clEnqueueWriteImage

Enqueues a command to write from a 2D or 3D image object to host memory.

```
cl_int clEnqueueWriteImage (cl_command_queue command_queue,
                            cl_mem image,
                            cl_bool blocking_write,
                            const size_t origin[3],
                            const size_t region[3],
                            size_t input_row_pitch,
                            size_t input_slice_pitch,
                            const void * ptr,
                            cl_uint num_events_in_wait_list,
                            const cl_event *event_wait_list,
                            cl_event *event)
```

Parameters

command_queue

Refers to the command-queue in which the write command will be queued. *command_queue* and image must be created with the same OpenCL contex

image

Refers to a valid 2D or 3D image object.

blocking_write

Indicates if the write operation is *blocking* or *non-blocking*.

If *blocking_write* is CL_TRUE the OpenCL implementation copies the data referred to by *ptr* and enqueues the write command in the command-queue. The memory pointed to by *ptr* can be reused by the application after the **clEnqueueWriteImage** call returns.

If *blocking_write* is CL_FALSE the OpenCL implementation will use *ptr* to perform a nonblocking write. As the write is non-blocking the implementation can return immediately. The memory pointed to by *ptr* cannot be reused by the application after the call returns. The *event* argument returns an event object which can be used to query the execution status of the write command. When the write command has completed, the memory pointed to by *ptr* can then be reused by the application.

origin

Defines the (*x, y, z*) offset in pixels in the image from where to write or write. If *image* is a 2D image object, the z value given by *origin*[2] must be 0.

region

Defines the (*width*, *height*, *depth*) in pixels of the 2D or 3D rectangle being write or written. If *image* is a 2D image object, the *depth* value given by *region*[2] must be 1.

input_row_pitch

The length of each row in bytes. This value must be greater than or equal to the element size in bytes * *width*. If *input_row_pitch* is set to 0, the appropriate row pitch is calculated based on the size of each element in bytes multiplied by *width*.

input_slice_pitch

Size in bytes of the 2D slice of the 3D region of a 3D image being written. This must be 0 if *image* is a 2D image. This value must be greater than or equal to *row_pitch* * *height*. If *input_slice_pitch* is set to 0, the appropriate slice pitch is calculated based on the *row_pitch* * *height*.

ptr

The pointer to a buffer in host memory where image data is to be written to.

event_wait_list , num_events_in_wait_list

Specify events that need to complete before this particular command can be executed. If *event_wait_list* is NULL, then this particular command does not wait on any event to complete. If *event_wait_list* is NULL, *num_events_in_wait_list* must be 0. If *event_wait_list* is not NULL, the list of events pointed to by *event_wait_list* must be valid and *num_events_in_wait_list* must be greater than 0. The events specified in *event_wait_list* act as synchronization points. The context associated with events in *event_wait_list* and *command_queue* must be the same.

event

Returns an event object that identifies this particular write command and can be used to query or queue a wait for this particular command to complete. *event* can be NULL in which case it will not be possible for the application to query the status of this command or queue a wait for this command to complete.

Notes

Calling **clEnqueueWriteImage** to update the latest bits in a region of the image object with the *ptr* argument value set to *host_ptr* + (*origin*[2] * *image slice pitch + origin[1] * image row pitch + origin[0] * bytes per pixel*), where *host_ptr* is a pointer to the memory region specified when the image object being written is created with `CL_MEM_USE_HOST_PTR`, must meet the following requirements in order to avoid undefined behavior:

- The host memory region being written contains the latest bits when the enqueued write command begins execution.

- The *input_row_pitch* and *input_slice_pitch* argument values in **clEnqueueWriteImage** must be set to the image row pitch and slice pitch.
- The image object is not mapped.
- The image object is not used by any command-queue until the write command has finished execution.

Errors

clEnqueueWriteImage return CL_SUCCESS if the function is executed successfully. Otherwise, it returns one of the following errors.

- CL_INVALID_COMMAND_QUEUE if *command_queue* is not a valid command-queue.
- CL_INVALID_CONTEXT if the context associated with *command_queue* and *image* are not the same or if the context associated with *command_queue* and events in *event_wait_list* are not the same.
- CL_INVALID_MEM_OBJECT if *image* is not a valid image object.
- CL_INVALID_VALUE if the region being write or written specified by *origin* and *region* is out of bounds or if *ptr* is a NULL value.
- CL_INVALID_VALUE if *image* is a 2D image object and *origin*[2] is not equal to 0 or *region*[2] is not equal to 1 or *slice_pitch* is not equal to 0.
- CL_INVALID_EVENT_WAIT_LIST if *event_wait_list* is NULL and *num_events_in_wait_list* greater than 0, or *event_wait_list* is not NULL and *num_events_in_wait_list* is 0, or if event objects in *event_wait_list* are not valid events.
- CL_MEM_OBJECT_ALLOCATION_FAILURE if there is a failure to allocate memory for data store associated with *image*.
- CL_OUT_OF_HOST_MEMORY if there is a failure to allocate resources required by the OpenCL implementation on the host.

Specification

OpenCL Specification

Also see

clEnqueueReadImage, clEnqueueCopyImage