Debugging and Analyzing Programs using the Intercept Layer for OpenCL™ Applications

Ben Ashbaugh

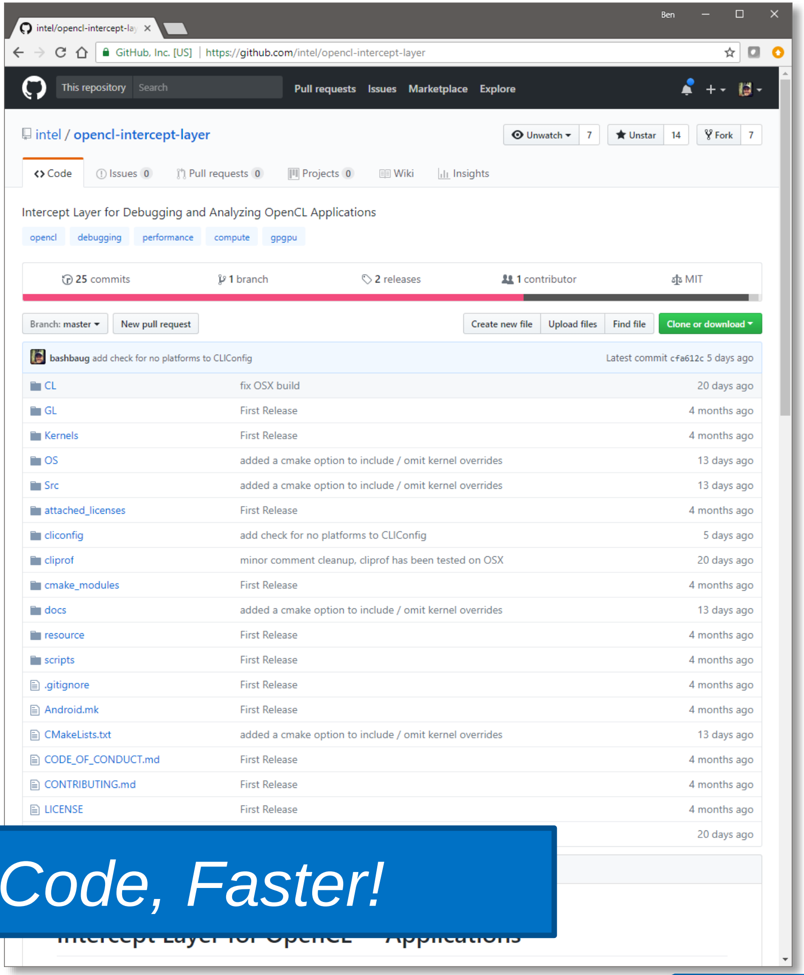
IWOCL 2018

https://github.com/intel/opencl-intercept-layer

Why am I here?

Intercept Layer for OpenCL Applications

- Debug and Analyze OpenCL Applications
- Open Source, Permissive License
- Works with Any* OpenCL Implementation
- Requires No Application Modifications
- Thin, Fast, Easy to Install / Uninstall
- Community Contributions are Welcome and Encouraged!



intel / opencl-intercept-layer

Intercept Layer for Debugging and Analyzing OpenCL Applications

opencl debugging performance compute gpgpu

25 commits | 1 branch | 2 releases | 1 contributor | MIT

bashbaug add check for no platforms to CLIConfig — Latest commit cfa612c 5 days ago

CL	fix OSX build	20 days ago
GL	First Release	4 months ago
Kernels	First Release	4 months ago
OS	added a cmake option to include / omit kernel overrides	13 days ago
Src	added a cmake option to include / omit kernel overrides	13 days ago
attached_licenses	First Release	4 months ago
cliconfig	add check for no platforms to CLIConfig	5 days ago
cliprof	minor comment cleanup, cliprof has been tested on OSX	20 days ago
cmake_modules	First Release	4 months ago
docs	added a cmake option to include / omit kernel overrides	13 days ago
resource	First Release	4 months ago
scripts	First Release	4 months ago
.gitignore	First Release	4 months ago
Android.mk	First Release	4 months ago
CMakeLists.txt	added a cmake option to include / omit kernel overrides	13 days ago
CODE_OF_CONDUCT.md	First Release	4 months ago
CONTRIBUTING.md	First Release	4 months ago
LICENSE	First Release	4 months ago
		20 days ago

Develop Fast OpenCL Code, Faster!

Agenda

History

How it Works

What it Can Do

Implementation Details

Possible Next Steps

Wrap Up

History

(2009-Present)

Initial Requests:

I’m debugging an application. Can you modify the driver to print the OpenCL APIs that are called?

Yeah, no problem.

One week later…

Can you print the API arguments too?

Sure, I think we can add that.

More requests:

I’m debugging the GPU compiler. Can you modify the driver to dump OpenCL kernels to a file?

Yeah, that’s not too hard.

Great.

Can you also make it work for the CPU OpenCL implementation?

I’m not sure – I think so?

Fantastic. Can you make it work for [third party competitor]?

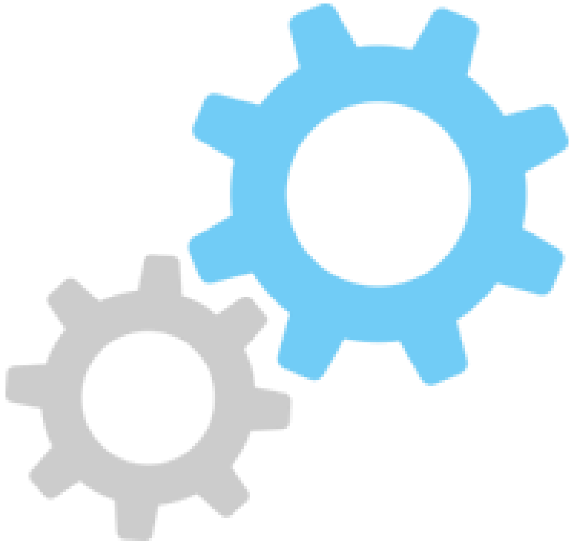
Meanwhile:

Our Driver Team was also adding instrumentation:

- Flush or Finish After Enqueue
- Assert on OpenCL Errors
- Timing API Calls
- More …

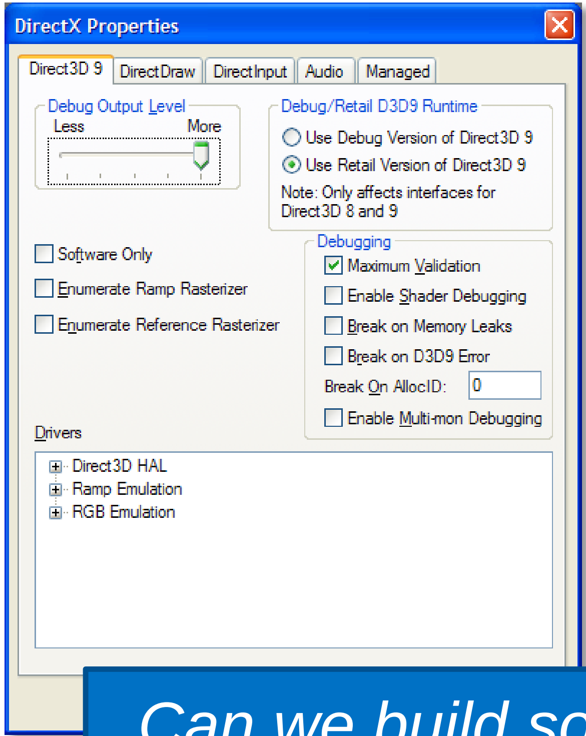
But:

- Required driver modifications!



Is there a better way to add these capabilities?

Prior Work from Graphics APIs:


DirectX Properties

Direct3D 9 | DirectDraw | DirectInput | Audio | Managed

Debug Output Level
Less — More

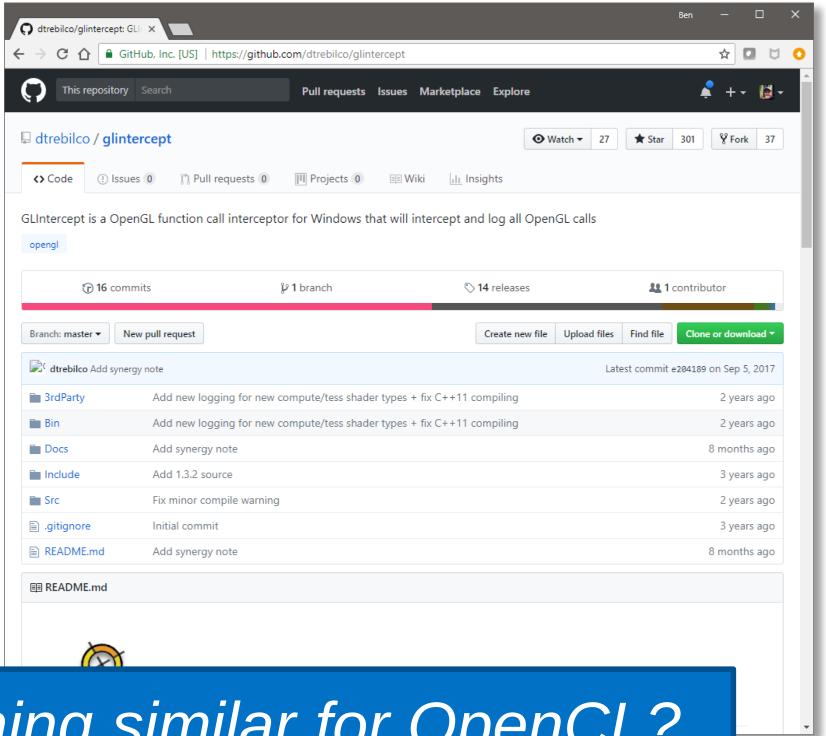
Debug/Retail D3D9 Runtime
- ○ Use Debug Version of Direct3D 9
- ● Use Retail Version of Direct3D 9

Note: Only affects interfaces for Direct3D 8 and 9

- ☐ Software Only
- ☐ Enumerate Ramp Rasterizer
- ☐ Enumerate Reference Rasterizer

Debugging
- ☑ Maximum Validation
- ☐ Enable Shader Debugging
- ☐ Break on Memory Leaks
- ☐ Break on D3D9 Error
- Break On AllocID: 0
- ☐ Enable Multi-mon Debugging

Drivers
- Direct3D HAL
- Ramp Emulation
- RGB Emulation


dtrebilco/glintercept: GL

GitHub, Inc. [US] | https://github.com/dtrebilco/glintercept

dtrebilco / glintercept

Watch 27 | Star 301 | Fork 37

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Insights

GLIntercept is a OpenGL function call interceptor for Windows that will intercept and log all OpenGL calls

opengl

16 commits | 1 branch | 14 releases | 1 contributor

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

dtrebilco Add synergy note — Latest commit e204189 on Sep 5, 2017

3rdParty	Add new logging for new compute/tess shader types + fix C++11 compiling	2 years ago
Bin	Add new logging for new compute/tess shader types + fix C++11 compiling	2 years ago
Docs	Add synergy note	8 months ago
Include	Add 1.3.2 source	3 years ago
Src	Fix minor compile warning	2 years ago
.gitignore	Initial commit	3 years ago
README.md	Add synergy note	8 months ago

README.md

Can we build something similar for OpenCL?

How It Works

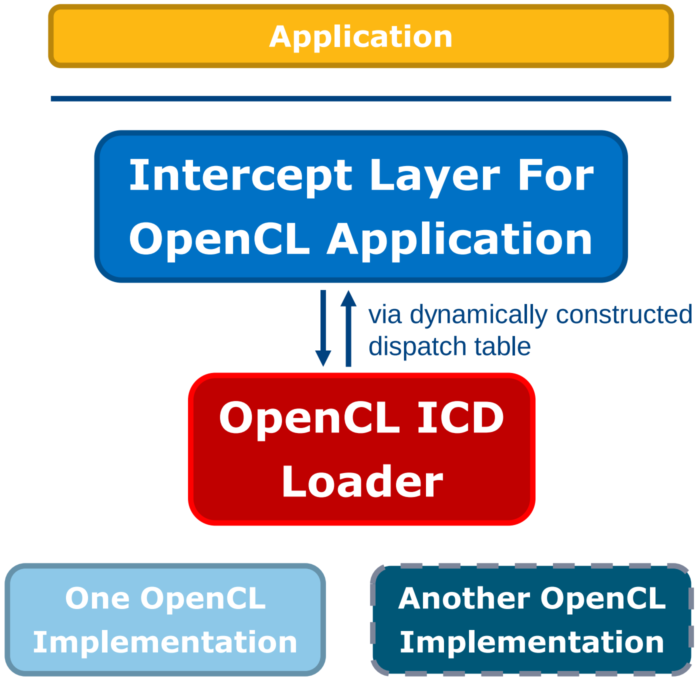
Intercept Layer for OpenCL Applications

Architecture: How it Works*:

- Inserts between Application and OpenCL ICD Loader
- Constructs Dispatch Table During Initialization
- Passes Through API Calls… or not!

Philosophies:

- Focus on Features that Solve Problems
  - For OpenCL Implementers
  - For OpenCL Developers
- Support Any OpenCL Device on Any Platform
- Be Invisible By Default


Application
Intercept Layer For OpenCL Application
via dynamically constructed dispatch table
OpenCL ICD Loader
One OpenCL Implementation
Another OpenCL Implementation

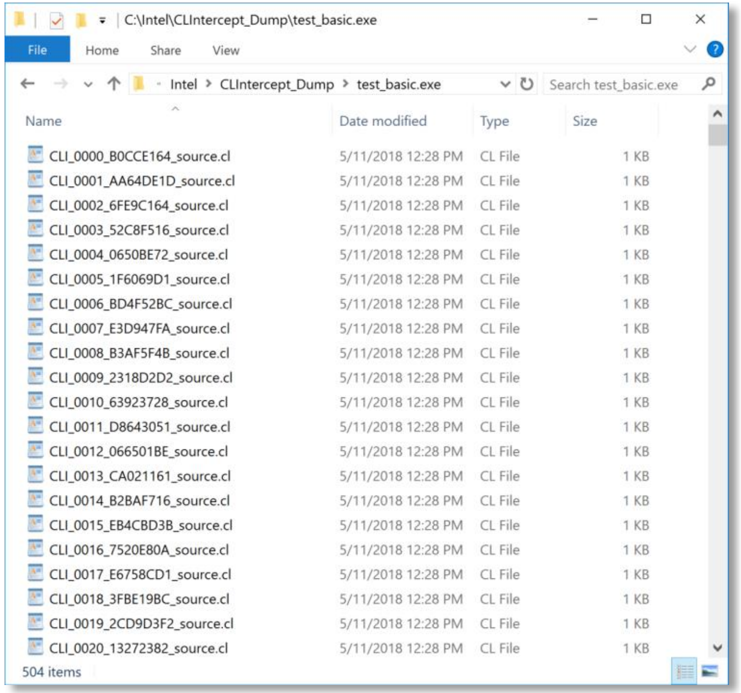
* Typical usage on Windows and Linux, OSX is a little different.

What It Can Do - Examples

Call and Error Logging

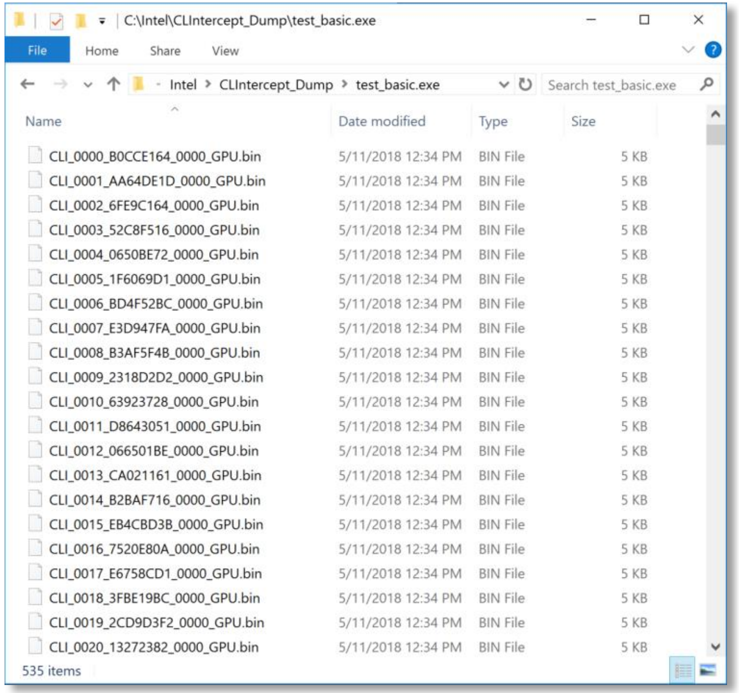
```
>>>> clGetPlatformIDs
<<<< clGetPlatformIDs
>>>> clGetPlatformIDs
<<<< clGetPlatformIDs
>>>> clGetDeviceIDs: platform = [ NVIDIA CUDA ], device_type = CL_DEVICE_TYPE_ALL (FFFFFFFF)
<<<< clGetDeviceIDs
>>>> clGetDeviceIDs: platform = [ Intel(R) OpenCL ], device_type = CL_DEVICE_TYPE_ALL (FFFFFFFF)
<<<< clGetDeviceIDs
>>>> clCreateContextFromType: properties = [ CL_CONTEXT_PLATFORM = Intel(R) OpenCL ], ...
ERROR! clCreateContextFromType returned CL_DEVICE_NOT_FOUND (-1)
<<<< clCreateContextFromType: returned 00000000
>>>> clCreateContextFromType: properties = [ CL_CONTEXT_PLATFORM = Intel(R) OpenCL ], ...
<<<< clCreateContextFromType: returned 00E97068
>>>> clGetContextInfo: param_name = CL_CONTEXT_DEVICES (00001081)
<<<< clGetContextInfo
>>>> clGetContextInfo: param_name = CL_CONTEXT_DEVICES (00001081)
<<<< clGetContextInfo
>>>> clCreateCommandQueue: device = [ Intel(R) Core(TM) i7-2600K CPU @ 3.40GHz (CL_DEVICE_TYPE_CPU) ]
<<<< clCreateCommandQueue: returned 05B038F8
>>>> clGetContextInfo: param_name = CL_CONTEXT_DEVICES (00001081)
<<<< clGetContextInfo
>>>> clGetContextInfo: param_name = CL_CONTEXT_DEVICES (00001081)
<<<< clGetContextInfo
>>>> clCreateProgramWithSource: context = 00E97068, count = 1
<<<< clCreateProgramWithSource: returned 04572EA8, program number = 0000
```

Dumping Program Source (and Binaries!)


C:\Intel\CLIntercept_Dump\test_basic.exe
File
Home
Share
View
Intel > CLIntercept_Dump > test_basic.exe
Search test_basic.exe

Name	Date modified	Type	Size
CLI_0000_B0CCE164_source.cl	5/11/2018 12:28 PM	CL File	1 KB
CLI_0001_AA64DE1D_source.cl	5/11/2018 12:28 PM	CL File	1 KB
CLI_0002_6FE9C164_source.cl	5/11/2018 12:28 PM	CL File	1 KB
CLI_0003_52C8F516_source.cl	5/11/2018 12:28 PM	CL File	1 KB
CLI_0004_0650BE72_source.cl	5/11/2018 12:28 PM	CL File	1 KB
CLI_0005_1F6069D1_source.cl	5/11/2018 12:28 PM	CL File	1 KB
CLI_0006_BD4F52BC_source.cl	5/11/2018 12:28 PM	CL File	1 KB
CLI_0007_E3D947FA_source.cl	5/11/2018 12:28 PM	CL File	1 KB
CLI_0008_B3AF5F4B_source.cl	5/11/2018 12:28 PM	CL File	1 KB
CLI_0009_2318D2D2_source.cl	5/11/2018 12:28 PM	CL File	1 KB
CLI_0010_63923728_source.cl	5/11/2018 12:28 PM	CL File	1 KB
CLI_0011_D8643051_source.cl	5/11/2018 12:28 PM	CL File	1 KB
CLI_0012_066501BE_source.cl	5/11/2018 12:28 PM	CL File	1 KB
CLI_0013_CA021161_source.cl	5/11/2018 12:28 PM	CL File	1 KB
CLI_0014_B2BAF716_source.cl	5/11/2018 12:28 PM	CL File	1 KB
CLI_0015_EB4CBD3B_source.cl	5/11/2018 12:28 PM	CL File	1 KB
CLI_0016_7520E80A_source.cl	5/11/2018 12:28 PM	CL File	1 KB
CLI_0017_E6758CD1_source.cl	5/11/2018 12:28 PM	CL File	1 KB
CLI_0018_3FBE19BC_source.cl	5/11/2018 12:28 PM	CL File	1 KB
CLI_0019_2CD9D3F2_source.cl	5/11/2018 12:28 PM	CL File	1 KB
CLI_0020_13272382_source.cl	5/11/2018 12:28 PM	CL File	1 KB

504 items


C:\Intel\CLIntercept_Dump\test_basic.exe
File
Home
Share
View
Intel > CLIntercept_Dump > test_basic.exe
Search test_basic.exe

Name	Date modified	Type	Size
CLI_0000_B0CCE164_0000_GPU.bin	5/11/2018 12:34 PM	BIN File	5 KB
CLI_0001_AA64DE1D_0000_GPU.bin	5/11/2018 12:34 PM	BIN File	5 KB
CLI_0002_6FE9C164_0000_GPU.bin	5/11/2018 12:34 PM	BIN File	5 KB
CLI_0003_52C8F516_0000_GPU.bin	5/11/2018 12:34 PM	BIN File	5 KB
CLI_0004_0650BE72_0000_GPU.bin	5/11/2018 12:34 PM	BIN File	5 KB
CLI_0005_1F6069D1_0000_GPU.bin	5/11/2018 12:34 PM	BIN File	5 KB
CLI_0006_BD4F52BC_0000_GPU.bin	5/11/2018 12:34 PM	BIN File	5 KB
CLI_0007_E3D947FA_0000_GPU.bin	5/11/2018 12:34 PM	BIN File	5 KB
CLI_0008_B3AF5F4B_0000_GPU.bin	5/11/2018 12:34 PM	BIN File	5 KB
CLI_0009_2318D2D2_0000_GPU.bin	5/11/2018 12:34 PM	BIN File	5 KB
CLI_0010_63923728_0000_GPU.bin	5/11/2018 12:34 PM	BIN File	5 KB
CLI_0011_D8643051_0000_GPU.bin	5/11/2018 12:34 PM	BIN File	5 KB
CLI_0012_066501BE_0000_GPU.bin	5/11/2018 12:34 PM	BIN File	5 KB
CLI_0013_CA021161_0000_GPU.bin	5/11/2018 12:34 PM	BIN File	5 KB
CLI_0014_B2BAF716_0000_GPU.bin	5/11/2018 12:34 PM	BIN File	5 KB
CLI_0015_EB4CBD3B_0000_GPU.bin	5/11/2018 12:34 PM	BIN File	5 KB
CLI_0016_7520E80A_0000_GPU.bin	5/11/2018 12:34 PM	BIN File	5 KB
CLI_0017_E6758CD1_0000_GPU.bin	5/11/2018 12:34 PM	BIN File	5 KB
CLI_0018_3FBE19BC_0000_GPU.bin	5/11/2018 12:34 PM	BIN File	5 KB
CLI_0019_2CD9D3F2_0000_GPU.bin	5/11/2018 12:34 PM	BIN File	5 KB
CLI_0020_13272382_0000_GPU.bin	5/11/2018 12:34 PM	BIN File	5 KB

535 items

Can also Modify and/or Inject Modified Program Source or Binaries!

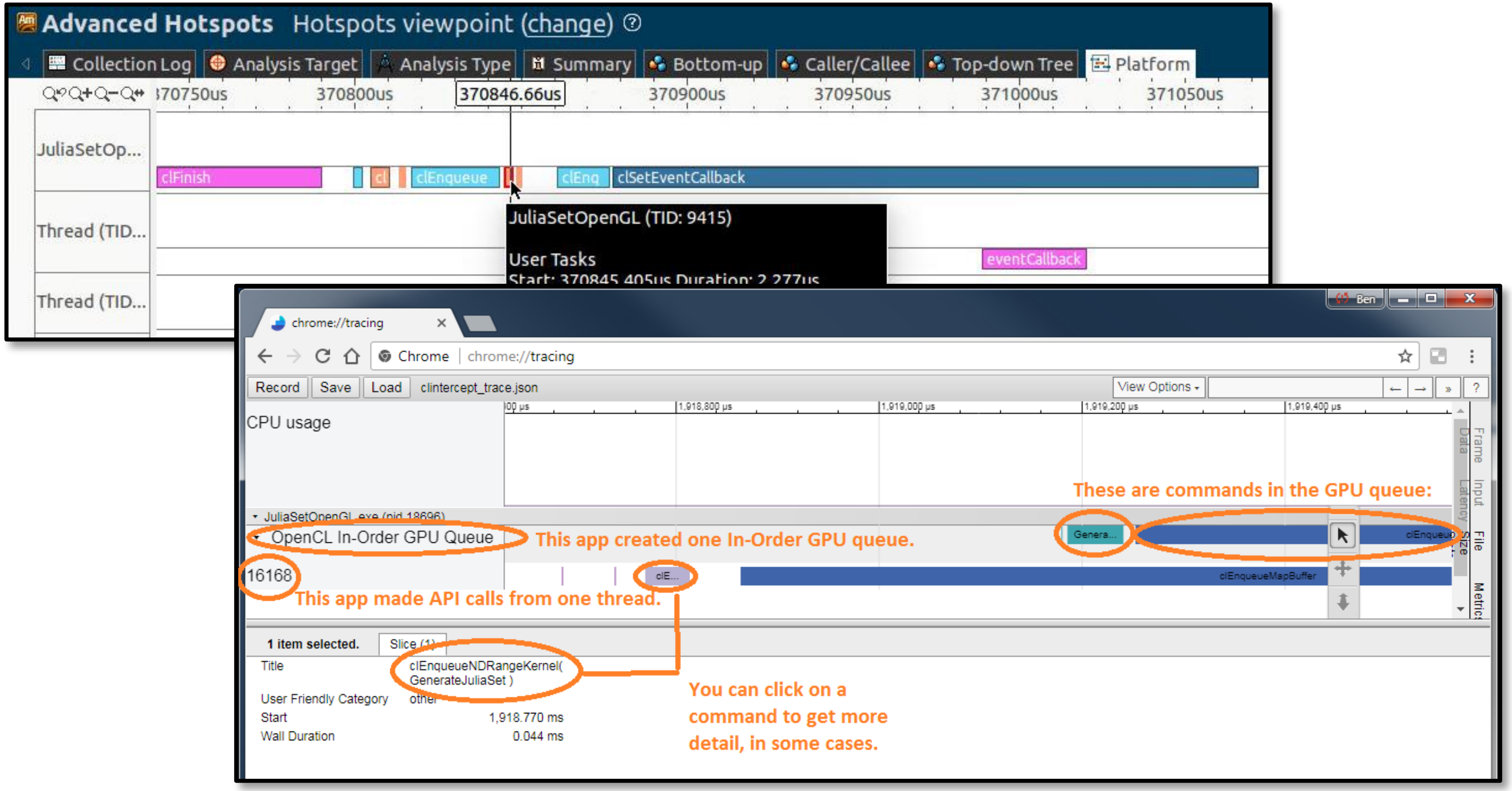
Host API Performance Timing

```
Host Performance Timing Results:

                            Function Name, Calls, Average (ns),   Min (ns),   Max (ns)
                           clBuildProgram,     3,    711065926,   22172160, 1634864192
                           clCreateBuffer,    23,      2234125,       2113,   36218573
                     clCreateCommandQueue,     1,        25054,      25054,      25054
                          clCreateContext,     1,    123618277,  123618277,  123618277
                          clCreateImage2D,     2,      8600269,    4682137,   12518402
                           clCreateKernel,     6,         7898,       2113,      14489
                clCreateProgramWithSource,     3,        24551,       4829,      51617
   clEnqueueNDRangeKernel( AdvancePaths ), 18036,        36967,      22941,   61064301
           clEnqueueNDRangeKernel( Init ),     1,      7529273,    7529273,    7529273
clEnqueueNDRangeKernel( InitFrameBuffer ),     1,      1095145,    1095145,    1095145
      clEnqueueNDRangeKernel( Intersect ), 18036,        25952,      15998,   24253177
        clEnqueueNDRangeKernel( Sampler ), 18036,        29856,      15696,     218847
                      clEnqueueReadBuffer,  2288,      3758695,     123158,   10236648
                                 clFinish,     2,      4723341,     717519,    8729163
                                  clFlush, 18036,        31018,      21432,     374003
                           clGetDeviceIDs,     4,         1811,        301,       5735
...
```

Device Command Performance Timing

```
Device Performance Timing Results:

Total Time (ns): 123904875200

       Function Name,  Calls,   Time (ns), Time (%), Average (ns), Min (ns), Max (ns)
        AdvancePaths,  18036, 28203368032,   22.76%,      1563726,  1388096,  1761472
                Init,      1,     8600000,    0.01%,      8600000,  8600000,  8600000
     InitFrameBuffer,      1,      155712,    0.00%,       155712,   155712,   155712
           Intersect,  18036, 79765237056,   64.38%,      4422556,  3248832,  5297600
             Sampler,  18036, 14307721664,   11.55%,       793286,    75712,  1182400
 clEnqueueReadBuffer,   2288,  1619792736,    1.31%,       707951,    39904,  4220992
```

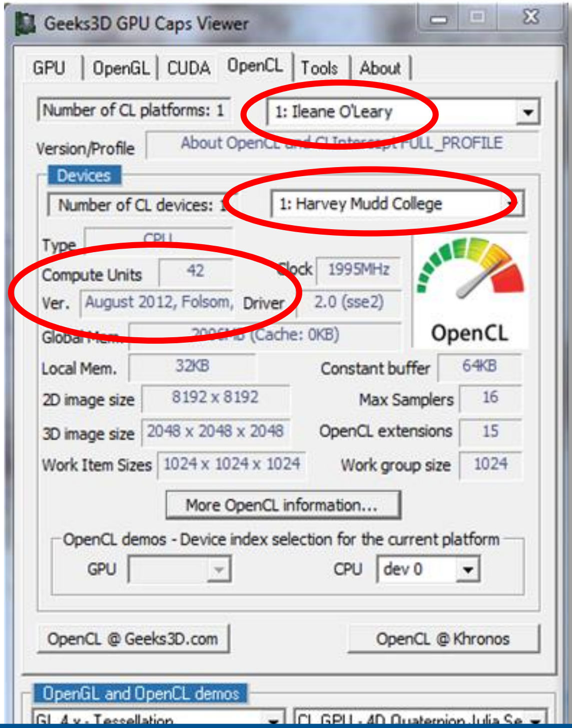
Performance Timing on VTune and Chrome*



1 item selected. Slice (1)

Title	clEnqueueNDRangeKernel(GenerateJuliaSet)
User Friendly Category	other
Start	1,918.770 ms
Wall Duration	0.044 ms

Platform and Device Queries



Geeks3D GPU Caps Viewer

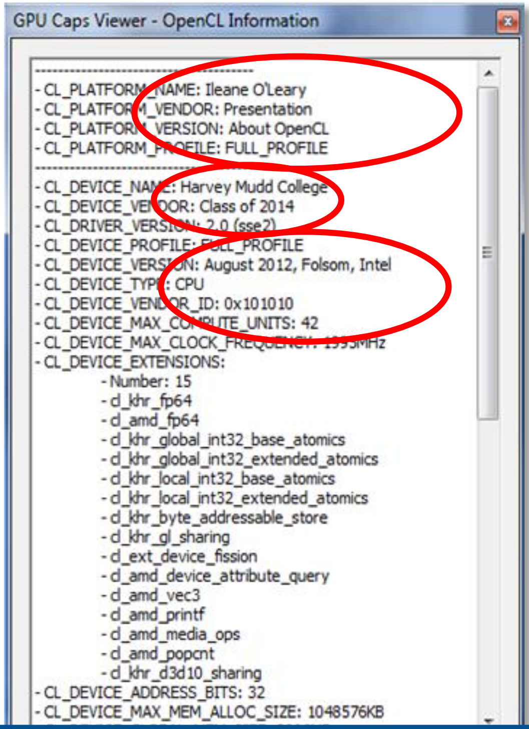
Number of CL platforms: 1	1: Ileane O'Leary
Version/Profile	About OpenCL and CLIntercept FULL_PROFILE

Devices

Number of CL devices: 1	1: Harvey Mudd College		
Type	CPU		
Compute Units	42	Clock	1995MHz
Ver.	August 2012, Folsom,	Driver	2.0 (sse2)
Global Mem.	2006MB (Cache: 0KB)		
Local Mem.	32KB	Constant buffer	64KB
2D image size	8192 x 8192	Max Samplers	16
3D image size	2048 x 2048 x 2048	OpenCL extensions	15
Work Item Sizes	1024 x 1024 x 1024	Work group size	1024

OpenCL demos - Device index selection for the current platform

GPU
CPU dev 0



GPU Caps Viewer - OpenCL Information

```
-------------------------------------------------
- CL_PLATFORM_NAME: Ileane O'Leary
- CL_PLATFORM_VENDOR: Presentation
- CL_PLATFORM_VERSION: About OpenCL
- CL_PLATFORM_PROFILE: FULL_PROFILE
-------------------------------------------------
- CL_DEVICE_NAME: Harvey Mudd College
- CL_DEVICE_VENDOR: Class of 2014
- CL_DRIVER_VERSION: 2.0 (sse2)
- CL_DEVICE_PROFILE: FULL_PROFILE
- CL_DEVICE_VERSION: August 2012, Folsom, Intel
- CL_DEVICE_TYPE: CPU
- CL_DEVICE_VENDOR_ID: 0x101010
- CL_DEVICE_MAX_COMPUTE_UNITS: 42
- CL_DEVICE_MAX_CLOCK_FREQUENCY: 1995MHz
- CL_DEVICE_EXTENSIONS:
        - Number: 15
        - cl_khr_fp64
        - cl_amd_fp64
        - cl_khr_global_int32_base_atomics
        - cl_khr_global_int32_extended_atomics
        - cl_khr_local_int32_base_atomics
        - cl_khr_local_int32_extended_atomics
        - cl_khr_byte_addressable_store
        - cl_khr_gl_sharing
        - cl_ext_device_fission
        - cl_amd_device_attribute_query
        - cl_amd_vec3
        - cl_amd_printf
        - cl_amd_media_ops
        - cl_amd_popcnt
        - cl_khr_d3d10_sharing
- CL_DEVICE_ADDRESS_BITS: 32
- CL_DEVICE_MAX_MEM_ALLOC_SIZE: 1048576KB
```

Explore how applications respond to different query responses!

Implementation Details:
OpenCL API Learnings and Insights

OpenCL APIs from a Layering Perspective

Most things went really well!

Features that made life easy:

- Built-in Reference Counting and Object Queries
- Standard Event Profiling, Standard Program Binaries
- Online Compilation

Features that made things complicated:

- NULL Local Work Size: Need “what happened” queries!
- Out-of-Order Queues – especially with Device Timing
- Device-side Only Controls (kernel attributes)
  - Easier to permute Host-side Controls (build options)

Intel-Specific Enhancements

Intel Specific Enhancement: Driver Diagnostics

cl_intel_driver_diagnostics: Intel Extension

- Extends Context Callback to Include **GOOD** / **BAD** / **INFORMATIONAL** Diagnostics
- Use the Intercept Layer for OpenCL Applications to Enable and Log Diagnostics

```
>>>> clCreateBuffer: flags = CL_MEM_WRITE_ONLY | CL_MEM_ALLOC_HOST_PTR (12), ...
=======> Context Callback (private_info = 00AFF728, cb = 4):
Performance hint: clCreateBuffer needs to allocate memory for buffer. For subsequent
operations the buffer will share the same physical memory with CPU.
<======= End of Context Callback
<<<< clCreateBuffer: returned 0573E000
...
>>>> clEnqueueMapBuffer: [ map count = 0 ] queue = 03254850, buffer = 0573E000, ...
=======> Context Callback (private_info = 00AFF214, cb = 4):
Performance hint: clEnqueueMapBuffer call on a buffer 0573E000 will not require any data
copy as buffer shares the same physical memory with CPU.
<======= End of Context Callback
<<<< clEnqueueMapBuffer: [ map count = 1 ] returned 04702000
```

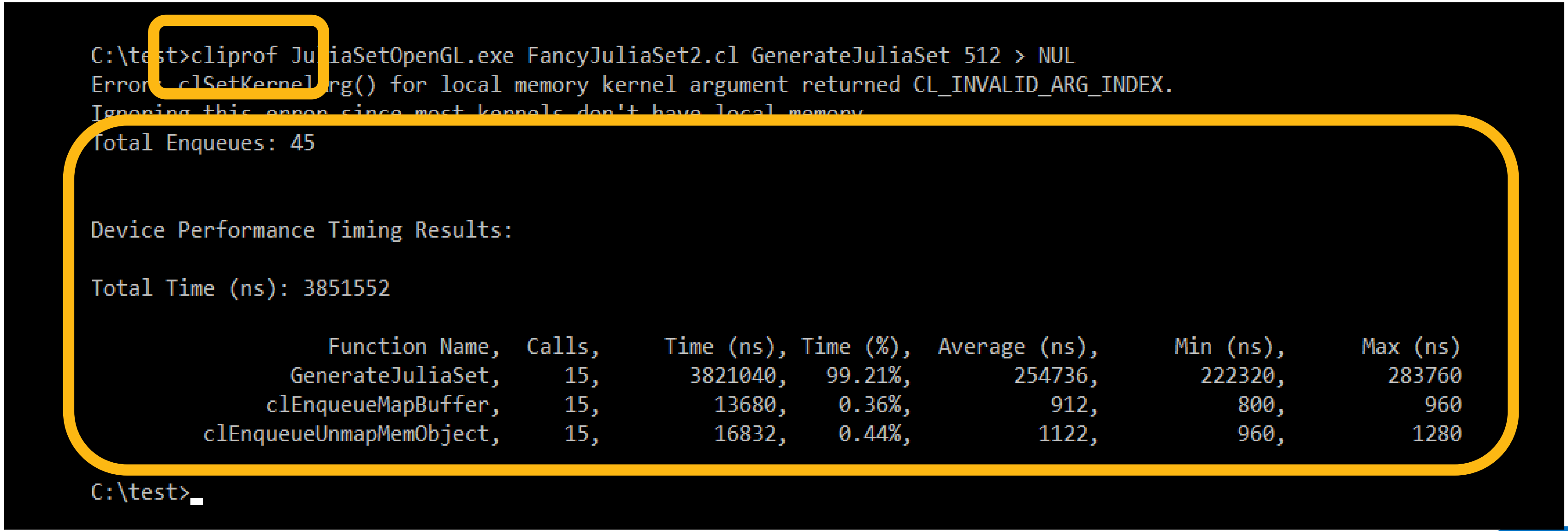
Future Work

Future Work – Short Term

Continue to stay use-case driven: How to find bugs and fix them faster?

- Example: OpenCL Object Leak Detection

Improve Usability and Accessibility

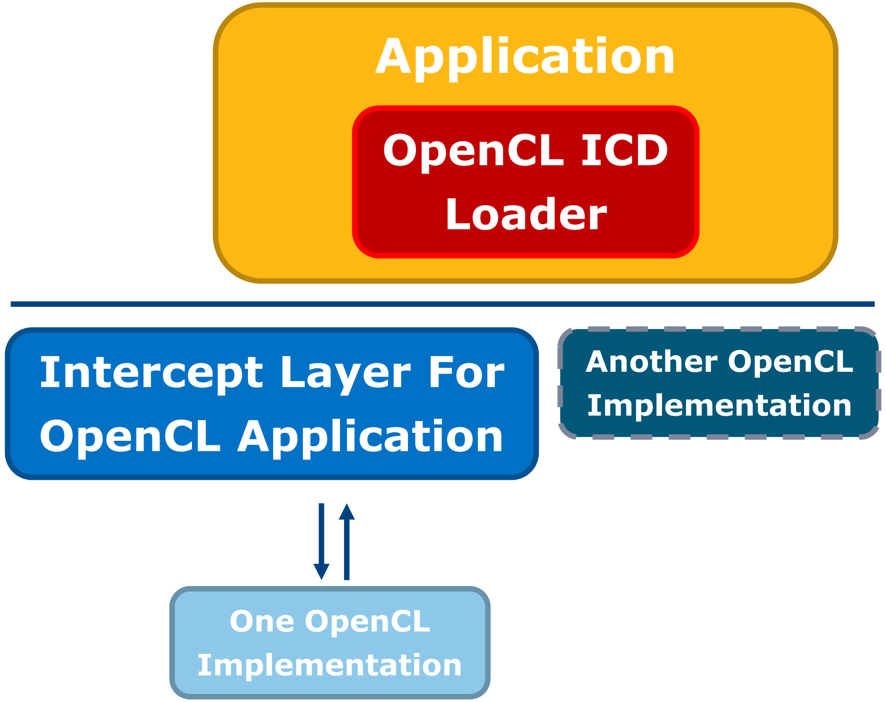


```
C:\test>cliprof JuliaSetOpenGL.exe FancyJuliaSet2.cl GenerateJuliaSet 512 > NUL
Error: clSetKernelArg() for local memory kernel argument returned CL_INVALID_ARG_INDEX.
Ignoring this error since most kernels don't have local memory.
Total Enqueues: 45


Device Performance Timing Results:

Total Time (ns): 3851552

           Function Name,  Calls,     Time (ns), Time (%),  Average (ns),       Min (ns),       Max (ns)
        GenerateJuliaSet,     15,      3821040,    99.21%,        254736,        222320,        283760
        clEnqueueMapBuffer,   15,        13680,     0.36%,           912,           800,           960
    clEnqueueUnmapMemObject,  15,        16832,     0.44%,          1122,           960,          1280

C:\test>
```

Future Work – Longer Term

Android Support ☺

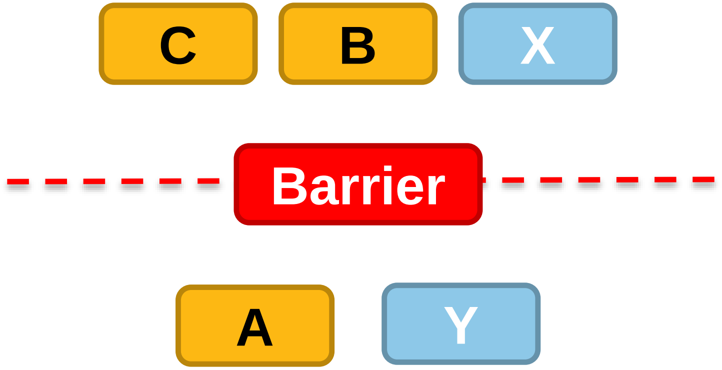
Intercept Layer as an ICD: Work with applications that statically link to the ICD loader.

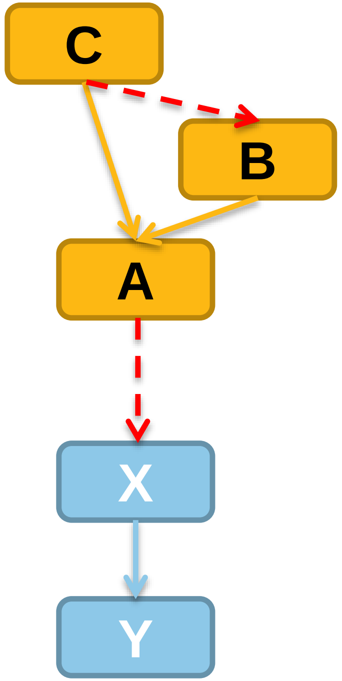

Application
OpenCL ICD Loader
Intercept Layer For OpenCL Application
Another OpenCL Implementation
One OpenCL Implementation

Future Work – Longer Term

Log and Analyze Graphs of OpenCL Commands

- Especially Important with Out-of-Order Queues
- Can we plot graphs of commands?
- Can we time device execution of subgraphs?





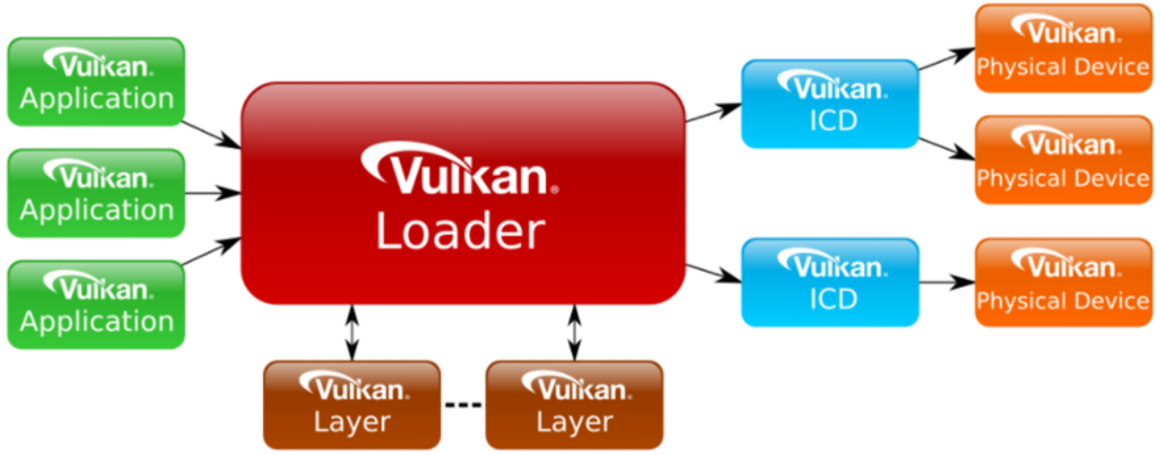
Future Work – Longer Term

Automatic Reproducer Generation

- Very Limited Capture-Playback
- One Kernel + Inputs + Params
- See Fossilize for Vulkan

Speaking of Vulkan…

- Lots of Layer Prior Art
- Steal with Pride?



Wrap Up

Summary and Call to Action

Try the Intercept Layer for OpenCL Applications!

- Debug and Analyze OpenCL programs faster!
- Send Issues and Pull Requests!

Grow the OpenCL Ecosystem with Layers

- Layers are an important part of the OpenCL ecosystem

To the Khronos OpenCL Working Group: Design the API with layers in mind!

To OpenCL Users: Use layers, evangelize layers, build layers!

Thank you!

- ben.ashbaugh@intel.com

Acknowledgements

Thanks to Michal Mrozek, Michael Carroll, Mike Kinsner, and Adam Herr for reviewing these slides.

Thanks to everyone from Intel who has used or contributed to the Intercept Layer for OpenCL Applications!

Useful Links:

Intercept Layer for OpenCL Applications:

https://github.com/intel/opencl-intercept-layer

Vulkan Loader and Layers:

https://github.com/KhronosGroup/Vulkan-LoaderAndValidationLayers

Legal Notice and Disclaimers

Legal Disclaimer and Optimization Notice