clEnqueueReadBuffer

Enqueue commands to read from a buffer object to host memory.

```
cl_int clEnqueueReadBuffer (cl_command_queue command_queue,
                            cl_mem buffer,
                            cl_bool blocking_read,
                            size_t offset,
                            size_t cb,
                            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 read command will be queued. *command_queue* and *buffer* must be created with the same OpenCL context.

buffer

Refers to a valid buffer object.

blocking_read

Indicates if the read operations are *blocking* or non-blocking. If *blocking_read* is CL_TRUE i.e. the read command is blocking, **clEnqueueReadBuffer** does not return until the buffer data has been read and copied into memory pointed to by *ptr*.

If *blocking_read* is CL_FALSE i.e. the read command is *non-blocking*, **clEnqueueReadBuffer** queues a *non-blocking* read command and returns. The contents of the buffer that *ptr* points to cannot be used until the read command has completed. The *event* argument returns an event object which can be used to query the execution status of the read command. When the read command has completed, the contents of the buffer that *ptr* points to can be used by the application.

offset

The offset in bytes in the buffer object to read from.

cb

The size in bytes of data being read.

ptr

The pointer to buffer in host memory where data is to be read into.

event_wait_list , num_events_in_wait_list

event_wait_list and *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 read 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 **clEnqueueReadBuffer** to read a region of the buffer object with the *ptr* argument value set to *host_ptr* + *offset*, where *host_ptr* is a pointer to the memory region specified when the buffer object being read is created with CL_MEM_USE_HOST_PTR, must meet the following requirements in order to avoid undefined behavior:

- All commands that use this buffer object have finished execution before the read command begins execution
- The buffer object is not mapped
- The buffer object is not used by any command-queue until the read command has finished execution

Errors

clEnqueueReadBuffer returns 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 *buffer* 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 *buffer* is not a valid buffer object.

- CL_INVALID_VALUE if the region being read specified by (*offset*, *cb*) is out of bounds or if *ptr* is a NULL value.
- 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 *buffer*.
- 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

clEnqueueCopyBuffer, clEnqueueWriteBuffer