GPU Web 2017-12-13

Chair: Corentin
Scribe: Ken
Location: Google Hangout

Minutes from last meeting

TL;DR

- Status updates
    - Google at the feasibility of adding a rendering backend to UnrealEngine
    - Microsoft: Ben started working on a spec for HLSL
    - Mozilla studied the portability of queries for gfx-rs
- Shading language and schedule
    - Seems to be consensus to no block the API on a new language.
    - Everyone would be OK contributing to HLSL.
    - To decide between HLSL and SPIR-V we could do a bake-off.
    - Updating the DirectX Compiler for WebGPU would work for Microsoft but not others, so there will be another open-source implementation.
    - Concern the bake-off only delay the decision on the shading language.
    - Discussion on whether using a bytecode helps avoid classes of bugs.
    - Discussion that there should be multiple implementations of the compiler.
    - Suggestion that an HLSL-- would be better than HLSL for interop.
    - Group action item to make a document exploring the tradeoffs.
    - We shouldn't create another HLSL CG and do things in WebGPU to avoid organizational complexity.

Tentative agenda

- Shading languages
- Agenda for next meeting

Attendance

- Apple
    - Myles C. Maxfield
    - Theresa O'Connor
- Google
    - Corentin Wallez

    - David Neto
    - James Darpinian
    - Kai Ninomiya
    - Ken Russell
- Microsoft
    - Ben Constable
    - Chas Boyd
    - Rafael Cintron
- Mozilla
    - Dzmitry Malyshau
    - Jeff Gilbert
    - Markus Siglreithmaier
- Yandex
    - Kirill Dmitrenko

Shading Languages

- MM: proposal that's been uploaded is for Secure HLSL, which isn't what we agreed upon a few weeks ago ("HLSL proper" – without Secure HLSL's new features)
    - Uploaded as a reference for going forward
    - CW: think we agreed to go with "an" HLSL, but without the doc, was hard to discuss
    - MM: thought we could discuss / standardize the features that are in core HLSL
- KN: not comfortable blocking the release of this API on the standardization of a whole new language, but think we should keep discussing it. If we can release Secure HLSL before we release a V1 in this group, think it's OK if the group switches to it.
- MM: think there's a natural progression from HLSL -> HLSL + some new things
    - Let's start with the smaller subset
    - KN: agree
- CW: has high-level concerns about exposing something beyond what's necessary to cover the hardware features
    - Secure HLSL adds some syntactic / complex features that are solely for developer productivity which would make it harder for browsers to be interoperable. Wouldn't help developers since they would transpile anyway
    - MM: disagree. The reason we agreed upon a human writeable language is that we want humans to write it.
    - Converse of making things easier on web developers is making life harder on web developers
- MM: what about a bake-off? Standardize both SPIR-V w/constraints and HLSL, and gather more information from developers? Then we could choose one as the winner, or if that solution works well, keep it
- BC: what sort of timescale were you thinking about for the MVP or V1?

    - CW: we want to release earlier than 5 years from now. Goal is to catch up with native APIs
    - CW: have been hoping we could have an MVP in the next year and a V1 in the year after.
    - MM: this problem is something our “shared library” – “ANGLE next” or whatever – will help with shipping things fast
- BC: when I hear these timeframes I don’t think it’s over-ambitious that we could have both of these things fleshed out to a certain degree. For MSFT, most of the work will be adding a shader model to the compiler to target Secure HLSL, and then add bounds checking for the hardware, etc.
    - Apple’s given the impression that you’ve been able to build a fairly rapid prototype. Have a lot of stuff running?
        - MM: that’s accurate
    - E.g., don’t mind there being a cutoff deadline, if it’s ready by this point then we can use it, don’t if not.
- RC: largely agree. Yes, will have to augment HLSL if we choose it. But will have to augment SPIR-V if that’s the language of preference. Things will have to change either way if so.
    - CB: the kinds of work items we’re talking about are minimal compared to the normal daily work we’re doing anyway.
- CW: it’s understood that adding the new shader model to dxc is fine. But we don’t think dxc is OK to use by anyone but Microsoft.
- MM: agree. If we’re going to contribute most of an HLSL compiler from scratch it’ll be open source.
- DN: there’s also a SPIR-V backend for dxc. Two ways to get SPIR-V that would run on Vulkan. Plan on reconciling these. There is a test suite. Usually, the test suite takes longer to write.
- BC: have started working on the spec. Trying to avoid turning a computer spec into language. Having both a spec and conformance suite together is ideal. Advantage of the tests is that they shouldn’t be platform specific anyway, can be contributed by anyone. Can be aspirational – cover things we want to work or don’t want to work
- DN: agree
- MM: agree
- BC: welcome questions about HLSL and/or how to spec and support it. Agree that we need a spec for HLSL for the purpose we’re talking about using it for.
- MM: Apple’s interested in contributing to it too.
- CW: no matter what the decision on shading language is, Google will be interested too.
- KR: How do we get back to our lessons of using GLSL for WebGL? Preprocessor issues, token length, etc. All the issues you get when using source text as the input vs. a bytecode format. Still think we should go with the IR as it eliminates a whole class of problems. What’s the criterion on deciding whether to have text vs. bytecode.
- MM: trying to get to the bottom of these beliefs is the point of this bakeoff
- JG: don’t think it’ll address these concerns

- MM: getting the real, raw data for how things will work is possible a good way forward
- JG: it's movement but not necessarily a way forward
- KN: even if we do this, we can't see the future of years of developers using the thing. Or all of the weird shaders people come up with or interoperability issues. We won't find all the bugs until far in the future. Useful to have concrete data, but don't think it gets us to our end goal. Also don't think that discussing philosophy moves us forward.
- CB: have we discussed all of the issues that are on the issue tracker?
    - Looks like the number of passes that we'll apply to SPIR-V is: all templates applied, structured control flow, CSE, etc. Would be great to have a conversation about what the differences look like between HLSL and this.
- JG: seems to me the big crux we can't agree on is whether we want an immediately-consumed human text format.
- TOC: or a binary format.
- CB: maybe there are ASCII-readable
- CW: perhaps we think interop is more imp't than Apple does. Maybe best to have a tradeoffs document.
- CB: think a lot of the things removed by SPIR-V maybe can be removed by the HLSL variant.
- JG: SPIR-V is easy to validate, easy to consume. Having a reduced HLSL is a cool idea, but think it shouldn't be a blocker for us. Don't think it's useful to have a bakeoff and push this into the future.
- BC: on some platforms SPIR-V isn't easy to consume.
- JG: the syntax is basically trivial.
- BC: have experienced building one of these compilers. Syntax was trivial.
- JG: nearly a decade into WebGL we're still dealing with errors in GLSL compilers.
- BC: the syntax was the smallest concern. Apple's and Microsoft's platforms don't take GLSL in.
- JG: even on Vulkan we're going to recompile the SPIR-V.
- KN: we're going to do the same amount of validation no matter which language is ingested
- JD: it's easy for one team to write a GLSL compiler. It's not easy for 5 teams to write interoperable GLSL compilers. Easy for that to be done with SPIR-V
- KN: it's not that it's difficult. It's that it hasn't been solved in years of development of WebGL. And we have only two compilers, soon to be one.
- CW: Olli from NVIDIA is still fixing GLSL compilation issues in ANGLE today.
- BC: all software has bugs. What would be instructive for me is: someone shows me five bugs that typify the kinds of issues we're talking about.
- KR: In previous meeting tried to dig in the ANGLE bug tracker to dig specific issues. Scoping rule mismatches, expression complexity, token length...
- TOC: All browsers ship a Javascript engine.
- JG: invite people who are skeptical about SPIR-V to just look at the example in the SPIR-v spec. SSA makes it super simple. Think I could write the compiler and validator in a week.

- CB: what is likely to be the objective toward making multiple implementations of the compiler front-end?
- CW / KN:
- KR: Goal isn't to necessarily have multiple compiler frontends. The goal is to remove part of the complexity and error surface of the language.
- CB: so if there are multiple frontends and they all spit out SPIR-V then all browsers can read it?
- JG: yes.
- CB: and if it's invalid then the browser rejects it?
- CW: yes. And all browsers would be required to reject invalid shaders.
- KN: re: comparison w JavaScript: yes, there are multiple languages that have multiple compilers. C, C++, JavaScript, Python. But there is way more JS / C++ in the world and people hammering on the compilers than for any shading language. The amount of testing in the wild that our compiler gets is probably 1/10,000th the native compiler will get. Just b/c there are multiple interoperable JS compilers doesn't mean that we'll be able to reach this.
- CB: didn't realize that we need to have a new ecosystem of compilers spring up. Thought we could put things up on Github.
- JG: the hybridization effect of having multiple implementations improves those implementations as well as the spec.
- BC: the way I'm thinking about this: don't think you can deny there's less work ingesting a binary IR than text.
- MM: there are certainly cases where ingesting a binary IR format will be more work than text.
- JG: this won't be the case with SPIR-V.
- MM: dominator analysis for example.
- DN: for two basic blocks, if all paths through block B go through A, then A dominates B. This is so it's easier to process.
- MM: this is a place where if you know the oracle tells you it's valid, it's easy. The trouble is when you have to implement the oracle
- DN: this replaces scoping rules. We have this analysis in the validator our team maintains. Anyone can pick it up.
- MM: making this point just to disprove the statement that IR-level languages are easier or harder to validate than a text-level language
- BC: trying to make the point that it takes more effort to ingest text than IR. But what's the delta complexity to take in text vs. IR? And is it worth the developer impact / benefits?
- CW: our concerns about interoperability could be an "HLSL-minus-minus" - a much simpler HLSL. Could make the argument that this is better than a bytecode because it's as simple to ingest, but allows web authors to write it without running a compiler.
    - But if you then add more features like generics, the complexity begins to grow. Adds features that will manifest in bugs in implementations.

    - With the high-level shading language we're pushing for evolution of the shading languages. Don't think this community group should be the ones pushing in that area.
- JG: think it's good to push forward for a better solution for shaders. Just think that tying this to the API is the wrong way. Do it as a library. Can undo things later.
- DN: got strong feedback from the ISVs that they wanted control of the shader language side as well as the hardware side, to get better control of their toolchains. That's the reason Vulkan went with SPIR-V. It's the same motivation here.
- DN: Chas suggested something we can produce, like a feature comparison.
- CB: there are a lot of passes that are done. We can see where these passes are done in the HLSL compiler vs. the SPIR-V compiler and validation.
- CW: that's one data point we should produce. Ultimately it seems that we should do more: for all aspects of the shading language, agree where the tradeoff lies between a bytecode and HLSL. For bugs, we can all agree that it's simpler to ingest a bytecode. But ingesting a bytecode may have other tradeoffs.
    - If we still can't agree, we can have a bakeoff.
    - Should ask the W3C TAG if they have any strong opinion.
- MM: we can do this. If we do, it has to be a collaborative document.
- CW: for the next shading language meeting, we should expose all of these points and try to decide where the group stands on all of them.
- MM: agree
- BC: maybe the high-level language discussions should be split off into a separate group.
- CW: so having an HLSL CG inside the W3C?
    - BC: sub-charter of what we're talking about here
    - TOC: the overhead of a separate CG would be pretty problematic. If the idea is that the overhead of these discussions is too high for this call, we can have a separate call in the same CG.
    - MM: it'll be targeted at WebGPU, so should be in this CG.
- TOC: we're sort of already doing separate meetings for this anyway, in the form of every other meeting.
- BC: seems since there isn't an HLSL spec it doesn't make sense to have conversations about it
- CW: even if HLSL doesn't have a spec, in discussions we're assuming it will have a good spec and test suite. We know enough about HLSL that we can reason about it. Don't worry about that aspect.

Action Items

- Create list of aspects, pros/cons of shading languages
- If there are aspects we don't agree on, can discuss and try to converge.

Agenda for next meeting

- Thinking about cancelling the next two meetings. Should next meeting be on January 3 or 10?
    - Think January 3 works for everyone.
    - No specific agenda right now. If you have ideas then send them.
    - Think we should discuss high-level goals and customers.
    - API, but no shading language topics.
- Open agenda