Tiling Format for Intel® Iris™ Graphics Extension for Instant Access
By Dave Bookout

Introduction
Intel® Iris™ graphics extension for instant access for 4th generation Intel® Core™ and later processors allows direct access to memory allocated for the GPU. The extension provides a mechanism for specifying which buffers will be shared and locking the memory for reading and writing from CPU side code.

This sample demonstrates how to properly access the memory from the CPU to maintain performance with the sample code for computing the tiled addressing scheme. This sample has been updated to include support for the MAP_TILE_TYPE_TILE_Y_NO_CSX_SWIZZLE format.

Intel® Iris™ Graphics Extension for Instant Access
The extension for instant access provides a mechanism for writing directly to resources allocated for the GPU without passing through the additional staging texture. The extension requires some additional setup code and overrides the Direct3D* functions CopyResource and Map. To utilize the extension, some host-side initialization code is required. Sample code is provided in the IGFXExtensionsHelper.h/cpp files. Before using any extension, the Init function must be called.

To utilize the extension for instant access, two buffers are created, each with a call to SetDirectAccessResourceExtension immediately before the buffer creation calls. One buffer is created as a normal texture, but the call to SetDirectAccessResourceExtension marks it as lockable from the CPU. The other buffer is created as a staging buffer, but the call to SetDirectAccessResourceExtension means that the buffer will be used to directly access GPU memory so no additional memory is allocated for the staging buffer. Finally, the Direct3D CopyResource function is called with the two buffers as parameters to bind them together.

To get CPU-side access to the resource, the application calls the Direct3D Map function on the staging buffer. The data returned in the D3D11_MAPPED_SUBRESOURCE structure is RESOURCE_SHARED_MAP_DATA structure. This new structure contains information about the GPU resource including the CPU-side pointer to the memory and the tiling format.

Sample Implementation
The sample demonstrates how to convert between linear memory (for example, the memory layout of a mapped staging texture, y × width + x) and the tiled memory layout for the resources available through the extension. The sample demonstrates this conversion for the MAP_TILE_TYPE_TILE_Y and MAP_TILE_TYPE_TILE_Y_NO_CSX_SWIZZLE formats. The tiles are column major with 16 bytes per row and 32 rows and are 128 bits wide. For the tiled address 4 bits specify the x position within a column; 5 bits specify the y position in the column (32 rows per tile), and the higher order bits specifying x position of the column (number of bits dependent on texture width) and tile row. For the MAP_TILE_TYPE_TILE_Y layout, alternate columns in a tile (columns 1, 3, 5,...) have an additional swizzle where 4 rows are swapped.
Figure 1. MAP_TILE_TYPE_TILE_Y layout showing additional swizzle of alternating columns.
The memory accessed through the DRA extension is "write combined memory," which is intended for high write throughput. When writing to the memory sequentially, several writes can be combined into a single more efficient write. The functions show the performance of writing in tile order versus writing in linear order and how to convert between the two. An additional highly optimized method (provided by Axel Mamode) demonstrates an efficient method for copying linear memory to tiled memory.

Generally, writing in tile order is fastest. The optimized linear version provides an efficient method for calculating the correct tiled address and writes to memory in large enough sequential chunks that write combining still works well. Unfortunately, the optimizations for write combined memory make reading from that memory slow.

Additional Resources
Fabian Giesen's blog ("The ryg blog") is a good source of information for tiling/swizzling and write combined memory, particularly these posts: "Texture tiling and swizzling" (https://fgiesen.wordpress.com/2011/01/17/texture-tiling-and-swizzling/) and "Write combining is not your friend" (https://fgiesen.wordpress.com/2013/01/29/write-combining-is-not-your-friend/)

Notices

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.

The products and services described may contain defects or errors known as errata which may cause deviations from published specifications. Current characterized errata are available on request.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or by visiting www.intel.com/design/literature.htm.

Intel, the Intel logo, Intel Core, and Iris are trademarks of Intel Corporation in the U.S. and/or other countries.

*Other names and brands may be claimed as the property of others

© 2015 Intel Corporation.