

OpenCL™ Language

OpenCL™ Tutorial, PPAM 2009

Dominik Behr | September 13th, 2009



OpenCL™ language is subset of ISO C99

With exceptions:

- No recursion, no function pointers, no standard headers
- No bitfields
- No extern, static, auto, register
- No writes to pointers or arrays of datatypes smaller than int



Address space

- `__global` – memory allocated from global address space, images are global by default
- `__constant` – is like global, but read only
- `__local` – memory shared by work-group
- `__private` – private per work-item memory
- `__read_only` – only for images
- `__write_only` – only for images

Kernel args have to be global, constant or local.

Can't assign to different pointer type.



Functions

- `__kernel` – only functions with `__kernel` attribute can be “called” as kernels from OpenCL™ API
- `__attribute__((work_group_size_hint(X, Y, Z)))` – hint for workgroup size if not specified in `EnqueueNDRangeKernel`
- `__attribute__((reqd_work_group_size(X, Y, Z)))` – required workgroup size
- `__attribute__((vec_type_hint(<typen>)))` - autovectorize



Workgroups

- `uint get_work_dim () (1 to 3)`
- `size_t get_global_size (uint dimindx)`
- `size_t get_global_id (uint dimindx)`
- `size_t get_local_size (uint dimindx)`
- `size_t get_local_id (uint dimindx)`
- `size_t get_num_groups (uint dimindx)`
- `size_t get_group_id (uint dimindx)`

$\text{num_groups} * \text{local_size} = \text{global_size}$

$\text{local_id} + \text{group_id} * \text{local_size} = \text{global_id}$

$\text{global_size} \% \text{local_size} = 0$



Data Types

- bool
- char (**cl_char**)
- unsigned char, uchar (**cl_uchar**)
- short (**cl_short**)
- unsigned short, ushort (**cl_ushort**)
- int (**cl_int**)
- unsigned int, uint (**cl_uint**)
- long (64-bit) (**cl_long**)
- unsigned long, ulong (64-bit) (**cl_ulong**)
- float (**cl_float**)
- half (**cl_half**)
- size_t (CL_DEVICE_ADDRESS_BITS)
- ptrdiff_t (CL_DEVICE_ADDRESS_BITS)
- intptr_t
- uintptr_t



Vector types

- Vector size $N = 2, 4, 8, 16$
- `charN` – `cl_charN`
- `ucharN` – `cl_ucharN`
- `shortN` – `cl_ushortN`
- `ushortN` – `cl_ushortN`
- `intN` – `cl_intN`
- `uintN` – `cl_uintN`
- `longN` – `cl_longN`
- `ulongN` – `cl_ulongN`
- `halfN` – `cl_halfN`
- `floatN` – `cl_floatN`

Access using `.xyzw` for `vec2` and `vec4`, can swizzle, like `"a.xy"`

Or `s0` to `sF`
(`0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F`, not case sensitive)
like `"a.s01"`

Or `.lo` (`.odd`) and `.hi` (`.even`), can chain them like `a.lo.lo`



Synchronization

barrier() function. All work-items must reach the barrier before they execute further. It must be encountered by all work-items in work-group.

Flags: LOCAL_MEM_FENCE, GLOBAL_MEM_FENCE – flush and ensure ordering for local or global memory.

mem_fence(), **read_mem_fence()**, **write_mem_fence()** – ensure memory loads and stores ordering within work-item.



Other

OpenCL™ language specifies much more useful built in functions, type conversions, (no) exception handling, rounding, precision of arithmetic operations and math functions.

Also there is whole set of image operations with different formats, filtering etc.

Read the spec.



Trademark Attribution

AMD, the AMD Arrow logo and combinations thereof are trademarks of Advanced Micro Devices, Inc. in the United States and/or other jurisdictions. OpenCL is trademark of Apple Inc. used under license to the Khronos Group Inc. Other names used in this presentation are for identification purposes only and may be trademarks of their respective owners.

©2009 Advanced Micro Devices, Inc. All rights reserved.

