Name

CL_QCOM_EXT_HOST_PTR

Name Strings

cl_qcom_ext_host_ptr

Contact

ssusheel at quicinc dot com
dgarcia at qti dot qualcomm dot com

Contributors

Balaji Calidas, QUALCOMM
David Garcia, QUALCOMM
Rajeev Rao, QUALCOMM
Sushmita Susheelendra, QUALCOMM Innovation Center Inc.

Status

SHIPPING

Number

OpenCL Extension #21

Dependencies

OpenCL 1.1 is required.

This extension is written against the OpenCL 1.1 specification

Overview

This extension extends the functionality provided by
clCreateBuffer, clCreateImage2D, clCreateImage3D.
It allows applications to specify a new flag CL_MEM_EXT_HOST_PTR_QCOM
which enables the driver to map external memory allocations
--to be defined in future layered extensions--

to the device's address space and thus avoiding having to
copy data back and forth between the host and the device.

IP Status

No known IP claims.

New Procedures and Functions

```
cl_int clGetDeviceImageInfoQCOM(cl_device_id            device,
                                size_t                  image_width,
                                size_t                  image_height,
                                const cl_image_format   *image_format,
                                cl_image_pitch_info_qcom param_name,
                                size_t                  param_value_size,
                                void                    *param_value,
                                size_t                  *param_value_size_ret);
```

New Types

```
typedef cl_uint cl_image_pitch_info_qcom;
```

New Tokens

Accepted by the <param_name> argument of clGetDeviceInfo

```
CL_DEVICE_EXT_MEM_PADDING_IN_BYTES_QCOM    0x40A0
CL_DEVICE_PAGE_SIZE_QCOM                   0x40A1
```

Accepted by the <flags> argument of clCreateBuffer, clCreateImage2D and clCreateImage3D:

```
CL_MEM_EXT_HOST_PTR_QCOM                   (1 << 29)
```

Accepted by the <host_ptr> argument of clCreateBuffer, clCreateImage2D and clCreateImage3D:

```
typedef struct _cl_mem_ext_host_ptr
{
    // Type of external memory allocation.
    // Legal values will be defined in layered extensions.
    cl_uint  allocation_type;

    // Host cache policy for this external memory allocation.
```

```
    cl_uint  host_cache_policy;

  } cl_mem_ext_host_ptr;
```

Accepted values for cl_mem_ext_host_ptr::host_cache_policy:

```
  CL_MEM_HOST_UNCACHED_QCOM               0x40A4
  CL_MEM_HOST_WRITEBACK_QCOM              0x40A5
  CL_MEM_HOST_WRITETHROUGH_QCOM           0x40A6
  CL_MEM_HOST_WRITE_COMBINING_QCOM        0x40A7
```

Accepted by the <param_name> argument of clGetDeviceImageInfoQCOM

```
  CL_IMAGE_ROW_ALIGNMENT_QCOM             0x40A2
  CL_IMAGE_SLICE_ALIGNMENT_QCOM           0x40A3
```

Additions to Chapter 5.2.1 of the OpenCL 1.1 Specification
(Creating Buffer Objects)

Add the following token to Table 5.3 (clCreateBuffer List of supported cl_mem_flags values):

```
  CL_MEM_EXT_HOST_PTR_QCOM          This flag is valid only when used together with CL_MEM_USE_HOST_PTR.
                                    If specified, it indicates that the <host_ptr> argument provided by
                                    the application is actually a pointer to cl_mem_ext_host_ptr.
```

When CL_MEM_EXT_HOST_PTR_QCOM is enabled in the <flags> argument, then <host_ptr> is interpreted as a pointer to cl_mem_ext_host_ptr.
The application must then initialize cl_mem_ext_host_ptr::allocation_type to the allowed token values defined in future layered extensions.

The application must also initialize cl_mem_ext_host_ptr::host_cache_policy to one of CL_MEM_HOST_UNCACHED_QCOM, CL_MEM_HOST_WRITEBACK_QCOM,
CL_MEM_HOST_WRITETHROUGH_QCOM, or CL_MEM_HOST_WRITE_COMBINING_QCOM according to the cache policy used in the host for this memory allocation.

Add the following token to Table 4.3 (clGetDeviceInfo OpenCL Device Queries):

```
  CL_DEVICE_EXT_MEM_PADDING_IN_BYTES_QCOM    Returns the amount of memory padding that the application must add
                                             to the end of every external allocation that will be used in conjunction with
                                             CL_MEM_EXT_HOST_PTR_QCOM.

  CL_DEVICE_PAGE_SIZE_QCOM                   Returns the device's page size.
```

The application may query the row and slice pitch values using clGetDeviceImageInfoQCOM
and provide the queried values or any other supported value to clCreateImage2D and clCreateImage3D

when using CL_MEM_EXT_HOST_PTR_QCOM.

A supported value for row pitch and slice pitch is defined respectively as:

* Any value greater than or equal to CL_IMAGE_ROW_PITCH that is also a multiple of CL_IMAGE_ROW_ALIGNMENT_QCOM
* Any value greater than or equal to CL_IMAGE_SLICE_PITCH that is also a multiple of CL_IMAGE_SLICE_ALIGNMENT_QCOM

Additions to Section 5.3 after clGetImageInfo

An application that creates OpenCL image objects with the CL_MEM_EXT_HOST_PTR_QCOM flag can invoke the following function to query the required row pitch, slice pitch and alignment for a particular device:

```
cl_int clGetDeviceImageInfoQCOM(cl_device_id            device,
                                size_t                  image_width,
                                size_t                  image_height,
                                const cl_image_format   *image_format,
                                cl_image_pitch_info_qcom param_name,
                                size_t                  param_value_size,
                                void                    *param_value,
                                size_t                  *param_value_size_ret);
```

device - is a valid device

image_width - width of the image in image elements (pixels)

image_height - height of the image in image elements (pixels)

image_row_pitch - scan-line pitch in bytes

param_name - specifies the information to query. The list of supported param_name types and the information returned in param_value by clGetImageInfo is described in table 5.XXX

param_value - is a pointer to memory where the appropriate result being queried is returned. If param_value is NULL, it is ignored.

param_value_size - is used to specify the size in bytes of memory pointed to by param_value. This size must be >= size of return type as described in table 5.8.

param_value_size_ret - returns the actual size in bytes of data being queried by param_value. If param_value_size_ret is NULL, it is ignored.

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

CL_INVALID_VALUE - if param_name is not valid, or if size in bytes specified by param_value_size
is < size of return type for that param_value and param_value is not NULL.
CL_INVALID_MEM_OBJECT - if image is a not a valid image object.
CL_OUT_OF_RESOURCES - if there is a failure to allocate resources required by the OpenCL implementation on the device.
CL_OUT_OF_HOST_MEMORY - if there is a failure to allocate resources required by the OpenCL implementation on the host.

Table 5.XXX

List of supported param_names by clGetDeviceImageInfoQCOM

cl_image_pitch_info_qcom	Return Type	Info returned in param_value
CL_IMAGE_ROW_PITCH	cl_uint	Returns the image row pitch supported by this device
CL_IMAGE_ROW_ALIGNMENT_QCOM	cl_uint	Returns the image row pitch alignment supported by this device
CL_IMAGE_SLICE_PITCH	cl_uint	Returns the image slice pitch supported by this device
CL_IMAGE_SLICE_ALIGNMENT_QCOM	cl_uint	Returns the image slice pitch alignment supported by this device

Additions to Section 5.3.1. (Creating Image Objects) at the end
of the list of errors returned by clCreateImage2D and clCreateImage3D

CL_INVALID_VALUE if <flags> has CL_MEM_EXT_HOST_PTR_QCOM enabled and yet
CL_MEM_USE_HOST_PTR is not enabled.

CL_INVALID_VALUE if <flags> has CL_MEM_EXT_HOST_PTR_QCOM enabled and if
<image_row_pitch> and/or <image_slice_pitch> fail to match the requirements
of section 5.3

CL_INVALID_VALUE if any of the fields in the struct pointed at by <host_ptr> are invalid.

Issues

None.

Revision History

Revision 1, 2013/05/27: Initial version.