Device Memory

1. Memory Data

Memory in our device can be classified as follow:

* **Video memory:** memory in graphics card.
* **System memory:** main memory.

And we have several generally usage:

* **Static data:** we only initialize it once.
* **CPU to GPU data:** we will modify it and need to update it often.
* **GPU to CPU data:** we need a coherent memory space. And the space is cached at system memory.

1. Device Memory

As mentioned at chapter 1, there are two memory spaces in our device. One is **system memory**, another is **video memory**. GPU can access video memory **directly** and access system memory **via driver.** So we can call memory allocated from video memory as **device local memory** and call allocated memory from system memory as **host memory**. But we need to notice one thing – GPU access host memory slower than device memory. Therefore, we should use host memory when we transfer data to device memory or small the data that maybe modify frequently.

There are memory types and heaps in our device. And each heap will specify its usage about memory. Memory type is used to tell us the usage about memory.

* 1. Memory Type

Memory type is used to specify memory usage. There are several memory types as follow:

|  |  |
| --- | --- |
| Memory Type | Description |
| VK\_MEMORY\_PROPERTY\_DEVICE\_LOCAL\_BIT (0x00000001) | The memory can be accessed efficiently. It’s usually in video memory heap. |
| VK\_MEMORY\_PROPERTY\_HOST\_VISIBLE\_BIT (0x00000002) | The memory can be mapped. It means the system memories can be seen by GPU.  Across PCIe® bus, reads cached on GPU. So we need call flush after memcpy. |
| VK\_MEMORY\_PROPERTY\_HOST\_COHERENT\_BIT (0x00000004) | GPU access through PCIe®. So we don’t need to call flush after memcpy. CPU maybe can’t access. |
| VK\_MEMORY\_PROPERTY\_HOST\_CACHED\_BIT (0x00000008) | GPU access through PCIe®. Cached property include coherent. CPU can access. |
| VK\_MEMORY\_PROPERTY\_LAZILY\_ALLOCATED\_BIT (0x00000010) | ???? |
| VK\_MEMORY\_PROPERTY\_PROTECTED\_BIT (0x00000020) | Mutually exclusive with host memory. But ???? |

One memory can follow one group of properties. Usual group list as follow :

* DEVICE\_LOCAL\_BIT
* HOST\_VISIBLE\_BIT
* HOST\_VISIBLE\_BIT | HOST\_COHERENT\_BIT | HOST\_CACHED\_BIT
* DEVICE\_LOCAL\_BIT | HOST\_VISIBLE\_BIT(AMD only. 256MB on windows.)
  1. Memory Heap

Heap is the pool that is used to store memory. And the number of heaps is different between different devices. Each heap keeps particular type memory. For example, there are two heaps in my PC that it use 16GB RAM and graphics card Nvidia 1060 6GB. All memory type related with host memory is allocated from heap which size is approximately with DRAM. And all memory type related with device local memory is allocated from heap which size is approximately with graphics cards.

There are three heap flag in Vulkan :

* **0 :** DRAM heap.
* **VK\_MEMORY\_HEAP\_DEVICE\_LOCAL\_BIT** **:** Video memory heap.
* **VK\_MEMORY\_HEAP\_MULTI\_INSTANCE\_BIT:** Video memory heap. Allocate this memory at all graphics devices.(SLI ????)

1. Memory Type and Heap in Vulkan

We can get memory information from physical device. API **vkGetPhysicalDeviceMemoryProperties** is used to get all supported **memory types** and **heaps** in this graphics device. The input of the API is structure **VkPhysicalDeviceMemoryProperties**. In the structure, we can see what **memory type will be allocated from which heap,** and **heap size and its property**.

1. Buffer Creation in Vulkan
2. Data transfer
3. Compare with OpenGL
4. Reference

[1] Khronos PPT <https://www.khronos.org/assets/uploads/developers/library/2018-vulkan-devday/03-Memory.pdf>

[2] Nvidia Vulkan

<https://developer.nvidia.com/vulkan-memory-management>

[3] AMD Vulkan

<https://gpuopen.com/vulkan-device-memory/>

[4] OpenGL Insight

<https://www.seas.upenn.edu/~pcozzi/OpenGLInsights/OpenGLInsights-AsynchronousBufferTransfers.pdf>