Getting Started with CodeXL

AMD Developer Tools Team
Advanced Micro Devices, Inc.

Table of Contents

Introduction........................................................................................................................................... 2
Install CodeXL...................................................................................................................................... 2
Validate CodeXL installation............................................................................................................... 3
CodeXL help ......................................................................................................................................... 5
Run the Teapot Sample project ......................................................................................................... 5
Basic debugging .................................................................................................................................. 7
Source Code view ............................................................................................................................... 8
Breakpoint view .................................................................................................................................. 8
Watch and Locals views ..................................................................................................................... 10
Debug Explorer view .......................................................................................................................... 11
Call Stack view ................................................................................................................................... 13
Function Calls History view .............................................................................................................. 13
Debugged Process Events view ......................................................................................................... 13
Memory view ....................................................................................................................................... 14
Statistics view ..................................................................................................................................... 14
System information ............................................................................................................................. 15
Project settings ................................................................................................................................. 16
Profile mode ....................................................................................................................................... 16
CPU profiling ....................................................................................................................................... 17
GPU profiling ....................................................................................................................................... 19
Profile Session Explorer view ........................................................................................................... 21
Kernel Analyzer .................................................................................................................................. 21
Start KernelAnalyzer2...................................................................................................................... 21
Open kernel source............................................................................................................................. 22
Build Options ...................................................................................................................................... 24
Build the kernel ................................................................................................................................... 25
Known issues ........................................................................................................................................ 27
Support .............................................................................................................................................. 27
Introduction

AMD CodeXL™ is a coherent and innovative tool suite with a unified user interface. It lets you harness the benefits of AMD CPUs, GPUs, and APUs with a single program. You can use its powerful APU/GPU debugging, CPU and GPU profiling, and static OpenCL™ kernel analysis capabilities to find bugs and to optimize application performance, giving you easy access to the new era of heterogeneous computing. AMD CodeXL is available as a standalone application for Windows® and Linux® and as a Microsoft® Visual Studio® extension for Windows.

This document describes CodeXL system requirements, how to install CodeXL, and how to get started using CodeXL. It then describes how to get started using the kernel analysis tool KernelAnalyzer2. Finally, it explains how to find information about known CodeXL issues and how to contact AMD for support.

Install CodeXL

This section describes how to install CodeXL. The installation details are system-specific (Windows or Linux), but once you have installed and started CodeXL its operation is system-independent.

OpenCL profiling and OpenCL kernel debugging requirements:
• An AMD GPU (Radeon HD5000 or newer) or APU
• AMD Catalyst Driver, release 12.8 or later
  o Recommended version: Catalyst 12.12 (available late December 2012)
  o See: support.amd.com/US/GPUDOWNLOAD/Pages/index.aspx

OpenCL API-level debugging requirement:
• A working OpenCL/OpenGL configuration (AMD or other)

CPU profiling requirements:
• Time-based profiling: any x64 or AMD X64 (x86-64) CPU or APU
• Event-based profiling (EBP) and instruction-based sampling (IBS):
  an AMD CPU or APU

Windows system requirements:
• Microsoft Windows 7 (32 bit or 64 bit)
• [Optional] Microsoft Visual Studio 2010

Linux system requirements:
• Red Hat Enterprise Linux 6.0 U2 64-bit or later
or
• Ubuntu 11.10 64-bit or later
Then install it as described below for your system.

For Windows:

```
Download the .exe file CodeXL*.exe for Windows (32-bit or 64-bit). When the download completes, click on the .exe file to install CodeXL.
```

The installer will walk you through the installation process.

The CodeXL Visual Studio extension is part of the Windows .exe installer package and is installed by default. Choose “Custom” installation and deselect the Visual Studio extension if you do not want to install it.

For Linux:

```
Download the 64-bit Linux RPM package CodeXL*.rpm.
```

- For CentOS or Red Hat, install the .rpm directly:

```
$ sudo rpm -Uvh --nodeps CodeXL*.rpm
```

- For Ubuntu, either install using the rpm command, or alternatively convert the .rpm to a .deb package:

```
Either:
$ sudo apt-get install rpm
$ sudo rpm -Uvh --nodeps CodeXL*.rpm
or:
$ sudo apt-get install alien
$ sudo alien -k CodeXL*.rpm
```

**Validate CodeXL installation**

Launch the CodeXL standalone application (or Visual Studio, if you are using the VS CodeXL extension) after you finish CodeXL installation.

For Windows:

- Windows >> All Programs should have a new folder entry: AMD Developer Tools >> AMD CodeXL.
- An AMD CodeXL shortcut should appear on the desktop.
- Control Panel should show AMD CodeXL in its list of installed programs.
Double-click on the CodeXL desktop shortcut or select CodeXL from the program menu.

The CodeXL standalone application should start.

For Windows using the Visual Studio plugin:

Launch Microsoft Visual Studio.

The VS GUI should appear. Verify that AMD CodeXL is installed:

Select Help >> About Microsoft Visual Studio from the menu bar.

AMD CodeXL should be listed under Installed products. The VS menu bar should include a CodeXL pull-down menu.

For Linux:

Either:

- Add /opt/amd/CodeXL/bin/ to your PATH and launch:
  
  $ PATH=/opt/amd/CodeXL/bin:$PATH
  
  $ CodeXL

or:

- Navigate to /opt/amd/CodeXL/bin/ and launch:
  
  $ cd /opt/amd/CodeXL/bin/
  
  $ ./CodeXL

The CodeXL standalone application should start.

When you finish the installation and start CodeXL as described above, the CodeXL GUI window should appear. The CodeXL Debug Explorer view notes:

No project loaded
Screen displays generated by the version of CodeXL you install may differ slightly from the screenshots shown in this document.

**CodeXL help**

Click on the **AMD CodeXL User Guide** link on the CodeXL startup Home Page or select **Help >> View Help** from the CodeXL toolbar.

This brings up a CodeXL Help window:

![CodeXL Help window](image)

CodeXL Help gives some of the same information provided by this document, but it also includes more detailed specifics about CodeXL views and modules.

To bring up the CodeXL Help window for the Visual Studio extension:

Select **CodeXL >> Help >> View Help** from the VS menu.

**Run the Teapot Sample project**

The CodeXL distribution includes a sample project that displays a smoking teapot. The project uses OpenCL kernels to solve Navier-Stokes equations. It shares a 3D texture between OpenCL and OpenGL, copies a density field grid into the 3D texture, and renders the smoke using OpenGL. This document uses the teapot project to demonstrate the use of CodeXL.
For the Visual Studio extension:

Select CodeXL >> Open Teapot Sample Project from the VS toolbar.

Visual Studio should display the teapot sample project:

Screenshots in the remainder of this document show the standalone version of CodeXL. The Visual Studio version would look similar, but within a VS window rather than a CodeXL window.

For Windows or Linux:

Click on the Load the Teapot Sample link to load the teapot sample project.

The CodeXL Debug Explorer view now shows:

AMDTTeaPot - Not running

The CodeXL window also displays several other views, but since the program is not running they do not display any information yet.

To run the teapot sample program:
The program begins execution, and soon it displays a rotating smoking teapot in a separate window:

![Teapot Image]

To stop the program:

Select Debug >> Stop Debugging from the taskbar, or just click on the black square taskbar Stop button, or click on the close button in the upper right corner of the teapot window.

**Basic debugging**

The CodeXL GPU Debugger lets you examine the runtime behavior of your OpenCL/OpenGL application in detail. You can use the information it provides to find bugs and to improve application performance. You can debug OpenCL kernels, inspect variable values across different work items and work groups, inspect call stacks, and so on.

This quick start guide presumes that you are familiar with the use of a GUI debugger, so it gives only a quick introduction to the basic features of debugging with CodeXL.

Two buttons at the far left of the CodeXL taskbar let you select debug mode or profile mode:

Hover over any taskbar button to display a brief pop-up help description.

The next taskbar buttons control program execution during debugging:
These controls are (left to right): start, frame step, draw step, step over, step in, step out, break, and stop debugging. You can also perform these actions from the taskbar Debug pull-down menu or by using function keys.

The next taskbar buttons show or hide various views:

These buttons are (left to right): Properties, CodeXL Debug Explorer, Function Calls History, Debugged Process Events, Call Stack, Locals, Watch, OpenGL™ State Variables, OpenCL Multi-Watch (1,2,3), Breakpoints, Memory, Statistics, and CodeXL Profile Session Explorer.

You can resize views, drag and drop views to rearrange them, or move them to a separate window. The next sections of this guide describe individual CodeXL views in more detail.

**Source Code view**

Source Code views display C, C++, or OpenCL code.

Start the teapot program as described above, then hit the Break button to interrupt it.

A Source Code view displays the source file where the break occurred, with a yellow arrow indicating the current line number (here line 2687 in file `amdtteapottoclsnomesystem.cpp`):

**Breakpoint view**

The Breakpoint view shows active breakpoints. Initially, the Breakpoint view shows no breakpoints:
Now add a breakpoint:

Double-click on Double-click to add or remove breakpoints.

A new Breakpoints window appears:

Select the API Functions tab to set a breakpoint on an API function, or select the Kernel Functions tab to set a breakpoint on a kernel function.

When program execution hits a breakpoint, a Source view displays the line where the breakpoint occurs. A yellow arrow indicates the current location. A red dot next to the line number indicates a set breakpoint:
Watch and Locals views

A Watch view shows the values and types of program variables you specify, and a Locals view displays the values and types of local variables in a kernel. In the image above, the Watch view displays the value of variable `dPlaneDist`. The Locals view displays the values of all local variables in the current kernel (here `computeIntersection` in `tpVolumeSlicing.cl`). For a structured variable, click on the triangle left of the variable name to see the name and value of each member.

A Multi-Watch view lets you compare the values of an OpenCL kernel variable across work items and work groups.
The Debug Explorer view displays OpenCL-allocated objects and OpenCL/OpenGL shared contexts.

For example, clicking on Texture 2 in the view above brings up its properties:

Click on Vertex Buffer object VBO 1 to display its data, with a variety of available drop-down menu display and format options in the right-hand panel.
For example, double-click on Vertex Shader 1 under Shaders to bring up a Source Code view of its source file `tpVertexShader.glsl`. Or double-click on Depth buffer to bring up an Image view of the depth buffer:

You can manipulate an Image view with the image manipulation buttons on the CodeXL toolbar:
These allow you to select, zoom in, zoom out, pan, enable R/G/B/alpha channels, enable grayscale mode, enable color invert mode, original size, best fit, and rotate CCW/CW. Hovering over the image displays pixel-specific information (position and color) in the Image Information panel.

Alternatively, select the Data view tab of the depth buffer to display the buffer as raw spreadsheet data rather than as an image.

**Call Stack view**

The Call Stack view displays a combined C/C++/OpenCL call stack.

**Function Calls History view**

The Function Calls History view displays a log of OpenCL API calls.

Click on a function call to display call details in a Properties view.

**Debugged Process Events view**
The Debugged Process Events view displays process events.

**Memory view**

The Memory view summarizes memory use.

**Statistics view**

The Statistics view provides statistical information about the program. Select a tab to choose among options, such as Function Types:
or Function Calls:

![Function Calls](image)

**System information**

Select Tools >> System Information from the CodeXL toolbar to display system information.

![System Information](image)

Tabs let you select among several categories of information. The example above displays OpenCL device information, showing a GPU device and a CPU device.
Project settings

Select File >> Project Settings from the drop-down File menu to edit the settings for a project.

The project must be stopped to display project settings. CodeXL Help information provides more details about project settings.

Profile mode

CodeXL profile mode makes CodeXL a powerful performance analysis tool. It supports CPU and GPU profiling to provide program performance data. CodeXL profiling does not require modifications to your source code or project. Profiling does not require recompilation, except that CPU profiling requires compilation with debugging enabled. Profiling lets you find performance hotspots and issues, determine the top data transfer and kernel execution operations, and identify problems such as failed API calls and resource leaks. You can use profiling to improve application performance through proper synchronization, bottleneck elimination, and load balancing.

CodeXL provides several different flavors of profiling. CPU profiling modes let you assess program performance, use instruction-based sampling (IBS) or time-based sampling (TBS), or investigate branching, data access, instruction access, or L2 cache access. GPU profiling provides application trace and performance counter modes.
This document provides a quick introduction to CPU and GPU profiling. For further details, see the CodeXL Help information.

**CPU profiling**

To profile a program:

- First, click on the profiling mode taskbar button.
- Then use the Profile drop-down menu to select the desired flavor of profiling, such as Profile >> CPU: Assess Performance for CPU performance profiling.

Start the program and let it run as long as you wish; the bottom of the CodeXL window displays the elapsed clock time. When you wish, close the application (e.g., for the teapot example: click on the 'x' in the upper right corner of the teapot window). For CPU performance profiling, a CPU Profile Overview appears:

![CPU Profile Overview](image)

Click on a link on one of the bottom lines of the Profile Overview to display more details about modules:
or about the call chain:

The System Data tab gives detailed performance data. Drop-down menus give you many options for aggregating, viewing, and separating performance data.
GPU profiling

For GPU application trace profiling:

First click on the profiling mode taskbar button, then select Profile >> GPU: Application Trace from the Profile drop-down menu.

Run the program, then let it complete or terminate it. An Application Trace view appears with a timeline of information about program execution. The timeline shows the created OpenCL contexts and command queues and the relationships between them.

Hold down <Ctrl> and click+drag on a section of the timeline to select a subrange of the timeline, or click+drag to shift the timeline display left or right. You can also zoom in/out with the mouse wheel or with +/- keys. Selecting a small subrange lets you zoom in to see details about each event.

Hover over an event to display a pop-up with additional information.

This example shows a COPY_BUFFER_TO_IMAGE data transfer event at 7752.980 ms on the timeline. The popup provides detailed timing data.

The Summary tab provides several options for viewing profiling data: API, context, kernel, top 10 data transfer, top 10 kernel, warnings/errors. The example below shows a Top 10 Kernel Summary:
In addition to a list of warnings and errors, the Warning(s)/Error(s) summary includes a helpful list of best practice recommendations to improve program performance. The example below notes issues with blocking write calls and small global work size.

The Performance Counters view in a GPU Performance Counters profile gives kernel performance details, including global work size and time.

This mode collects performance counters from the GPU or APU for each kernel dispatched to the device. It also displays statistics from the shader compiler for each kernel dispatched. The performance counters and statistics can be used to discover kernel bottlenecks.
A pull-down on the right side lets you select OpenCL source (CL), intermediate language (IL), or instruction set architecture (ISA) code.

Profile Session Explorer view

Use the Profile Session Explorer view to switch between profiling sessions.

This view lists all profiling sessions for the current project. Double-click on on a session to display its data, or right-click to rename or delete it. You can import profiling data by right-clicking or by dragging/dropping a session data file to the Profile Session Explorer.

Kernel Analyzer

AMD APP KernelAnalyzer2 analyzes the performance of OpenCL kernels for AMD GPUs. It gives accurate kernel performance estimates and lets you view kernel compilation results and assembly code for multiple GPUs, without requiring actual GPU hardware.

Start KernelAnalyzer2

KernelAnalyzer2 is installed along with CodeXL.

You can launch KernelAnalyzer2 from within CodeXL:
Alternatively, you can launch KernelAnalyzer2 directly from the operating system. For Windows: Install CodeXL as described above, then:

Select from the Windows program menu:
All Programs >> AMD Developer Tools >> AMD APP KernelAnalyzer2 >> AMD App KernelAnalyzer2

For Linux: Install CodeXL as described above, then navigate to the KernelAnalyzer2 directory and invoke it:

```
$ cd /opt/AMD/AMDPKZ KernelAnalyzerV2/AMDPKZ*/x86/
# 32-bit
$ cd /opt/AMD/AMDPKZ KernelAnalyzerV2/AMDPKZ*/x86_64/
# 64-bit
$ ./AMDAPPKernelAnalyzer2
```

Alternatively, add the KernelAnalyzer2 directory to your PATH and then invoke it.

For either Windows or Linux, a KernelAnalyzer2 window appears:

The window contains three panels: a kernel source panel at top left, a kernel assembly code panel at top right, and a build output/statistics panel at bottom.

**Open kernel source**

Select File >> Open from the KernelAnalyzer2 toolbar, then navigate to a kernel source file.
For example, you can use OpenCL source `tpAdvectFieldScalar.cl` from the teapot example; it resides in Windows directory

\Program Files\AMD\AMD CodeXL\Examples\Teapot\res

or in Linux directory

/opt/AMD/CodeXL/bin/examples/Teapot/AMDTTeaPotLib/AMDTTeaPotLib/res
The source file appears in the source panel:

![Source File Screenshot]

You can also drag and drop a kernel source into the source panel.

**Build Options**

Select **Build >> Options** to bring up the Build Options window.

Its ASICs tab contains a list of devices by series:

![Build Options Screenshot]

Use the checkboxes to select or deselect an entire series, or click on a small triangle at left to expand. The R700 (HD 4000) series does not support the teapot example kernel, so it is deselected here:

![Build Options Details Screenshot]
For the `tpAdvectFieldScalar.cl` kernel, enter the following options:
- `-D GRID_NUM_CELLS_X=64`  
- `-D GRID_NUM_CELLS_Y=64`  
- `-D GRID_NUM_CELLS_Z=64`  
- `-D GRID_INV_SPACING=1.000000f`  
- `-D GRID_SPACING=1.000000f`  
- `-D GRID_SHIFT_X=6`  
- `-D GRID_SHIFT_Y=6`  
- `-D GRID_SHIFT_Z=6`  
- `-D GRID_STRIDE_Y=64`  
- `-D GRID_STRIDE_SHIFT_Y=6`  
- `-D GRID_STRIDE_Z=4096`  
- `-D GRID_STRIDE_SHIFT_Z=12`

Here `path_to_example_src` should be e.g. `\Program Files\AMD\AMD CodeXL\Examples\Teapot\` for Windows or `/opt/AMD/CodeXL/bin/examples/Teapot/AMDTTeaPotLib/AMDTTeaPotLib/` for Linux.

Hit OK to exit from the Build Options window.

**Build the kernel**

After you set build options:

```
Select Build >> Build to build the kernel.
```

Compilation output appears in the Output tab. The example below shows successful builds (no warnings or errors) for 17 of 17 devices. The right panel displays a drop-down list of kernel names at the top, with tabs below to display the intermediate language (IL) or instruction set architecture (ISA) code for each device. Click on a tab to display, or double-click to display in a new window. Right-click in the code pane and select Save as to save the IL or ISA code as a text file.
You can export the output as a binary file with File >> Export Binaries.

The statistics tab gives detailed statistics for each device.

| Device | Statistics | ThreadsPerBlock | WarpsPerBlock | MaxBlocks | L1Size | L2Size | L3Size | L1Cache | L2Cache | L3Cache | RegsActive | RegsActiveX | RegsActiveY | RegsActiveZ | RegsActiveZ
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Float</td>
<td>216</td>
<td>64</td>
<td>256</td>
<td>16</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4096</td>
<td>4096</td>
<td>4096</td>
<td>4096</td>
<td>4096</td>
<td></td>
</tr>
<tr>
<td>Integer</td>
<td>216</td>
<td>64</td>
<td>256</td>
<td>16</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4096</td>
<td>4096</td>
<td>4096</td>
<td>4096</td>
<td>4096</td>
<td></td>
</tr>
<tr>
<td>Complex</td>
<td>216</td>
<td>64</td>
<td>256</td>
<td>16</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4096</td>
<td>4096</td>
<td>4096</td>
<td>4096</td>
<td>4096</td>
<td></td>
</tr>
</tbody>
</table>
Known issues

For a list of known CodeXL issues, check the release notes on the CodeXL web page and the AMD Developer Tools CodeXL forum:

devgurus.amd.com/groups/codexl

Support

AMD general developer support and training website:

developer.amd.com/Support/Pages/default.aspx

GPU developer tools website:

developer.amd.com/tools/pages/default.aspx

OpenCL Zone:

developer.amd.com/Resources/hc/OpenCLZone/programming/Pages/default.aspx

*AMD Accelerated Parallel Processing OpenCL Programming Guide*: 

For GPU development issues relating to other AMD tools, see the AMD GPU Developer Tools Forum:

devgurus.amd.com/community/gpu_developer_tools

To report a specific problem or request help with AMD CodeXL, visit the CodeXL Forum at:

devgurus.amd.com/groups/codexl