Neil Trevett
Vice President Mobile Ecosystem at NVIDIA | President of Khronos
SIGGRAPH, Vancouver 2014
Khronos Standards

Visual Computing
- 3D Graphics
- Heterogeneous Parallel Computing

3D Asset Handling
- 3D authoring asset interchange
- 3D asset transmission format with compression

Sensor Processing
- Vision Acceleration
- Camera Control
- Sensor Fusion

Acceleration in HTML5
- 3D in browser - no Plug-in
- Heterogeneous computing for JavaScript

Over 100 companies defining royalty-free APIs to connect software to silicon
Motivation for WebCL

• Parallel acceleration for compute-intensive web applications
  - Portable and efficient access to heterogeneous multicore devices in JavaScript

• Typical Use Cases
  - 3D asset codecs, video codecs and processing, imaging and vision processing
  - Physics for WebGL games, Online data visualization, Augmented Reality

• WebCL 1.0 specification officially released at GDC March 2014
  - https://www.khronos.org/webcl
OpenCL - Portable Heterogeneous Computing

- Portable Heterogeneous programming of diverse compute resources
  - Targeting supercomputers -> embedded systems -> mobile devices

- One code tree can be executed on CPUs, GPUs, DSPs and hardware
  - Dynamically interrogate system load and balance work across available processors

- OpenCL = Two APIs and C-based Kernel language
  - Platform Layer API to query, select and initialize compute devices
  - Kernel language - Subset of ISO C99 + language extensions
  - C Runtime API to build and execute kernels across multiple devices
WebCL - Heterogeneous Computing for Web

- WebCL defines JavaScript binding to the OpenCL APIs
  - Enables initiation of Kernels from within the browser

- Kernels are still written in OpenCL C
  - Analogous to WebGL shaders use GLSL
  - Enables re-use of kernels across OpenCL and WebCL
WebGL/WebCL Ecosystem

Content downloaded from the Web

Middleware can make WebGL and WebCL accessible to non-expert programmers E.g. three.js library: [http://threejs.org/](http://threejs.org/) used by majority of WebGL content

Browser provides WebGL and WebCL
Alongside other HTML5 technologies
No plug-in required

OS Provided Drivers
WebGL uses OpenGL ES 2.0 or Angle for OpenGL ES 2.0 over DX9
WebCL uses OpenCL 1.X

Low-level APIs provide a powerful foundation for a rich JavaScript middleware ecosystem
WebCL Programming

1. Query host for OpenCL devices
2. Create a context to associate OpenCL devices
3. Create programs for execution on one or more associated devices
4. Select kernels to execute from the programs
5. Create memory objects accessible from the host and/or the device
6. Copy memory data to the device as needed
7. Provide kernels to command queue for execution
8. Copy results from the device to the host
WebCL Architecture

- HTMLWindow or WorkerUtils implements WebCLEnvironment
  - If WebCL object not present, then WebCL is not supported

- Three Layers
  - Platform
  - Compiler
  - Runtime
WebCL - Designed-in Architectural Security

• Leverages OpenCL 1.2 robustness/security extensions
  - Context Termination: to prevent DoS from long running kernels
  - Memory Initialization: no leakage from out of bounds memory access

• API and Language Restrictions
  - To ensure no unsafe behavior is possible

• WebCL Kernel Validator
  - Static and dynamic kernel checking
WebCL Language Restrictions

- WebCL adds OpenCL C language restrictions for security
- Structures are not supported as kernel arguments
- Kernels name must be less than 256 characters
- Mapping of CL memory objects into host memory space is not supported
- Program binaries are not supported (for enhanced security)
- Some OpenCL API functions & builtin functions may require translation
WebCL Kernel Validator

- Open source on GitHub
  - [https://github.com/KhronosGroup/webcl-validator](https://github.com/KhronosGroup/webcl-validator)

- Cross platform compatible C API + library + command line client
  - Runs on Windows (Visual Studio), OSX (XCode) and Linux (Makefiles)

- Parses and validates kernel code against specification
  - Checks that no WebCL restrictions are broken - with extensive error messages

- Verifies that all memory accesses are inside valid memory areas
  - If not validated at compile-time, low overhead run-time checks injected in code

- Works on all OpenCL devices
  - Input WebCL C code -> Output is valid OpenCL C code
  - Source to source transformation based on Clang 3.4
Validation Overhead Example

- NBody Force Simulation

<table>
<thead>
<tr>
<th>No Protection</th>
<th>Protected</th>
</tr>
</thead>
<tbody>
<tr>
<td>190 GFlops</td>
<td>148 GFlops</td>
</tr>
<tr>
<td>22% Overhead</td>
<td>190</td>
</tr>
</tbody>
</table>

© Copyright Khronos Group 2014 - Page 12
WebCL Test Suite

- Source of tests openly available on GitHub
  - https://github.com/KhronosGroup/WebCL-conformance/

- Full API Test Coverage
  - WebCL requires a conformant underlying OpenCL on the host system
  - Input and output validation, and error handling checks for all methods in the API
  - WebCL validator conformance tests

- Tests for security and robustness through exercising...
  - Data leakage, Denial of service, Out of bounds memory accesses

- Test Suite Framework:
  - Allows selective execution of tests: skip tests per category or individually
  - Supports selection of available OpenCL platforms and devices
  - Allows different execution modes: relaxed and strict

- Implementers must still become official adopters
  - If they wish to use the WebCL trademark
WebCL Implementations

1. Nokia - Firefox build with integrated WebCL
   - Firefox extension, open sourced May 2011
     - Mozilla Public License 2.0
     - https://github.com/toaarnio/webcl-firefox
2. Samsung - uses WebKit, open sourced June 2011 (BSD)
   - https://github.com/SRA-SiliconValley/webkit-webcl
3. Motorola Mobility: Uses Node.js, open sourced April ‘12 (BSD)
   - https://github.com/Motorola-Mobility/node-webcl
4. AMD - uses Chromium (open source)
   - https://github.com/amd/Chromium-WebCL
     - WebCL Kernel Validator (open source)
       https://github.com/KhronosGroup/webcl-validator
     - OpenCL to WebCL Translator
       https://github.com/wolfviking0/webcl-translator

Based on Iñigo Quilez, Shader Toy

Based on Apple QJulia

Based on Iñigo Quilez, Shader Toy

http://fract.ured.me/
Chromium with WebCL

- Sandboxed architecture

node-webcl

- A cross-platform (Mac, Windows, Linux) implementation of WebCL 1.0 using Node.JS (uses Google Chrome V8 JavaScript engine)
  - Many examples: ray casting, image processing...
  - Passing WebCL conformance tests with workarounds for bugs in OpenCL drivers

- Extensions
  - Node’s Buffers for faster processing vs. TypedArrays
  - Multi-window and monitor support thanks to GLFW
  - Fully compatible with Node.JS async callbacks
  - OpenCL 1.1 & 1.2 features not (yet) in WebCL 1.0
  - Experimental features for future WebCL specifications

- WebGL interoperability and rendering with node-webgl, cross-platform
  - node-webcl v.1.0 to be released soon at https://github.com/Motorola-Mobility/node-webcl
Web Acceleration APIs

- Khronos and W3C liaison for Web APIs
  - Leverage proven native APIs
  - Fast API development/deployment
  - Designed by hardware community
  - Familiar foundation reduces developer learning curve

Native APIs shipping or Khronos working group
JavaScript API shipping, acceleration being developed or work underway
Possible future JavaScript APIs or acceleration