1

OpenCL: Questions about global memory reads, using host pointer buffers, and private memory

(self.gpgpu)

OG-Mudbone 於 1 年前 * 發表

I am trying to determine the read/write speed between processing elements and global memory on an Adreno330. I'm launching a single work item that does 1,000,000 float reads in kernel A and 1,000,000 float write in kernel B. (Therefore 4MB each way).

HOST

```
// Create arrays on host (CPU/GPU unified memory)
int size = 1000000;
float *writeArray = new float[size];
float *readArray = new float[size];
for (int i = 0; i<size; ++i){
   readArray[i] = i;
   writeArray[i] = i;
}

// Initial value = 0.0
LOGD("Before read : %f", *readArray);
LOGD("Before write : %f", *writeArray);

// Create device buffer;
cl_mem readBuffer = clCreateBuffer(
      openCLObjects.context,
      CL_MEM_READ_WRITE | CL_MEM_USE_HOST_PTR,
      size * sizeof(cl_float),
      readArray,
      &err );
cl_mem writeBuffer = clCreateBuffer(
```

```
        openCLObjects.context,
        CL_MEM_READ_WRITE | CL_MEM_USE_HOST_PTR,
        size * sizeof(cl_float),
        writeArray,
        &err );

//Set kernel arguments
size_t globalSize[3] = {1,1,1};
err = clSetKernelArg(openCLObjects.readTest, 0, sizeof(cl_mem), &readBuffer);
err = clSetKernelArg(openCLObjects.writeTest, 0, sizeof(cl_mem), &writeBuffer);

// Launch kernels
err = clEnqueueNDRangeKernel(openCLObjects.queue, openCLObjects.readTest, 3, NULL, globalSize, NUL
clFinish(openCLObjects.queue);
err = clEnqueueNDRangeKernel(openCLObjects.queue, openCLObjects.writeTest, 3, NULL, globalSize, NUI
clFinish(openCLObjects.queue);

// Expected result = 7.11
clReleaseMemObject(readBuffer);
clReleaseMemObject(writeBuffer);
LOGD("After read: %f", *readArray); // After read: 0.0 (??)
LOGD("After write: %f", *writeArray);
```

KERNELS

```
kernel void readTest(__global float* array)
{
   float privateArray[1000000];
   for (int i = 0; i < 1000000; ++i)
   {
      privateArray[i] = array[i];
   }
}

kernel void writeTest(__global float* array)
```

2 · 2 留言

Profiling OpenCL on nvidia cards?

```
{
    for (int i = 0; i < 1000000; ++i){
        array[i] = 7.11;
    }
}
```

Results via AdrenoProfiler: readTest: Global loads: 0 bytes Global stores: 0 bytes Runtime: 0.010 ms

writeTest: Global loads: 0 bytes Global stores: 4000000 bytes Runtime: 65 ms

My questions:

1. Why doesn't readTest do any memory loads? If I change it to array[i] = array[i]+1 then it does 4m reads and 4m writes (120ms) which makes sense. If memory is loaded but never nothing is written back, does the compiler skip it?
2. Why does am I not reading the updated values of the arrays after the process completes? If I call enqueuMapBuffer just before printing the results, I see the correct values. I understand why this would be necessary for pinned memory but I thought the purpose of CL_MEM_USE_HOST_PTR was that the work items are modifying actual arrays allocated on the host.
3. To my understanding, if I were to declare a private variable within the kernel, it will be stored in private memory (registers?) There are no available specs and I have not been able to find a way to measure the amount of private memory available to a processing element. Any suggestions on how? I'm sure 4mb is much too large, so what is happening with the memory in the readTest kernel. Is privateArray just being stored on the global mem (unified DRAM?) Are private values stored on the local if they don't fit in registers, and global if they don't fit in local? (8kb local in my case.) I can't seem to find an thorough explanation for private memory.

Sorry for the lengthy post, I really appreciate any information anyone could provide.

5 留言　分享

所有 5 則留言

排序依據: **最佳**

[–] **useless_panda**　2 指標 1 年前

Qualcomm will optimize away any memory read when they detect the results are not used. So that is probably why your read kernel did nothing. This is a good thing.

As for your writing issue... according to the Adreno SDK's OpenCL Programming Guide (80-N8592-1 L, section 3.3.2.3 "Critical memory optimizations to consider"), I think the only methods to perform zero-copy is to use CL_MEM_ALLOC_HOST_PTR, the Ion memory

extension, QTI Android native buffer extension, or the EGL image path.

When your kernel uses more private memory than available, the memory will be allocated in other memory region with more room. In the worst case scenario your private memory will be stored in the global memory as you guessed, so all your private memory r/w become global memory r/w. Sometimes they might be able to squeeze it in the cache or local memory. Anyways I just rely on their OpenCL scrubber to watch for my private memory going on the global memory space.

There are some stuff you must be aware of when dealing with global R/W (on Adreno GPUs). I was gonna write a bunch of stuff but that would just be repeating exactly what's in section 3.3.3.1 "Memory load/store" and section 3.3.3.2 "Vectorization" from the programmer's guide.

Note: It is possible that using image mem obj can be much faster for your memory reads. Image objs are accessed via the hardware texture pipeline via the L1 cache (section 3.3.2.2 "Memory objects" from programmer guide).

永久連結 **embed**

載入更多留言 (4 回覆)

More from r/gpgpu

HIDE

2 Profiling OpenCL on nvidia cards? (self.gpgpu)

kwhali 於 9 天前 發表

2 留言 **分享**

π