WebGPU environment specification for SPIR-V

dneto@, cwallez@
Status: Draft

Introduction

SPIR-V modules are useful only when embedded in an execution environment such as Vulkan, OpenGL, OpenCL, and potentially WebGPU. The validity rules for a SPIR-V module are expressed as a combination of specifications:

- Some version of the SPIR-V specification, often called the "core" spec.
- An *environment specification* which specifies extra rules as demanded by the embedding execution environment. For example, Appendix A of the Vulkan specification is Appendix A: Vulkan Environment for SPIR-V
- Specifications for SPIR-V extended instruction sets permitted by the environment spec, e.g. SPIR-V Extended Instructions for GLSL
- Specifications for SPIR-V extensions permitted by the environment spec, e.g. SPV_KHR_storage_buffer_storage_class

This document is a skeleton for a WebGPU environment specification for SPIR-V. Since WebGPU is a graphic environment like Vulkan, this spec is inspired from the Appendix A of the Vulkan specification and adds WebGPU-specific security and portability constraints.

The environment spec only describe constraints on a SPIR-V module taken in isolation. Validation of the "linking" of the module with other parts of a pipeline (other modules, pipeline layout, input layout...) will be done in the main body of the WebGPU spec. Also not addressed yet are the precision of mathematical operations, image format and how data provided by the application is interpreted as SPIR-V (endianness, SPIR-V magic number etc.).

Capabilities

Allowed capabilities are the following:
- Matrix - Allows using OpTypeMatrix
- Shader - Allows the Vertex, Fragment and GLCompute execution models
- Sampled1D and Image1D - Allows using 1D textures
- ImageQuery - Allows querying information about textures
- DerivativeControl - Allows precise fwidth(), dfdx() and dfdy() builtins.

Allowed if the API supports “texel storage buffers”:

- SampledBuffer / ImageBuffer - Allows using “texel storage buffers”

It is an error for a SPIR-V module to declare an OpCapability that isn’t allowed.

Validation rules

In addition to the validations rules in section 2.16 of the SPIR-V specification, SPIR-V module must satisfy the following constraints.

These are WebGPU-specific:

- OpVariable with storage class “Output”, “Private” or “Function” must have initializers.
- OpVectorShuffle component literals must not be FFFFFFFF
- OpUndef must not be used.

These are present in Vulkan too:

- Entry points must have no return value an no argument.
- The call graph for entry points must be acyclic.
- The module must use the logical addressing model.
- Constraints on allowed instruction “execution scope” and “memory scope”
- Constraints on allowed variable “storage classes”
    - The PushConstant storage class is permitted only if WebGPU supports a push-constants feature.
- Constraints on allowed builtins and their uses.
- OriginLowerLeft and PixelCenterInteger execution mode must not be used.
- Constraints on allowed image types and OpSampleImage.
- Constraints on allowed decorations.
- OpTypeRuntimeArray must only be used for the last member of an OpTypeStruct with Uniform storage class and a BufferBlock decoration.
- Compute shaders must have LocalSize or an object with WorkgroupSize.
- Atomic instructions must return 32bit integers.

Robust resource access

Execution of SPIR-V modules in WebGPU must satisfy constraints in addition to the SPIR-V validation described above. For security, WebGPU must provide robust resource access as discussed in this issue. Validation rules above ensure all variables are initialized in the SPIR-V code or with the bindings with the exception of OpVariable with a Workgroup storage class. These must act as if initialized to 0 at the beginning of the entry point.