clEnqueueNDRangeKernel

Enqueues a command to execute a kernel on a device.

```
cl_int clEnqueueNDRangeKernel (cl_command_queue command_queue,
                               cl_kernel kernel,
                               cl_uint work_dim,
                               const size_t *global_work_offset,
                               const size_t *global_work_size,
                               const size_t *local_work_size,
                               cl_uint num_events_in_wait_list,
                               const cl_event *event_wait_list,
                               cl_event *event)
```

Parameters

command_queue

A valid command-queue. The kernel will be queued for execution on the device associated with *command_queue*.

kernel

A valid kernel object. The OpenCL context associated with *kernel* and *command_queue* must be the same.

work_dim

The number of dimensions used to specify the global work-items and work-items in the work-group. *work_dim* must be greater than zero and less than or equal to three.

global_work_offset

Must currently be a NULL value. In a future revision of OpenCL, *global_work_offset* can be used to specify an array of *work_dim* unsigned values that describe the offset used to calculate the global ID of a work-item instead of having the global IDs always start at offset (0, 0,... 0).

global_work_size

Points to an array of *work_dim* unsigned values that describe the number of global work-items in *work_dim* dimensions that will execute the kernel function. The total number of global work-items is computed as *global_work_size*[0] *...* *global_work_size*[*work_dim* - 1].

The values specified in *global_work_size* cannot exceed the range given by the `sizeof(size_t)` for the device on which the kernel execution will be enqueued. The `sizeof(size_t)` for a device can be determined using `CL_DEVICE_ADDRESS_BITS` in the table of OpenCL Device Queries for

clGetDeviceInfo. If, for example, CL_DEVICE_ADDRESS_BITS = 32, i.e. the device uses a 32-bit address space, size_t is a 32-bit unsigned integer and global_work_size values must be in the range 1 .. 2^32 - 1. Values outside this range return a CL_OUT_OF_RESOURCES error.

local_work_size

Points to an array of *work_dim* unsigned values that describe the number of work-items that make up a work-group (also referred to as the size of the work-group) that will execute the kernel specified by *kernel*. The total number of work-items in a work-group is computed as *local_work_size*[0] *... * *local_work_size*[*work_dim* - 1]. The total number of work-items in the work-group must be less than or equal to the CL_DEVICE_MAX_WORK_GROUP_SIZE value specified in table of OpenCL Device Queries for clGetDeviceInfo and the number of work-items specified in *local_work_size*[0],... *local_work_size*[*work_dim* - 1] must be less than or equal to the corresponding values specified by CL_DEVICE_MAX_WORK_ITEM_SIZES[0],.... CL_DEVICE_MAX_WORK_ITEM_SIZES[*work_dim* - 1]. The explicitly specified *local_work_size* will be used to determine how to break the global work-items specified by *global_work_size* into appropriate work-group instances. If *local_work_size* is specified, the values specified in *global_work_size*[0],... *global_work_size*[*work_dim* - 1] must be evenly divisable by the corresponding values specified in *local_work_size*[0],... *local_work_size*[*work_dim* - 1].

The work-group size to be used for *kernel* can also be specified in the program source using the __attribute__((reqd_work_group_size(X, Y, Z)))qualifier. In this case the size of work group specified by *local_work_size* must match the value specified by the reqd_work_group_size __attribute__ qualifier.

local_work_size can also be a NULL value in which case the OpenCL implementation will determine how to be break the global work-items into appropriate work-group instances.

See note for more information.

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 kernel execution instance. Event objects are unique and can be used to identify a particular

kernel execution instance later on. If *event* is NULL, no event will be created for this kernel execution instance and therefore it will not be possible for the application to query or queue a wait for this particular kernel execution instance.

Notes

Work-group instances are executed in parallel across multiple compute units or concurrently on the same compute unit.

Each work-item is uniquely identified by a global identifier. The global ID, which can be read inside the kernel, is computed using the value given by *global_work_size* and *global_work_offset*. In OpenCL 1.0, the starting global ID is always (0, 0, ... 0). In addition, a work-item is also identified within a work-group by a unique local ID. The local ID, which can also be read by the kernel, is computed using the value given by *local_work_size*. The starting local ID is always (0, 0, ... 0).

Errors

Returns CL_SUCCESS if the kernel execution was successfully queued. Otherwise, it returns one of the following errors:

- CL_INVALID_PROGRAM_EXECUTABLE if there is no successfully built program executable available for device associated with *command_queue*.
- CL_INVALID_COMMAND_QUEUE if *command_queue* is not a valid command-queue.
- CL_INVALID_KERNEL if *kernel* is not a valid kernel object.
- CL_INVALID_CONTEXT if context associated with *command_queue* and *kernel* is not the same or if the context associated with *command_queue* and events in *event_wait_list* are not the same.
- CL_INVALID_KERNEL_ARGS if the kernel argument values have not been specified.
- CL_INVALID_WORK_DIMENSION if *work_dim* is not a valid value (i.e. a value between 1 and 3).
- CL_INVALID_WORK_GROUP_SIZE if *local_work_size* is specified and number of work-items specified by *global_work_size* is not evenly divisable by size of work-group given by *local_work_size* or does not match the work-group size specified for *kernel* using the __attribute__((reqd_work_group_size(X, Y, Z))) qualifier in program source.
- CL_INVALID_WORK_GROUP_SIZE if *local_work_size* is specified and the total number of work-items in the work-group computed as *local_work_size*[0] *... *local_work_size*[*work_dim* - 1] is greater than the

value specified by CL_DEVICE_MAX_WORK_GROUP_SIZE in the table of OpenCL Device Queries for clGetDeviceInfo.

- CL_INVALID_WORK_GROUP_SIZE if *local_work_size* is NULL and the __attribute__((reqd_work_group_size(X, Y, Z))) qualifier is used to declare the work-group size for *kernel* in the program source.

- CL_INVALID_WORK_ITEM_SIZE if the number of work-items specified in any of *local_work_size*[0], ... *local_work_size*[*work_dim* - 1] is greater than the corresponding values specified by CL_DEVICE_MAX_WORK_ITEM_SIZES[0], ... CL_DEVICE_MAX_WORK_ITEM_SIZES[*work_dim* - 1].

- CL_INVALID_GLOBAL_OFFSET if *global_work_offset* is not NULL.

- CL_OUT_OF_RESOURCES if there is a failure to queue the execution instance of *kernel* on the command-queue because of insufficient resources needed to execute the kernel. For example, the explicitly specified *local_work_size* causes a failure to execute the kernel because of insufficient resources such as registers or local memory. Another example would be the number of read-only image args used in *kernel* exceed the CL_DEVICE_MAX_READ_IMAGE_ARGS value for device or the number of write-only image args used in *kernel* exceed the CL_DEVICE_MAX_WRITE_IMAGE_ARGS value for device or the number of samplers used in *kernel* exceed CL_DEVICE_MAX_SAMPLERS for device.

- CL_MEM_OBJECT_ALLOCATION_FAILURE if there is a failure to allocate memory for data store associated with image or buffer objects specified as arguments to *kernel*.

- CL_INVALID_EVENT_WAIT_LIST if *event_wait_list* is NULL and *num_events_in_wait_list* > 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_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

clCreateCommandQueue, clGetDeviceInfo, clEnqueueNativeKernel, clEnqueueTask, Work-Item Functions