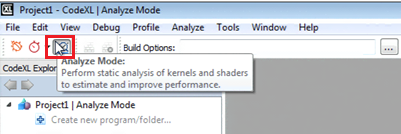
## Static Analyzer Tutorial

### Switching to Analyze mode

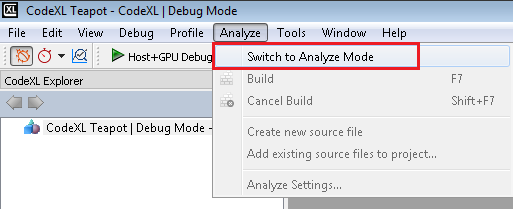
Option 1- Analyze mode button:

Click on the Analyze Mode button in the CodeXL Mode toolbar:



Option 2- Main menu:

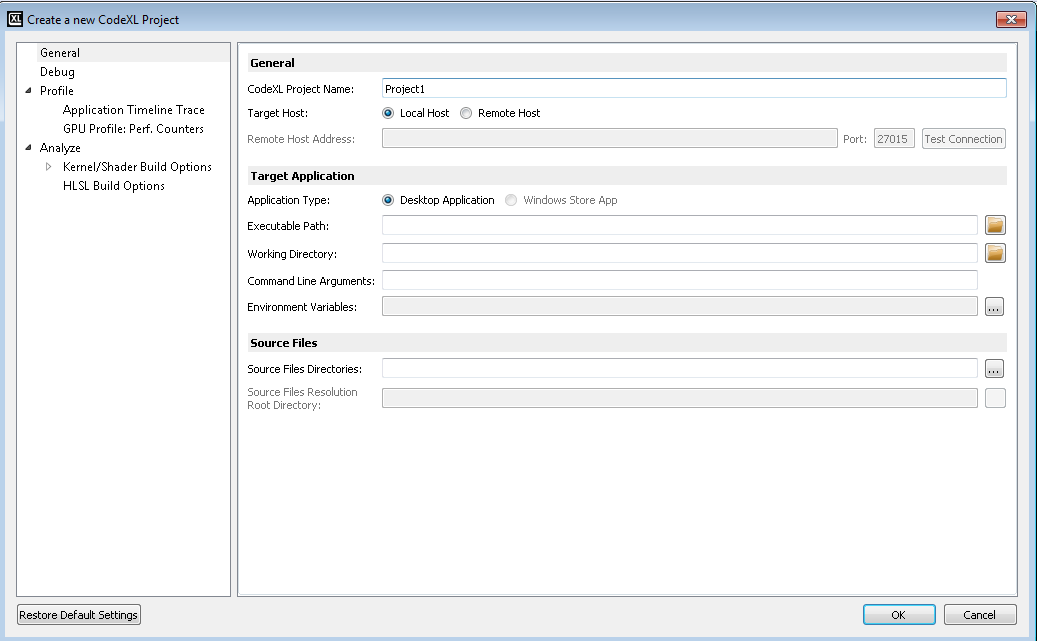
Open the Analyze menu from menu bar and select the ‘Switch to Analyze Mode’ command:



After you switch to Analyze mode, you can also create a new project, open a previously saved project, or load the Teapot or Matrix Multiply samples.

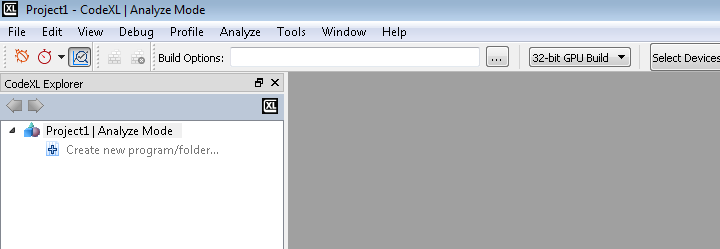
### Creating a new project for Analysis

Click on the “File->Create Project”, or use the Ctrl+N shortcut. The following CodeXL Project Settings dialog will appear:



Rename the project, and click on the OK.

After the new project has been created, the CodeXL Analyzer Explorer Tree should appear in the left pane:



### Working with the new CodeXL Analyzer Explorer Tree

If you are familiar with the former versions of the Analyzer, you probably noticed that the tree has a different structure than the one used in previous versions. Let’s examine the structure of the new CodeXL Analyzer Explorer:

1. **Programs and Folders:** before describing how to technically create Programs and Folders, let’s first discuss what those objects are, and why they can be useful.
2. **Programs (OpenGL, Vulkan):**

As of version 2.0, CodeXL can compile and link together multiple source files for OpenGL and Vulkan. This is especially important when different shaders have mutual impact on one another’s ISA and performance statistics. To provide that type of support, CodeXL Analyzer introduced the concept of a Program. There are two types of Programs in CodeXL 2.0:

* Rendering Programs
* Compute Programs

A Rendering Program represents a graphics pipeline, and can have a single shader attached to each of its stages:

* Vertex
* Tessellation Control
* Tessellation Evaluation
* Geometry
* Fragment

A Compute Program represents a compute pipeline, and can have a single compute shader attached to its single stage.

When you build a program that has multiple shaders attached to it, all shaders are being compiled and linked together. This way, you get more accurate ISA and performance statistics than those generated using previous versions of CodeXL.

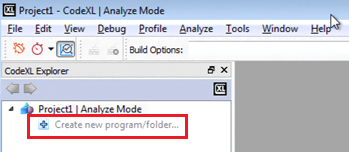
1. **Folders (OpenCL, DirectX):**

Folders are logical containers of source files. When you build a folder that has multiple source files attached to it, the source files are simply being built one after the other. Unlike programs, there is no kind of interdependency between the source files in a given folder: when a folder is being built, each source file is being compiled independently. Folders can be used to organize the project, by serving as a logical separator. They can also be used to ease the process of comparing build results, since now the build results are being maintained per-folder: you can create two different Folders, each containing the same source files, but have a different configuration (for example, create two DirectX Folders, each with a different shader model). After building the two Folders, you can toggle between the performance statistics of the two Folders to see the differences.

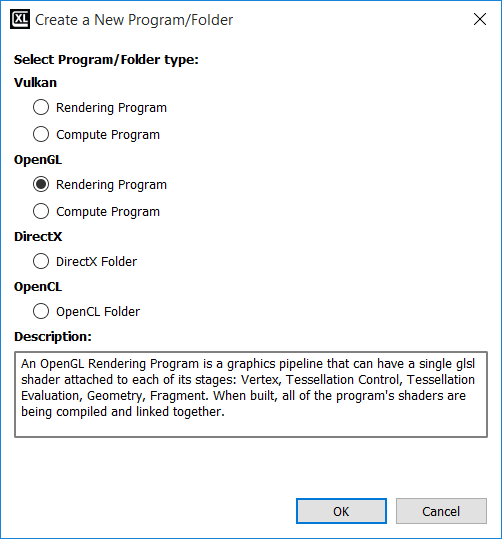
You may ask yourself why CodeXL does not support the concept of DirectX Programs, just like it does for OpenGL and Vulkan. This is a good point. Supporting DirectX Programs is at a high priority in the Analyzer’s roadmap, and we will do our best to add that feature in the upcoming versions of the product.

**Creating a new Program or Folder**

To create a new Program or a Folder, double-click on the “Create new program/folder” item in CodeXL Analyzer Explorer Tree:

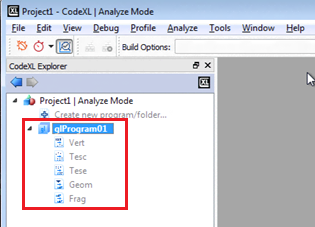


Then, the following dialog would pop-up:



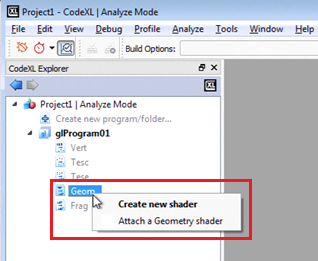
Select the Program/Folder type of choice, and click OK.

Then, the empty Program/Folder would appear in the Explorer Tree. For Example, if you choose an OpenGL Rendering Program, you will see an empty OpenGL Rendering Program created:



### Working with Programs

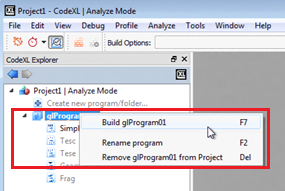
After creating a new program, you will see that it contains an empty placeholder for every pipeline stage. Right-click on any stage to add an existing shader or create a new one:



Note: You can also double-click on a stage to create a new shader and automatically attach it to that Program’s stage.

As you can see in the above screenshot, we attached SimpleVertexShader.vs as the vertex shader to our OpenGL Rendering Program, and it was also automatically added to the Source Files pool. We can now drag SimpleVertexShader.vs from the Source Files pool and drop it on the stage node of any Program that we may add to the project, to reuse SimpleVertexShader.vs (there is no dependency in the build process between different Programs).

To build the program, right-click on it and select the Build option, or use the F7 shortcut:

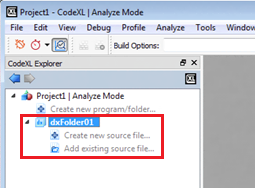


You can also select the Program and manually click on the Build button in the Analyzer toolbar:

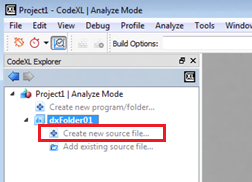


### Working with Folders

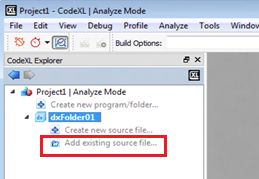
After creating a new OpenCL or DirectX Folder, an empty Folder would be listed in the Explorer Tree:



To create a new source file, and automatically add it to the Folder, double-click on the “Create new source file item…” item of the folder:

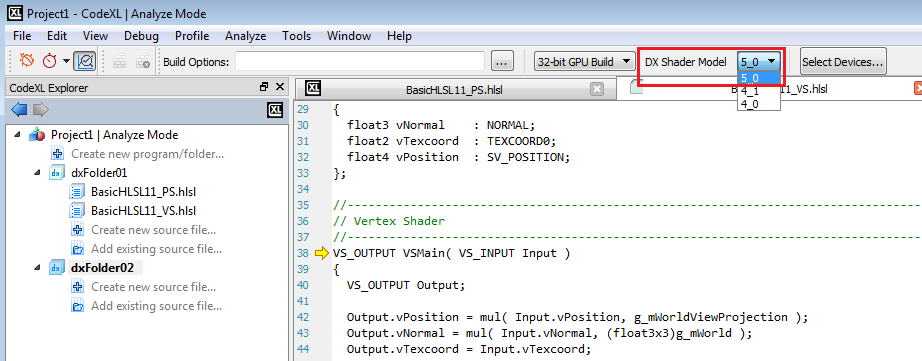


To add an existing source file, and automatically add it to the Folder, double-click on the “Add existing source file item…” item of the folder:



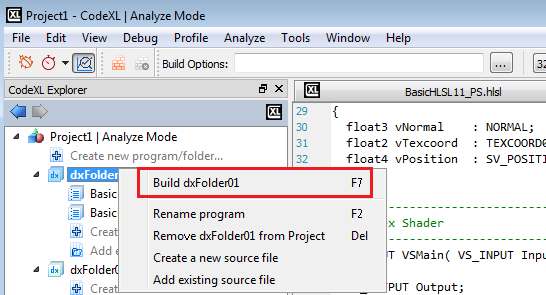
To configure the build properties of a source file under a specific Folder, click on that source file and use the Analyzer toolbar’s Type and Entry point drop-down lists. The first sets the type of the shader and the latter specifies the specific target shader (among the shaders in the source file). This configuration is Folder-specific. That is, the same source file can be set with different properties under different Folders. CodeXL will remember those configurations for you.

To configure the build properties of the Folder, click on the Folder and adjust the enabled items in the Analyzer toolbar. For CodeXL 2.0, this is only relevant to the DX Shader Model property of DX Folders:

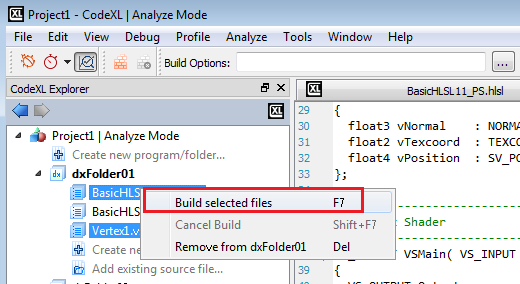


Once set, the DX Shader Model value will hold for all the shaders in the selected Folder. For example, if you choose 5\_0 as the DX Shader Model, any D3D vertex shader in that Folder will be compiled using shader model vs\_5\_0.

To build the whole Folder, right-click on it and select the Build item:

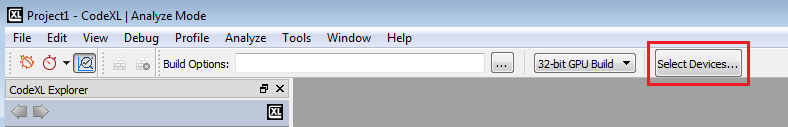


Unlike the case with Programs, Folders are more flexible as they allow you to build selected source files, without being required to build the whole Folder. To build selected source files, click on the selected source files under the program, while holding the Ctrl key. Then, right-click on one of the selected files and select the build option:

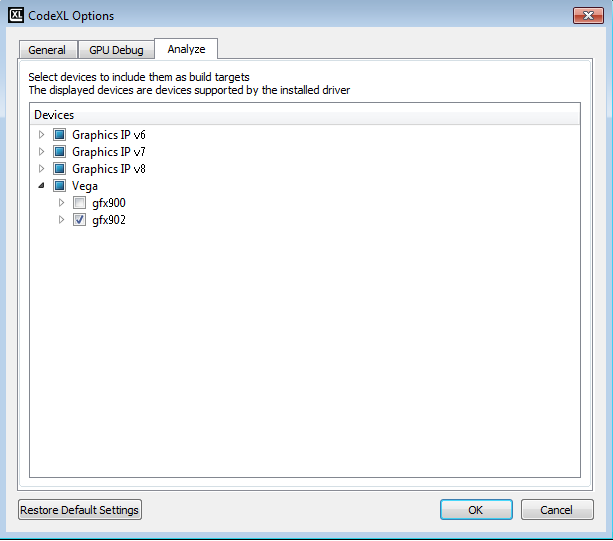


### Selecting target devices

CodeXL Analyzer can target a variety of devices, independent of the device that is physically installed on your system. To select the target devices, for which the build would be performed, first click on the Select Devices button in the Analyzer toolbar:

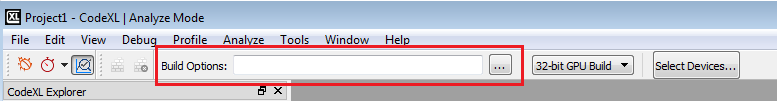


Then, the CodeXL Options dialog would pop-pup with its Analyze tab activated. The devices are grouped by generations. You can use the check boxes to select and remove devices:



### Build Options- Defining OpenCL and DirectX build options

In the Static Analyze toolbar, you can define specific OpenCL or HLSL build options:



The Build Options box is a place to set compiler build flags such as –x clc++ or –o3. Any compiler build flag can be placed in this box.

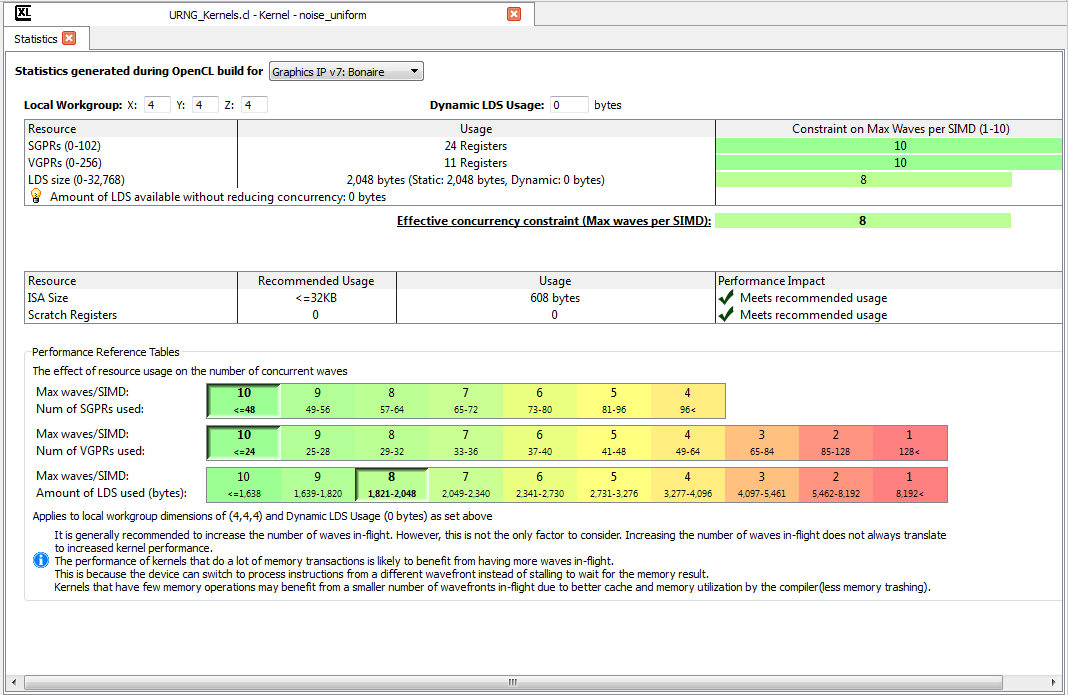
You can set the build options by typing the options directly in the designated text box or by using the OpenCL/HLSL Build Options dialog.

### The Statistics and Analysis Data

Kernels Statistic view

The kernel statistics tab gives detailed statistics for the selected kernel for each target device.   
To open the statistics tab, expand the desired kernel in the project tree, and double-click the Statistics node:

Statistics page for devices GCN devices:

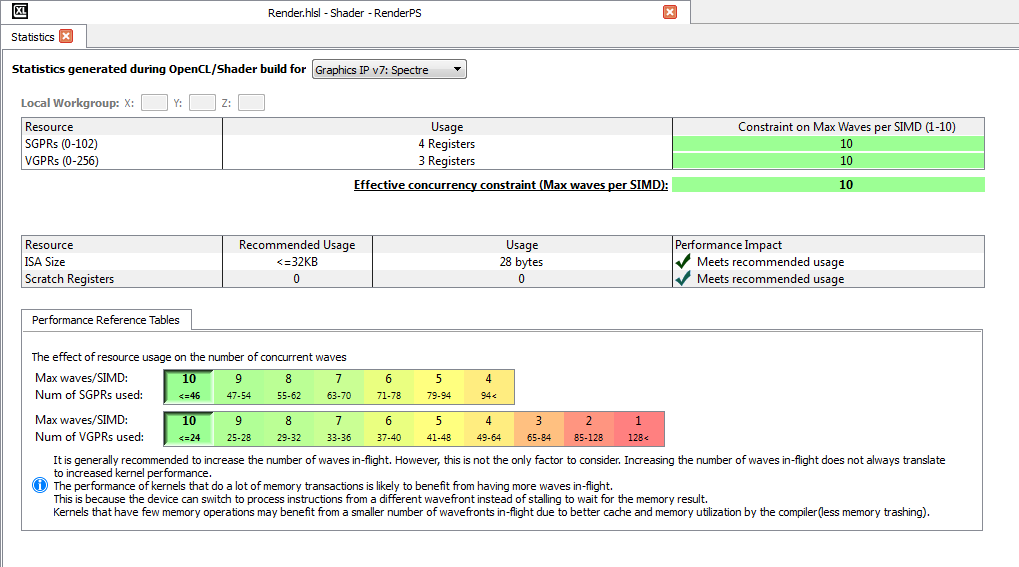


This view is focused on giving the programmer the wave constraints based on the SGPRs, VGPRs and LDS size.

Shaders Statistic view

The shader statistics tab gives detailed statistics for the selected shader for each target device.   
To open the statistics tab, expand the desired shader in the project tree, and double-click the Statistics node:  
**Note:** the statistics will be available only for V6 generation and later.

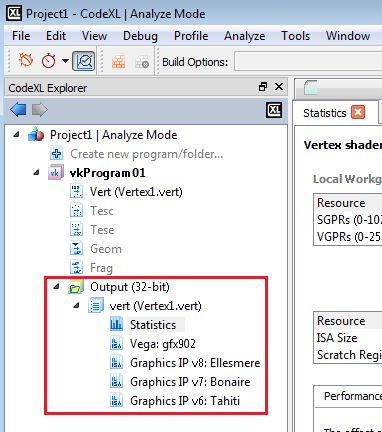
Shaders Statistics page:



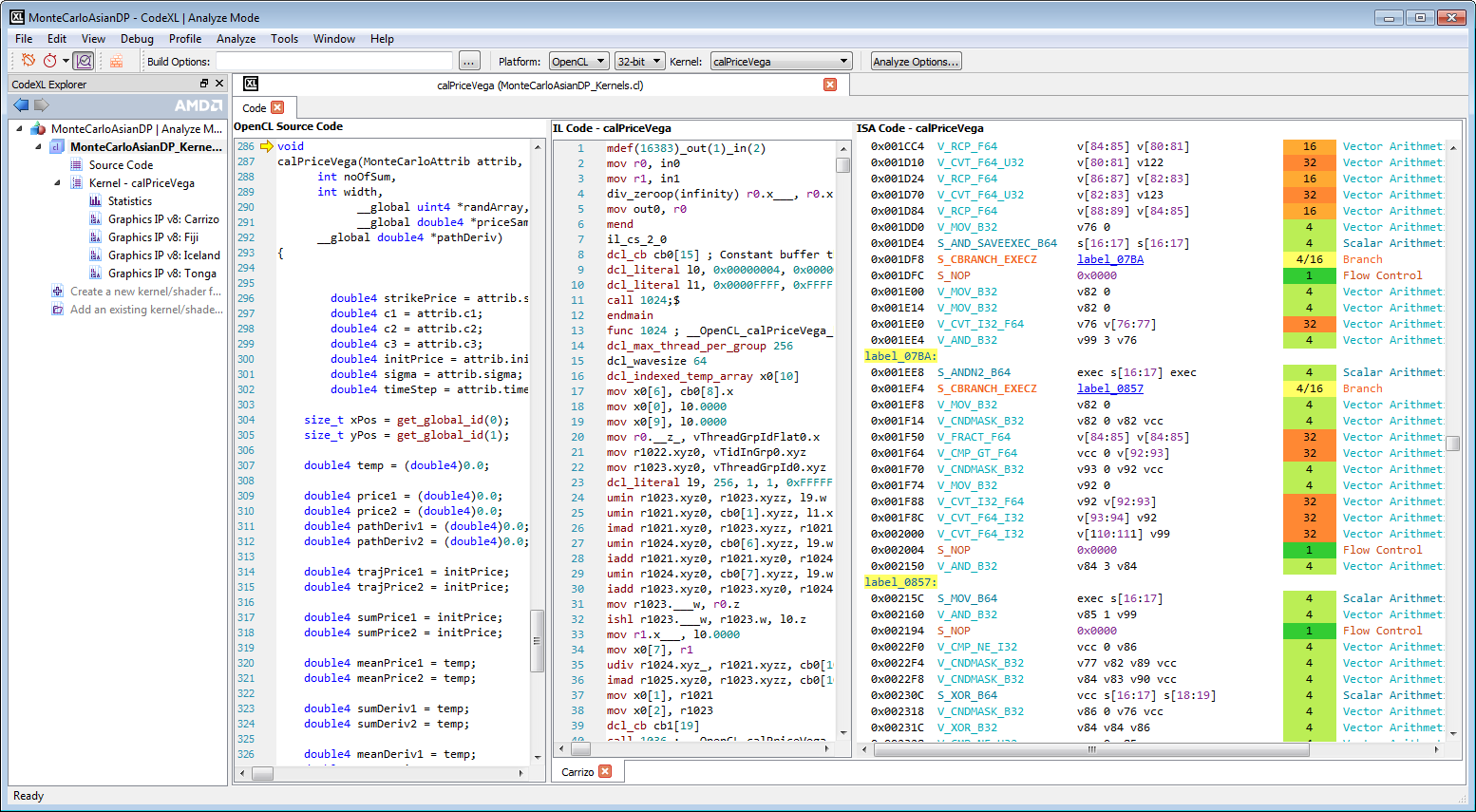
This view is focused on giving the programmer the wave constraints based on the SGPRs and VGPRs.

### Viewing compilation output: IL and ISA

The performance statistics tab will be opened automatically when the build process is over. To view the compilation output, double click the node of the desired ASIC in the explorer tree, under the Program/Folder and configuration (32-bit or 64-bit):

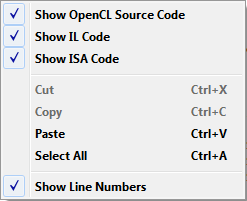


This action will open a tab containing the source code, the AMD IL and the ISA:



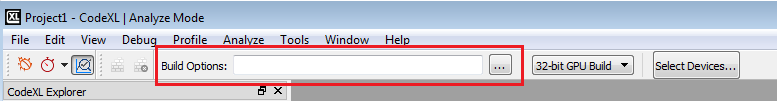
Opening several build results of different devices for the same kernel/shader will open different IL/ISA tabs in the same “Source Code/IL/ISA” view as can be seen in the above image.

The context menu enables showing/hiding different parts of the view, enables showing/hiding line numbers, and enables different edit actions depending on the selection of where the context menu was opened and whether the view section is editable or not.



Build Options- Defining compilation options

In the Static Analyze toolbar, there is a space where you can define specific kernel/shader build options:



Build Options Dialog

This dialog will help you choose the correct kernel/shader build options for you and hopefully will prevent making spelling mistakes while typing the options manually.

To open the dialog, press The  Button. The dialog will be opened.   
For OpenCL build options - you can browse between the ”General & Optimization” tab and the ”Other” tab to view all the available options.   
For DirectX build options – choose the “HLSL Build Options” tab.

Once you choose an option, the option text is displayed in the text box marked ”.. Build Command Line” that appears below.   
This string will also appear in the menu bar after you click the OK button.

Typing the option in the text box will also mark it in the appropriate area in the dialog, or select it in the appropriate drop box.

