These are the Commands that could be entered into the Command Buffer, I:

- vkCmdPushDescriptorSetKHR
- vkCmdSetViewport
- vkCmdSetViewportWScalingNV
- vkCmdSetStencilWriteMask
- vkCmdSetStencilCompareMask
- vkCmdSetStencilReference
- vkCmdSetDiscardRectangleEXT
- vkCmdSetDeviceMaskKHX
- vkCmdSetDepthBias
- vkCmdResetEvent
- vkCmdPushConstants
- vkCmdProcessCommandsNVX
- vkCmdPipelineBarrier
- vkCmdNextSubpass
- vkCmdEndRenderPass
- vkCmdEndQuery
- vkCmdDraw
- vkCmdDispatchIndirect
- vkCmdDebugMarkerInsertEXT
- vkCmdDebugMarkerEndEXT
- vkCmdDebugMarkerBeginEXT
- vkCmdCopyQueryPoolResults
- vkCmdCopyImageToBuffer
- vkCmdCopyImage
- vkCmdClearDepthStencilImage
- vkCmdClearAttachments
- vkCmdBindIndexBuffer
- vkCmdBindDescriptorSets
- vkCmdBeginQuery
- vkCmdExecuteCommands

From the Command Buffer Notes:

1. We want all of these commands to be able to run "side-by-side", but if we do that, surely there will be race conditions!

vkCmdPipelineBarrier | Function Call

A Pipeline Barrier is a way to establish a memory dependency between commands that were submitted before the barrier and commands that are submitted after the barrier.

- srcStageMask
- dstStageMask
- memoryBarrierCount
- bufferMemoryBarrierCount
- imageMemoryBarrierCount
- VK_DEPENDENCY_BY_REGION_BIT
- Vulkan Barrier

Pipeline Stage Flags – Where in the Pipeline is this Memory being Accessed?

- VK_PIPELINE_STAGE_TOP_OF_PIPE_BIT
- VK_PIPELINE_STAGE_BOTTOM_OF_PIPE_BIT
- VK_PIPELINE_STAGE_DRAW_INDIRECT_BIT
- VK_PIPELINE_STAGE_VERTEX_SHADER_BIT
- VK_PIPELINE_STAGE_FRAGMENT_SHADER_BIT
- VK_PIPELINE_STAGE_TESSELLATION_CONTROL_SHADER_BIT
- VK_PIPELINE_STAGE_TESSELLATION_EVALUATION_SHADER_BIT
- VK_PIPELINE_STAGE_GEOMETRY_SHADER_BIT
- VK_PIPELINE_STAGE_COMPUTE_SHADER_BIT
- VK_PIPELINE_STAGE_VERTEX_INPUT_BIT
- VK_PIPELINE_STAGE_VERTEX_OUTPUT_BIT
- VK_PIPELINE_STAGE_FRAGMENT_INPUT_BIT
- VK_PIPELINE_STAGE_EARLY_FRAGMENT_TESTS_BIT
- VK_PIPELINE_STAGE_LATE_FRAGMENT_TESTS_BIT
- VK_PIPELINE_STAGE_COLOR_ATTACHMENT_OUTPUT_BIT
- VK_PIPELINE_STAGE_ALL_GRAPHICS_BIT

The Scenario

- src cars
- dst cars

Note: the concept of an in-order pipeline is accurate, but really the src and dst triggering action only depends on the name of the street where you are right now!
Example: Don’t overwrite a texture image until we are done using it

```
vmb.dstAccessMask = ??;
vmb.srcAccessMask = ??;
vmb.pNext = nullptr;
vmb.sType = VK_STRUCTURE_TYPE_MEMORY_BARRIER;
```

The Scenario

```
src cars
```
```
dst cars
```
```
dat cars
```
Example: Don't read a buffer back to the host until a shader is done writing it

- VK_PIPELINE_STAGE_TOP_OF_PIPE_BIT
- VK_PIPELINE_STAGE_DRAW_INDIRECT_BIT
- VK_PIPELINE_STAGE_VERTEX_INPUT_BIT
- VK_PIPELINE_STAGE_VERTEX_SHADER_BIT
- VK_PIPELINE_STAGE_TESSELLATION_CONTROL_SHADER_BIT
- VK_PIPELINE_STAGE_TESSELLATION_EVALUATION_SHADER_BIT
- VK_PIPELINE_STAGE_GEOMETRY_SHADER_BIT
- VK_PIPELINE_STAGE_FRAGMENT_SHADER_BIT
- VK_PIPELINE_STAGE_EARLY_FRAGMENT_TESTS_BIT
- VK_PIPELINE_STAGE_LATE_FRAGMENT_TESTS_BIT
- VK_PIPELINE_STAGE_COLOR_ATTACHMENT_OUTPUT_BIT
- VK_PIPELINE_STAGE_COMPUTE_SHADER_BIT
- VK_PIPELINE_STAGE_TRANSFER_BIT
- VK_PIPELINE_STAGE_BOTTOM_OF_PIPE_BIT
- VK_PIPELINE_STAGE_HOST_BIT
- VK_PIPELINE_STAGE_ALL_GRAPHICS_BIT
- VK_PIPELINE_STAGE_ALL_COMMANDS_BIT

The Scenario

Buffer Memory Barrier

```cpp
VkBufferMemoryBarrier vbmb;
vbmb.sType = VK_STRUCTURE_TYPE_BUFFER_MEMORY_BARRIER;
vbmb.pNext = nullptr;
vbmb.srcAccessMask = ??;
vbmb.dstAccessMask = ??;
vbmb.srcQueueFamilyIndex = VK_QUEUE_FAMILY_IGNORED;
vbmb.dstQueueFamilyIndex = VK_QUEUE_FAMILY_IGNORED;
vbmb.buffer = ??;
vbmb.offset = 0;
vbmb.size = VK_WHOLE_SIZE;
```

Image Memory Barrier

```cpp
VkImageMemoryBarrier vimb;

vimb.sType = VK_STRUCTURE_TYPE_IMAGE_MEMORY_BARRIER;
vimb.pNext = nullptr;
vimb.srcAccessMask = ??;
vimb.dstAccessMask = ??;
vimb.oldLayout = ??;
vimb.newLayout = ??;
vimb.srcQueueFamilyIndex = VK_QUEUE_FAMILY_IGNORED;
vimb.dstQueueFamilyIndex = VK_QUEUE_FAMILY_IGNORED;
vimb.image = ??;
vimb.subresourceRange = visr;
```

Image Subresource Range

```cpp
VkImageSubresourceRange visr;
visr.aspectMask = VK_IMAGE_ASPECT_COLOR_BIT;
visr.baseMipLevel = 0;
visr.levelCount = 1;
visr.baseArrayLayer = 0;
visr.layerCount = 1;
```

VkImageLayout – How an Image gets Laid Out in Memory depends on How it will be Used

- VK_IMAGE_LAYOUT_UNDEFINED
- VK_IMAGE_LAYOUT_SHARED_PRESENT_KHR
- VK_IMAGE_LAYOUT_PRESENT_SRC_KHR
- VK_IMAGE_LAYOUT_COLOR_ATTACHMENT_OPTIMAL
- VK_IMAGE_LAYOUT_DEPTH_STENCIL_ATTACHMENT_OPTIMAL
- VK_IMAGE_LAYOUT_TRANSFER_SRC_OPTIMAL
- VK_IMAGE_LAYOUT_TRANSFER_DST_OPTIMAL
- VK_IMAGE_LAYOUT_SHADER_READ_ONLY_OPTIMAL
- VK_IMAGE_LAYOUT_TRANSFER_SRC_OPTIMAL
- VK_IMAGE_LAYOUT_TRIANGLES
- VK_IMAGE_LAYOUT_INDEX

One use of `vkCmdPipelineBarrier()` is to simply change the layout of an image.