AsyncCompute

AsyncCompute is a hardware feature that interleaves different GPU tasks to improve efficiency.

The Rendering Hardware Interface (RHI) now supports asynchronous compute (**AsyncCompute**) for Xbox One. This is a good way to utilize unused GPU resources (Compute Units (CUs), registers and bandwidth), by running **dispatch()** calls asynchronously with the rendering. Async compute uses a separate context, and we provide RHI functions to synchronize the rendering and compute contexts. Dr PIX is useful for identifying areas which might benefit from async compute. For example, if half the CUs are unused during a particular rendering pass, those CUs could potentially be utilized by an asynchronous compute job.

Async compute has some restrictions:

Buffers with UAV counters are not supported (this is a limitation of the XDK, and will generate a D3D warning) Async compute jobs do not show up in PIX GPU captures (although they do appear in Timing Captures), PIX only captures the graphics immediate context (this is a platform limitation). * Automatic pipeline flushes are not provided by the driver. You need to explicitly call RHICSManualGpuFlush if flushes are needed(e.g. if one dispatch is dependent on the previous one).

API

RHIBeginAsyncComputeJob_DrawThread *(EAsyncComputePriority Priority) Begins an async compute job from the rendering thread. The priority is used to determine the number of shader resources to allocate for the job (via ID3D11XComputeContext::SetLimits). There are currently two priorities available, AsyncComputePriority_High and AsyncComputePriority_Default.* **RHIEndAsyncComputeJob_DrawThread** Ends an async compute job. Returns a 32-bit Fence Index which can be used for synchronization, or -1 if compute is disabled or there is no active compute job * **RHIGraphicsWaitOnAsyncComputeJob** Inserts n command in the graphics immediate context to block until an async job is complete. Passing -1 causes this to early-out.

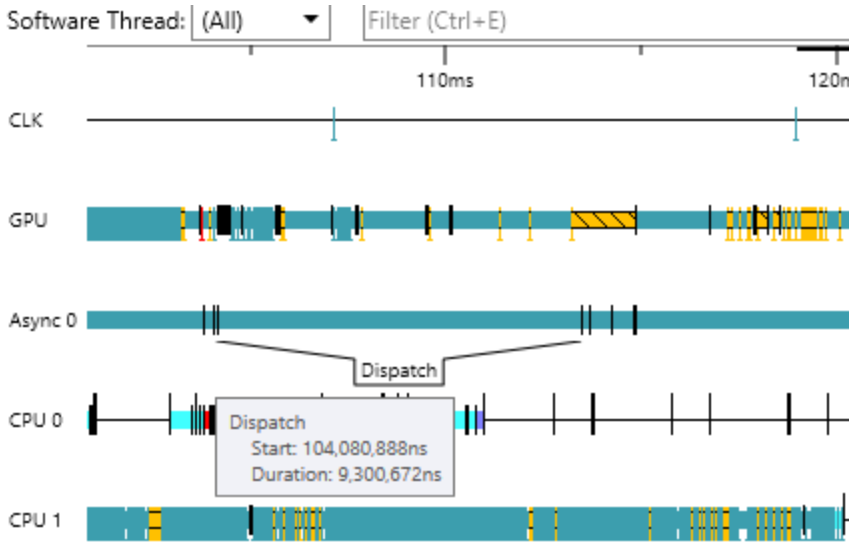
Between calls to RHIBeginAsyncComputeJob_DrawThread and RHIEndAsyncComputeJob_DrawThread, the RHI will be in the async compute state. During this time, supported RHI commands will be executed via the async compute context. Unsupported RHI functions will assert.

Disabling/Enabling

Async compute can be enabled or disabled at compile time with the #define USE_ASYNC_COMPUTE_CONTEXT. It can be disabled at runtime with the r.AsyncCompute console variable. When async compute is disabled, dispatches within async compute blocks are executed synchronously on the graphics command buffer. USE_ASYNC_COMPUTE_CONTEXT is defined to 0 on PC, since it's not supported in D3D11.1.

PIX

Async compute context jobs are not captured in GPU captures, so these captures can give a misleading picture of GPU performance when async compute is enabled. For graphics debugging purposes, async compute should be disabled using the above cvar. Async compute is supported by PIX timing captures. These show up in the timeline like this:



Credit

This feature was implemented by Lionhead Studios. We integrated it and indend to make use of it as a tool to optimize XboxOne rendering.