Built-in Variables:

```
dim3  gridDim;   // Total number Thread Blocks
uint3 blockIdx;  // index of a block in a grid
dim3  blockDim;  // Number threads per Thread Block
uint3 threadIdx; // index of a thread in a block (a "local id")
int   warpSize;  // generally 32
```

CUDA Thread Geometry

row : 2 col : 3 sheet : 1

blockIdx.y = 2
blockIdx.x = 3
blockIdx.z = 1

blockIdx = (2, 3, 1)

threadIdx = (1, 2, 0)

threadIdx.y = 1
threadIdx.x = 2
threadIdx.z = 0

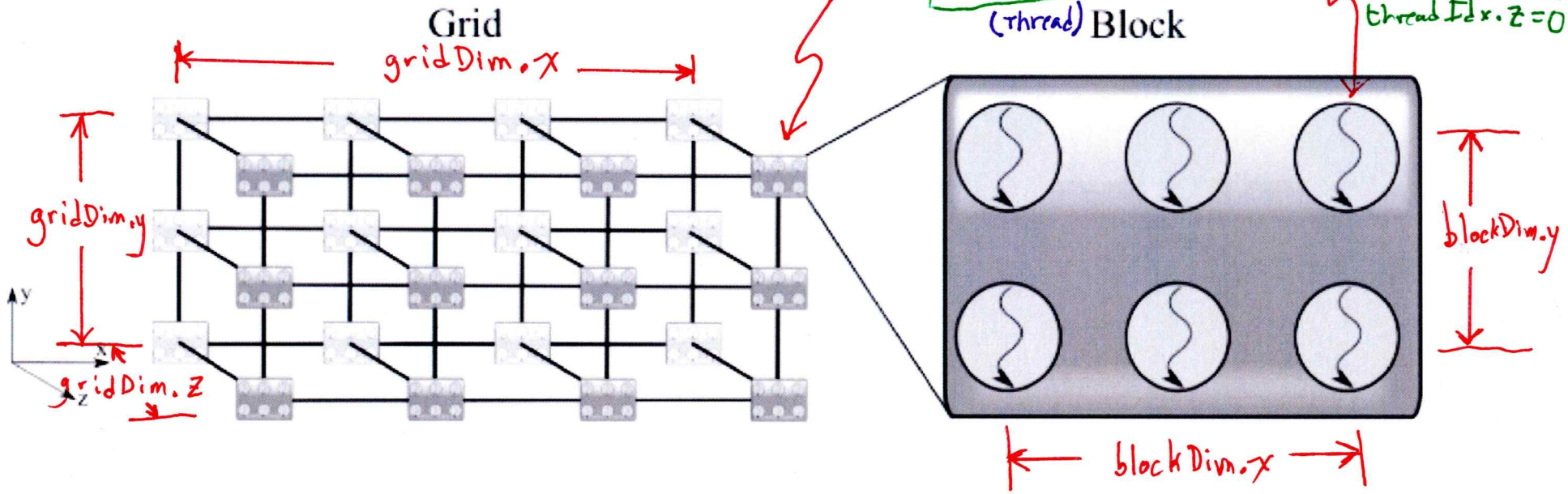

Figure 6.3: An example of the grid/block hierarchy used to describe a set of threads that will be spawned by CUDA. The figure illustrates a 4x3x2 grid made by 3x2 blocks. The grid connections are there only for illustrative purposes.

OpenCL Thread Geometry

Built-in functions:

```
uint   get_work_dim();
size_t get_num_groups(uint dim);// gridDim.c
size_t get_group_id (uint dim); // blockIdx.c
size_t get_local_size(uint dim); // blockDim.c
size_t get_local_id (uint dim); // threadIdx.c
```

```
size_t get_global_size(uint dim);
    // gridDim.c * blockDim.c
size_t get_global_id (uint dim);
    // blockIdx.c * blockDim.c +
                    threadIdx.c
```

c = { dim=0 ⇒ y (row); dim=1 ⇒ x (col); dim=2 ⇒ z (sheet) }

~~Grid~~ NDRange* or index range

~~(Thread) Block~~ Work Group

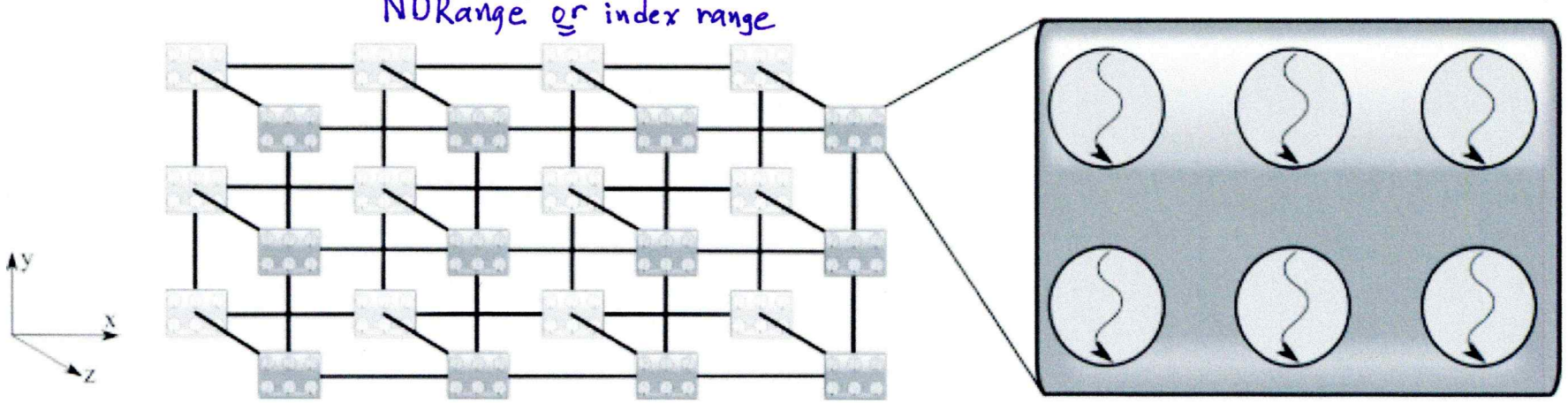

Figure 6.3: An example of the grid/block hierarchy used to describe a set of threads that will be spawned by CUDA. The figure illustrates a 4x3x2 grid made by 3x2 blocks. The grid connections are there only for illustrative purposes.

KernelFunc<<<...>>>(...); ⇒ clEnqueueNDRangeKernel(...)*