
8. Adapteva Epiphany SDK for Epiphany and Parallella support [Optional]

Please take note that libelf 1.x branch found on most Linux distributions is *not a valid substitute* for libelf-0.8.13 since they lack the required features and exhibit undocumented broken behavior.

---

4 Installation

4.1 Installation Overview

The COPRTHR SDK may be installed from pre-compiled binaries for selected platforms or built from source. It may be necessary to install one or more 3rd-party packages as part of the installation process as described above in the previous section [Support and Requirements)(#support_and_requirements).

For Linux and FreeBSD installation a configure script is provided to customize the package. Detailed descriptions of the various options are described at the end of this section.

4.2 Download the Release

Source and binary releases of the COPRTHR SDK are available from the download page (http://www.browndeertechnology.com/

Alternatively, the latest updates and development branch may be found on the github project page (http://www.github.com/browndeer/coprthr).

4.3 Linux and FreeBSD Installation

The following generic steps can be used to install the COPRTHR SDK on typical Linux and FreeBSD systems. For a full description of configure options and suggested system-specific configurations, see section 2.5 and 2.6. Additionally, details for more complex platform configurations can be found in in specific Application Notes.

1) From the root coprthr directory, type,
   ./configure [options]
2) Build the package, type,
   gmake
3) Install the package, type,
   gmake install
4) Set the appropriate paths to use the headers and libraries when building your own applications.

4.4 Windows 7 Installation

Run the Windows installer (libstdcl-1.4.0-win7-install.msi) and set the appropriate paths to use the headers and library from your application.

4.5 Configure Options

When building from source the configure script supports the following options:

–**prefix=/path/to/target-install-dir** set the root directory for

installation

–**enable-clgl** enable clgl support (default=yes)

–**enable-clete** enable clete expression template acceleration (default=yes)

–**enable-debug-libs** enable building debug versions of librarires (default=yes)

–**enable-clcc** enable building clcc tools (default=yes)

–**enable-libcoprthr** enable libcoprthr (default=yes)

–**enable-libcoprthrcc** enable libcoprthrcc (default=yes)

–**enable-libocl** enable libocl (default=yes)

–**enable-libclrpc** enable libclrpc (default=yes)

–**enable-libcoprthr-ncpu** enable libcoprthr ncpu (default=yes)

–**enable-fortran** enable fortran bindings (default=no)

–**enable-silent** enable silent build (default=no)

–**enable-debug** enable debug messages (default=no)

–**enable-epiphany** enable epiphany support (default=no)

–**enable-libocl-hook** enable libocl hook support (default=no)

–**enable-emek-build** enable build for Epiphany EMEK (default=no)

–**enable-old-esdk** enable old eSDK API for Epiphany (default=no)

–**enable-user-install** enable installation by user without roo permission (default=no)

–**enable-android-cross-compile** enable cross-compile for Android (default=no)

–**enable-mic-cross-compile** enable cross-compile for MIC (default=no)

–**with-opencl-platform=NAMELIST** specify default platform selection as a comma seperated list

–**with-opencl-include=DIR** specify OpenCL include path

–**with-opencl-lib=DIR** specify OpenCL lib path

–**with-fortran=PROG** Use fortran compiler PROG

–**with-lib=DIR** set path for prerequisite libs

–**with-libelf=DIR** specify path to libelf

–**with-opencl-icd-path=DIR** set custom path for ICD files

–**with-libevent=DIR** set path for libevent

–**with-libconfig=DIR** set path for libconfig

–**with-max-clmesg-level=LEVEL** set max clmesg level

–**with-default-clmesg-level=LEVEL** set default clmesg level

–**with-esdk=DIR** set path for esdk

–**with-android-ndk=DIR** set path for android NDK

–**with-android-platform=DIR** set path for android platform

–**with-android-arch=DIR** set path for android arch

–**with-lib-mic=DIR** set path for MIC libraries

4.6 Suggested System-Specific Configurations

The following are just a few examples of common system-specific configurations to be used as a guide for installing COPRTHR SDK on your specific platform. In very many cases, simply using ./configure will work if you have installed 3rd-party software in standard locations.

Linux using only AMD APP SDK for CPU/GPU

```
./configure --with-opencl-platform=amdapp
```

Linux using AMD APP SDK for CPU/GPU and COPRTHR OpenCL for CPU also

```
./configure --enable-libcoprthr --with-opencl-platform=amdapp,coprthr
```

Linux using only Nvidia OCL for GPU

```
./configure --enable-libcoprthr --with-opencl-platform=nvidia
```

Linux using only Intel OCL for CPU/GPU

```
./configure --with-opencl-platform=intel
```

Linux or FreeBSD standalone installation with COPRTHR OCL for CPU (x86)

```
./configure --enable-libcoprthr
```

Linux standalone installation with COPRTHR OCL for CPU (ARM)

```
./configure --enable-libcoprthr
```

Parallella/Epiphany support

```
./configure --enable-epiphany
```

---

5 Important Notes

- This release has been updated to use the Epiphany SDK version 5 and supports production Parallella platforms.
- On Windows 7 platforms the function init_stdcl() must be called to initialize the STDCL API. This call is unnecessary for Linux and FreeBSD, but there is no harm in including it since it will default to an empty macro.
- The version of the boost library (1.47.0) used here for development unfortunately has a very minor bug that prevents it from working with MSVS 2010. A fix to the boost package is provided in this release. Just follow the instructions under the `msvs2010/boost_1_47_0-multi_array-iterator-fix/` directory. (All you need to do is copy over a replacement for iterator.hpp.)
- Some example kernels include the stdcl.h header. Including this header for OpenCL kernels allows the use of alternate syntax that avoids breaking C. Programmers are encouraged to begin this practice now. A future release will formalize syntax corrections to OpenCL to produce a C compliant language for programming thread functions (kernels). The alternate syntax will be completely backward compatable with OpenCL.

---

6 More Information

Additional information and examples may be found in the COPRTHR Primer version 1.6.

Document revision 1.6.0.0