One-Time Command for Copying

```
VkCommandBuffer
VkBuffer
VkBuffer
```

```
void vkCmdCopyBuffer(
    VkCommandBuffer      commandBuffer,
    VkBuffer             srcBuffer,
    VkBuffer             dstBuffer,
    uint32_t             regionCount,
    const VkBufferCopy*  pRegions
);
```

```
typedef struct VkBufferCopy {
    VkDeviceSize    srcOffset;
    VkDeviceSize    dstOffset;
    VkDeviceSize    size;
} VkBufferCopy;
```

```
typedef struct VkImageSubresourceLayers {
    VkImageAspectFlags    aspectMask;
    uint32_t              mipLevel;
    uint32_t              baseArrayLayer;
    uint32_t              layerCount;
} VkImageSubresourceLayers;
```

```
typedef struct VkBufferImageCopy {
    VkDeviceSize              bufferOffset;
    uint32_t                  bufferRowLength;
    uint32_t                  bufferImageHeight;
    VkImageSubresourceLayers  imageSubresource;
    VkOffset3D                imageOffset;
    VkExtent3D                imageExtent;
} VkBufferImageCopy;
```

```
enum VkImageAspectFlagBits
VK_IMAGE_ASPECT_COLOR_BIT = 0x00000001,
VK_IMAGE_ASPECT_DEPTH_BIT = 0x00000002,
VK_IMAGE_ASPECT_STENCIL_BIT = 0x00000004,
VK_IMAGE_ASPECT_METADATA_BIT = 0x00000008,
VK_IMAGE_ASPECT_PLANE_0_BIT = 0x00000010,
VK_IMAGE_ASPECT_PLANE_1_BIT = 0x00000020,
VK_IMAGE_ASPECT_PLANE_2_BIT = 0x00000040,
VK_IMAGE_ASPECT_NONE = 0,
VK_IMAGE_ASPECT_MEMORY_PLANE_0_BIT_EXT = 0x00000080,
VK_IMAGE_ASPECT_MEMORY_PLANE_1_BIT_EXT = 0x00000100,
VK_IMAGE_ASPECT_MEMORY_PLANE_2_BIT_EXT = 0x00000200,
VK_IMAGE_ASPECT_MEMORY_PLANE_3_BIT_EXT = 0x00000400,
VK_IMAGE_ASPECT_PLANE_0_BIT_KHR = VK_IMAGE_ASPECT_PLANE_0_BIT,
VK_IMAGE_ASPECT_PLANE_1_BIT_KHR = VK_IMAGE_ASPECT_PLANE_1_BIT,
VK_IMAGE_ASPECT_PLANE_2_BIT_KHR = VK_IMAGE_ASPECT_PLANE_2_BIT,
VK_IMAGE_ASPECT_NONE_KHR = VK_IMAGE_ASPECT_NONE,
```

```
typedef enum VkImageLayout {
    VK_IMAGE_LAYOUT_UNDEFINED = 0,
    VK_IMAGE_LAYOUT_GENERAL = 1,
    VK_IMAGE_LAYOUT_COLOR_ATTACHMENT_OPTIMAL = 2,
    VK_IMAGE_LAYOUT_DEPTH_STENCIL_ATTACHMENT_OPTIMAL = 3,
    VK_IMAGE_LAYOUT_DEPTH_STENCIL_READ_ONLY_OPTIMAL = 4,
    VK_IMAGE_LAYOUT_SHADER_READ_ONLY_OPTIMAL = 5,
    VK_IMAGE_LAYOUT_TRANSFER_SRC_OPTIMAL = 6,
    VK_IMAGE_LAYOUT_TRANSFER_DST_OPTIMAL = 7,
    VK_IMAGE_LAYOUT_PREINITIALIZED = 8,
    ...
} VkImageLayout;
```

```
VkCommandBuffer
VkBuffer
VkImage
```

```
void vkCmdCopyBufferToImage(
    VkCommandBuffer          commandBuffer,
    VkBuffer                 srcBuffer,
    VkImage                  dstImage,
    VkImageLayout            dstImageLayout,
    uint32_t                 regionCount,
    const VkBufferImageCopy* pRegions
);
```

```
void vkCmdCopyImageToBuffer(
    VkCommandBuffer          commandBuffer,
    VkImage                  srcImage,
    VkImageLayout            srcImageLayout,
    VkBuffer                 dstBuffer,
    uint32_t                 regionCount,
    const VkBufferImageCopy* pRegions
);
```

```
typedef struct VkImageSubresourceLayers {
    VkImageAspectFlags    aspectMask;
    uint32_t              mipLevel;
    uint32_t              baseArrayLayer;
    uint32_t              layerCount;
} VkImageSubresourceLayers;
```

```
typedef struct VkImageBlit {
    VkImageSubresourceLayers    srcSubresource;
    VkOffset3D                  srcOffsets[2];
    VkImageSubresourceLayers    dstSubresource;
    VkOffset3D                  dstOffsets[2];
} VkImageBlit;
```

```
VkCommandBuffer
VkImage
VkImage
```

```
typedef enum VkFilter {
    VK_FILTER_NEAREST,
    VK_FILTER_LINEAR,
    VK_FILTER_CUBIC_EXT,
    VK_FILTER_CUBIC_IMG,
} VkFilter;
```

```
void vkCmdBlitImage(
    VkCommandBuffer      commandBuffer,
    VkImage              srcImage,
    VkImageLayout        srcImageLayout,
    VkImage              dstImage,
    VkImageLayout        dstImageLayout,
    uint32_t             regionCount,
    const VkImageBlit*   pRegions,
    VkFilter             filter
);
```