**Graphics Feature Status**

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

**Driver Bug Workarounds**

* clear\_uniforms\_before\_first\_program\_use
* decode\_encode\_srgb\_for\_generatemipmap
* disable\_accelerated\_av1\_encode
* disable\_direct\_composition\_sw\_video\_overlays
* disable\_dynamic\_video\_encode\_framerate\_update
* enable\_bgra8\_overlays\_with\_yuv\_overlay\_support
* enable\_webgl\_timer\_query\_extensions
* exit\_on\_context\_lost
* force\_rgb10a2\_overlay\_support\_flags
* max\_msaa\_sample\_count\_4
* msaa\_is\_slow
* no\_downscaled\_overlay\_promotion
* supports\_two\_yuv\_hardware\_overlays
* disabled\_extension\_GL\_KHR\_blend\_equation\_advanced
* disabled\_extension\_GL\_KHR\_blend\_equation\_advanced\_coherent

**Problems Detected**

* Some drivers are unable to reset the D3D device in the GPU process sandbox
*Applied Workarounds: exit\_on\_context\_lost*
* Clear uniforms before first program use on all platforms: [124764](http://crbug.com/124764), [349137](http://crbug.com/349137)
*Applied Workarounds: clear\_uniforms\_before\_first\_program\_use*
* On Intel GPUs MSAA performance is not acceptable for GPU rasterization: [527565](http://crbug.com/527565), [1298585](http://crbug.com/1298585)
*Applied Workarounds: msaa\_is\_slow*
* Disable KHR\_blend\_equation\_advanced until cc shaders are updated: [661715](http://crbug.com/661715)
*Applied Workarounds: disable(GL\_KHR\_blend\_equation\_advanced), disable(GL\_KHR\_blend\_equation\_advanced\_coherent)*
* Decode and Encode before generateMipmap for srgb format textures on Windows: [634519](http://crbug.com/634519)
*Applied Workarounds: decode\_encode\_srgb\_for\_generatemipmap*
* Expose WebGL's disjoint\_timer\_query extensions on platforms with site isolation: [808744](http://crbug.com/808744), [870491](http://crbug.com/870491)
*Applied Workarounds: enable\_webgl\_timer\_query\_extensions*
* Enable HDR video playing through overlay on Intel: [1062184](http://crbug.com/1062184)
*Applied Workarounds: force\_rgb10a2\_overlay\_support\_flags*
* Intel GPUs fail to report BGRA8 overlay support: [1119491](http://crbug.com/1119491)
*Applied Workarounds: enable\_bgra8\_overlays\_with\_yuv\_overlay\_support*
* 8x MSAA for WebGL contexts is slow on Win Intel: [1145793](http://crbug.com/1145793)
*Applied Workarounds: max\_msaa\_sample\_count\_4*
* Promote 2 videos to hardware overlays on Windows Intel platforms: [1105618](http://crbug.com/1105618)
*Applied Workarounds: supports\_two\_yuv\_hardware\_overlays*
* Disable software overlays for Intel GPUs. All Skylake+ devices support hw overlays, older devices peform poorly.: [1192748](http://crbug.com/1192748)
*Applied Workarounds: disable\_direct\_composition\_sw\_video\_overlays*
* Intel GPUs do not promote downscaled overlays: [1245835](http://crbug.com/1245835)
*Applied Workarounds: no\_downscaled\_overlay\_promotion*
* AVC/AV1 hardware encoder MFT output bitrate incorrect upon framerate update on Intel GPUs.: [1295815](http://crbug.com/1295815)
*Applied Workarounds: disable\_dynamic\_video\_encode\_framerate\_update*
* Disable hardware MFT Av1 encoder on machines with multiple GPUs: [1367038](http://crbug.com/1367038)
*Applied Workarounds: disable\_accelerated\_av1\_encode*

**ANGLE Features**

* **allowCompressedFormats** (Frontend workarounds): Enabled: true
*Allow compressed formats*
* **cacheCompiledShader** (Frontend features) [anglebug:7036](http://anglebug.com/7036): Disabled
*Enable to cache compiled shaders*
* **disableAnisotropicFiltering** (Frontend workarounds): Disabled
*Disable support for anisotropic filtering*
* **disableDrawBuffersIndexed** (Frontend features) [anglebug:7724](http://anglebug.com/7724): Disabled
*Disable support for OES\_draw\_buffers\_indexed and EXT\_draw\_buffers\_indexed*
* **disableProgramBinary** (Frontend features) [anglebug:5007](http://anglebug.com/5007): Disabled
*Disable support for GL\_OES\_get\_program\_binary*
* **disableProgramCachingForTransformFeedback** (Frontend workarounds): Disabled
*On some GPUs, program binaries don't contain transform feedback varyings*
* **dumpShaderSource** (Frontend features) [anglebug:7760](http://anglebug.com/7760): Disabled
*Write shader source to temp directory*
* **emulatePixelLocalStorage** (Frontend features) [anglebug:7279](http://anglebug.com/7279): Disabled: false
*Emulate ANGLE\_shader\_pixel\_local\_storage using shader images*
* **enableCaptureLimits** (Frontend features) [anglebug:5750](http://anglebug.com/5750): Disabled
*Set the context limits like frame capturing was enabled*
* **enableProgramBinaryForCapture** (Frontend features) [anglebug:5658](http://anglebug.com/5658): Disabled
*Even if FrameCapture is enabled, enable GL\_OES\_get\_program\_binary*
* **enableShaderSubstitution** (Frontend workarounds) [anglebug:7761](http://anglebug.com/7761): Disabled
*Check the filesystem for shaders to use instead of those provided through glShaderSource*
* **forceDepthAttachmentInitOnClear** (Frontend workarounds) [anglebug:7246](https://anglebug.com/7246): Disabled: isAMD
*Force depth attachment initialization on clear ops*
* **forceGlErrorChecking** (Frontend features) <https://issuetracker.google.com/220069903>: Disabled
*Force GL error checking (i.e. prevent applications from disabling error checking*
* **forceInitShaderVariables** (Frontend features): Disabled
*Force-enable shader variable initialization*
* **forceRobustResourceInit** (Frontend features) [anglebug:6041](http://anglebug.com/6041): Disabled
*Force-enable robust resource init*
* **loseContextOnOutOfMemory** (Frontend workarounds): Enabled: true
*Some users rely on a lost context notification if a GL\_OUT\_OF\_MEMORY error occurs*
* **scalarizeVecAndMatConstructorArgs** (Frontend workarounds) [1165751](http://crbug.com/1165751): Disabled: false
*Always rewrite vec/mat constructors to be consistent*
* **singleThreadedTextureDecompression** (Frontend workarounds): Disabled
*Disables multi-threaded decompression of compressed texture formats*
* **addMockTextureNoRenderTarget** (D3D workarounds) [anglebug:2152](http://anglebug.com/2152): Disabled: isIntel && capsVersion >= IntelDriverVersion(160000) && capsVersion < IntelDriverVersion(164815)
*On some drivers when rendering with no render target, two bugs lead to incorrect behavior*
* **allowClearForRobustResourceInit** (D3D workarounds) [941620](http://crbug.com/941620): Enabled: true
*Some drivers corrupt texture data when clearing for robust resource initialization.*
* **allowES3OnFL100** (D3D workarounds): Disabled: false
*Allow ES3 on 10.0 devices*
* **allowTranslateUniformBlockToStructuredBuffer** (D3D workarounds) [anglebug:3682](http://anglebug.com/3682): Enabled: IsWin10OrGreater()
*There is a slow fxc compile performance issue with dynamic uniform indexing if translating a uniform block with a large array member to cbuffer.*
* **callClearTwice** (D3D workarounds) [655534](https://crbug.com/655534): Disabled: isIntel && isSkylake && capsVersion >= IntelDriverVersion(160000) && capsVersion < IntelDriverVersion(164771)
*Using clear() may not take effect*
* **depthStencilBlitExtraCopy** (D3D workarounds) [anglebug:1452](http://anglebug.com/1452): Disabled
*Bug in some drivers triggers a TDR when using CopySubresourceRegion from a staging texture to a depth/stencil*
* **disableB5G6R5Support** (D3D workarounds): Disabled: (isIntel && capsVersion >= IntelDriverVersion(150000) && capsVersion < IntelDriverVersion(154539)) || isAMD
*Textures with the format DXGI\_FORMAT\_B5G6R5\_UNORM have incorrect data*
* **disableRasterizerOrderViews** (D3D workarounds) [anglebug:7279](http://anglebug.com/7279): Disabled
*Disable ROVs for testing*
* **emulateIsnanFloat** (D3D workarounds) [650547](https://crbug.com/650547): Disabled: isIntel && isSkylake && capsVersion >= IntelDriverVersion(160000) && capsVersion < IntelDriverVersion(164542)
*Using isnan() on highp float will get wrong answer*
* **emulateTinyStencilTextures** (D3D workarounds): Disabled: isAMD && !(deviceCaps.featureLevel < D3D\_FEATURE\_LEVEL\_10\_1)
*1x1 and 2x2 mips of depth/stencil textures aren't sampled correctly*
* **enableTimestampQueries** (D3D workarounds): Disabled
*Enable timestamp on GL\_EXT\_disjoint\_timer\_query extension*
* **expandIntegerPowExpressions** (D3D workarounds): Enabled: true
*The HLSL optimizer has a bug with optimizing 'pow' in certain integer-valued expressions*
* **flushAfterEndingTransformFeedback** (D3D workarounds): Disabled: isNvidia
*Some drivers sometimes write out-of-order results to StreamOut buffers when transform feedback is used to repeatedly write to the same buffer positions*
* **forceAtomicValueResolution** (D3D workarounds) [anglebug:3246](http://anglebug.com/3246): Disabled: isNvidia
*On some drivers the return value from RWByteAddressBuffer.InterlockedAdd does not resolve when used in the .yzw components of a RWByteAddressBuffer.Store operation*
* **getDimensionsIgnoresBaseLevel** (D3D workarounds): Disabled: isNvidia
*Some drivers do not take into account the base level of the texture in the results of the HLSL GetDimensions builtin*
* **mrtPerfWorkaround** (D3D workarounds): Enabled: true
*Some drivers have a bug where they ignore null render targets*
* **preAddTexelFetchOffsets** (D3D workarounds): Enabled: isIntel
*HLSL's function texture.Load returns 0 when the parameter Location is negative, even if the sum of Offset and Location is in range*
* **rewriteUnaryMinusOperator** (D3D workarounds): Disabled: isIntel && (isBroadwell || isHaswell) && capsVersion >= IntelDriverVersion(150000) && capsVersion < IntelDriverVersion(154624)
*Evaluating unary minus operator on integer may get wrong answer in vertex shaders*
* **selectViewInGeometryShader** (D3D workarounds): Disabled: !deviceCaps.supportsVpRtIndexWriteFromVertexShader
*The viewport or render target slice will be selected in the geometry shader stage for the ANGLE\_multiview extension*
* **setDataFasterThanImageUpload** (D3D workarounds): Enabled: !(isIvyBridge || isBroadwell || isHaswell)
*Set data faster than image upload*
* **skipVSConstantRegisterZero** (D3D workarounds): Disabled: isNvidia
*In specific cases the driver doesn't handle constant register zero correctly*
* **useInstancedPointSpriteEmulation** (D3D workarounds): Disabled: isFeatureLevel9\_3
*Some D3D11 renderers do not support geometry shaders for pointsprite emulation*
* **useSystemMemoryForConstantBuffers** (D3D workarounds) [593024](https://crbug.com/593024): Enabled: isIntel
*Copying from staging storage to constant buffer storage does not work*
* **zeroMaxLodWorkaround** (D3D workarounds): Disabled: isFeatureLevel9\_3
*Missing an option to disable mipmaps on a mipmapped texture*

**DAWN Info**

**<Integrated GPU> D3D12 backend - Intel(R) Iris(R) Xe Graphics**
[Default 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\_d3d12\_resource\_heap\_tier2:**<https://crbug.com/dawn/27>*: Enable support for resource heap tier 2. Resource heap tier 2 allows mixing of texture and buffers in the same heap. This allows better heap re-use and reduces fragmentation.*
* **use\_d3d12\_render\_pass:**<https://crbug.com/dawn/36>*: Use the D3D12 render pass API introduced in Windows build 1809 by default. On versions of Windows prior to build 1809, or when this toggle is turned off, Dawn will emulate a render pass.*
* **use\_d3d12\_residency\_management:**<https://crbug.com/dawn/193>*: Enable residency management. This allows page-in and page-out of resource heaps in GPU memory. This component improves overcommitted performance by keeping the most recently used resources local to the GPU. Turning this component off can cause allocation failures when application memory exceeds physical device memory.*
* **disallow\_unsafe\_apis:**<http://crbug.com/1138528>*: Produces validation errors on API entry points or parameter combinations that aren't considered secure yet.*
* **d3d12\_split\_buffer\_texture\_copy\_for\_rows\_per\_image\_paddings:**<https://crbug.com/dawn/1289>*: D3D12 requires more buffer storage than it should when rowsPerImage is greater than copyHeight, which means there are pure padding row(s) on each image. In this situation, the buffer used for B2T/T2B copy might be big enough according to WebGPU's spec but it doesn't meet D3D12's requirement, then we need to workaround it via split the copy operation into two copies, in order to make B2T/T2B copy being done correctly on D3D12.*
* **d3d12\_force\_clear\_copyable\_depth\_stencil\_texture\_on\_creation:**<https://crbug.com/dawn/1487>*: Always clearing copyable depth stencil textures when creating them instead of skipping the initialization when the entire subresource is the copy destination as a workaround on Intel D3D12 drivers.*
* **d3d12\_dont\_set\_clear\_value\_on\_depth\_texture\_creation:**<https://crbug.com/dawn/1487>*: Don't set D3D12\_CLEAR\_VALUE when creating depth textures with CreatePlacedResource() or CreateCommittedResource() as a workaround on Intel Gen12 D3D12 drivers.*
* **d3d12\_allocate\_extra\_memory\_for\_2d\_array\_color\_texture:**<https://crbug.com/dawn/949>*: Memory allocation for 2D array color texture may be smaller than it should be on D3D12 on some Intel devices. So texture access can be out-of-bound, which may cause critical security issue. We can workaround this security issue via allocating extra memory and limiting its access in itself.*
* **apply\_clear\_big\_integer\_color\_value\_with\_draw:**<https://crbug.com/dawn/537>*: Apply the clear value of the color attachment with a draw call when load op is 'clear'. This toggle is enabled by default on D3D12 backends when we set large integer values (> 2^24 or < -2^24 for signed integer formats) as the clear value of a color attachment with 32-bit integer or unsigned integer formats because D3D12 APIs only support using float numbers as clear values, while a float number cannot always precisely represent an integer that is greater than 2^24 or smaller than -2^24). This toggle is also enabled on Intel GPUs on Metal backend due to a driver issue on Intel Metal driver.*

[WebGPU Forced 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.*

[Supported Features]

* texture-compression-bc
* pipeline-statistics-query
* timestamp-query
* timestamp-query-inside-passes
* depth-clip-control
* depth32float-stencil8
* chromium-experimental-dp4a
* indirect-first-instance
* shader-f16
* rg11b10ufloat-renderable
* dawn-internal-usages
* multiplanar-formats
* dawn-native

**<Discrete GPU> D3D12 backend - NVIDIA GeForce RTX 3070 Ti Laptop GPU**
[Default 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\_d3d12\_resource\_heap\_tier2:**<https://crbug.com/dawn/27>*: Enable support for resource heap tier 2. Resource heap tier 2 allows mixing of texture and buffers in the same heap. This allows better heap re-use and reduces fragmentation.*
* **use\_d3d12\_render\_pass:**<https://crbug.com/dawn/36>*: Use the D3D12 render pass API introduced in Windows build 1809 by default. On versions of Windows prior to build 1809, or when this toggle is turned off, Dawn will emulate a render pass.*
* **use\_d3d12\_residency\_management:**<https://crbug.com/dawn/193>*: Enable residency management. This allows page-in and page-out of resource heaps in GPU memory. This component improves overcommitted performance by keeping the most recently used resources local to the GPU. Turning this component off can cause allocation failures when application memory exceeds physical device memory.*
* **disallow\_unsafe\_apis:**<http://crbug.com/1138528>*: Produces validation errors on API entry points or parameter combinations that aren't considered secure yet.*
* **d3d12\_split\_buffer\_texture\_copy\_for\_rows\_per\_image\_paddings:**<https://crbug.com/dawn/1289>*: D3D12 requires more buffer storage than it should when rowsPerImage is greater than copyHeight, which means there are pure padding row(s) on each image. In this situation, the buffer used for B2T/T2B copy might be big enough according to WebGPU's spec but it doesn't meet D3D12's requirement, then we need to workaround it via split the copy operation into two copies, in order to make B2T/T2B copy being done correctly on D3D12.*
* **apply\_clear\_big\_integer\_color\_value\_with\_draw:**<https://crbug.com/dawn/537>*: Apply the clear value of the color attachment with a draw call when load op is 'clear'. This toggle is enabled by default on D3D12 backends when we set large integer values (> 2^24 or < -2^24 for signed integer formats) as the clear value of a color attachment with 32-bit integer or unsigned integer formats because D3D12 APIs only support using float numbers as clear values, while a float number cannot always precisely represent an integer that is greater than 2^24 or smaller than -2^24). This toggle is also enabled on Intel GPUs on Metal backend due to a driver issue on Intel Metal driver.*

[WebGPU Forced 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.*

[Supported Features]

* texture-compression-bc
* pipeline-statistics-query
* timestamp-query
* timestamp-query-inside-passes
* depth-clip-control
* depth32float-stencil8
* chromium-experimental-dp4a
* indirect-first-instance
* shader-f16
* rg11b10ufloat-renderable
* dawn-internal-usages
* multiplanar-formats
* dawn-native

**<CPU> D3D12 backend - Microsoft Basic Render Driver**
[Default 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\_d3d12\_resource\_heap\_tier2:**<https://crbug.com/dawn/27>*: Enable support for resource heap tier 2. Resource heap tier 2 allows mixing of texture and buffers in the same heap. This allows better heap re-use and reduces fragmentation.*
* **use\_d3d12\_render\_pass:**<https://crbug.com/dawn/36>*: Use the D3D12 render pass API introduced in Windows build 1809 by default. On versions of Windows prior to build 1809, or when this toggle is turned off, Dawn will emulate a render pass.*
* **use\_d3d12\_residency\_management:**<https://crbug.com/dawn/193>*: Enable residency management. This allows page-in and page-out of resource heaps in GPU memory. This component improves overcommitted performance by keeping the most recently used resources local to the GPU. Turning this component off can cause allocation failures when application memory exceeds physical device memory.*
* **disallow\_unsafe\_apis:**<http://crbug.com/1138528>*: Produces validation errors on API entry points or parameter combinations that aren't considered secure yet.*
* **d3d12\_split\_buffer\_texture\_copy\_for\_rows\_per\_image\_paddings:**<https://crbug.com/dawn/1289>*: D3D12 requires more buffer storage than it should when rowsPerImage is greater than copyHeight, which means there are pure padding row(s) on each image. In this situation, the buffer used for B2T/T2B copy might be big enough according to WebGPU's spec but it doesn't meet D3D12's requirement, then we need to workaround it via split the copy operation into two copies, in order to make B2T/T2B copy being done correctly on D3D12.*
* **apply\_clear\_big\_integer\_color\_value\_with\_draw:**<https://crbug.com/dawn/537>*: Apply the clear value of the color attachment with a draw call when load op is 'clear'. This toggle is enabled by default on D3D12 backends when we set large integer values (> 2^24 or < -2^24 for signed integer formats) as the clear value of a color attachment with 32-bit integer or unsigned integer formats because D3D12 APIs only support using float numbers as clear values, while a float number cannot always precisely represent an integer that is greater than 2^24 or smaller than -2^24). This toggle is also enabled on Intel GPUs on Metal backend due to a driver issue on Intel Metal driver.*

[WebGPU Forced 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.*

[Supported Features]

* texture-compression-bc
* pipeline-statistics-query
* timestamp-query
* timestamp-query-inside-passes
* depth-clip-control
* depth32float-stencil8
* indirect-first-instance
* shader-f16
* rg11b10ufloat-renderable
* dawn-internal-usages
* multiplanar-formats
* dawn-native

**<Discrete GPU> Vulkan backend - NVIDIA GeForce RTX 3070 Ti Laptop GPU**
[Default 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\_unsafe\_apis:**<http://crbug.com/1138528>*: Produces validation errors on API entry points or parameter combinations that aren't considered secure yet.*
* **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 Forced 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.*

[Supported Features]

* texture-compression-bc
* pipeline-statistics-query
* timestamp-query
* timestamp-query-inside-passes
* depth-clip-control
* depth32float-stencil8
* chromium-experimental-dp4a
* indirect-first-instance
* rg11b10ufloat-renderable
* dawn-internal-usages
* dawn-native

**<CPU> Vulkan backend - SwiftShader Device (Subzero)**
[Default 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\_unsafe\_apis:**<http://crbug.com/1138528>*: Produces validation errors on API entry points or parameter combinations that aren't considered secure yet.*
* **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 Forced 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.*

[Supported Features]

* texture-compression-bc
* texture-compression-etc2
* texture-compression-astc
* timestamp-query
* timestamp-query-inside-passes
* depth-clip-control
* depth32float-stencil8
* indirect-first-instance
* rg11b10ufloat-renderable
* dawn-internal-usages
* dawn-native

**Version Information**

**Data exported**

2023-03-08T00:24:07.766Z

**Chrome version**

Chrome/111.0.5563.65

**Operating system**

Windows NT 10.0.22621

**Software rendering list URL**

https://chromium.googlesource.com/chromium/src/+/c710e93d5b63b7095afe8c2c17df34408078439d/gpu/config/software\_rendering\_list.json

**Driver bug list URL**

https://chromium.googlesource.com/chromium/src/+/c710e93d5b63b7095afe8c2c17df34408078439d/gpu/config/gpu\_driver\_bug\_list.json

**ANGLE commit id**

cd45d155bf4c

**2D graphics backend**

Skia/111 59932b057f281ddaeb0926ecfac55486270f8c51

**Command Line**

"C:\Program Files\Google\Chrome\Application\chrome.exe" --flag-switches-begin --flag-switches-end

**Driver Information**

**Initialization time**

369

**In-process GPU**

false

**Passthrough Command Decoder**

true

**Sandboxed**

true

**GPU0**

VENDOR= 0x8086, DEVICE=0x46a6, SUBSYS=0x0b541028, REV=12, LUID={0,93068}, DRIVER\_VENDOR=Intel, DRIVER\_VERSION=30.0.101.1298 \*ACTIVE\*

**GPU1**

VENDOR= 0x10de, DEVICE=0x24e0, SUBSYS=0x0b541028, REV=161, LUID={0,93909}, DRIVER\_VERSION=30.0.15.1338

**GPU2**

VENDOR= 0x1414, DEVICE=0x008c, LUID={0,93865}, DRIVER\_VERSION=10.0.22621.1194

**Optimus**

false

**AMD switchable**

false

**Desktop compositing**

Aero Glass

**Direct composition**

true

**Supports overlays**

true

**YUY2 overlay support**

SCALING

**NV12 overlay support**

SCALING

**BGRA8 overlay support**

SCALING

**RGB10A2 overlay support**

SCALING

**Driver D3D12 feature level**

D3D 12.1

**Driver Vulkan API version**

Vulkan API 1.3.0

**GPU CUDA compute capability major version**

0

**Pixel shader version**

5.0

**Vertex shader version**

5.0

**Max. MSAA samples**

16

**Machine model name**

**Machine model version**

**GL\_VENDOR**

Google Inc. (Intel)

**GL\_RENDERER**

ANGLE (Intel, Intel(R) Iris(R) Xe Graphics Direct3D11 vs\_5\_0 ps\_5\_0, D3D11-30.0.101.1298)

**GL\_VERSION**

OpenGL ES 2.0.0 (ANGLE 2.1.20370 git hash: cd45d155bf4c)

**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\_serialized\_context\_string GL\_ANGLE\_get\_tex\_level\_parameter GL\_ANGLE\_instanced\_arrays GL\_ANGLE\_lossy\_etc\_decode GL\_ANGLE\_memory\_size GL\_ANGLE\_multi\_draw GL\_ANGLE\_pack\_reverse\_row\_order GL\_ANGLE\_program\_cache\_control GL\_ANGLE\_provoking\_vertex GL\_ANGLE\_request\_extension GL\_ANGLE\_robust\_client\_memory GL\_ANGLE\_texture\_compression\_dxt3 GL\_ANGLE\_texture\_compression\_dxt5 GL\_ANGLE\_texture\_usage GL\_ANGLE\_translated\_shader\_source 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\_CHROMIUM\_sync\_query GL\_EXT\_EGL\_image\_external\_wrap\_modes GL\_EXT\_base\_instance GL\_EXT\_blend\_func\_extended GL\_EXT\_blend\_minmax GL\_EXT\_clip\_control GL\_EXT\_color\_buffer\_half\_float GL\_EXT\_debug\_label GL\_EXT\_debug\_marker 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\_multi\_draw\_indirect GL\_EXT\_multisampled\_render\_to\_texture GL\_EXT\_occlusion\_query\_boolean GL\_EXT\_polygon\_offset\_clamp GL\_EXT\_read\_format\_bgra GL\_EXT\_robustness GL\_EXT\_sRGB GL\_EXT\_shader\_texture\_lod 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\_norm16 GL\_EXT\_texture\_rg GL\_EXT\_texture\_storage GL\_EXT\_texture\_type\_2\_10\_10\_10\_REV GL\_EXT\_unpack\_subimage GL\_KHR\_debug GL\_KHR\_parallel\_shader\_compile GL\_NV\_EGL\_stream\_consumer\_external GL\_NV\_fence GL\_NV\_framebuffer\_blit GL\_NV\_pack\_subimage GL\_NV\_pixel\_buffer\_object GL\_OES\_EGL\_image GL\_OES\_EGL\_image\_external 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\_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\_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\_rgb8\_rgba8 GL\_OES\_standard\_derivatives GL\_OES\_surfaceless\_context 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\_texture\_stencil8 GL\_OES\_vertex\_array\_object GL\_WEBGL\_video\_texture

**Disabled Extensions**

GL\_KHR\_blend\_equation\_advanced GL\_KHR\_blend\_equation\_advanced\_coherent

**Disabled WebGL Extensions**

**Window system binding vendor**

Google Inc. (Intel)

**Window system binding version**

1.5 (ANGLE 2.1.20370 git hash: cd45d155bf4c)

**Window system binding extensions**

EGL\_EXT\_create\_context\_robustness EGL\_ANGLE\_d3d\_share\_handle\_client\_buffer EGL\_ANGLE\_d3d\_texture\_client\_buffer EGL\_ANGLE\_surface\_d3d\_texture\_2d\_share\_handle EGL\_ANGLE\_query\_surface\_pointer EGL\_ANGLE\_window\_fixed\_size EGL\_ANGLE\_keyed\_mutex EGL\_ANGLE\_surface\_orientation EGL\_ANGLE\_direct\_composition EGL\_NV\_post\_sub\_buffer EGL\_KHR\_create\_context EGL\_KHR\_image EGL\_KHR\_image\_base EGL\_KHR\_gl\_texture\_2D\_image EGL\_KHR\_gl\_texture\_cubemap\_image EGL\_KHR\_gl\_renderbuffer\_image EGL\_KHR\_get\_all\_proc\_addresses EGL\_KHR\_stream EGL\_KHR\_stream\_consumer\_gltexture EGL\_NV\_stream\_consumer\_gltexture\_yuv EGL\_ANGLE\_stream\_producer\_d3d\_texture EGL\_ANGLE\_create\_context\_webgl\_compatibility EGL\_CHROMIUM\_create\_context\_bind\_generates\_resource EGL\_CHROMIUM\_sync\_control 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\_ANGLE\_image\_d3d11\_texture EGL\_ANGLE\_create\_context\_backwards\_compatible EGL\_KHR\_no\_config\_context EGL\_KHR\_create\_context\_no\_error EGL\_KHR\_reusable\_sync

**Direct rendering version**

unknown

**Reset notification strategy**

0x8252

**GPU process crash count**

0

**gfx::BufferFormats supported for allocation and texturing**

R\_8: not supported, R\_16: not supported, RG\_88: not supported, RG\_1616: not supported, BGR\_565: not supported, RGBA\_4444: not supported, RGBX\_8888: not supported, RGBA\_8888: not supported, BGRX\_8888: not supported, BGRA\_1010102: not supported, RGBA\_1010102: not supported, BGRA\_8888: not supported, RGBA\_F16: not supported, YVU\_420: not supported, YUV\_420\_BIPLANAR: not supported, YUVA\_420\_TRIPLANAR: not 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**

GPU\_READ, SCANOUT

**RGBA\_8888**

GPU\_READ, SCANOUT

**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[2528732444] bounds=[0,0 1707x960], workarea=[0,0 1707x912], scale=1.5, rotation=0, panel\_rotation=0 internal.

**Color space (sRGB/no-alpha)**

{r:[0.6720, 0.3225], g:[0.2405, 0.7061], b:[0.1411, 0.3225], w:[0.3127, 0.3290]}, transfer:SRGB, matrix:RGB, range:FULL}

**Buffer format (sRGB/no-alpha)**

BGRX\_8888

**Color space (sRGB/alpha)**

{r:[0.6720, 0.3225], g:[0.2405, 0.7061], b:[0.1411, 0.3225], w:[0.3127, 0.3290]}, transfer:SRGB, matrix:RGB, range:FULL}

**Buffer format (sRGB/alpha)**

BGRA\_8888

**Color space (WCG/no-alpha)**

{r:[0.6720, 0.3225], g:[0.2405, 0.7061], b:[0.1411, 0.3225], w:[0.3127, 0.3290]}, transfer:SRGB, matrix:RGB, range:FULL}

**Buffer format (WCG/no-alpha)**

BGRX\_8888

**Color space (WCG/alpha)**

{r:[0.6720, 0.3225], g:[0.2405, 0.7061], b:[0.1411, 0.3225], w:[0.3127, 0.3290]}, transfer:SRGB, matrix:RGB, range:FULL}

**Buffer format (WCG/alpha)**

BGRA\_8888

**Color space (HDR/no-alpha)**

{r:[0.6720, 0.3225], g:[0.2405, 0.7061], b:[0.1411, 0.3225], w:[0.3127, 0.3290]}, transfer:SRGB, matrix:RGB, range:FULL}

**Buffer format (HDR/no-alpha)**

BGRX\_8888

**Color space (HDR/alpha)**

{r:[0.6720, 0.3225], g:[0.2405, 0.7061], b:[0.1411, 0.3225], w:[0.3127, 0.3290]}, transfer:SRGB, matrix:RGB, range:FULL}

**Buffer format (HDR/alpha)**

BGRA\_8888

**Color volume**

{r:[0.6720, 0.3225], g:[0.2405, 0.7061], b:[0.1411, 0.3225], 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**

240

**Video Acceleration Information**

**Decoding**

**Decode h264 baseline**

64x64 to 4096x4096 pixels

**Decode h264 main**

64x64 to 4096x4096 pixels

**Decode h264 high**

64x64 to 4096x4096 pixels

**Decode vp9 profile0**

64x64 to 8192x8192 pixels

**Decode vp9 profile2**

64x64 to 8192x8192 pixels

**Decode hevc main**

64x64 to 8192x8192 pixels

**Decode hevc main 10**

64x64 to 8192x8192 pixels

**Decode av1 profile main**

64x64 to 8192x8192 pixels

**Decode hevc range extensions**

64x64 to 8192x8192 pixels

**Encoding**

**Encode h264 baseline**

32x32 to 1920x1088 pixels, and/or 30.000 fps.

**Encode h264 main**

32x32 to 1920x1088 pixels, and/or 30.000 fps.

**Encode h264 high**

32x32 to 1920x1088 pixels, and/or 30.000 fps.

**Encode vp9 profile0**

32x32 to 1920x1088 pixels, and/or 30.000 fps.

**Vulkan Information**

**Device Performance Information**

**Total Physical Memory (Gb)**

31

**Total Disk Space (Gb)**

933

**Hardware Concurrency**

20

**System Commit Limit (Gb)**

51

**D3D11 Feature Level**

12\_1

**Has Discrete GPU**

yes

**Intel GPU Generation**

12

**Software Rendering**

No

**Diagnostics**

**0**

**b3DAccelerationEnabled**

true

**b3DAccelerationExists**

true

**bAGPEnabled**

true

**bAGPExistenceValid**

true

**bAGPExists**

true

**bCanRenderWindow**

true

**bDDAccelerationEnabled**

true

**bDX12UEnabled**

false

**bDriverBeta**

false

**bDriverDebug**

false

**bDriverSigned**

false

**bDriverSignedValid**

false

**bNoHardware**

false

**dwBpp**

32

**dwDDIVersion**

12

**dwHeight**

1440

**dwRefreshRate**

240

**dwWHQLLevel**

0

**dwWidth**

2560

**iAdapter**

0

**lDriverSize**

34278720

**lMiniVddSize**

0

**szAGPStatusEnglish**

Enabled

**szAGPStatusLocalized**

Enabled

**szChipType**

Intel(R) Iris(R) Xe Graphics Family

**szD3DStatusEnglish**

Enabled

**szD3DStatusLocalized**

Enabled

**szDACType**

Internal

**szDDIVersionEnglish**

12

**szDDIVersionLocalized**

12

**szDDStatusEnglish**

Enabled

**szDDStatusLocalized**

Enabled

**szDX12UStatusEnglish**

Disabled

**szDX12UStatusLocalized**

Disabled

**szDXVAHDEnglish**

Supported

**szDXVAModes**

**szDescription**

Intel(R) Iris(R) Xe Graphics

**szDeviceId**

0x46A6

**szDeviceIdentifier**

{D7B78E66-05E6-11CF-8A76-312BA3C2D335}

**szDeviceName**

\\.\DISPLAY1

**szDisplayMemoryEnglish**

16354 MB

**szDisplayMemoryLocalized**

16354 MB

**szDisplayModeEnglish**

2560 x 1440 (32 bit) (240Hz)

**szDisplayModeLocalized**

2560 x 1440 (32 bit) (240Hz)

**szDriverAssemblyVersion**

30.0.101.1298

**szDriverAttributes**

Final Retail

**szDriverDateEnglish**

2022-01-09 4:00:00 PM

**szDriverDateLocalized**

1/9/2022 16:00:00

**szDriverLanguageEnglish**

English

**szDriverLanguageLocalized**

English

**szDriverModelEnglish**

WDDM 3.0

**szDriverModelLocalized**

WDDM 3.0

**szDriverName**

<>,C:\WINDOWS\System32\DriverStore\FileRepository\iigd\_dch.inf\_amd64\_c076c143c0fabf30\igd10iumd64.dll,C:\WINDOWS\System32\DriverStore\FileRepository\iigd\_dch.inf\_amd64\_c076c143c0fabf30\igd10iumd64.dll,C:\WINDOWS\System32\DriverStore\FileRepository\iigd\_dch.inf\_amd64\_c076c143c0fabf30\igd12umd64.dll

**szDriverNodeStrongName**

oem53.inf:5f63e5343e91ba36:iADLPD\_w10\_DS:30.0.101.1298:PCI\VEN\_8086&DEV\_46A6&SUBSYS\_0B541028

**szDriverSignDate**

Unknown

**szDriverVersion**

30.00.0101.1298

**szKeyDeviceID**

Enum\PCI\VEN\_8086&DEV\_46A6&SUBSYS\_0B541028&REV\_0C

**szKeyDeviceKey**

\Registry\Machine\System\CurrentControlSet\Control\Video\{3348A41C-B2E7-11ED-A3BE-EF0BE6CA7CB1}\0000

**szManufacturer**

Intel Corporation

**szMiniVdd**

unknown

**szMiniVddDateEnglish**

Unknown

**szMiniVddDateLocalized**

unknown

**szMonitorMaxRes**

Unknown

**szMonitorName**

Dell Alienware 15 LGD0690 Display

**szNotesEnglish**

No problems found.

**szNotesLocalized**

No problems found.

**szOverlayEnglish**

Not Supported

**szRankOfInstalledDriver**

00CF0001

**szRegHelpText**

Unknown

**szRevision**

Unknown

**szRevisionId**

0x000C

**szSubSysId**

0x0B541028

**szTestResultD3D7English**

Not run

**szTestResultD3D7Localized**

Not run

**szTestResultD3D8English**

Not run

**szTestResultD3D8Localized**

Not run

**szTestResultD3D9English**

Not run

**szTestResultD3D9Localized**

Not run

**szTestResultDDEnglish**

Not run

**szTestResultDDLocalized**

Not run

**szVdd**

unknown

**szVendorId**

0x8086

**1**

**b3DAccelerationEnabled**

true

**b3DAccelerationExists**

true

**bAGPEnabled**

true

**bAGPExistenceValid**

false

**bAGPExists**

false

**bCanRenderWindow**

false

**bDDAccelerationEnabled**

true

**bDX12UEnabled**

true

**bDriverBeta**

false

**bDriverDebug**

false

**bDriverSigned**

false

**bDriverSignedValid**

false

**bNoHardware**

false

**dwBpp**

0

**dwDDIVersion**

12

**dwHeight**

0

**dwRefreshRate**

0

**dwWHQLLevel**

0

**dwWidth**

0

**iAdapter**

0

**lDriverSize**

716392

**lMiniVddSize**

0

**szAGPStatusEnglish**

Enabled

**szAGPStatusLocalized**

Enabled

**szChipType**

NVIDIA GeForce RTX 3070 Ti Laptop GPU

**szD3DStatusEnglish**

Enabled

**szD3DStatusLocalized**

Enabled

**szDACType**

Integrated RAMDAC

**szDDIVersionEnglish**

12

**szDDIVersionLocalized**

12

**szDDStatusEnglish**

Enabled

**szDDStatusLocalized**

Enabled

**szDX12UStatusEnglish**

Enabled

**szDX12UStatusLocalized**

Enabled

**szDXVAHDEnglish**

Unknown

**szDXVAModes**

Unknown

**szDescription**

NVIDIA GeForce RTX 3070 Ti Laptop GPU

**szDeviceId**

0x24E0

**szDeviceIdentifier**

Unknown

**szDeviceName**

Unknown

**szDisplayMemoryEnglish**

24269 MB

**szDisplayMemoryLocalized**

24269 MB

**szDisplayModeEnglish**

Unknown

**szDisplayModeLocalized**

unknown

**szDriverAssemblyVersion**

30.0.15.1338

**szDriverAttributes**

Final Retail

**szDriverDateEnglish**

2022-07-06 4:00:00 PM

**szDriverDateLocalized**

7/6/2022 16:00:00

**szDriverLanguageEnglish**

English

**szDriverLanguageLocalized**

English

**szDriverModelEnglish**

WDDM 3.0

**szDriverModelLocalized**

WDDM 3.0

**szDriverName**

C:\WINDOWS\System32\DriverStore\FileRepository\nvdmegpu.inf\_amd64\_72a92fd725b79e70\nvldumdx.dll,C:\WINDOWS\System32\DriverStore\FileRepository\nvdmegpu.inf\_amd64\_72a92fd725b79e70\nvldumdx.dll,C:\WINDOWS\System32\DriverStore\FileRepository\nvdmegpu.inf\_amd64\_72a92fd725b79e70\nvldumdx.dll,C:\WINDOWS\System32\DriverStore\FileRepository\nvdmegpu.inf\_amd64\_72a92fd725b79e70\nvldumdx.dll

**szDriverNodeStrongName**

oem69.inf:0f066de39b945a6c:Section234:30.0.15.1338:pci\ven\_10de&dev\_24e0&subsys\_0b541028

**szDriverSignDate**

Unknown

**szDriverVersion**

30.00.0015.1338

**szKeyDeviceID**

Enum\PCI\VEN\_10DE&DEV\_24E0&SUBSYS\_0B541028&REV\_A1

**szKeyDeviceKey**

Unknown

**szManufacturer**

NVIDIA

**szMiniVdd**

unknown

**szMiniVddDateEnglish**

Unknown

**szMiniVddDateLocalized**

unknown

**szMonitorMaxRes**

Unknown

**szMonitorName**

Unknown

**szNotesEnglish**

No problems found.

**szNotesLocalized**

No problems found.

**szOverlayEnglish**

Unknown

**szRankOfInstalledDriver**

00CF0001

**szRegHelpText**

Unknown

**szRevision**

Unknown

**szRevisionId**

0x00A1

**szSubSysId**

0x0B541028

**szTestResultD3D7English**

Not run

**szTestResultD3D7Localized**

Not run

**szTestResultD3D8English**

Not run

**szTestResultD3D8Localized**

Not run

**szTestResultD3D9English**

Not run

**szTestResultD3D9Localized**

Not run

**szTestResultDDEnglish**

Not run

**szTestResultDDLocalized**

Not run

**szVdd**

unknown

**szVendorId**

0x10DE

**Log Messages**

* GpuProcessHost: The info collection GPU process exited normally. Everything is okay.