

Graphics Feature Status

=====

- * Canvas: Hardware accelerated
- * Canvas out-of-process rasterization: Enabled
- * Direct Rendering Display Compositor: Disabled
- * Compositing: Hardware accelerated
- * Multiple Raster Threads: Enabled
- * OpenGL: Enabled
- * Rasterization: Hardware accelerated
- * Raw Draw: Disabled
- * Skia Graphite: Disabled
- * Video Decode: Software only. Hardware acceleration disabled
- * Video Encode: Software only. Hardware acceleration disabled
- * Vulkan: Enabled
- * WebGL: Hardware accelerated
- * WebGL2: Hardware accelerated
- * WebGPU: Disabled

Driver Bug Workarounds

=====

- * adjust_src_dst_region_for_blitframebuffer
- * disable_discard_framebuffer
- * enable_webgl_timer_query_extensions
- * exit_on_context_lost
- * force_cube_complete
- * init_gl_position_in_vertex_shader
- * pack_parameters_workaround_with_pack_buffer
- * unpack_overlapping_rows_separately_unpack_buffer
- * use_virtualized_gl_contexts
- * disabled_extension_GL_KHR_blend_equation_advanced
- * disabled_extension_GL_KHR_blend_equation_advanced_coherent
- * disabled_extension_GL_MESA_framebuffer_flip_y

Problems Detected

=====

- * WebGPU has been disabled via blocklist or the command line.
Disabled Features: webgpu
- * Accelerated video encode has been disabled, either via blocklist,
about:flags or the command line.
Disabled Features: video_encode
- * Accelerated video decode has been disabled, either via blocklist,
about:flags or the command line.
Disabled Features: video_decode
- * Program link fails in NVIDIA Linux if gl_Position is not set:
(<http://crbug.com/286468>)
Applied Workarounds: init_gl_position_in_vertex_shader
- * MakeCurrent is slow on Linux with NVIDIA drivers:
(<http://crbug.com/449150>), (<http://crbug.com/514510>)
Applied Workarounds: use_virtualized_gl_contexts
- * NVIDIA fails glReadPixels from incomplete cube map texture:
(<http://crbug.com/518889>)
Applied Workarounds: force_cube_complete
- * Pack parameters work incorrectly with pack buffer bound:
(<http://crbug.com/563714>)
Applied Workarounds: pack_parameters_workaround_with_pack_buffer
- * Framebuffer discarding can hurt performance on non-tilers:
(<http://crbug.com/570897>)

Applied Workarounds: disable_discard_framebuffer

* Unpacking overlapping rows from unpack buffers is unstable on NVIDIA GL driver:

(<http://crbug.com/596774>)

Applied Workarounds: unpack_overlapping_rows_separately_unpack_buffer

* Disable KHR_blend_equation_advanced until cc shaders are updated:

(<http://crbug.com/661715>)

Applied Workarounds: disable(GL_KHR_blend_equation_advanced),
disable(GL_KHR_blend_equation_advanced_coherent)

* Expose WebGL's disjoint_timer_query extensions on platforms with site isolation:

(<http://crbug.com/808744>), (<http://crbug.com/870491>)

Applied Workarounds: enable_webgl_timer_query_extensions

* Some drivers can't recover after OUT_OF_MEM and context lost:

(<http://crbug.com/893177>)

Applied Workarounds: exit_on_context_lost

* adjust src/dst region if blitting pixels outside framebuffer on Linux NVIDIA:

(<http://crbug.com/830046>)

Applied Workarounds: adjust_src_dst_region_for_blitframebuffer

* Disable GL_MESA_framebuffer_flip_y for desktop GL:

(<http://crbug.com/964010>)

Applied Workarounds: disable(GL_MESA_framebuffer_flip_y)

ANGLE Features

=====

* allowCompressedFormats (Frontend workarounds): Enabled

condition: true

Allow compressed formats

* cacheCompiledShader (Frontend features) (<http://anglebug.com/7036>): Enabled

condition: true

Enable to cache compiled shaders

* disableAnisotropicFiltering (Frontend workarounds): Disabled

Disable support for anisotropic filtering

* disableDrawBuffersIndexed (Frontend features) (<http://anglebug.com/7724>): Disabled

Disable support for OES_draw_buffers_indexed and EXT_draw_buffers_indexed

* disableProgramBinary (Frontend features) (<http://anglebug.com/5007>): Disabled

Disabled

Disable support for GL_OES_get_program_binary

* disableProgramCaching (Frontend features) (<http://anglebug.com/1423136>): Disabled

Disabled

Disables saving programs to the cache

* disableProgramCachingForTransformFeedback (Frontend workarounds): Disabled

On some GPUs, program binaries don't contain transform feedback varyings

* dumpShaderSource (Frontend features) (<http://anglebug.com/7760>): Disabled

Write shader source to temp directory

* dumpTranslatedShaders (Frontend features) (<http://anglebug.com/8280>): Disabled

Disabled

Write translated shaders to temp directory

- * emulatePixelLocalStorage (Frontend features) (<http://anglebug.com/7279>): Enabled
 - condition: true
 - Emulate ANGLE_shader_pixel_local_storage using shader images
- * enableCaptureLimits (Frontend features) (<http://anglebug.com/5750>): Disabled
 - Set the context limits like frame capturing was enabled
- * enableProgramBinaryForCapture (Frontend features) (<http://anglebug.com/5658>): Disabled
 - Even if FrameCapture is enabled, enable GL_OES_get_program_binary
- * enableShaderSubstitution (Frontend workarounds) (<http://anglebug.com/7761>): Disabled
 - Check the filesystem for shaders to use instead of those provided through glShaderSource
- * enableTranslatedShaderSubstitution (Frontend workarounds) (<http://anglebug.com/8280>): Disabled
 - Check the filesystem for translated shaders to use instead of the shader translator's
- * forceDepthAttachmentInitOnClear (Frontend workarounds) (<https://anglebug.com/7246>): Disabled
 - Force depth attachment initialization on clear ops
- * forceGLErrorChecking (Frontend features) (<https://issuetracker.google.com/220069903>): Disabled
 - condition: (IsAndroid() && isSwiftShader)
 - Force GL error checking (i.e. prevent applications from disabling error checking)
- * forceInitShaderVariables (Frontend features): Disabled
 - Force-enable shader variable initialization
- * forceMinimumMaxVertexAttributes (Frontend features): Disabled
 - condition: false
 - Force the minimum GL_MAX_VERTEX_ATTRIBS that the context's client version allows.
- * forceRobustResourceInit (Frontend features) (<http://anglebug.com/6041>): Disabled
 - Force-enable robust resource init
- * linkJobIsThreadSafe (Frontend features) (<http://anglebug.com/8297>): Enabled
 - condition: true
 - If false, parts of the link job cannot be parallelized
- * loseContextOnOutOfMemory (Frontend workarounds): Enabled
 - condition: true
 - Some users rely on a lost context notification if a GL_OUT_OF_MEMORY error occurs
- * singleThreadedTextureDecompression (Frontend workarounds): Disabled
 - Disables multi-threaded decompression of compressed texture formats
- * uncurrentEglSurfaceUponSurfaceDestroy (Frontend workarounds) (<https://issuetracker.google.com/292285899>): Enabled
 - condition: true
 - Make egl surface uncurrent when calling eglDestroySurface(), if the surface is still bound by the context of current render thread
- * adjustClearColorPrecision (Vulkan workarounds)

(<https://issuetracker.google.com/292282210>): Disabled
condition: IsAndroid() && mFeatures.supportsLegacyDithering.enabled && isARM
Adjust normalized clear color precision based on framebuffer color channel bits count

- * allocateNonZeroMemory (Vulkan features) (<http://anglebug.com/4384>): Disabled
condition: false
Fill new allocations with non-zero values to flush out errors.
- * allowGenerateMipmapWithCompute (Vulkan features) (<http://anglebug.com/4551>): Enabled
condition: supportsSubgroupQuadOpsInComputeShader && mSubgroupExtendedTypesFeatures.shaderSubgroupExtendedTypes && maxComputeWorkGroupInvocations >= 256 && ((isAMD && !IsWindows()) || isNvidia || isSamsung)
Use the compute path to generate mipmaps on devices that meet the minimum requirements, and the performance is better.
- * allowHostImageCopyDespiteNonIdenticalLayout (Vulkan features): Disabled
condition: false
When using VK_EXT_host_image_copy, allow VK_IMAGE_USAGE_HOST_TRANSFER_BIT_EXT even if perf query indicates only optimalDeviceAccess, but not identicalMemoryLayout
- * allowMultisampledRenderToTextureEmulation (Vulkan workarounds) (<http://anglebug.com/8291>): Disabled
condition: isTileBasedRenderer || isSamsung
Allow emulation of EXT_multisampled_render_to_texture
- * appendAliasedMemoryDecorations (Vulkan workarounds) (b/266235549): Enabled
condition: true
Append aliased memory decoration to ssbo and image in SpirV if they are not declared with restrict memory qualifier in GLSL
- * asyncCommandBufferReset (Vulkan features) (<https://issuetracker.google.com/255411748>): Enabled
condition: true
Reset command buffer in async thread.
- * asyncCommandQueue (Vulkan features) (<http://anglebug.com/4324>): Disabled
condition: false
Use CommandQueue worker thread to dispatch work to GPU.
- * bottomLeftOriginPresentRegionRectangles (Vulkan workarounds): Disabled
condition: IsAndroid()
On some platforms present region rectangles are expected to have a bottom-left origin, instead of top-left origin as from spec
- * bresenhamLineRasterization (Vulkan features): Enabled
condition: mLineRasterizationFeatures.bresenhamLines == 1U
Enable Bresenham line rasterization via VK_EXT_line_rasterization extension
- * clampFragDepth (Vulkan workarounds) (<http://anglebug.com/3970>): Disabled
condition: isNvidia && !mFeatures.supportsDepthClampZeroOne.enabled
gl_FragDepth is not clamped when rendering to a floating point depth buffer without VK_EXT_depth_clamp_zero_one
- * clampPointSize (Vulkan workarounds) (<http://anglebug.com/2970>): Disabled
condition: isNvidia && nvidiaVersion.major < uint32_t(IsWindows()) ? 430 : 421)
The point size range reported from the API is inconsistent with the actual behavior

- * compressVertexData (Vulkan workarounds): Disabled
Compress vertex data to smaller data types when possible. Using this feature makes ANGLE non-conformant.
- * deferFlushUntilEndRenderPass (Vulkan workarounds)
(<https://issuetracker.google.com/issues/166475273>): Enabled
condition: !isQualcommProprietary
Allow glFlush to be deferred until renderpass ends
- * disableFlippingBlitWithCommand (Vulkan workarounds)
(<http://anglebug.com/3498>): Disabled
condition: IsAndroid() && isQualcommProprietary
vkCmdBlitImage with flipped coordinates blits incorrectly.
- * disallowMixedDepthStencilLoadOpNoneAndLoad (Vulkan workarounds)
(<http://anglebug.com/7370>): Disabled
condition: isARM && armDriverVersion < ARMDriverVersion(38, 1, 0)
Disallow use of LOAD_OP_NONE for only one of the depth or stencil aspects of a depth/stencil attachment
- * doubleDepthBiasConstantFactor (Vulkan workarounds): Disabled
condition: isIntel && !IsWindows()
Due to a Vulkan spec ambiguity, some drivers interpret depthBiasConstantFactor as half the expected value
- * eglColorspaceAttributePassthrough (Vulkan features)
(<https://anglebug.com/7319>): Disabled
condition: IsAndroid() && isSamsung
Support passthrough of EGL colorspace attribute values
- * emulateAdvancedBlendEquations (Vulkan features) (<http://anglebug.com/3586>): Disabled
condition: !mFeatures.supportsBlendOperationAdvanced.enabled && (isVenus || !isIntel)
Emulate GL_KHR_blend_equation_advanced
- * emulateDithering (Vulkan features) (<http://anglebug.com/6755>): Disabled
condition: IsAndroid() && !mFeatures.supportsLegacyDithering.enabled
Emulate OpenGL dithering
- * emulateR32fImageAtomicExchange (Vulkan workarounds)
(<http://anglebug.com/5535>): Enabled
condition: true
Emulate r32f images with r32ui to support imageAtomicExchange.
- * emulateTransformFeedback (Vulkan features) (<http://anglebug.com/3205>): Disabled
condition: !mFeatures.supportsTransformFeedbackExtension.enabled && vk::CanSupportTransformFeedbackEmulation(mPhysicalDeviceFeatures)
Emulate transform feedback as the VK_EXT_transform_feedback is not present.
- * emulatedPrerotation180 (Vulkan features) (<http://anglebug.com/4901>): Disabled
Emulate 180-degree prerotation.
- * emulatedPrerotation270 (Vulkan features) (<http://anglebug.com/4901>): Disabled
Emulate 270-degree prerotation.
- * emulatedPrerotation90 (Vulkan features) (<http://anglebug.com/4901>): Disabled
Emulate 90-degree prerotation.
- * enableAsyncPipelineCacheCompression (Vulkan workarounds)

```

(http://anglebug.com/4722): Enabled
    condition: true
    Enable compressing pipeline cache in a thread.

*   enableMultisampledRenderToTexture (Vulkan workarounds)
(http://anglebug.com/4937): Disabled
    condition: mFeatures.supportsMultisampledRenderToSingleSampled.enabled ||
mFeatures.supportsMultisampledRenderToSingleSampledGOOGLEX.enabled ||
(supportsIndependentDepthStencilResolve &&
mFeatures.allowMultisampledRenderToTextureEmulation.enabled)
    Expose EXT_multisampled_render_to_texture

*   enableParallelCompileAndLink (Vulkan features) (http://anglebug.com/8297):
Disabled
    Expose the GL_KHR_parallel_shader_compile extension

*   enablePipelineCacheDataCompression (Vulkan features)
(https://issuetracker.google.com/258207403): Enabled
    condition: true
    enable pipeline cache data compression.

*   enablePortabilityEnumeration (Vulkan workarounds)
(http://anglebug.com/8229): Disabled
    condition: mFeatures.supportsPortabilityEnumeration.enabled && IsApple()
    Enable use of VK_KHR_portability_enumeration extension

*   enablePreRotateSurfaces (Vulkan features) (http://anglebug.com/3502):
Disabled
    condition: IsAndroid()
    Enable Android pre-rotation for landscape applications

*   enablePrecisionQualifiers (Vulkan features) (http://anglebug.com/3078):
Enabled
    condition: !(IsPixel2(mPhysicalDeviceProperties.vendorID,
mPhysicalDeviceProperties.deviceID) && (mPhysicalDeviceProperties.driverVersion
< kPixel2DriverWithRelaxedPrecision)) && !
IsPixel4(mPhysicalDeviceProperties.vendorID, mPhysicalDeviceProperties.deviceID)
    Enable precision qualifiers in shaders

*   explicitlyCastMediumFloatTo16Bit (Vulkan workarounds)
(https://issuetracker.google.com/274859104): Disabled
    condition: isARM && !isVenus
    Explicitly cast medium floating point values to 16 bit

*   explicitlyEnablePerSampleShading (Vulkan workarounds)
(http://anglebug.com/6876): Disabled
    condition: isARM
    Explicitly enable per-sample shading if the fragment shader contains the
sample qualifier

*   exposeNonConformantExtensionsAndVersions (Vulkan workarounds)
(http://anglebug.com/5375): Disabled
    condition: kExposeNonConformantExtensionsAndVersions && !isVenus
    Expose GLES versions and extensions that are not conformant.

*   forceContinuousRefreshOnSharedPresent (Vulkan features)
(https://issuetracker.google.com/229267970): Disabled
    condition: false
    Force to create vulkan swapchain with continuous refresh on shared present

*   forceD16TexFilter (Vulkan workarounds) (http://anglebug.com/3452): Disabled
    condition: IsAndroid() && isQualcommProprietary
    VK_FORMAT_D16_UNORM does not support
    VK_FORMAT_FEATURE_SAMPLED_IMAGE_FILTER_LINEAR_BIT, which prevents

```

OES_depth_texture from being supported.

- * forceDelayedDeviceCreationForTesting (Vulkan workarounds)
(<https://anglebug.com/8300>): Disabled
Artificially defer device creation to after surface is enabled for testing multi-queue scenarios
- * forceDisableFullScreenExclusive (Vulkan workarounds)
(<http://anglebug.com/8215>): Disabled
Device needs VK_EXT_full_screen_exclusive explicitly disabled
- * forceFallbackFormat (Vulkan workarounds): Disabled
Force a fallback format for angle_end2end_tests
- * forceFragmentShaderPrecisionHighpToMediump (Vulkan workarounds)
(<https://issuetracker.google.com/184850002>): Disabled
condition: false
Forces highp precision in fragment shader to mediump.
- * forceMaxUniformBufferSize16KB (Vulkan workarounds)
(<https://issuetracker.google.com/161903006>): Disabled
condition: isQualcommProprietary && isAdreno540
Force max uniform buffer size to 16K on some device due to bug
- * forceNearestFiltering (Vulkan workarounds): Disabled
Force nearest filtering when sampling.
- * forceNearestMipFiltering (Vulkan workarounds): Disabled
Force nearest mip filtering when sampling.
- * forceSubmitImmutableTextureUpdates (Vulkan app workarounds)
(<http://anglebug.com/6929>): Disabled
Force submit updates to immutable textures
- * forceTextureLodOffset1 (Vulkan workarounds): Disabled
Increase the minimum texture level-of-detail by 1 when sampling.
- * forceTextureLodOffset2 (Vulkan workarounds): Disabled
Increase the minimum texture level-of-detail by 2 when sampling.
- * forceTextureLodOffset3 (Vulkan workarounds): Disabled
Increase the minimum texture level-of-detail by 3 when sampling.
- * forceTextureLodOffset4 (Vulkan workarounds): Disabled
Increase the minimum texture level-of-detail by 4 when sampling.
- * forceWaitForSubmissionToCompleteForQueryResult (Vulkan workarounds)
(<https://issuetracker.google.com/253522366>): Disabled
condition: isARM || (isNvidia && nvidiaVersion.major < 470u)
Force wait for submission to complete before calling getQueryResult(wait).
- * hasEffectivePipelineCacheSerialization (Vulkan features)
(<https://anglebug.com/7369>): Enabled
condition: !isSwiftShader && !nvVersionLessThan520
Whether the implementation serializes the Vulkan pipeline cache effectively. On some implementations, pipeline cache serialization returns no data, so there is no benefit to serializing it
- * limitSampleCountTo2 (Vulkan workarounds) (<http://anglebug.com/8162>): Disabled
Limit sample count to 2 to save memory on low end devices.
- * logMemoryReportCallbacks (Vulkan features): Disabled
condition: false

Log each callback from VK_EXT_device_memory_report

- * logMemoryReportStats (Vulkan features): Disabled
condition: false
Log stats from VK_EXT_device_memory_report each swap
- * mapUnspecifiedColorSpaceToPassThrough (Vulkan features): Disabled
condition: isVenus
Use VK_COLOR_SPACE_PASS_THROUGH_EXT for EGL_NONE or unspecified color spaces
- * mergeProgramPipelineCachesToGlobalCache (Vulkan workarounds) (<https://anglebug.com/7369>): Disabled
condition: !mFeatures.supportsGraphicsPipelineLibrary.enabled || (mFeatures.preferMonolithicPipelinesOverLibraries.enabled && libraryBlobsAreReusedByMonolithicPipelines)
Whether it's beneficial to merge the pipeline cache for the shaders subset of the pipeline into the monolithic pipeline cache. Only useful on platforms where monolithic pipelines can reuse blobs from partial pipelines
- * mutableMipmapTextureUpload (Vulkan features) (<https://anglebug.com/7308>): Disabled
condition:
canPreferDeviceLocalMemoryHostVisible(mPhysicalDeviceProperties.deviceType)
Enable uploading the previously defined mutable mipmap texture.
- * overrideSurfaceFormatRGB8ToRGBA8 (Vulkan workarounds) (<http://anglebug.com/6651>): Enabled
condition: true
Override surface format GL_RGB8 to GL_RGBA8
- * padBuffersToMaxVertexAttribStride (Vulkan workarounds) (<http://anglebug.com/4428>): Disabled
condition: isAMD || isSamsung
Vulkan considers vertex attribute accesses to count up to the last multiple of the stride. This additional access supports AMD's robust buffer access implementation. AMDVLK in particular will return incorrect values when the vertex access extends into the range that would be the stride padding and the buffer is too small. This workaround limits GL_MAX_VERTEX_ATTRIB_STRIDE to a maximum value and pads up every buffer allocation size to be a multiple of the maximum stride.
- * perFrameWindowSizeQuery (Vulkan workarounds) (<http://anglebug.com/3623>, <http://anglebug.com/3624>, <http://anglebug.com/3625>): Disabled
condition: IsAndroid() || isIntel || (IsWindows() && isAMD) || IsFuchsia() || isSamsung || displayVk->isWayland()
Vulkan swapchain is not returning VK_ERROR_OUT_OF_DATE when window resizing
- * permanentlySwitchToFramebufferFetchMode (Vulkan features): Disabled
condition: isTileBasedRenderer
Whether the context should permanently switch to framebuffer fetch mode on first encounter
- * persistentlyMappedBuffers (Vulkan features) (<http://anglebug.com/2162>): Enabled
condition: true
Persistently map buffer memory to reduce map/unmap IOCTL overhead.
- * preferAggregateBarrierCalls (Vulkan workarounds) (<http://anglebug.com/4633>): Enabled
condition: isImmediateModeRenderer
Single barrier call is preferred over multiple calls with fine grained

pipeline stage dependency information

- * preferCPUForBufferSubData (Vulkan features)
(<http://issuetracker.google.com/200067929>): Disabled
condition: isMaliJobManagerBasedGPU
Prefer use CPU to do bufferSubData instead of staged update.
- * preferDeviceLocalMemoryHostVisible (Vulkan features)
(<http://anglebug.com/7047>): Disabled
condition:
canPreferDeviceLocalMemoryHostVisible(mPhysicalDeviceProperties.deviceType)
Prefer adding HOST_VISIBLE flag for DEVICE_LOCAL memory when picking memory types
- * preferDrawClearOverVkCmdClearAttachments (Vulkan workarounds)
(<https://issuetracker.google.com/166809097>): Disabled
condition: isQualcommProprietary
On some hardware, clear using a draw call instead of vkCmdClearAttachments in the middle of render pass due to bugs
- * preferDriverUniformOverSpecConst (Vulkan features)
(<http://anglebug.com/7406>): Disabled
condition: (isQualcommProprietary && mPhysicalDeviceProperties.driverVersion < kPixel4DriverWithWorkingSpecConstSupport) || isARM || isPowerVR || isSwiftShader
Prefer using driver uniforms instead of specialization constants.
- * preferHostCachedForNonStaticBufferUsage (Vulkan features)
(<https://issuetracker.google.com/288119108>): Disabled
prefer host cached memory for non static buffer usage
- * preferLinearFilterForYUV (Vulkan features) (<https://anglebug.com/7382>): Disabled
condition: isVenus
Prefer to use VK_FILTER_LINEAR for VkSamplerYcbcrConversion
- * preferMonolithicPipelinesOverLibraries (Vulkan workarounds)
(<https://anglebug.com/7369>): Disabled
condition: !
mGraphicsPipelineLibraryProperties.graphicsPipelineLibraryFastLinking || isSwiftShader
Whether monolithic pipelines perform significantly better than libraries
- * preferSkippingInvalidateForEmulatedFormats (Vulkan workarounds)
(<http://anglebug.com/6860>): Enabled
condition: isImmediateModeRenderer
Skipping invalidate is preferred for emulated formats that have extra channels over re-clearing the image
- * preferSubmitAtFBOBoundary (Vulkan workarounds)
(<https://issuetracker.google.com/187425444>): Disabled
condition: isARM || isSwiftShader || isVenus
Submit commands to driver at each FBO boundary for performance improvements.
- * preferSubmitOnAnySamplesPassedQueryEnd (Vulkan workarounds)
(<https://issuetracker.google.com/250706693>): Disabled
condition: isTileBasedRenderer
Submit commands to driver when last GL_ANY_SAMPLES_PASSED query is made for performance improvements.
- * provokingVertex (Vulkan features): Enabled
condition: mProvokingVertexFeatures.provokingVertexLast == 1U
Enable provoking vertex mode via VK_EXT_provoking_vertex extension

- * retainSPIRVDebugInfo (Vulkan features) (<http://anglebug.com/5901>): Disabled
condition: getEnableValidationLayers()
Retain debug info in SPIR-V blob.
- * roundOutputAfterDithering (Vulkan workarounds) (<http://anglebug.com/6953>): Disabled
condition: isQualcomm
Round output after dithering to workaround a driver bug that rounds the output up
- * slowAsyncCommandQueueForTesting (Vulkan workarounds) (<https://anglebug.com/6574>): Disabled
Artificially slow down async command queue for threading testing
- * slowDownMonolithicPipelineCreationForTesting (Vulkan workarounds) (<https://anglebug.com/7369>): Disabled
Artificially slow down async monolithic pipeline creation for threading testing
- * supportsAndroidHardwareBuffer (Vulkan features): Disabled
VkDevice supports the VK_ANDROID_external_memory_android_hardware_buffer extension
- * supportsAndroidNativeFenceSync (Vulkan features) (<http://anglebug.com/2517>): Enabled
condition: (mFeatures.supportsExternalFenceFd.enabled && FencePropertiesCompatibleWithAndroid(externalFenceProperties) && mFeatures.supportsExternalSemaphoreFd.enabled && SemaphorePropertiesCompatibleWithAndroid(externalSemaphoreProperties))
VkDevice supports the EGL_ANDROID_native_fence_sync extension
- * supportsBindMemory2 (Vulkan features) (<https://anglebug.com/4966>): Enabled
condition: true
VkDevice supports the VK_KHR_bind_memory2 extension
- * supportsBlendOperationAdvanced (Vulkan features) (<http://anglebug.com/3586>): Enabled
condition: ExtensionFound("VK_EXT_blend_operation_advanced", deviceExtensionNames)
VkDevice supports VK_EXT_blend_operation_advanced extension.
- * supportsColorWriteEnable (Vulkan features) (<https://anglebug.com/7161>): Disabled
VkDevice supports VK_EXT_color_write_enable extension
- * supportsComputeTranscodeEtcToBc (Vulkan features): Disabled
condition: !mPhysicalDeviceFeatures.textureCompressionETC2 && kSupportTranscodeEtcToBc && (mSubgroupProperties.supportedOperations & kRequiredSubgroupOp) == kRequiredSubgroupOp && (limitsVk.maxTexelBufferElements >= kMaxTexelBufferSize)
supports compute shader transcode etc format to bc format
- * supportsCustomBorderColor (Vulkan features) (<http://anglebug.com/3577>): Enabled
condition: mCustomBorderColorFeatures.customBorderColors == 1U && mCustomBorderColorFeatures.customBorderColorWithoutFormat == 1U
VkDevice supports the VK_EXT_custom_border_color extension
- * supportsDepthClampZeroOne (Vulkan features) (<http://anglebug.com/3970>): Enabled
condition: mDepthClampZeroOneFeatures.depthClampZeroOne == 1U
VkDevice supports the VK_EXT_depth_clamp_zero_one extension

- * supportsDepthClipControl (Vulkan features) (<http://anglebug.com/5421>):
Enabled
condition: mDepthClipControlFeatures.depthClipControl == 1U
VkDevice supports VK_EXT_depth_clip_control extension.
- * supportsDepthClipEnable (Vulkan features) (<http://anglebug.com/3970>):
Enabled
condition: mDepthClipEnableFeatures.depthClipEnable == 1U
VkDevice supports the VK_EXT_depth_clip_enable extension.
- * supportsDepthStencilResolve (Vulkan features) (<http://anglebug.com/4836>):
Enabled
condition: mFeatures.supportsRenderpass2.enabled &&
mDepthStencilResolveProperties.supportedDepthResolveModes != 0
VkDevice supports the VK_KHR_depth_stencil_resolve extension with the
independentResolveNone feature
- * supportsExtendedDynamicState (Vulkan features) (<http://anglebug.com/5906>):
Enabled
condition: mExtendedDynamicStateFeatures.extendedDynamicState == 1U &&
dynamicStateWorks
VkDevice supports VK_EXT_extended_dynamic_state extension
- * supportsExtendedDynamicState2 (Vulkan features) (<http://anglebug.com/5906>):
Enabled
condition: mExtendedDynamicState2Features.extendedDynamicState2 == 1U &&
dynamicStateWorks
VkDevice supports VK_EXT_extended_dynamic_state2 extension
- * supportsExternalFenceCapabilities (Vulkan features): Enabled
condition: true
VkInstance supports the VK_KHR_external_fence_capabilities extension
- * supportsExternalFenceFd (Vulkan features) (<http://anglebug.com/2517>):
Enabled
condition: ExtensionFound("VK_KHR_external_fence_fd", deviceExtensionNames)
VkDevice supports the VK_KHR_external_fence_fd extension
- * supportsExternalFormatResolve (Vulkan features): Disabled
condition: false
VkDevice supports the VK_ANDROID_external_format_resolve extension
- * supportsExternalMemoryDmaBufAndModifiers (Vulkan features)
(<http://anglebug.com/6248>): Enabled
condition: ExtensionFound("VK_EXT_external_memory_dma_buf",
deviceExtensionNames) && ExtensionFound("VK_EXT_image_drm_format_modifier",
deviceExtensionNames)
VkDevice supports the VK_EXT_external_memory_dma_buf and
VK_EXT_image_drm_format_modifier extensions
- * supportsExternalMemoryFd (Vulkan features): Enabled
condition: ExtensionFound("VK_KHR_external_memory_fd", deviceExtensionNames)
VkDevice supports the VK_KHR_external_memory_fd extension
- * supportsExternalMemoryFuchsia (Vulkan features): Disabled
condition: ExtensionFound("VK_FUCHSIA_external_memory",
deviceExtensionNames)
VkDevice supports the VK_FUCHSIA_external_memory extension
- * supportsExternalMemoryHost (Vulkan features): Disabled
VkDevice supports the VK_EXT_external_memory_host extension
- * supportsExternalSemaphoreCapabilities (Vulkan features): Enabled
condition: true

VkInstance supports the VK_KHR_external_semaphore_capabilities extension

- * supportsExternalSemaphoreFd (Vulkan features): Enabled
condition: ExtensionFound("VK_KHR_external_semaphore_fd",
deviceExtensionNames)
VkDevice supports the VK_KHR_external_semaphore_fd extension
- * supportsExternalSemaphoreFuchsia (Vulkan features): Disabled
condition: ExtensionFound("VK_FUCHSIA_external_semaphore",
deviceExtensionNames)
VkDevice supports the VK_FUCHSIA_external_semaphore extension
- * supportsFilteringPrecision (Vulkan features): Disabled
condition: ExtensionFound("VK_GOOGLE_sampler_filtering_precision",
deviceExtensionNames)
VkDevice supports the VK_GOOGLE_sampler_filtering_precision extension
- * supportsFormatFeatureFlags2 (Vulkan features): Enabled
condition: ExtensionFound("VK_KHR_format_feature_flags2",
deviceExtensionNames)
VkDevice supports the VK_KHR_format_feature_flags2 extension
- * supportsFragmentShaderPixelInterlock (Vulkan features): Enabled
condition: mFragmentShaderInterlockFeatures.fragmentShaderPixelInterlock ==
1U
VkDevice supports the VK_EXT_fragment_shader_interlock extension and has
the fragmentShaderPixelInterlock feature
- * supportsFragmentShadingRate (Vulkan features) (<http://anglebug.com/7172>):
Enabled
condition: canSupportFragmentShadingRate(deviceExtensionNames)
VkDevice supports VK_KHR_fragment_shading_rate extension
- * supportsFullScreenExclusive (Vulkan features) (<http://anglebug.com/8215>):
Disabled
VkDevice supports the VK_EXT_full_screen_exclusive extension
- * supportsGGPFrameToken (Vulkan features): Disabled
VkDevice supports the VK_GGP_frame_token extension
- * supportsGeometryStreamsCapability (Vulkan features)
(<http://anglebug.com/3206>): Enabled
condition: mFeatures.supportsTransformFeedbackExtension.enabled &&
mTransformFeedbackFeatures.geometryStreams == 1U
Implementation supports the GeometryStreams SPIR-V capability.
- * supportsGetMemoryRequirements2 (Vulkan features)
(<https://anglebug.com/4830>): Enabled
condition: true
VkDevice supports the VK_KHR_get_memory_requirements2 extension
- * supportsGraphicsPipelineLibrary (Vulkan features)
(<https://anglebug.com/7369>): Enabled
condition: mGraphicsPipelineLibraryFeatures.graphicsPipelineLibrary == 1U &&
(!isNvidia || nvidiaVersion.major >= 531) && !isRADV
VkDevice supports the VK_EXT_graphics_pipeline_library extension
- * supportsHostImageCopy (Vulkan features): Disabled
condition: mHostImageCopyFeatures.hostImageCopy == 1U &&
mHostImageCopyProperties.identicalMemoryTypeRequirements && !IsFuchsia()
VkDevice supports the VK_EXT_host_image_copy extension
- * supportsHostQueryReset (Vulkan features) (<http://anglebug.com/6692>): Enabled
condition: mHostQueryResetFeatures.hostQueryReset == 1U

VkDevice supports VK_EXT_host_query_reset extension

* supportsImage2dViewOf3d (Vulkan features) (<https://anglebug.com/7320>):

Enabled

condition: mImage2dViewOf3dFeatures.image2DViewOf3D == 1U

VkDevice supports VK_EXT_image_2d_view_of_3d

* supportsImageCubeArray (Vulkan features) (<http://anglebug.com/3584>): Enabled

condition: mPhysicalDeviceFeatures.imageCubeArray == 1U

VkDevice supports the imageCubeArray feature properly

* supportsImageFormatList (Vulkan features) (<http://anglebug.com/5281>):

Enabled

condition: ExtensionFound("VK_KHR_image_format_list", deviceExtensionNames)

Enable VK_IMAGE_CREATE_MUTABLE_FORMAT_BIT by default for ICDs that support

VK_KHR_image_format_list

* supportsImagelessFramebuffer (Vulkan features) (<http://anglebug.com/7553>):

Enabled

condition: mImagelessFramebufferFeatures.imagelessFramebuffer == 1U &&

(vk::RenderPassCommandBuffer::ExecutesInline() || !isSamsung)

VkDevice supports VK_KHR_imageless_framebuffer extension

* supportsIncrementalPresent (Vulkan features): Disabled

condition: ExtensionFound("VK_KHR_incremental_present",

deviceExtensionNames)

VkDevice supports the VK_KHR_incremental_present extension

* supportsIndexTypeUint8 (Vulkan features) (<http://anglebug.com/4405>): Enabled

condition: mIndexTypeUint8Features.indexTypeUint8 == 1U

VkDevice supports the VK_EXT_index_type_uint8 extension

* supportsLegacyDithering (Vulkan features)

(<https://issuetracker.google.com/284462263>): Disabled

condition: mDitheringFeatures.legacyDithering == 1U

VkDevice supports the VK_EXT_legacy_dithering extension

* supportsLockSurfaceExtension (Vulkan features): Disabled

condition: IsAndroid()

Surface supports the EGL_KHR_lock_surface3 extension

* supportsLogicOpDynamicState (Vulkan features) (<http://anglebug.com/3862>):

Enabled

condition: mFeatures.supportsExtendedDynamicState2.enabled &&

mExtendedDynamicState2Features.extendedDynamicState2LogicOp == 1U && !(IsLinux()

&& isIntel && isMesaLessThan22_2) && !(IsAndroid() && isGalaxyS23)

VkDevice supports the logicOp feature of VK_EXT_extended_dynamic_state2 extension

* supportsMemoryBudget (Vulkan features): Enabled

condition: ExtensionFound("VK_EXT_memory_budget", deviceExtensionNames)

VkDevice supports the VK_EXT_memory_budget extension.

* supportsMixedReadWriteDepthStencilLayouts (Vulkan features)

(<https://anglebug.com/7899>): Enabled

condition: true

VkDevice supports the mixed read and write depth/stencil layouts

introduced by VK_KHR_maintenance2

* supportsMultiDrawIndirect (Vulkan features) (<http://anglebug.com/6439>):

Enabled

condition: mPhysicalDeviceFeatures.multiDrawIndirect == 1U

VkDevice supports the multiDrawIndirect extension

```

*   supportsMultisampledRenderToSingleSampled (Vulkan features)
(http://anglebug.com/4836): Disabled
    condition: mFeatures.supportsRenderpass2.enabled &&
mFeatures.supportsDepthStencilResolve.enabled &&
mMultisampledRenderToSingleSampledFeatures.multisampledRenderToSingleSampled ==
1U
    VkDevice supports the VK_EXT_multisampled_render_to_single_sampled
    extension

*   supportsMultisampledRenderToSingleSampledGOOGLEX (Vulkan features)
(http://anglebug.com/4836): Disabled
    condition: !mFeatures.supportsMultisampledRenderToSingleSampled.enabled &&
mFeatures.supportsRenderpass2.enabled &&
mFeatures.supportsDepthStencilResolve.enabled &&
mMultisampledRenderToSingleSampledFeaturesGOOGLEX.multisampledRenderToSingleSam-
led == 1U
    VkDevice supports the VK_GOOGLEX_multisampled_render_to_single_sampled
    extension

*   supportsMultiview (Vulkan features) (http://anglebug.com/6048): Enabled
    condition: mMultiviewFeatures.multiview == 1U
    VkDevice supports the VK_KHR_multiview extension

*   supportsPipelineCreationCacheControl (Vulkan features)
(http://anglebug.com/5881): Enabled
    condition:
mPipelineCreationCacheControlFeatures.pipelineCreationCacheControl && !
isSwiftShader
    VkDevice supports VK_EXT_pipeline_creation_cache_control extension

*   supportsPipelineCreationFeedback (Vulkan features)
(http://anglebug.com/5881): Enabled
    condition: ExtensionFound("VK_EXT_pipeline_creation_feedback",
deviceExtensionNames)
    VkDevice supports VK_EXT_pipeline_creation_feedback extension

*   supportsPipelineProtectedAccess (Vulkan features)
(https://anglebug.com/7714): Disabled
    condition: mPipelineProtectedAccessFeatures.pipelineProtectedAccess == 1U &&
mProtectedMemoryFeatures.protectedMemory == 1U
    VkDevice supports the VK_EXT_pipeline_protected_access extension

*   supportsPipelineRobustness (Vulkan features) (https://anglebug.com/5845):
Enabled
    condition: mPipelineRobustnessFeatures.pipelineRobustness == 1U &&
mPhysicalDeviceFeatures.robustBufferAccess
    VkDevice supports VK_EXT_pipeline_robustness extension

*   supportsPipelineStatisticsQuery (Vulkan features)
(http://anglebug.com/5430): Enabled
    condition: mPhysicalDeviceFeatures.pipelineStatisticsQuery == 1U
    VkDevice supports the pipelineStatisticsQuery feature

*   supportsPortabilityEnumeration (Vulkan features) (http://anglebug.com/8229):
Enabled
    condition: ExtensionFound("VK_KHR_portability_enumeration",
instanceExtensionNames)
    Vulkan supports VK_KHR_portability_enumeration extension

*   supportsPresentation (Vulkan features): Enabled
    condition: !displayVk->isGBM()
    VkDisplay supports presentation through a present family queue

*   supportsPrimitiveTopologyListRestart (Vulkan features)

```

```

(http://anglebug.com/3832): Enabled
condition:
mPrimitiveTopologyListRestartFeatures.primitiveTopologyListRestart == 1U
VkDevice supports VK_EXT_primitive_topology_list_restart extension.

* supportsPrimitivesGeneratedQuery (Vulkan features)
(http://anglebug.com/5430): Enabled
condition: mFeatures.supportsTransformFeedbackExtension.enabled &&
mPrimitivesGeneratedQueryFeatures.primitivesGeneratedQuery == 1U
VkDevice supports VK_EXT_primitives_generated_query extension

* supportsProtectedMemory (Vulkan features) (http://anglebug.com/3965):
Disabled
condition: mProtectedMemoryFeatures.protectedMemory == 1U && (!isARM ||
mPipelineProtectedAccessFeatures.pipelineProtectedAccess == 1U)
VkDevice supports protected memory

* supportsRasterizationOrderAttachmentAccess (Vulkan features)
(https://anglebug.com/7604): Disabled
condition: !isQualcomm &&
mRasterizationOrderAttachmentAccessFeatures.rasterizationOrderColorAttachmentAccess == 1U
VkDevice supports VK_EXT_rasterization_order_attachment_access extension

* supportsRenderPassLoadStoreOpNone (Vulkan features)
(http://anglebug.com/5371): Enabled
condition: ExtensionFound("VK_EXT_load_store_op_none", deviceExtensionNames)
VkDevice supports VK_EXT_load_store_op_none extension.

* supportsRenderPassStoreOpNone (Vulkan features) (http://anglebug.com/5055):
Disabled
condition: !mFeatures.supportsRenderPassLoadStoreOpNone.enabled &&
ExtensionFound("VK_QCOM_render_pass_store_ops", deviceExtensionNames)
VkDevice supports VK_QCOM_render_pass_store_ops extension.

* supportsRenderpass2 (Vulkan features): Enabled
condition: ExtensionFound("VK_KHR_create_renderpass2", deviceExtensionNames)
VkDevice supports the VK_KHR_create_renderpass2 extension

* supportsSampler2dViewOf3d (Vulkan features) (https://anglebug.com/7320):
Enabled
condition: mFeatures.supportsImage2dViewOf3d.enabled &&
mImage2dViewOf3dFeatures.sampler2DViewOf3D == 1U
VkDevice supports the sampler2DViewOf3D feature of
VK_EXT_image_2d_view_of_3d

* supportsSamplerMirrorClampToEdge (Vulkan features): Enabled
condition: ExtensionFound("VK_KHR_sampler_mirror_clamp_to_edge",
deviceExtensionNames)
VkDevice supports the VK_KHR_sampler_mirror_clamp_to_edge extension

* supportsShaderFloat16 (Vulkan features) (http://anglebug.com/4551): Enabled
condition: mShaderFloat16Int8Features.shaderFloat16 == 1U
VkDevice supports the VK_KHR_shader_float16_int8 extension and has the
shaderFloat16 feature

* supportsShaderFramebufferFetch (Vulkan features): Disabled
condition: (IsAndroid()) && isARM) ||
mFeatures.supportsRasterizationOrderAttachmentAccess.enabled
Whether the Vulkan backend supports coherent framebuffer fetch

* supportsShaderFramebufferFetchNonCoherent (Vulkan features): Disabled
condition: (IsAndroid()) && !(isARM || isQualcomm)) || isSwiftShader
Whether the Vulkan backend supports non-coherent framebuffer fetch

```

- * supportsShaderStencilExport (Vulkan features): Disabled
condition: ExtensionFound("VK_EXT_shader_stencil_export",
deviceExtensionNames)
VkDevice supports the VK_EXT_shader_stencil_export extension
- * supportsSharedPresentableImageExtension (Vulkan features): Enabled
condition: ExtensionFound("VK_KHR_shared_presentable_image",
deviceExtensionNames)
VkSurface supports the VK_KHR_shared_presentable_images extension
- * supportsSurfaceCapabilities2Extension (Vulkan features): Enabled
condition: ExtensionFound("VK_KHR_get_surface_capabilities2",
instanceExtensionNames)
VkInstance supports the VK_KHR_get_surface_capabilities2 extension
- * supportsSurfaceProtectedCapabilitiesExtension (Vulkan features): Enabled
condition: ExtensionFound("VK_KHR_surface_protected_capabilities",
instanceExtensionNames)
VkInstance supports the VK_KHR_surface_protected_capabilities extension
- * supportsSurfaceProtectedSwapchains (Vulkan features): Disabled
condition: IsAndroid()
VkSurface supportsProtected for protected swapchains
- * supportsSurfacelessQueryExtension (Vulkan features): Disabled
condition: ExtensionFound("VK_GOOGLE_surfaceless_query",
instanceExtensionNames) && !isMockICDEnabled()
VkInstance supports the VK_GOOGLE_surfaceless_query extension
- * supportsSwapchainMaintenance1 (Vulkan features) (<https://anglebug.com/7847>): Enabled
condition: mSwapchainMaintenance1Features.swapchainMaintenance1 == 1U
VkDevice supports the VK_EXT_surface_maintenance1 and
VK_EXT_swapchain_maintenance1 extensions
- * supportsTimelineSemaphore (Vulkan features): Enabled
condition: mTimelineSemaphoreFeatures.timelineSemaphore == 1U
VkDevice supports the VK_KHR_timeline_semaphore extension
- * supportsTimestampSurfaceAttribute (Vulkan features)
(<https://anglebug.com/7489>): Disabled
condition: IsAndroid() && ExtensionFound("VK_GOOGLE_display_timing",
deviceExtensionNames)
Platform supports setting frame timestamp surface attribute
- * supportsTransformFeedbackExtension (Vulkan features)
(<http://anglebug.com/3206>): Enabled
condition:
vk::CanSupportTransformFeedbackExtension(mTransformFeedbackFeatures)
Transform feedback uses the VK_EXT_transform_feedback extension.
- * supportsVertexInputDynamicState (Vulkan features)
(<https://anglebug.com/7162>): Disabled
VkDevice supports VK_EXT_vertex_input_dynamic_state extension
- * supportsYUVSamplerConversion (Vulkan features): Enabled
condition: mSamplerYcbcrConversionFeatures.samplerYcbcrConversion != 0U
VkDevice supports the VK_KHR_sampler_ycbcr_conversion extension
- * supportsYuvTarget (Vulkan features): Disabled
VkDevice supports VK_ANDROID_render_to_external_format and
VK_EXT_ycbcr_attachment

- * `swapbuffersOnFlushOrFinishWithSingleBuffer` (Vulkan features)
(<http://anglebug.com/6878>): Disabled
condition: `IsAndroid()`
Bypass `deferredFlush` with calling `swapbuffers` on flush or finish when in Shared Present mode

- * `syncMonolithicPipelinesToBlobCache` (Vulkan workarounds)
(<https://anglebug.com/7369>): Disabled
condition: `mFeatures.hasEffectivePipelineCacheSerialization.enabled && (hasNoPipelineWarmUp || canSyncLargeMonolithicCache)`
Whether it's beneficial to store monolithic pipelines in the blob cache when `VK_EXT_graphics_pipeline_library` is in use. Otherwise the libraries are stored only, and monolithic pipelines are recreated on every run

- * `useCullModeDynamicState` (Vulkan workarounds) (<http://anglebug.com/5906>): Enabled
condition: `mFeatures.supportsExtendedDynamicState.enabled && dynamicStateWorks`
Use the Cull Mode dynamic state from `VK_EXT_extended_dynamic_state`

- * `useDepthBiasEnableDynamicState` (Vulkan workarounds)
(<http://anglebug.com/5906>): Enabled
condition: `mFeatures.supportsExtendedDynamicState2.enabled`
Use the Depth Bias Enable dynamic state from `VK_EXT_extended_dynamic_state2`

- * `useDepthCompareOpDynamicState` (Vulkan workarounds)
(<http://anglebug.com/5906>): Enabled
condition: `mFeatures.supportsExtendedDynamicState.enabled`
Use the Depth Compare Op dynamic state from `VK_EXT_extended_dynamic_state`

- * `useDepthTestEnableDynamicState` (Vulkan workarounds)
(<http://anglebug.com/5906>): Enabled
condition: `mFeatures.supportsExtendedDynamicState.enabled`
Use the Depth Test Enable dynamic state from `VK_EXT_extended_dynamic_state`

- * `useDepthWriteEnableDynamicState` (Vulkan workarounds)
(<http://anglebug.com/5906>): Enabled
condition: `mFeatures.supportsExtendedDynamicState.enabled && dynamicStateWorks`
Use the Depth Write Enable dynamic state from `VK_EXT_extended_dynamic_state`

- * `useFrontFaceDynamicState` (Vulkan workarounds) (<http://anglebug.com/5906>): Enabled
condition: `mFeatures.supportsExtendedDynamicState.enabled`
Use the Front Face dynamic state from `VK_EXT_extended_dynamic_state`

- * `useMultipleDescriptorsForExternalFormats` (Vulkan workarounds)
(<http://anglebug.com/6141>): Enabled
condition: `true`
Return a default descriptor count for external formats.

- * `useNonZeroStencilWriteMaskStaticState` (Vulkan workarounds)
(<http://anglebug.com/7556>): Disabled
condition: `isARM && armDriverVersion < ARMDriverVersion(43, 0, 0)`
Work around a driver bug where 0 in stencil write mask static state would make the corresponding dynamic state malfunction in the presence of discard or alpha to coverage

- * `usePrimitiveRestartEnableDynamicState` (Vulkan workarounds)
(<http://anglebug.com/5906>): Enabled
condition: `mFeatures.supportsExtendedDynamicState2.enabled && dynamicStateWorks`

Use the Primitive Restart Enable dynamic state from
VK_EXT_extended_dynamic_state2

- * useRasterizerDiscardEnableDynamicState (Vulkan workarounds)
(<http://anglebug.com/5906>): Enabled
condition: mFeatures.supportsExtendedDynamicState2.enabled
Use the Rasterizer Discard Enable dynamic state from
VK_EXT_extended_dynamic_state2
- * useResetCommandBufferBitForSecondaryPools (Vulkan workarounds): Disabled
condition: isARM
Use VK_COMMAND_POOL_CREATE_RESET_COMMAND_BUFFER_BIT for initializing
SecondaryCommandPools when using VulkanSecondaryCommandBuffer.
- * useStencilOpDynamicState (Vulkan workarounds) (<http://anglebug.com/5906>):
Enabled
condition: mFeatures.supportsExtendedDynamicState.enabled
Use the Stencil Op dynamic state from VK_EXT_extended_dynamic_state
- * useStencilTestEnableDynamicState (Vulkan workarounds)
(<http://anglebug.com/5906>): Enabled
condition: mFeatures.supportsExtendedDynamicState.enabled
Use the Stencil Test Enable dynamic state from
VK_EXT_extended_dynamic_state
- * useVertexInputBindingStrideDynamicState (Vulkan workarounds)
(<http://anglebug.com/5906>): Enabled
condition: mFeatures.supportsExtendedDynamicState.enabled &&
dynamicStateWorks
Use the Vertex Input Binding Stride dynamic state from
VK_EXT_extended_dynamic_state
- * useVmaForImageSuballocation (Vulkan features): Enabled
condition: true
Utilize VMA for image memory suballocation.
- * varyingsRequireMatchingPrecisionInSpirv (Vulkan workarounds)
(<http://anglebug.com/7488>): Disabled
condition: isPowerVR
Add additional SPIRV instructions to make sure precision between shader
stages match with each other
- * waitIdleBeforeSwapchainRecreation (Vulkan workarounds)
(<http://anglebug.com/5061>): Disabled
condition: IsAndroid() && isARM
Before passing an oldSwapchain to VkSwapchainCreateInfoKHR, wait for queue
to be idle. Works around a bug on platforms which destroy oldSwapchain in
vkCreateSwapchainKHR.
- * warmUpPipelineCacheAtLink (Vulkan features) (<http://anglebug.com/5881>):
Enabled
condition: libraryBlobsAreReusedByMonolithicPipelines && !
isQualcommProprietary && !(IsLinux() && isIntel) && !(IsChromeOS()) &&
isSwiftShader) && !isVenus
Warm up the Vulkan pipeline cache at link time

DAWN Info
=====

<Discrete GPU> Vulkan backend - NVIDIA GeForce RTX 3060 Laptop GPU

[WebGPU Status]

* Available

[Adapter Supported Features]

* depth-clip-control
* depth32float-stencil8
* texture-compression-bc
* indirect-first-instance
* rg11b10float-renderable
* bgra8unorm-storage
* float32filterable
* dawn-internal-usages
* dawn-multi-planar-formats
* dawn-native
* implicit-device-synchronization
* surface-capabilities
* transient-attachments
* norm16texture-formats

[Enabled Toggle Names]

* lazy_clear_resource_on_first_use:
(<https://crbug.com/dawn/145>):
Clears resource to zero on first usage. This initializes the resource so that no dirty bits from recycled memory is present in the new resource.

* use_temporary_buffer_in_texture_to_texture_copy:
(<https://crbug.com/dawn/42>):
Split texture-to-texture copy into two copies: copy from source texture into a temporary buffer, and copy from the temporary buffer into the destination texture when copying between compressed textures that don't have block-aligned sizes. This workaround is enabled by default on all Vulkan drivers to solve an issue in the Vulkan SPEC about the texture-to-texture copies with compressed formats. See #1005 (<https://github.com/KhronosGroup/Vulkan-Docs/issues/1005>) for more details.

* vulkan_use_d32s8:
(<https://crbug.com/dawn/286>):
Vulkan mandates support of either D32_FLOAT_S8 or D24_UNORM_S8. When available the backend will use D32S8 (toggle to on) but setting the toggle to off will make it use the D24S8 format when possible.

* vulkan_use_s8:
(<https://crbug.com/dawn/666>):
Vulkan has a pure stencil8 format but it is not universally available. When this toggle is on, the backend will use S8 for the stencil8 format, otherwise it will fallback to D32S8 or D24S8.

* disallow_spirv:
(<https://crbug.com/1214923>):
Disallow usage of SPIR-V completely so that only WGSL is used for shader modules. This is useful to prevent a Chromium renderer process from successfully sending SPIR-V code to be compiled in the GPU process.

* use_placeholder_fragment_in_vertex_only_pipeline:
(<https://crbug.com/dawn/136>):
Use a placeholder empty fragment shader in vertex only render pipeline. This toggle must be enabled for OpenGL ES backend, the Vulkan Backend, and serves as a workaround by default enabled on some Metal devices with Intel GPU to ensure the depth result is correct.

* timestamp_quantization:
(<https://crbug.com/dawn/1800>):

Enable timestamp queries quantization to reduce the precision of timers that can be created with timestamp queries.

- * use_vulkan_zero_initialize_workgroup_memory_extension:
(<https://crbug.com/dawn/1302>):
Initialize workgroup memory with OpConstantNull on Vulkan when the Vulkan extension VK_KHR_zero_initialize_workgroup_memory is supported.
- * vulkan_use_image_robust_access_2:
(<https://crbug.com/tint/1890>):
Disable Tint robustness transform on textures when VK_EXT_robustness2 is supported and robustImageAccess2 == VK_TRUE.
- * vulkan_use_buffer_robust_access_2:
(<https://crbug.com/tint/1890>):
Disable index clamping on the runtime-sized arrays on buffers in Tint robustness transform when VK_EXT_robustness2 is supported and robustBufferAccess2 == VK_TRUE.

[WebGPU Required Toggles - enabled]

- * disallow_spirv:
(<https://crbug.com/1214923>):
Disallow usage of SPIR-V completely so that only WGSL is used for shader modules. This is useful to prevent a Chromium renderer process from successfully sending SPIR-V code to be compiled in the GPU process.
- * timestamp_quantization:
(<https://crbug.com/dawn/1800>):
Enable timestamp queries quantization to reduce the precision of timers that can be created with timestamp queries.

<CPU> Vulkan backend - SwiftShader Device (Subzero)

[WebGPU Status]

- * Blocklisted

[Adapter Supported Features]

- * depth-clip-control
- * depth32float-stencil8
- * texture-compression-bc
- * texture-compression-etc2
- * texture-compression-astc
- * indirect-first-instance
- * rg11b10float-renderable
- * bgra8unorm-storage
- * float32filterable
- * dawn-internal-usages
- * dawn-multi-planar-formats
- * dawn-native
- * implicit-device-synchronization
- * surface-capabilities
- * transient-attachments

[Enabled Toggle Names]

- * lazy_clear_resource_on_first_use:
(<https://crbug.com/dawn/145>):
Clears resource to zero on first usage. This initializes the resource so that no dirty bits from recycled memory is present in the new resource.

- * `use_temporary_buffer_in_texture_to_texture_copy:`
<https://crbug.com/dawn/42>):
 Split texture-to-texture copy into two copies: copy from source texture into a temporary buffer, and copy from the temporary buffer into the destination texture when copying between compressed textures that don't have block-aligned sizes. This workaround is enabled by default on all Vulkan drivers to solve an issue in the Vulkan SPEC about the texture-to-texture copies with compressed formats. See #1005 (<https://github.com/KhronosGroup/Vulkan-Docs/issues/1005>) for more details.

- * `vulkan_use_d32s8:`
<https://crbug.com/dawn/286>):
 Vulkan mandates support of either `D32_FLOAT_S8` or `D24_UNORM_S8`. When available the backend will use `D32S8` (toggle to on) but setting the toggle to off will make it use the `D24S8` format when possible.

- * `vulkan_use_s8:`
<https://crbug.com/dawn/666>):
 Vulkan has a pure stencil8 format but it is not universally available. When this toggle is on, the backend will use `S8` for the stencil8 format, otherwise it will fallback to `D32S8` or `D24S8`.

- * `disallow_spirv:`
<https://crbug.com/1214923>):
 Disallow usage of SPIR-V completely so that only WGSL is used for shader modules. This is useful to prevent a Chromium renderer process from successfully sending SPIR-V code to be compiled in the GPU process.

- * `use_placeholder_fragment_in_vertex_only_pipeline:`
<https://crbug.com/dawn/136>):
 Use a placeholder empty fragment shader in vertex only render pipeline. This toggle must be enabled for OpenGL ES backend, the Vulkan Backend, and serves as a workaround by default enabled on some Metal devices with Intel GPU to ensure the depth result is correct.

- * `timestamp_quantization:`
<https://crbug.com/dawn/1800>):
 Enable timestamp queries quantization to reduce the precision of timers that can be created with timestamp queries.

- * `use_vulkan_zero_initialize_workgroup_memory_extension:`
<https://crbug.com/dawn/1302>):
 Initialize workgroup memory with `OpConstantNull` on Vulkan when the Vulkan extension `VK_KHR_zero_initialize_workgroup_memory` is supported.

[WebGPU Required Toggles - enabled]

-
- * `disallow_spirv:`
<https://crbug.com/1214923>):
 Disallow usage of SPIR-V completely so that only WGSL is used for shader modules. This is useful to prevent a Chromium renderer process from successfully sending SPIR-V code to be compiled in the GPU process.

 - * `timestamp_quantization:`
<https://crbug.com/dawn/1800>):
 Enable timestamp queries quantization to reduce the precision of timers that can be created with timestamp queries.

Version Information

=====

Data exported	: 2023-12-15T14:10:44.847Z
Chrome version	: Chrome/120.0.6099.109
Operating system	: Linux 6.6.6-arch1-1

Software rendering list URL:
https://chromium.googlesource.com/chromium/src/+3419140ab665596f21b385ce136419fde0924272/gpu/config/software_rendering_list.json
Driver bug list URL :
https://chromium.googlesource.com/chromium/src/+3419140ab665596f21b385ce136419fde0924272/gpu/config/gpu_driver_bug_list.json
ANGLE commit id : 4ae5f681dfe6
2D graphics backend : Skia/120 349c1179c43ef46f2804404952b9460dc007d76a
Command Line : /opt/google/chrome/google-chrome --gpu-testing-vendor-id=0x1002 --gpu-testing-device-id=0x1638 --flag-switches-begin --flag-switches-end --origin-trial-disabled-features=WebGPU

Driver Information

=====

Initialization time : 197
In-process GPU : false
Passthrough Command Decoder : true
Sandboxed : false
GPU0 : VENDOR= 0x1002, DEVICE=0x1638,
DRIVER_VENDOR=NVIDIA, DRIVER_VERSION=545.29.6.0
GPU1 : VENDOR= 0x10de, DEVICE=0x2520,
DRIVER_VENDOR=NVIDIA, DRIVER_VERSION=545.29.6.0 *ACTIVE*
Optimus : false
AMD switchable : false
GPU CUDA compute capability major version: 0
Pixel shader version : 1.00
Vertex shader version : 1.00
Max. MSAA samples : 8
Machine model name :
Machine model version :
GL implementation parts : (gl=egl-angle,angle=vulkan)
Display type : ANGLE_VULKAN
GL_VENDOR : Google Inc. (NVIDIA)
GL_RENDERER : ANGLE (NVIDIA, Vulkan 1.3.260 (NVIDIA NVIDIA GeForce RTX 3060 Laptop GPU (0x00002520)), NVIDIA-545.29.6.0)
GL_VERSION : OpenGL ES 2.0.0 (ANGLE 2.1.22152 git hash: 4ae5f681dfe6)
GL_EXTENSIONS : GL_AMD_performance_monitor
GL_ANGLE_base_vertex_base_instance
GL_ANGLE_base_vertex_base_instance_shader_builtin GL_ANGLE_client_arrays
GL_ANGLE_depth_texture GL_ANGLE_framebuffer_blit
GL_ANGLE_framebuffer_multisample GL_ANGLE_get_image
GL_ANGLE_get_serialized_context_string GL_ANGLE_get_tex_level_parameter
GL_ANGLE_instanced_arrays GL_ANGLE_logic_op GL_ANGLE_memory_object_flags
GL_ANGLE_memory_size GL_ANGLE_multi_draw GL_ANGLE_pack_reverse_row_order
GL_ANGLE_polygon_mode GL_ANGLE_program_cache_control
GL_ANGLE_read_only_depth_stencil_feedback_loops
GL_ANGLE_relaxed_vertex_attribute_type GL_ANGLE_request_extension
GL_ANGLE_rgbx_internal_format GL_ANGLE_robust_client_memory
GL_ANGLE_robust_fragment_shader_output GL_ANGLE_texture_compression_dxt3
GL_ANGLE_texture_compression_dxt5 GL_ANGLE_texture_usage GL_ANGLE_vulkan_image
GL_APPLE_clip_distance GL_CHROMIUM_bind_generates_resource
GL_CHROMIUM_bind_uniform_location GL_CHROMIUM_color_buffer_float_rgb
GL_CHROMIUM_color_buffer_float_rgba GL_CHROMIUM_copy_compressed_texture
GL_CHROMIUM_copy_texture GL_CHROMIUM_lose_context
GL_EXT_EGL_image_external_wrap_modes GL_EXT_base_instance
GL_EXT_blend_func_extended GL_EXT_blend_minmax GL_EXT_buffer_storage
GL_EXT_clip_control GL_EXT_color_buffer_half_float
GL_EXT_compressed_ETC1_RGB8_sub_texture GL_EXT_copy_image GL_EXT_debug_label
GL_EXT_debug_marker GL_EXT_depth_clamp GL_EXT_discard_framebuffer
GL_EXT_disjoint_timer_query GL_EXT_draw_buffers GL_EXT_draw_elements_base_vertex
GL_EXT_float_blend GL_EXT_frag_depth GL_EXT_instanced_arrays
GL_EXT_map_buffer_range GL_EXT_memory_object GL_EXT_memory_object_fd
GL_EXT_multi_draw_indirect GL_EXT_multisample_compatibility

GL_EXT_occlusion_query_boolean GL_EXT_polygon_offset_clamp
 GL_EXT_read_format_bgra GL_EXT_robustness GL_EXT_sRGB GL_EXT_sRGB_write_control
 GL_EXT_semaphore GL_EXT_semaphore_fd GL_EXT_separate_shader_objects
 GL_EXT_shader_non_constant_global_initializers GL_EXT_shader_texture_lod
 GL_EXT_shadow_samplers GL_EXT_texture_border_clamp
 GL_EXT_texture_compression_bptc GL_EXT_texture_compression_dxt1
 GL_EXT_texture_compression_rgtc GL_EXT_texture_compression_s3tc_srgb
 GL_EXT_texture_filter_anisotropic GL_EXT_texture_format_BGRA8888
 GL_EXT_texture_mirror_clamp_to_edge GL_EXT_texture_norm16 GL_EXT_texture_rg
 GL_EXT_texture_sRGB_decode GL_EXT_texture_storage
 GL_EXT_texture_type_2_10_10_10_REV GL_EXT_unpack_subimage
 GL_KHR_blend_equation_advanced GL_KHR_debug GL_NV_depth_buffer_float2
 GL_NV_fence GL_NV_framebuffer_blit GL_NV_pack_subimage GL_NV_pixel_buffer_object
 GL_NV_polygon_mode GL_NV_read_depth GL_NV_read_depth_stencil GL_NV_read_stencil
 GL_OES_EGL_image GL_OES_EGL_image_external GL_OES_EGL_sync
 GL_OES_compressed_EAC_R11_signed_texture
 GL_OES_compressed_EAC_R11_unsigned_texture
 GL_OES_compressed_EAC_RG11_signed_texture
 GL_OES_compressed_EAC_RG11_unsigned_texture GL_OES_compressed_ETC1_RGB8_texture
 GL_OES_compressed_ETC2_RGB8_texture GL_OES_compressed_ETC2_RGBA8_texture
 GL_OES_compressed_ETC2_punchthroughA_RGBA8_texture
 GL_OES_compressed_ETC2_punchthroughA_sRGB8_alpha_texture
 GL_OES_compressed_ETC2_sRGB8_alpha8_texture GL_OES_compressed_ETC2_sRGB8_texture
 GL_OES_depth24 GL_OES_depth32 GL_OES_depth_texture GL_OES_depth_texture_cube_map
 GL_OES_draw_elements_base_vertex GL_OES_element_index_uint
 GL_OES_fbo_render_mipmap GL_OES_get_program_binary GL_OES_mapbuffer
 GL_OES_packed_depth_stencil GL_OES_primitive_bounding_box GL_OES_rgb8_rgba8
 GL_OES_sample_shading GL_OES_standard_derivatives GL_OES_surfaceless_context
 GL_OES_texture_3D GL_OES_texture_border_clamp GL_OES_texture_float
 GL_OES_texture_float_linear GL_OES_texture_half_float
 GL_OES_texture_half_float_linear GL_OES_texture_npot GL_OES_vertex_array_object
 GL_OES_vertex_half_float GL_QCOM_shading_rate
 Disabled Extensions : GL_KHR_blend_equation_advanced
 GL_KHR_blend_equation_advanced_coherent GL_MESA_framebuffer_flip_y
 Disabled WebGL Extensions :
 Window system binding vendor : Google Inc. (NVIDIA)
 Window system binding version : 1.5 (ANGLE 2.1.22152 git hash: 4ae5f681dfe6)
 Window system binding extensions: EGL_EXT_create_context_robustness
 EGL_ANGLE_surface_orientation EGL_KHR_create_context EGL_KHR_image
 EGL_KHR_image_base EGL_EXT_image_gl_colorspace EGL_KHR_gl_colorspace
 EGL_KHR_gl_texture_2D_image EGL_KHR_gl_texture_cubemap_image
 EGL_KHR_gl_texture_3D_image EGL_KHR_gl_renderbuffer_image
 EGL_KHR_get_all_proc_addresses EGL_KHR_fence_sync EGL_KHR_wait_sync
 EGL_ANGLE_create_context_webgl_compatibility
 EGL_CHROMIUM_create_context_bind_generates_resource
 EGL_KHR_swap_buffers_with_damage EGL_EXT_pixel_format_float
 EGL_KHR_surfaceless_context EGL_ANGLE_display_texture_share_group
 EGL_ANGLE_display_semaphore_share_group EGL_ANGLE_create_context_client_arrays
 EGL_ANGLE_program_cache_control EGL_ANGLE_robust_resource_initialization
 EGL_ANGLE_create_context_extensions_enabled EGL_ANDROID_blob_cache
 EGL_ANDROID_recordable EGL_ANDROID_native_fence_sync
 EGL_ANGLE_create_context_backwards_compatible EGL_KHR_no_config_context
 EGL_IMG_context_priority EGL_KHR_create_context_no_error
 EGL_EXT_image_dma_buf_import EGL_EXT_image_dma_buf_import_modifiers
 EGL_KHR_reusable_sync EGL_EXT_buffer_age EGL_KHR_mutable_render_buffer
 EGL_ANGLE_create_surface_swap_interval EGL_ANGLE_vulkan_image
 EGL_KHR_partial_update
 XDG_CURRENT_DESKTOP : GNOME
 XDG_SESSION_TYPE : wayland
 GDMSESSION : gnome
 Ozone platform : x11
 Direct rendering version : unknown
 Reset notification strategy : 0x8252
 GPU process crash count : 1

gfx::BufferFormats supported for allocation and texturing: R_8: supported, R_16: supported, RG_88: supported, RG_1616: supported, BGR_565: supported, RGBA_4444: supported, RGBX_8888: supported, RGBA_8888: supported, BGRX_8888: supported, BGRA_1010102: supported, RGBA_1010102: supported, BGRA_8888: supported, RGBA_F16: supported, YVU_420: not supported, YUV_420_BIPLANAR: not supported, YUVA_420_TRIPLANAR: supported, P010: not supported

Compositor Information

=====

Tile Update Mode: One-copy

Partial Raster : Enabled

GpuMemoryBuffers Status

=====

R_8	: Software only
R_16	: Software only
RG_88	: Software only
RG_1616	: Software only
BGR_565	: Software only
RGBA_4444	: Software only
RGBX_8888	: Software only
RGBA_8888	: Software only
BGRX_8888	: Software only
BGRA_1010102	: Software only
RGBA_1010102	: Software only
BGRA_8888	: Software only
RGBA_F16	: Software only
YVU_420	: Software only
YUV_420_BIPLANAR	: Software only
YUVA_420_TRIPLANAR	: Software only
P010	: Software only

Display(s) Information

=====

Info	: Display[33] bounds=[0,0 1920x1080], workarea=[0,32 1920x1048], scale=1, rotation=0, panel_rotation=0 internal detected
Color space (all)	: {primaries:BT709, transfer:SRGB, matrix:RGB, range:FULL}
Buffer format (all)	: BGRA_8888
Color volume	: {name:'srgb', r:[0.6400, 0.3300], g:[0.3000, 0.6000], b:[0.1500, 0.3300], w:[0.3127, 0.3290]}
SDR white level in nits	: 203
HDR relative maximum luminance	: 1
Bits per color component	: 8
Bits per pixel	: 24
Refresh Rate in Hz	: 119.9301528930664

Video Acceleration Information

=====

Decoding:

Encoding:

Vulkan Information

=====

```
info: {
  "apiVersion": "1.3.268",
  "usedApiVersion": "1.1.0",
  "instanceExtensions": {
    "VK_KHR_device_group_creation": 1,
    "VK_KHR_display": 23,
    "VK_KHR_external_fence_capabilities": 1,
    "VK_KHR_external_memory_capabilities": 1,
    "VK_KHR_external_semaphore_capabilities": 1,
```



```

"VK_KHR_get_display_properties2": 1,
"VK_KHR_get_physical_device_properties2": 2,
"VK_KHR_get_surface_capabilities2": 1,
"VK_KHR_surface": 25,
"VK_KHR_surface_protected_capabilities": 1,
"VK_KHR_wayland_surface": 6,
"VK_KHR_xcb_surface": 6,
"VK_EXT_acquire_drm_display": 1,
"VK_EXT_debug_report": 10,
"VK_EXT_debug_utils": 2,
"VK_EXT_direct_mode_display": 1,
"VK_EXT_display_surface_counter": 1,
"VK_EXT_surface_maintenance1": 1,
"VK_EXT_swapchain_colorspace": 4,
"VK_KHR_portability_enumeration": 1,
"VK_LUNARG_direct_driver_loading": 1
},
"enabledInstanceExtensions": [
  "VK_EXT_debug_report",
  "VK_EXT_surface_maintenance1",
  "VK_EXT_swapchain_colorspace",
  "VK_KHR_get_surface_capabilities2",
  "VK_KHR_surface",
  "VK_KHR_surface_protected_capabilities",
  "VK_KHR_xcb_surface"
],
"instanceLayers": [
  {
    "layerName": "VK_LAYER_VALVE_steam_overlay_32",
    "specVersion": 4206799,
    "implementationVersion": "0.0.1",
    "description": "Steam Overlay Layer"
  },
  {
    "layerName": "VK_LAYER_VALVE_steam_overlay_64",
    "specVersion": 4206799,
    "implementationVersion": "0.0.1",
    "description": "Steam Overlay Layer"
  },
  {
    "layerName": "VK_LAYER_VALVE_steam_fossilize_32",
    "specVersion": 4206799,
    "implementationVersion": "0.0.1",
    "description": "Steam Pipeline Caching Layer"
  },
  {
    "layerName": "VK_LAYER_VALVE_steam_fossilize_64",
    "specVersion": 4206799,
    "implementationVersion": "0.0.1",
    "description": "Steam Pipeline Caching Layer"
  },
  {
    "layerName": "VK_LAYER_MANGOHUD_overlay_x86_64",
    "specVersion": 4206592,
    "implementationVersion": "0.0.1",
    "description": "Vulkan Hud Overlay"
  },
  {
    "layerName": "VK_LAYER_MANGOAPP_overlay",
    "specVersion": 4206592,
    "implementationVersion": "0.0.1",
    "description": "Mangoapp Layer"
  },
  {

```

```

    "layerName": "VK_LAYER_NV_optimus",
    "specVersion": 4206852,
    "implementationVersion": "0.0.1",
    "description": "NVIDIA Optimus layer"
  }
],
"physicalDevices": [
  {
    "properties": {
      "apiVersion": "1.3.260",
      "driverVersion": "-479.116.384",
      "vendorID": 4318,
      "deviceID": 9504,
      "deviceType": 2,
      "deviceName": "NVIDIA GeForce RTX 3060 Laptop GPU",
      "pipelineCacheUUID": "67de3fe5-2f6e-7d75-238d-8751348e200f",
      "limits": {
        "maxImageDimension1D": 32768,
        "maxImageDimension2D": 32768,
        "maxImageDimension3D": 16384,
        "maxImageDimensionCube": 32768,
        "maxImageArrayLayers": 2048,
        "maxTexelBufferElements": 134217728,
        "maxUniformBufferRange": 65536,
        "maxStorageBufferRange": 4294967295,
        "maxPushConstantsSize": 256,
        "maxMemoryAllocationCount": 4294967295,
        "maxSamplerAllocationCount": 4000,
        "bufferImageGranularity": 1,
        "sparseAddressSpaceSize": 1,
        "maxBoundDescriptorSets": 32,
        "maxPerStageDescriptorSamplers": 1048576,
        "maxPerStageDescriptorUniformBuffers": 1048576,
        "maxPerStageDescriptorStorageBuffers": 1048576,
        "maxPerStageDescriptorSampledImages": 1048576,
        "maxPerStageDescriptorStorageImages": 1048576,
        "maxPerStageDescriptorInputAttachments": 1048576,
        "maxPerStageResources": 4294967295,
        "maxDescriptorSetSamplers": 1048576,
        "maxDescriptorSetUniformBuffers": 1048576,
        "maxDescriptorSetUniformBuffersDynamic": 15,
        "maxDescriptorSetStorageBuffers": 1048576,
        "maxDescriptorSetStorageBuffersDynamic": 16,
        "maxDescriptorSetSampledImages": 1048576,
        "maxDescriptorSetStorageImages": 1048576,
        "maxDescriptorSetInputAttachments": 1048576,
        "maxVertexInputAttributes": 32,
        "maxVertexInputBindings": 32,
        "maxVertexInputAttributeOffset": 2047,
        "maxVertexInputBindingStride": 2048,
        "maxVertexOutputComponents": 128,
        "maxTessellationGenerationLevel": 64,
        "maxTessellationPatchSize": 32,
        "maxTessellationControlPerVertexInputComponents": 128,
        "maxTessellationControlPerVertexOutputComponents": 128,
        "maxTessellationControlPerPatchOutputComponents": 120,
        "maxTessellationControlTotalOutputComponents": 4216,
        "maxTessellationEvaluationInputComponents": 128,
        "maxTessellationEvaluationOutputComponents": 128,
        "maxGeometryShaderInvocations": 32,
        "maxGeometryInputComponents": 128,
        "maxGeometryOutputComponents": 128,
        "maxGeometryOutputVertices": 1024,
        "maxGeometryTotalOutputComponents": 1024,

```

```
"maxFragmentInputComponents": 128,
"maxFragmentOutputAttachments": 8,
"maxFragmentDualSrcAttachments": 1,
"maxFragmentCombinedOutputResources": 4294967295,
"maxComputeSharedMemorySize": 49152,
"maxComputeWorkGroupCount": [
  2147483647,
  65535,
  65535
],
"maxComputeWorkGroupInvocations": 1024,
"maxComputeWorkGroupSize": [
  1024,
  1024,
  64
],
"subPixelPrecisionBits": 8,
"subTexelPrecisionBits": 8,
"mipmapPrecisionBits": 8,
"maxDrawIndexedIndexValue": 4294967295,
"maxDrawIndirectCount": 4294967295,
"maxSamplerLodBias": 15,
"maxSamplerAnisotropy": 16,
"maxViewports": 16,
"maxViewportDimensions": [
  32768,
  32768
],
"viewportBoundsRange": [
  -65536,
  65536
],
"viewportSubPixelBits": 8,
"minMemoryMapAlignment": 64,
"minTexelBufferOffsetAlignment": 1,
"minUniformBufferOffsetAlignment": 1,
"minStorageBufferOffsetAlignment": 1,
"minTexelOffset": -8,
"maxTexelOffset": 7,
"minTexelGatherOffset": -32,
"maxTexelGatherOffset": 31,
"minInterpolationOffset": -0.5,
"maxInterpolationOffset": 0.4375,
"subPixelInterpolationOffsetBits": 4,
"maxFramebufferWidth": 32768,
"maxFramebufferHeight": 32768,
"maxFramebufferLayers": 2048,
"framebufferColorSampleCounts": 15,
"framebufferDepthSampleCounts": 15,
"framebufferStencilSampleCounts": 31,
"framebufferNoAttachmentsSampleCounts": 31,
"maxColorAttachments": 8,
"sampledImageColorSampleCounts": 15,
"sampledImageIntegerSampleCounts": 15,
"sampledImageDepthSampleCounts": 15,
"sampledImageStencilSampleCounts": 31,
"storageImageSampleCounts": 15,
"maxSampleMaskWords": 1,
"timestampComputeAndGraphics": true,
"timestampPeriod": 1,
"maxClipDistances": 8,
"maxCullDistances": 8,
"maxCombinedClipAndCullDistances": 8,
"discreteQueuePriorities": 2,
```

```

    "pointSizeRange": [
        1,
        2047.9375
    ],
    "lineWidthRange": [
        1,
        64
    ],
    "pointSizeGranularity": 0.0625,
    "lineWidthGranularity": 0.0625,
    "strictLines": true,
    "standardSampleLocations": true,
    "optimalBufferCopyOffsetAlignment": 1,
    "optimalBufferCopyRowPitchAlignment": 1,
    "nonCoherentAtomSize": 1
},
"sparseProperties": {
    "residencyStandard2DBlockShape": true,
    "residencyStandard2DMultisampleBlockShape": true,
    "residencyStandard3DBlockShape": true,
    "residencyAlignedMipSize": false,
    "residencyNonResidentStrict": true
}
},
"extensions": {
    "VK_KHR_16bit_storage": 1,
    "VK_KHR_8bit_storage": 1,
    "VK_KHR_acceleration_structure": 13,
    "VK_KHR_bind_memory2": 1,
    "VK_KHR_buffer_device_address": 1,
    "VK_KHR_cooperative_matrix": 2,
    "VK_KHR_copy_commands2": 1,
    "VK_KHR_create_renderpass2": 1,
    "VK_KHR_dedicated_allocation": 3,
    "VK_KHR_deferred_host_operations": 4,
    "VK_KHR_depth_stencil_resolve": 1,
    "VK_KHR_descriptor_update_template": 1,
    "VK_KHR_device_group": 4,
    "VK_KHR_draw_indirect_count": 1,
    "VK_KHR_driver_properties": 1,
    "VK_KHR_dynamic_rendering": 1,
    "VK_KHR_external_fence": 1,
    "VK_KHR_external_fence_fd": 1,
    "VK_KHR_external_memory": 1,
    "VK_KHR_external_memory_fd": 1,
    "VK_KHR_external_semaphore": 1,
    "VK_KHR_external_semaphore_fd": 1,
    "VK_KHR_format_feature_flags2": 2,
    "VK_KHR_fragment_shader_barycentric": 1,
    "VK_KHR_fragment_shading_rate": 2,
    "VK_KHR_get_memory_requirements2": 1,
    "VK_KHR_global_priority": 1,
    "VK_KHR_image_format_list": 1,
    "VK_KHR_imageless_framebuffer": 1,
    "VK_KHR_maintenance1": 2,
    "VK_KHR_maintenance2": 1,
    "VK_KHR_maintenance3": 1,
    "VK_KHR_maintenance4": 2,
    "VK_KHR_maintenance5": 1,
    "VK_KHR_map_memory2": 1,
    "VK_KHR_multiview": 1,
    "VK_KHR_pipeline_executable_properties": 1,
    "VK_KHR_pipeline_library": 1,
    "VK_KHR_present_id": 1,

```

"VK_KHR_present_wait": 1,
"VK_KHR_push_descriptor": 2,
"VK_KHR_ray_query": 1,
"VK_KHR_ray_tracing_maintenance1": 1,
"VK_KHR_ray_tracing_pipeline": 1,
"VK_KHR_ray_tracing_position_fetch": 1,
"VK_KHR_relaxed_block_layout": 1,
"VK_KHR_sampler_mirror_clamp_to_edge": 3,
"VK_KHR_sampler_ycbcr_conversion": 14,
"VK_KHR_separate_depth_stencil_layouts": 1,
"VK_KHR_shader_atomic_int64": 1,
"VK_KHR_shader_clock": 1,
"VK_KHR_shader_draw_parameters": 1,
"VK_KHR_shader_float16_int8": 1,
"VK_KHR_shader_float_controls": 4,
"VK_KHR_shader_integer_dot_product": 1,
"VK_KHR_shader_non_semantic_info": 1,
"VK_KHR_shader_subgroup_extended_types": 1,
"VK_KHR_shader_subgroup_uniform_control_flow": 1,
"VK_KHR_shader_terminate_invocation": 1,
"VK_KHR_shared_presentable_image": 1,
"VK_KHR_spirv_1_4": 1,
"VK_KHR_storage_buffer_storage_class": 1,
"VK_KHR_swapchain": 70,
"VK_KHR_swapchain_mutable_format": 1,
"VK_KHR_synchronization2": 1,
"VK_KHR_timeline_semaphore": 2,
"VK_KHR_uniform_buffer_standard_layout": 1,
"VK_KHR_variable_pointers": 1,
"VK_KHR_video_decode_h264": 8,
"VK_KHR_video_decode_h265": 7,
"VK_KHR_video_decode_queue": 7,
"VK_KHR_video_queue": 8,
"VK_KHR_vulkan_memory_model": 3,
"VK_KHR_workgroup_memory_explicit_layout": 1,
"VK_KHR_zero_initialize_workgroup_memory": 1,
"VK_EXT_4444_formats": 1,
"VK_EXT_attachment_feedback_loop_dynamic_state": 1,
"VK_EXT_attachment_feedback_loop_layout": 2,
"VK_EXT_blend_operation_advanced": 2,
"VK_EXT_border_color_swizzle": 1,
"VK_EXT_buffer_device_address": 2,
"VK_EXT_calibrated_timestamps": 2,
"VK_EXT_color_write_enable": 1,
"VK_EXT_conditional_rendering": 2,
"VK_EXT_conservative_rasterization": 1,
"VK_EXT_custom_border_color": 12,
"VK_EXT_depth_bias_control": 1,
"VK_EXT_depth_clamp_zero_one": 1,
"VK_EXT_depth_clip_control": 1,
"VK_EXT_depth_clip_enable": 1,
"VK_EXT_depth_range_unrestricted": 1,
"VK_EXT_descriptor_buffer": 1,
"VK_EXT_descriptor_indexing": 2,
"VK_EXT_discard_rectangles": 2,
"VK_EXT_display_control": 1,
"VK_EXT_dynamic_rendering_unused_attachments": 1,
"VK_EXT_extended_dynamic_state": 1,
"VK_EXT_extended_dynamic_state2": 1,
"VK_EXT_extended_dynamic_state3": 2,
"VK_EXT_external_memory_dma_buf": 1,
"VK_EXT_external_memory_host": 1,
"VK_EXT_fragment_shader_interlock": 1,
"VK_EXT_global_priority": 2,

"VK_EXT_global_priority_query": 1,
"VK_EXT_graphics_pipeline_library": 1,
"VK_EXT_host_image_copy": 1,
"VK_EXT_host_query_reset": 1,
"VK_EXT_image_2d_view_of_3d": 1,
"VK_EXT_image_drm_format_modifier": 2,
"VK_EXT_image_robustness": 1,
"VK_EXT_image_sliced_view_of_3d": 1,
"VK_EXT_image_view_min_lod": 1,
"VK_EXT_index_type_uint8": 1,
"VK_EXT_inline_uniform_block": 1,
"VK_EXT_line_rasterization": 1,
"VK_EXT_load_store_op_none": 1,
"VK_EXT_memory_budget": 1,
"VK_EXT_memory_priority": 1,
"VK_EXT_mesh_shader": 1,
"VK_EXT_multi_draw": 1,
"VK_EXT_mutable_descriptor_type": 1,
"VK_EXT_non_seamless_cube_map": 1,
"VK_EXT_opacity_micromap": 2,
"VK_EXT_pageable_device_local_memory": 1,
"VK_EXT_pci_bus_info": 2,
"VK_EXT_physical_device_drm": 1,
"VK_EXT_pipeline_creation_cache_control": 3,
"VK_EXT_pipeline_creation_feedback": 1,
"VK_EXT_pipeline_library_group_handles": 1,
"VK_EXT_pipeline_robustness": 1,
"VK_EXT_post_depth_coverage": 1,
"VK_EXT_primitive_topology_list_restart": 1,
"VK_EXT_primitives_generated_query": 1,
"VK_EXT_private_data": 1,
"VK_EXT_provoking_vertex": 1,
"VK_EXT_queue_family_foreign": 1,
"VK_EXT_robustness2": 1,
"VK_EXT_sample_locations": 1,
"VK_EXT_sampler_filter_minmax": 2,
"VK_EXT_scalar_block_layout": 1,
"VK_EXT_separate_stencil_usage": 1,
"VK_EXT_shader_atomic_float": 1,
"VK_EXT_shader_demote_to_helper_invocation": 1,
"VK_EXT_shader_image_atomic_int64": 1,
"VK_EXT_shader_module_identifier": 1,
"VK_EXT_shader_object": 1,
"VK_EXT_shader_subgroup_ballot": 1,
"VK_EXT_shader_subgroup_vote": 1,
"VK_EXT_shader_viewport_index_layer": 1,
"VK_EXT_subgroup_size_control": 2,
"VK_EXT_swapchain_maintenance1": 1,
"VK_EXT_texel_buffer_alignment": 1,
"VK_EXT_tooling_info": 1,
"VK_EXT_transform_feedback": 1,
"VK_EXT_vertex_attribute_divisor": 3,
"VK_EXT_vertex_input_dynamic_state": 2,
"VK_EXT_ycbcr_2plane_444_formats": 1,
"VK_EXT_ycbcr_image_arrays": 1,
"VK_NV_clip_space_w_scaling": 1,
"VK_NV_compute_shader_derivatives": 1,
"VK_NV_cooperative_matrix": 1,
"VK_NV_copy_memory_indirect": 1,
"VK_NV_corner_sampled_image": 2,
"VK_NV_coverage_reduction_mode": 1,
"VK_NV_cuda_kernel_launch": 2,
"VK_NV_dedicated_allocation": 1,
"VK_NV_dedicated_allocation_image_aliasing": 1,

```

"VK_NV_device_diagnostic_checkpoints": 2,
"VK_NV_device_diagnostics_config": 2,
"VK_NV_device_generated_commands": 3,
"VK_NV_device_generated_commands_compute": 2,
"VK_NV_fill_rectangle": 1,
"VK_NV_fragment_coverage_to_color": 1,
"VK_NV_fragment_shader_barycentric": 1,
"VK_NV_fragment_shading_rate_enums": 1,
"VK_NV_framebuffer_mixed_samples": 1,
"VK_NV_geometry_shader_passthrough": 1,
"VK_NV_inherited_viewport_scissor": 1,
"VK_NV_linear_color_attachment": 1,
"VK_NV_low_latency": 1,
"VK_NV_low_latency2": 1,
"VK_NV_memory_decompression": 1,
"VK_NV_mesh_shader": 1,
"VK_NV_optical_flow": 1,
"VK_NV_ray_tracing": 3,
"VK_NV_ray_tracing_invocation_reorder": 1,
"VK_NV_ray_tracing_motion_blur": 1,
"VK_NV_representative_fragment_test": 2,
"VK_NV_sample_mask_override_coverage": 1,
"VK_NV_scissor_exclusive": 2,
"VK_NV_shader_image_footprint": 2,
"VK_NV_shader_sm_builtins": 1,
"VK_NV_shader_subgroup_partitioned": 1,
"VK_NV_shading_rate_image": 3,
"VK_NV_viewport_array2": 1,
"VK_NV_viewport_swizzle": 1,
"VK_NVX_binary_import": 1,
"VK_NVX_image_view_handle": 2,
"VK_NVX_multiview_per_view_attributes": 1,
"VK_AMD_buffer_marker": 1
},
"features": {
  "robustBufferAccess": true,
  "fullDrawIndexUint32": true,
  "imageCubeArray": true,
  "independentBlend": true,
  "geometryShader": true,
  "tessellationShader": true,
  "sampleRateShading": true,
  "dualSrcBlend": true,
  "logicOp": true,
  "multiDrawIndirect": true,
  "drawIndirectFirstInstance": true,
  "depthClamp": true,
  "depthBiasClamp": true,
  "fillModeNonSolid": true,
  "depthBounds": true,
  "wideLines": true,
  "largePoints": true,
  "alphaToOne": true,
  "multiViewport": true,
  "samplerAnisotropy": true,
  "textureCompressionETC2": false,
  "textureCompressionASTC_LDR": false,
  "textureCompressionBC": true,
  "occlusionQueryPrecise": true,
  "pipelineStatisticsQuery": true,
  "vertexPipelineStoresAndAtomics": true,
  "fragmentStoresAndAtomics": true,
  "shaderTessellationAndGeometryPointSize": true,
  "shaderImageGatherExtended": true,

```

```

"shaderStorageImageExtendedFormats": true,
"shaderStorageImageMultisample": true,
"shaderStorageImageReadWithoutFormat": true,
"shaderStorageImageWriteWithoutFormat": true,
"shaderUniformBufferArrayDynamicIndexing": true,
"shaderSampledImageArrayDynamicIndexing": true,
"shaderStorageBufferArrayDynamicIndexing": true,
"shaderStorageImageArrayDynamicIndexing": true,
"shaderClipDistance": true,
"shaderCullDistance": true,
"shaderFloat64": true,
"shaderInt64": true,
"shaderInt16": true,
"shaderResourceResidency": true,
"shaderResourceMinLod": true,
"sparseBinding": true,
"sparseResidencyBuffer": true,
"sparseResidencyImage2D": true,
"sparseResidencyImage3D": true,
"sparseResidency2Samples": true,
"sparseResidency4Samples": true,
"sparseResidency8Samples": true,
"sparseResidency16Samples": true,
"sparseResidencyAliased": true,
"variableMultisampleRate": true,
"inheritedQueries": true
},
"featureSamplerYcbcrConversion": true,
"featureProtectedMemory": false,
"queueFamilies": [
{
"queueFlags": 15,
"queueCount": 16,
"timestampValidBits": 64,
"minImageTransferGranularity": {
"width": 1,
"height": 1,
"depth": 1
}
},
{
"queueFlags": 12,
"queueCount": 2,
"timestampValidBits": 64,
"minImageTransferGranularity": {
"width": 1,
"height": 1,
"depth": 1
}
},
{
"queueFlags": 14,
"queueCount": 8,
"timestampValidBits": 64,
"minImageTransferGranularity": {
"width": 1,
"height": 1,
"depth": 1
}
},
{
"queueFlags": 44,
"queueCount": 1,
"timestampValidBits": 32,

```



```

        "minImageTransferGranularity": {
            "width": 1,
            "height": 1,
            "depth": 1
        }
    },
    {
        "queueFlags": 268,
        "queueCount": 1,
        "timestampValidBits": 64,
        "minImageTransferGranularity": {
            "width": 1,
            "height": 1,
            "depth": 1
        }
    }
]
}

```

Device Performance Information

=====

Log Messages

=====

```

[83219:83219:1215/151039.938429:WARNING:sandbox_linux.cc(400)] :
InitializeSandbox() called with multiple threads in process gpu-process.
[83219:83219:1215/151040.034308:ERROR:vulkan_swap_chain.cc(404)] :
vkQueuePresentKHR() failed: -10000001004
GpuProcessHost: The GPU process exited with code 8704.
[83418:83418:1215/151040.226392:WARNING:sandbox_linux.cc(400)] :
InitializeSandbox() called with multiple threads in process gpu-process.
[83418:83418:1215/151044.722869:ERROR:gl_utils.cc(412)] : [.WebGL-
0x29bc02111c00]GL Driver Message (OpenGL, Performance, GL_CLOSE_PATH_NV, High):
GPU stall due to ReadPixels

```