Memory Management and error handling

MEMORY_MANAGEMENT Folder

Memory allocation and set

- use cudaMalloc(&pointer, memSize)
  - &pointer is a *pointer to pointer* to alloc heap memory on device
- cudaMemSet(pointer, value, memSize)

Copy between CPU and GPU

- with cudaMemcpy(destination_pointer, source_pointer, memSize, direction_of_the_copy)
  - direction can be cudaMemcpyHostToDevice or cudaMemcpyDeviceToHost
- Use is even not necessary since 6.0

Copy between device & device

- with cudaMemcpy(destination_pointer, source_pointer, memSize, direction_of_the_copy)
  - direction : cudaMemCpyDeviceToDevice
  - Much faster

Free memory

- with cudaFree(pointer)
- or cudaFreeHost(pointer) with pointer allocated with cudaHostAlloc

Unified Memory Adressing (UVA)

- No need to specify where the ressource is
- Needs :
  - 64 bits
  - Fermi
  - >= CUDA 4.0
- Example : cudaMemcpy(dst, src, memsize, cudaMemcpyDefault)
- Check if capable with
  - cudaDeviceProp prop;
  - cudaGetDeviceProperties(&prop);
  - printf(«Addressing capable : %d\n», prop.unifiedAddressing);

Page-locked memory

http://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html#page-locked-host-memory

- cudaHostAlloc(&pointer, memsize, flag) : allows « pinned » memory
- cudaFreeHost
- Flags :
  - cudaHostAllocWriteCombined : faster transfers.
  - cudaHostAllocMapped : transparent read & write (no need of cudaMemcpy)
- Illustrated in MEMORY_MANAGEMENT/pinMemTransfer

Asynchronous transfers / executions

http://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html#asynchronous-concurrent-execution

- using cudaMemcpyAsync (cudaMemsetAsync…)
- Only on devices that support it
- Only form / to Page-Locked host memory

Error checking

http://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html#error-checking

- Similar to parallel libraries :
  - ✤ Functions return a code (unsigned int)
  - ✤ Corresponds to an error string from a table
- Idea : wrap functions calls with macro checking the code
- Everything in /usr/local/cuda/samples/common/inc/helper_cuda.h and wrappers in formation folder COMMON/commons.cuh

Cases

- Try a simple sequence
  - ✤ memory allocation
  - ✤ copy between host and device
  - ✤ back to host
  - ✤ free memory
- Try some errors : allocating too much memory, wrong direction of copy… what happens ?