
# Khronos Group
# Request for Proposals

# LLVM SPIR-V Backend

## November 2024

## Re-issued with Clarifications May 2025

### Notice



### Trademarks

# 1. Background

The SPIR-V target within LLVM, also known as the LLVM SPIR-V Backend, facilitates code generation for the OpenCL SPIR-V binary format, as outlined in the SPIR-V specification [1]. This Backend was upstreamed into the LLVM repository in 2022 and is on track to become a stable, permanent target. Historically, the Khronos LLVM/SPIR-V Bi-Directional Translator has served as the official tool for generating SPIR-V from LLVM IR. However, due to ongoing developments and contributions from various SPIR-V vendors, the Backend is anticipated to fully supplant the Translator.

The goal of this project is to address and integrate specific functionalities that are currently available in the Translator project but missing from the Backend.

# 2. Methodology

Khronos has a fixed budget for this project, with payment due on project completion.

This RFP is being circulated to all Khronos members, and publicly, any interested party is welcome to respond.

A Khronos Slack channel will be used for project communications that any interested Khronos member may join. Short status and progress reports should be prepared for the weekly OpenCL and SPIR-V Working Group meetings, and the biweekly OpenCL Tooling meetings and circulated on the email lists for those groups.

All code development shall take in place in the public LLVM project repository [3], and the Khronos LLVM/SPIR-V Bi-Directional Translator repository [4], using GitHub pull requests and issues reviewed by OpenCL and SPIR Working Group members.

# 3. Scope

## 3.1 Work Areas

The scope of this Request for Proposals (RFP) is concentrated on aligning the capabilities of the experimental LLVM SPIR-V Backend with those currently supported by the Khronos LLVM/SPIR-V Bi-Directional Translator. This involves several areas of work, some involving the LLVM's Integrated Testing (LIT) framework, outlined below in priority order:

1. **Port existing tests:** port existing LIT tests from the Translator to test the Backend and contribute current Backend LITs back to the Khronos repository. This will help identify future areas of work to ensure functional parity. This excludes any DWARF test cases since LLVM does not support "reverse translation" from SPIR-V to LLVM IR.
2. **Implement and test LLVM intrinsics lowering methods:** implement and test missing intrinsics, and correct current implementations, by running existing LIT and Translator tests or creating new tests as necessary. Any created tests shall be contributed back to the Khronos repository.
3. **Implement and test SPIR-V extensions:** implement and test missing SPIR-V extensions, and correct current implementations, by running existing LIT and Translator tests or creating new tests as necessary. Any created tests shall be contributed back to the Khronos repository.
4. **Adding missing OpenCL builtin function TableGen definitions**: TableGen definitions should be added to the current Backend implementation of the GlobalISel lowering flow if these would impact 1-3 above. However, significant rework or refactoring of CodeGen passes and lowering methods is NOT expected.

## 3.2 Priority Intrinsics and Extensions

Effort should be focused on LLVM intrinsics and SPIR-V extensions that are currently missing, untested, or incorrectly/inefficiently implemented in the Backend, including:

- Intrinsics include *llvm.experimental.constrained.** (arithmetic, comparison, and conversion), *llvm.debugtrap*, *llvm.memmove*, *llvm.experimental.noalias.scope.decl*, *llvm.fptosi*, *llvm.is.fpclass*, *llvm.frexp*, *llvm.instrprof.**, and *llvm.nearbyint*.

- Extensions: *SPV_KHR_integer_dot_product, SPV_KHR_non_semantic_info (with a focus on implementing NonSemantic.Shader.DebugInfo.100 and NonSemantic.Shader.DebugInfo.200 extended instruction sets), SPV_EXT_image_raw10_raw12, SPV_EXT_relaxed_printf_string_address_space, SPV_INTEL_arbitrary_precision_fixed_point, SPV_INTEL_arbitrary_precision_floating_point, ~~SPV_INTEL_arithmetic_fence~~, SPV_INTEL_bindless_images, SPV_INTEL_blocking_pipes~~, and SPV_INTEL_complex_float_mul_div~~*.

## 3.3 Clarifications

The following questions were received on the original RFP, and the following clarifications are being shared here for all respondents to update their responses if desired.

*1. The Debug Info tests may require rewriting the DWARF tests (these tests in the SPIR-V LLVM Translator repository) to SPIR-V tests. Have you assessed this and is this implicitly included in the expected effort?*
No, we do not expect that these tests would be ported to the SPIR-V backend and LLVM repository. Especially since we do not provide any "reverse translator" from SPIR-V to LLVM IR as part of the SPIR-V backend.

*2. There is no specification for the extensions SPV_INTEL_complex_float_mul_div and SPV_INTEL_arithmetic_fence. Will documentation be provided in time to be able to write tests for these?*
- SPV_INTEL_arithmetic_fence was recently replaced by SPV_EXT_arithmetic_fence and the RFP is modified accordingly above
- SPV_INTEL_complex_float_mul_div was implemented in the PR1442 and is defined in PR5824. However, this extension is now removed from the RFP.

*3. We found two categories of tests that we assumed not to be implemented, based on how the goals and scope were defined: a) tests for extensions that aren't supported. These are around 100 additional tests, b) tests that are not missing, but implemented differently. Is this assumption correct?*
Yes, this is correct. Many of the current tests were based on the tests in the Translator repository and parts were skipped in cases already covering a feature in another test or implemented differently due to issues discovered in the SPIR-V Backend (to prevent these from reappearing). Porting existing test cases from the translator repository does not necessarily need to mean copying the exact file/folder or even test case structure - any systematic solution is acceptable. However, it is important to identify the missing parts, bugs, and organize it all in a way that will allow us to continue adding new cases without duplication in both projects.

*4. What to do with the SPV_EXT_image_raw10_raw12 extension depends on OpImageQueryFormat, which is not present?*
SPV_EXT_image_raw10_raw12 defines two more tokens for the "Image Channel Data Type" section which are already a part of the standard,
https://registry.khronos.org/SPIR-V/specs/unified1/SPIRV.html#_image_channel_data_type, see 19 UnsignedIntRaw10EXT and 20 UnsignedIntRaw12EXT.

*5. What is exactly required for SPV_KHR_non_semantic_info? Does this require implementing `--spirv-preserve-auxdata` or is it out of scope for that reason?*
We don't think this is the case. LLVM optimizations and backend transformations should already preserve the information. However, we can follow up on this question if there are any specific concerns or unforeseen issues with the implementation. Bertrand Wlodarczyk worked on this extension, but there is still a long way to go and a wider context, including our plans to have eventually support for "NonSemantic.Shader.DebugInfo.200" that is a work in progress, see https://github.com/KhronosGroup/SPIRV-Registry/pull/186.

*6. Is there a difference between SPV_INTEL_arithmetic_fence and SPV_EXT_arithmetic_fence? The latter is already implemented in LLVM.*
SPV_INTEL_arithmetic_fence was recently replaced by SPV_EXT_arithmetic_fence in the SPIR-V LLVM Translator and the extension is already implemented in the SPIR-V backend.

*7. Some features are already implemented in the Intel LLVM fork. Can these simply be reused?*
Yes, of course! Maintaining one implementation and sharing the details will make everything much easier.

# 4. Deliverables

## 4.1 Project Plan

A project plan outlining the tasks to be completed within the available time and budget, after mutually agreeing priorities with the OpenCL Working Group.

**Acceptance Criteria:** signoff from the OpenCL Working Group that the plan is practical and correctly prioritized.

## 4.2 LIT Tests Ported from the Khronos LLVM/SPIR-V Bi-Directional Translator

Pull requests for LIT tests ported from the Translator repository with updated checks, and additional verification and validation runs. Enhanced tests should also be pushed back to the Translator repository if relevant.

**Acceptance Criteria:** discussion, code review, and acceptance by the OpenCL Working Group.

## 4.3 LLVM Intrinsics Implementation and Tests

Pull requests for LLVM intrinsic implementations, with associated tests, pushed to the LLVM project repository on an ongoing basis.

**Acceptance Criteria:** discussion, code review, and acceptance by the OpenCL Working Group.

## 4.4 SPIR-V Extension Implementation and Tests

Pull requests with the implementation of SPIR-V extensions, using minimum-implementation tests from the Khronos repository to verify accuracy and completeness.

**Acceptance Criteria:** discussion, code review, and acceptance by the OpenCL Working Group.

# 5. Schedule and Budget

Khronos has a fixed budget of $40K USD for this project and expects work to be completed within three months of starting work, with payment on completion and acceptance of all deliverables as agreed by the OpenCL Working Group.

# 6. Selection Process

Khronos shall designate a Khronos RFP Manager and will use an RFP email list (opencl-rfp@lists.khronos.org) that can be used to contact the RFP Manager and all other OpenCL Working Group members involved in the bid selection process. No Khronos member making a bid shall be on the RFP list. Any company considering making a bid in response to the RFP should notify the RFP list as soon as possible. Any potential bidder may request additional information and submit questions directly to the RFP manager or on the RFP email list. Any additional Khronos information and RFP clarifications will be distributed equally to all potential bidders.

All bidders should provide the following information in the format of their choice:

- Proposed schedule, assuming work starts in July 2025.
- Confirmation that if your bid is accepted, you are willing to work under the terms of the Khronos Contractor Agreement [2].
- Any issues or risk factors that they wish to highlight.
- Supporting materials, including background materials about their company, highlighting experience and expertise relevant to this project.

Updated RFP responses are requested by **5PM PT on Friday June 6th 2025** and should be sent to the RFP list. Bidders may update their bid as they wish before the submission deadline. In exceptional circumstances a requested submission deadline extension may be issued to all bidders at Khronos' discretion.

Khronos will evaluate all bids and select the winning bid based on timescales, and relevant experience and expertise.

Khronos expects to announce the selected bid two weeks after the submission deadline and will immediately notify all bidders and enter contract negotiations with the selected bidder to finalize deliverables and payment schedule. Khronos will immediately notify all other bidders once contract negotiations are complete. If contractual agreement cannot be reached, Khronos may select an alternative bidder.

Work can start immediately when the contract is negotiated and executed by both parties.

## 7. Contractors Agreement

The selected contractor will be required to execute the 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 a mutually executed Contractor's Agreement.

It is important that contractors understand that, under the terms of the Contractors Agreement, Khronos will assess progress on a regular basis and reserves the right to terminate or renegotiate the contract in the event of insufficient progress or other issues.

## 8. References

[1] SPIR-V Specifications
https://www.khronos.org/registry/SPIR-V/

[2] Khronos Contractors Agreement template
https://members.khronos.org/document/dl/30507

[3] LLVM project repository
https://github.com/llvm/llvm-project

[4] Khronos LLVM/SPIR-V Bi-Directional Translator repository
https://github.com/KhronosGroup/SPIRV-LLVM-Translator