WebCL™ Kernel Validator

Request for Quotations

Notice
ALL KHRONOS SPECIFICATIONS AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, “MATERIALS”) ARE BEING PROVIDED “AS IS.” KHRONOS MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Information furnished is believed to be accurate and reliable. However, Khronos assumes no responsibility for the consequences of use of such information or for any infringement of patents or other rights of third parties that may result from its use. No license is granted by implication or otherwise under any patent or patent rights of Khronos. Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied.

Trademarks
Khronos, WebCL and WebGL, and associated logos are trademarks or registered trademarks of Khronos Group Inc. OpenCL is a trademark of Apple Inc. and used under license by Khronos. All other product names, trademarks, and/or company names are used solely for identification and belong to their respective owners.

Copyright
© 2013 The Khronos Group
All rights reserved.
1. Background

The WebCL™ API is designed with security as a primary requirement. The Validator is a security tool to enforce compliance of WebCL™ kernels with the defined WebCL™ language restrictions, and prevent out of bounds memory accesses. The WebCL™ kernel Validator will enforce out of bounds memory protection and perform syntactic validation of WebCL™ kernels.

2. Requirements

The project will deliver the design, implementation and documentation for a Validator for WebCL™ kernels. The Validator will take WebCL™ kernels as input, and generate validated OpenCL™ kernels as output, effectively acting as a WebCL-C source to OpenCL-C source trans-compiler. The validator will analyze the input WebCL™ kernels to determine violations of WebCL™ kernel behavior and language restrictions. The results from the analysis will also be used to determine any necessary instrumentation to bring the WebCL™ kernels in compliance with security and syntactic requirements of the WebCL™ API.

3. Deliverables and Acceptance Criteria

The scope of the WebCL™ kernel Validator project will include the following functionalities and deliverables:

A. Parsing/Syntactic and Semantic Validation consisting of a compiler front end for the C-99 based WebCL™ kernels. During this validation process, the C code is converted to an abstract syntax tree, and syntactic validation is undertaken to check for robustness. Syntax and semantic validation will provide list of error messages.

B. Analysis and Instrumentation involves static analysis of dynamic memory accesses, to find potential out of bounds memory accesses. Abstract syntax tree is instrumented for range checking.

C. Translation from abstract syntax tree, to instrument the resulting OpenCL-C.

D. Unit tests used to test the instrumented code, to ensure correctness of functionality, and conformance to secure coding practices.
4. **Recommended Approach**

Recommended approaches for each of the aspects are noted below.

A. **Parsing/Syntactic and Semantic Validation** involves compiler front end for the C-99 based WebCL kernels. A recommended approach is noted below:

   - C99 based approach:
     i. Using Clang, which currently has partial OpenCL™ support:
     ii. Start from Clang (using flags during compilation for extensions to enable OpenCL™ support for OpenCL™ vector types).

   ❖ Estimated maximum duration; 7 weeks

B. **Analysis & Instrumentation:**

   - Abstract Syntax Tree (AST)-based analysis:
     • Analysis checks, consisting of static checks and modification of AST.

   ❖ Estimated maximum duration: 6 weeks

C. **Translation:** Abstract Syntax Tree (AST) to OpenCL C conversion/translation.

   ❖ Estimated maximum duration: 4 weeks

D. **Unit tests** for testing the modifications made (through instrumentation) by the validator, for conformance.

   ❖ Estimated maximum duration: 3 weeks
5. **High Level Design**

The WebCL C Front End will consist of a modified Clang front end. It will take WebCL™ kernels as input, and will perform the following functions. The output will be an AST:

- Static analysis;
- Dynamic analysis;
- Fat pointer creation and checking;
- Array bounds checking.

Following is the summary of the WebCL™ security requirements and behavioral restrictions, which the validator will verify and enforce:

- Pointer casting
- Pointer clamping
- Index clamping
- Memory initialization (zeroing memory)
- Reject any vendor-specific or otherwise non-portable features
Behavioral Restrictions

<table>
<thead>
<tr>
<th>Pointer casting</th>
<th>Pointer clamping</th>
<th>Index clamping</th>
<th>Memory initialization</th>
<th>Remove compiler attributes not present in spec</th>
<th>Local memory allocation and initialization in inner scope</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trivial to find instruction in AST Visit + Smart pointers</td>
<td>Trivial to find instruction in AST Visit + Smart pointers</td>
<td>Trivial to find instruction in AST Visit + Smart pointers</td>
<td>Trivial in AST Visit with rewriter</td>
<td>Trivial in AST Visit</td>
<td>Trivial in AST Visit with rewriter</td>
</tr>
</tbody>
</table>

Stage III

Preprocessor

Indexer

Optimize AST

Libs:
- libBasic
- libLex
- libParse
- libAST
- libSema
- libAnalysis

- Source-to-source (S2S) translation is now a one-to-one mapping.
- Static analysis now possible.
- Optimizations on the AST

WebCL Validator (Stage III)

Analysis and Transformation

(Required)

<table>
<thead>
<tr>
<th>Pointer &amp; Scoping Restrictions</th>
<th>SMART Pointers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add smart checks to optimize execution and avoid clamps when they are not required.</td>
<td>Use static single assignment form and meta information to add protections (memory size, scope).</td>
</tr>
</tbody>
</table>

Libs:
- libBasic
- libLex
- libParse
- libAST
- libSema
- libAnalysis

- Source-to-source (S2S) translation is now a one-to-one mapping.
- Static analysis now possible.
- Optimizations on the AST

Analysis and Transformation

(Optionalal Optimizations)

<table>
<thead>
<tr>
<th>Static Analysis</th>
<th>Dynamic Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not needed from security point of view. Only required for optimizations (speed and memory)</td>
<td>Add smart checks to optimize execution and avoid clamps when they are not required</td>
</tr>
</tbody>
</table>

WebCL Validator (Stage III)
6. Project Scoping and Priorities
We hope to achieve complete feature set implementation, testing and documentation, with a scoping estimate of no more than 20 man weeks.

7. Schedule & Checkpoints
Below are the suggested project milestones. We will assess progress on a bi-weekly basis, so the feature coverage timeline below is only a rough guideline to the order in which we expect to have validator and tests written. Please provide detailed milestone dates that you can commit to delivering, using the format noted in section 8.

- April 12, 2013: Contractor selection and start of work;
- Completion of "Parsing" feature, and delivery of accompanying software and documentation.
- Completion of "Syntactic and Semantic Validation" functionality, and delivery of accompanying software and documentation.
- Completion of "Analysis and Instrumentation" related features, and delivery of accompanying software and documentation.
- Completion of "Translation", and delivery of accompanying software and documentation.
- Completion of Unit Testing and "Validator Conformance Tests". Delivery of complete feature set and documentation and release Validator version 1.0 to public

8. Milestone Details
RFQ respondents should provide detailed milestone plan that they can meet, according to the resources they are able to apply, and to bid an amount to compete. Details related to the milestones, inclusive of description and design, implementation details, and proposed start and end dates for each milestone should be provided.

<table>
<thead>
<tr>
<th>Parsing/Syntactic and Semantic Validation (Compiler front end for the C-99 based WebCL™ kernels)</th>
<th>Start Date</th>
<th>End Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description and Design</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Implementation Details:</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Analysis & Instrumentation (AST-based)

<table>
<thead>
<tr>
<th>Description and Design:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Implementation Details:</td>
</tr>
</tbody>
</table>

### Translation (AST to OpenCL C conversion)

<table>
<thead>
<tr>
<th>Description and Design:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Implementation Details:</td>
</tr>
</tbody>
</table>

### Unit Tests (Test Plan and Unit Tests for Conformance Testing)

<table>
<thead>
<tr>
<th>Description and Design:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Implementation Details:</td>
</tr>
</tbody>
</table>

#### 9. Khronos Contractor Agreement

Contractors wishing to respond to this RFQ that are not Khronos members and require access to Khronos confidential information will be required to execute the standard Khronos membership agreement (with membership fees waived) to gain access to confidential materials for the sole purpose of responding to this RFQ.

The selected contractor will be required to execute the standard Khronos Contractors Agreement with milestones and costs entered into Exhibit B and Contractor Disclosures entered into Exhibit C.

No work shall begin, and Khronos shall be liable for no costs or expenses, until the selected contractor is in receipt of an executed contractor’s agreement.

It is important that contractors understand that we will be assessing progress on a regular basis, and we reserve the right to terminate or renegotiate the contract in the event insufficient progress is being made.
10. RFQ Responses

The RFQ response materials will form the basis for detailed milestone and cost negotiations for the final contract with the selected vendor or vendors. Please provide the following information in the format of your choice:

- Identification of deliverables on which you wish to bid;
- Proposed schedule, highlighting any differences from the suggested milestones in Section 7, using the format noted in section 8;
- The hourly cost for engineering resources from your company, the minimum and maximum number of hours you can commit to this project on a bi-weekly basis, and a description of the qualification of the engineering resource(s) which would be used;
- The total project cost to Khronos;
- A description of your familiarity (if any) with WebCL™, and CLang, and of any test development you have previously performed related to WebCL™, OpenCL™ and/or WebGL™,
- An indication you are willing to work under the Khronos Contractor Agreement;
- Any particular issues or risk factors that you wish to highlight;
- Supporting materials, including background materials about your company, highlighting other relevant experience and expertise for this project.