AMD **Accelerated** Parallel Processing TECHNOLOGY

AMD APP SDK v2.8

FAQ

1 General Questions

1. **Do I need to use additional software with the SDK?**

   To run an OpenCL™ application, you must have an OpenCL™ runtime on your system. If your system includes a recent AMD discrete GPU, or an APU, you also should install the latest Catalyst™ drivers, which can be downloaded from AMD.com. Information on supported devices can be found at developer.amd.com/appsdk. If your system does not include a recent AMD discrete GPU, or APU, the SDK installs a CPU-only OpenCL™ run-time.

   Also, we recommend using the debugging profiling and analysis tools contained in the AMD CodeXL heterogeneous compute tools suite.

2. **Which versions of the OpenCL™ standard does this SDK support?**

   AMD APP SDK 2.8 supports development of applications using the OpenCL™ Specification v 1.2. As all OpenCL™ 1.1 APIs are supported within OpenCL™ 1.2, you also can develop OpenCL™ 1.1-compliant applications.

3. **Will applications developed to execute on OpenCL™ 1.1 still operate in an OpenCL™ 1.2 environment?**

   OpenCL™ is designed to be backwards compatible. The OpenCL™ 1.2 run-time delivered with the AMD Catalyst drivers run any OpenCL™ 1.1-compliant application. However, an OpenCL™ 1.2-compliant application will not execute on an OpenCL™ 1.1 run-time if APIs only supported by OpenCL™ 1.2 are used.

4. **Does AMD provide any additional OpenCL™ samples, other than those contained within the SDK?**

   The most recent versions of all of the samples contained within the SDK are also available for individual download from the developer.amd.com/appsdk “Samples & Demos” page. This page also contains additional samples that either were too large to include in the SDK, or which have been developed since the most recent SDK release. Check this web page for new, updated, or large samples.

5. **How often can I expect to get AMD APP SDK updates?**

   Developers can expect that the AMD APP SDK may be updated two to three times a year. Actual release intervals may vary depending on available new features and product updates. AMD is committed to providing developers with regular updates to allow them to take advantage of the latest developments in AMD APP technology.

6. **What is the difference between the CPU and GPU components of OpenCL™ that are bundled with the AMD APP SDK?**

   The CPU component uses the compatible CPU cores in your system to accelerate your OpenCL™ compute kernels; the GPU component uses the compatible GPU cores in your system to accelerate your OpenCL™ compute kernels.

7. **What CPUs does the AMD APP SDK v2.8 with OpenCL™ 1.2 support work on?**

   The CPU component of OpenCL™ bundled with the AMD APP SDK works with any x86 CPU with SSE3 or later, as well as SSE2.x or later. AMD CPUs have supported SSE3 (and later) since 2005. Some examples of AMD CPUs that support SSE3 (or later) are the AMD Athlon™ 64 (starting with the Venice/San Diego steppings), AMD Athlon™ 64 X2, AMD Athlon™ 64 FX (starting with San Diego stepping), AMD Opteron™ (starting with E4 stepping), AMD Sempron™ (starting with Palermo stepping), AMD Phenom™, AMD Turion™ 64, and AMD Turion™ 64 X2.

8. **What APUs and GPUs does the AMD APP SDK v2.8 with OpenCL™ 1.2 support work on?**

   For the list of supported APUs and GPUs, see the AMD APP SDK v2.8 System Requirements list at: http://developer.amd.com/appsdk

9. **Can my OpenCL™ code run on GPUs from other vendors?**

   At this time, AMD does not plan to have the AMD APP SDK support GPU products from other vendors; however, since OpenCL™ is an industry standard programming interface, programs written in OpenCL™ 1.2 can be recompiled and run with any OpenCL-compliant compiler and runtime.

10. **What version of MS Visual Studio is supported?**

    The AMD APP SDK v2.8 with OpenCL™ 1.2 supports Microsoft® Visual Studio® 2008 Professional Edition, Microsoft® Visual Studio® 2010 Professional Edition, and Microsoft® Visual Studio 2012.

11. **Is it possible to run multiple AMD APP applications (compute and graphics) concurrently?**

    Multiple AMD APP applications can be run concurrently, as long as they do not access the same GPU at the same time. AMD APP applications that attempt to access the same GPU at the same time are automatically serialized by the runtime system.

12. **Which graphics driver is required for the current AMD APP SDK v2.8 with OpenCL™ 1.2 CPU support?**

    For the minimum required graphics driver, see the AMD APP SDK v2.8 System Requirements list at: http://developer.amd.com/appsdk. In general, it is advised that you update your system to use the most recent graphics drivers that are available for it.

13. **How does OpenCL™ compare to other APIs and programming platforms for parallel computing, such as OpenMP and MPI? Which one should I use?**

    OpenCL™ is designed to target parallelism within a single system and provide portability to multiple different types of devices (GPUs, multi-core CPUs, etc.). OpenMP targets multi-core

CPUs and SMP systems. MPI is a message passing protocol most often used for communication between nodes; it is a popular parallel programming model for clusters of machines. Each programming model has its advantages. It is anticipated that developers mix APIs, for example programming a cluster of machines with GPUs with MPI and OpenCL.

14. If I write my code on the CPU version, does it work on the GPU version, or do I have to make changes.

Assuming the size limitations for CPUs is considered, the code works on both the CPU and GPU components. Performance tuning, however, is different for each.

15. What is the precision of mathematical operations?

See Chapter 7, “OpenCL Numerical Compliance,” of the OpenCL™ 1.2 Specification for exact mathematical operations precision requirements.

http://developer.amd.com/support/KnowledgeBase/Lists/KnowledgeBase/DispForm.aspx?ID=88

16. Are byte-addressable stores supported?

Byte-addressable stores are supported.

17. Are long integers supported?

Yes, 64-bit integers are supported.

18. Are operations on vectors supported?

Yes, operations on vectors are supported.

19. Is swizzling supported?

Yes, swizzling (the rearranging of elements in a vector) is supported.

20. How does one verify if OpenCL™ has been installed correctly on the system?

Run `clinfo` from the command-line interface. The clinfo tool shows the available OpenCL™ devices on a system.

21. How do I know if I have installed the latest version of the AMD APP SDK

On installation of the SDK, if an internet connection is available, the installer states whether or not a newer SDK is available in the file `VersionInfo.txt` in directory `C:\Program Files (x86)\AMD APP\docs\` . Alternatively, you can check for the latest available version of the AMD APP SDK at `http://developer.amd.com/appsdk`.

2 Optimizations

22. How do I use constants in a kernel for best performance?

For performance using constants, highest to lowest performance is achieved using:

- Literal values
- Constant pointer with compile time constants indexing.
- Constant pointer with runtime constant indexing that is the same for all threads.

- Constant pointer with linear access indexing that is the same for all threads.
- Constant pointer with linear access indexing that is different between threads.
- Constant pointer with random access indexing.

23. Why are literal values the fastest way to use constants?

Up to 96 bits of literal values are embedded in the instruction; thus, in theory, there is no limit on the number of usable literals. In practice, the limit is 16K unique literals in a compilation unit.

24. Why does `a * b + c` not generate a mad instruction?

Depending on the hardware and the floating point precision, the compiler may not generate a mad instruction for the computation of `a * b + c` due to the floating-point precision requirements in the OpenCL™ specification. Here, developers who want to exploit the mad instruction for performance can replace that computation with the `mad()` built-in function in OpenCL.

25. What is the preferred work-group size?

The preferred work-group size on the AMD platform is a multiple of 64.

3 OpenCL™ Questions

26. What is OpenCL™?

OpenCL™ (Open Computing Language) is the first truly open and royalty-free programming standard for general-purpose computations on heterogeneous systems. OpenCL™ lets programmers preserve their expensive source code investment and easily target both multi-core CPUs and the latest GPUs, such as those from AMD.

Developed in an open standards committee with representatives from major industry vendors, OpenCL™ gives users a cross-vendor, non-proprietary solution for accelerating their applications on their CPU and GPU cores.

27. How much does the AMD OpenCL™ development platform cost?

AMD bundles support for OpenCL™ as part of its AMD APP SDK product offering. The AMD APP SDK is offered to developers and users free of charge.

28. What operating systems does the AMD APP SDK v2.8 with OpenCL™ 1.2 support?

AMD APP SDK v2.8 runs on 32-bit and 64-bit versions of Windows and Linux. For the exact list of supported operating systems, see the AMD APP SDK v2.8 System Requirements list at: http://developer.amd.com/appsdk

29. Can I write an OpenCL™ application that works on both CPU and GPU?

Applications that program to the core OpenCL™ 1.2 API and kernel language should be able to target both CPUs and GPUs. At runtime, the appropriate device (CPU or GPU) must be selected by the application.

30. Does the AMD OpenCL™ compiler automatically vectorize for SSE on the CPU?

The CPU component of OpenCL™ that is bundled with the AMD APP SDK takes advantage of SSE3 instructions on the CPU. It also takes advantage of the AVX instructions where supported. In addition to AVX, OpenCL™ math library functions also leverage XOP and FMA4 capabilities on CPUs that support them.

31. Does the AMD APP SDK v2.8 with OpenCL™ 1.2 support work on multiple GPUs (ATI CrossFire)?

OpenCL™ applications can explicitly invoke separate compute kernels on multiple compatible GPUs in a single system. The partitioning of the algorithm to multiple parallel compute kernels must be done by the developer. It is recommended that ATI CrossFire be turned off in most system configurations so that AMD APP applications can access all available GPUs in the system.

ATI CrossFire technology allows multiple AMD GPUs to work together on a single graphics-rendering task. This method does not apply to AMD APP computational tasks because it is not compatible with the compute model used for AMD APP applications.

32. Can I ship pre-compiled OpenCL™ application binaries that work on either CPU or GPU?

By using OpenCL™ runtime APIs, developers can write OpenCL™ applications that can detect the available compatible CPUs and GPUs in the system. This lets developers pre-compile applications into binaries that dynamically work on either CPUs or GPUs that execute on targeted devices. Including LLVM IR in the binary provides a means for the binary to support devices for which the application was not explicitly pre-compiled.

33. Is the OpenCL™ double precision optional extension supported?

The Khronos and AMD double precision extensions are supported on certain devices. Your application can use the OpenCL™ API to query if this functionality is supported on the device in use.

34. Is it possible to write OpenCL™ code that scales transparently over multiple devices?

For OpenCL™ programs that target only multi-core CPUs, scaling can be done transparently; however, scaling across multiple GPUs requires the developer to explicitly partition the algorithm into multiple compute kernels, as well as explicitly launch the compute kernels onto each compatible GPU.

35. What should I do if I get wrong results on the Apple platform with AMD devices?

Apple handles support for the Apple platform; please contact them.

36. Is it possible to dynamically index into a vector?

No, this is not possible because a vector is not an array, but a representation of a hardware register.

37. What is the difference between `local int a[4]` and `int a[4]`?

`local int a[4]` uses hardware local memory, which is a small, low-latency, high-bandwidth memory; `int a[4]` uses per-thread hardware scratch memory, which is located in uncached global memory.

38. Why does using a barrier cause the max kernel work-group size to drop to 64 on HD4XXX chips?

The supported HD4XXX chips do not have a hardware barrier, so the OpenCL™ runtime cannot execute more than a single wavefront per group to satisfy the OpenCL™ memory consistency model. Note that HD4XXX device support is EOL. Catalyst drivers no longer include support for these devices. See the OpenCL™ SDK driver and compatibility page for more details.

39. How come my program runs slower in OpenCL™ than in CUDA/Brook+/IL?

When comparing performance, it is better to compare code optimized for our OpenCL™ platform against code optimized against another vendor's OpenCL™ platforms. By comparing the same toolchain on different vendors, you can find out which vendors hardware works the best for your problem set.

40. Why can I not use texture on RV7XX devices in OpenCL?

RV7XX devices do not support all of the texture modes and precision requirements that OpenCL™ requires. Since textures are mapped to images in OpenCL™ and is an “all or nothing” approach, we do not support images on RV7XX devices; thus, there is no access to textures. Note that HD4XXX device support is EOL. Catalyst drivers no longer include support for these devices. See the OpenCL™ SDK driver and compatibility page for more details.

41. Why do read-write images not exist in OpenCL?

OpenCL™ has a memory consistency model that requires certain constraints (see the *OpenCL Specification* for more information). Since images are special functional hardware units, they are different for reading and writing. This is different from pointers, which for the most part use the same hardware units and can guarantee the consistency that OpenCL™ requires.

42. Does prefetching work on the GPU?

Prefetch is not needed on the GPU because the hardware has a built-in mechanism to hide latency when many work-groups are running. The LDS can be used as a software-controlled cache.

43. How do you determine the max number of concurrent work-groups?

The maximum number of concurrent work-groups is determined by resource usage. This includes number of registers, amount of LDS space, and number of threads per work group. There is no way to directly specify the number of registers used by a kernel. Each SIMD has a 64-wide register file, with each column consisting of 256 x 32 x 4 registers.

44. Is it possible to tell OpenCL™ not to use all CPU Cores?

Yes, use the `device fission` extension.

4 OpenCL™ Optimizations

45. What is more efficient, the ternary operator `?:` or the `select` function?

The select function compiles to the single cycle instruction, `cmov_logical`; in most cases, `?:` also compiles to the same instruction. In some cases, when memory is in one of the operands, the `?:` operator is compiled down to an IF/ELSE block. An IF/ELSE block takes more than a single instruction to execute.

46. What is the difference between 24-bit and 32-bit integer operations?

24-bit operations are faster because they use floating point hardware and can execute on all compute units. Many 32-bit integer operations also run on all stream processors, but if both a 24-bit and a 32-bit version exist for the same instruction, the 32-bit instruction executes only one per cycle.

5 Hardware Information

47. How are 8/16-bit operations handled in hardware?

The 8/16-bit operations are emulated with 32-bit registers.

48. Do 24-bit integers exist in hardware?

No, there are 24-bit instructions, such as `MUL24`/`MAD24`, but the smallest integer in hardware registers is 32-bits.

49. What are the benefits of using 8/16-bit types over 32-bit integers?

8/16-bit types take less memory than a 32-bit integer type, increasing the amount of data you are able to load with a single instruction. The OpenCL™ compiler up-converts to 32-bits on load and down-converts to the correct type on store.

50. What is the difference between a GPR and a shared register?

Although they are physically equivalent, the difference is whether the register offset in the hardware is absolute to the register file or relative to the wavefront ID.

51. How often are wavefronts created?

Wavefronts are created by the hardware to execute as long as resources are available. If they are created but cannot execute immediately, they are put in a wait queue where they stay until currently running wavefronts are finished.

52. What is the maximum number of wavefronts?

The maximum number of wavefronts is determined by which resource limits the number of wavefronts that can be spawned. This can be the number of registers, amount of local memory, required stack size, or other factors. Compute shader with local memory usage has a hard cap at 16 wavefronts.

53. Why do I get blue or black screens when executing longer running kernels?

The GPU is not a preemptable device. If you are running the GPU as your display device, ensure that a compute program does not use the GPU past a certain time limit set by Windows. Exceeding the time limit causes the watchdog timer to trigger; this can result in undefined program results.

54. What is the cost of a clause switch?

In general, the latency of a clause switch is around 40 cycles. Note that this is relevant only for Evergreen and Northern Island devices.

55. How can I hide clause switch latency?

By executing multiple wavefronts in parallel. Note that this is relevant only for Evergreen and Northern Island devices.

56. How can I reduce clause switches?

Clause switches are almost directly related to source program control flow. By reducing source program control flow, clause switches can also be reduced. This is only relevant for Evergreen and Northern Islands devices.

57. How does the hardware execute a loop on a wavefront?

The loop only ends execution for a wavefront once every thread in the wavefront breaks out of the loop. Once a thread breaks out of the loop, all of its execution results are masked, but the execution still occurs.

58. How does flow control work with wavefronts?

There are no flow control units for each individual thread, so the whole wavefront must execute the branch if any thread in the wavefront executes the branch. If the condition is false, the results are not written to memory, but the execution still occurs.

59. What is the constant buffer size on GPU hardware?

64 kB.

60. What happens with out-of-bound memory operations?

Writes are dropped, and reads return a pre-defined value.

61. For 7XX devices, why does 64x1 give bad performance in OpenCL™ kernels?

One of the reasons is because of how the caches are setup on RV7XX devices. The caches are optimized to work in a tiled mode, not in linear mode (which is the mode OpenCL™ kernels use). To get optimal cache re-use from the texture in compute shader mode on RV7XX devices, reblock your thread IDs. A 16x4, 8x8, or 4x16 should give you good enough blocking to get similar cache performance as your pixel shader kernel. This is because a cacheline can be thought of as a 4x2 block of data coming in at once. So, for pixel shaders, 64 threads are blocked in a 8x8 block that uses exactly eight cache lines. For OpenCL™ kernels, your 64x1 block pattern uses 16 cache lines, but only uses half the data in each cache line. Note that HD4XXX device support is EOL. Catalyst drivers no longer include

support for these devices. See the OpenCL™ SDK driver and compatibility page for more details.

62. What is unique about the LDS in HD4XXX devices, and what are its performance characteristics?

The LDS in the HD 4XXX devices is an owner's write model with limited applications. When used correctly, it has very similar performance characteristics to the L1 cache, but the user gains control over what data exists in the memory. The LDS_Transpose sample in the SDK uses the LDS in the HD 4XXX devices very efficiently. Note that HD4XXX device support is EOL. Catalyst drivers no longer include support for these devices. See the OpenCL™ SDK driver and compatibility page for more details.

6 Microsoft® Visual Studio®

63. Can I use the SDK with Microsoft® Visual Studio® 2012 Express?

Due to limitations in Microsoft® Visual Studio® Express, it is only possible to use build files for individual samples. Microsoft® Visual Studio® Express does not support building of all of the samples at the same time. The project files that build all of the samples are only supported by full versions of Microsoft® Visual Studio® 2008, 2010, or 2012.

7 Bolt

64. What GPUs is Bolt optimized for?

Bolt is optimized for AMD "Southern Islands" family of GPUs.

65. Which GPU compute technologies are supported by Bolt?

The preview version of Bolt uses OpenCL™ as its compute back-end; however, future releases should support both C++ AMP and OpenCL™ compute back-ends.

66. Are the Bolt APIs final?

The APIs in the Bolt preview are not final. The Bolt API should be stable and should use formal deprecation procedures from Bolt release 1.0 onwards. Please help us to improve Bolt; we encourage you to provide suggestions on how we can improve the Bolt API.

67. Does Bolt require linking to a library?

Yes, Bolt includes a small, static library that must be linked into a user program to properly resolve symbols. Since Bolt is a template library, almost all functionality is written in header files, but OpenCL™ follows an online compilation model which makes sense to include in a library and not in header files.

68. Does Bolt depend on any other libraries?

Yes, Bolt contains dependencies on Boost. All dependent header files and pre-compiled libraries are available in the Bolt SDK.

69. What algorithms does Bolt currently support?

Transform, reduce, transform_reduce, count, count_if, inclusive_scan, exclusive_scan, and sort.

70. What version of OpenCL™ does Bolt currently require?

Bolt uses features available in OpenCL™ v1.2.

71. Does Bolt require the AMD OpenCL™ runtime?

Yes, Bolt requires the use of templates in kernel code. At the time of this writing, AMD is the only vendor to provide this support.

72. Which Catalyst package should I use with Bolt?

Generally speaking, downloading and installing the latest Catalyst package contains the most recent OpenCL™ runtime. As of the time of this writing, the recommended Catalyst package is 12.10.

73. Which license is Bolt licensed under?

The Apache License, Version 2.0.

74. When should I use device_vector vs regular host memory?

bolt::cl::device_vector is used to manage device-local memory and can deliver higher performance on discrete GPU system. However, the host memory interfaces eliminate the need to create and manage device_vectors. If memory is re-used across multiple Bolt calls or is referenced by other kernels, using device_vector delivers higher performance

75. How do I measure the performance of OpenCL™ Bolt library calls?

The first time that a Bolt library routine is called, the runtime calls the OpenCL™ compiler to compile the kernel. Each unique template instantiation reuses the same OpenCL™ compilation, so two Bolt calls with the same functors are compiled only one time. When measuring the performance of Bolt, it is important to exclude the first call (with the compilation overhead) from the timing measurement. Here are two ways to time Bolt function calls:

The first example pulls the call outside the timer loop.

```
std::vector<int> a;
bolt::cl::transform(a.begin, a.end, z.begin(), bolt::cl::negate<int>);
startTimer(...);
for (int i=0; i<100; i++) {
    bolt::cl::transform(a.begin, a.end, z.begin(), bolt::cl::negate<int>);
}
stopTimer(...);
```

A second alternative is to explicitly exclude the first compilation from the timing region:

```
// Explicitly start timer after first iteration:
std::vector<int> a,z;
for (int i=0; i<100; i++) {
    if (i==1) {
        startTimer(...)
    }
    bolt::cl::transform(a.begin(), a.end(), z.begin(), bolt::cl::negate<int>);
```

```
}
stopTimer(...);
```

76. The Bolt library APIs accept host data structures such as std::vector(). Are these copied to the GPU or kept in host memory?

For host data structures, Bolt creates an OpenCL™ buffer with the `CL_MEM_USE_HOST_PTR` flag and uses the OpenCL™ map and unmap APIs to manage the host and device access. On the AMD OpenCL™ implementation, this results in data being kept in host memory when the host memory is aligned on 256-byte boundary. In other cases, the runtime copies the data as needed to the GPU. See section 4.5.2 of the *AMD Accelerated Parallel Processing OpenCL Programming Guide* for more information on memory optimization. Higher-performance may be obtained by aligning the host data structures to 256-byte boundaries or by using the device_vector class for finer control over the allocation properties.

77. Bolt APIs also accept device_vector inputs. When should use device_vectors for my inputs?

The bolt::cl::device_vector interface is a thin wrapper around the cl::Buffer class; it provides an interface similar to std::vector. Also, the device_vector optionally accepts a flags argument that is passed to the OpenCL™ buffer creation. The following two flags can be useful:

`CL_MEM_ALLOC_HOST_POINTER`: Forces memory to be allocated in host memory, even on discrete GPUs. GPUs access to the data is slower (over the PCIe bus), but the allocation avoids the overhead of copying data to, and from, the GPU. This can be useful when the data is used in only a single Bolt call before being processed again on the CPU.

`CL_MEM_USE_PERSISTENT_MEM_AMD`: Provides device-resident zero copy memory. Use this for data that is write-only by the CPU. See section 4.5.2 of the *AMD Accelerated Parallel Processing OpenCL Programming Guide* for more information.

8 C++ AMP

78. What is C++ AMP?

C++ Accelerated Massive Parallelism (C++ AMP) accelerates execution of C++ code by taking advantage of data-parallel hardware, such as the compute units on and APU. C++ AMP is an open specification that extends regular C++ through the use of the `restrict` keyword to denote portions of code to be executed on the accelerated device.

79. What AMD devices support C++ AMP?

Any AMD APU or GPU that supports Microsoft® DirectX 11 Feature Level 11.0 and higher.

80. Can you run a C++ AMP program on a CPU?

You can run a C++ AMP program on a CPU, but its usage is not recommended in a production environment. For more information, see:
http://blogs.msdn.com/b/nativeconcurrency/archive/2012/03/10/cpu-accelerator-in-c-amp.aspx

81. Where can I get support for C++ AMP?

C++ AMP is supported by the C++ compiler in Microsoft® Visual Studio® 2012.

82. Is C++ AMP supported on Linux?

At this time, C++ AMP is not supported on Linux; however, as C++ AMP is defined as an open standard, Microsoft has opened the door for implementations on operating systems other than Windows.

83. Where can I learn more about C++ AMP?

There is an MSDN page in C++ AMP. Also, the book *C++ AMP: Accelerated Massive Parallelism with Microsoft Visual C++*, by Kate Gregory and Ade Miller, may be useful.

9 Aparapi

84. What is Aparapi?

Aparapi is an API for expressing data parallel algorithms in Java and a runtime component capable of converting Java bytecode to OpenCL™ for execution on the GPU. If Aparapi cannot execute on the GPU at runtime, it executes the developer's algorithm in a Java thread pool.

For appropriate workloads, this extends Java's 'Write Once Run Anywhere' to include GPU devices."

85. What AMD devices support Aparapi?

Any AMD APU, CPU, or GPU that supports OpenCL™ 1.1 and higher will work. See the list of AMD APP SDK v2.8 System Requirements list at: http://developer.amd.com/appsdk.

86. Where can I get support for Aparapi?

See: http://code.google.com/p/aparapi/

87. Is Aparapi supported on Linux?

Yes.

88. Which JDK/JRE is recommended for Aparapi?

The Oracle JDK/JRE, although not a requirement for Aparapi, is highly recommended for performance and for stability reasons.

89. Where can I learn more about Aparapi?

You can find out more about Aparapi from: http://code.google.com/p/aparapi/

Contact

Advanced Micro Devices, Inc.
One AMD Place
P.O. Box 3453
Sunnyvale, CA, 94088-3453
Phone: +1.408.749.4000

For AMD Accelerated Parallel Processing:

URL: developer.amd.com/appsdk
Developing: developer.amd.com/
Forum: developer.amd.com/openclforum