**Memory management** is one of the most important tasks in any program. Below is a brief explanation of the various ways OpenCL connects the device and host memory.

**Memory mapping between ARM and FPGA:**

Global memory – It is the main means of communicating reads and writes of data between host and device. Their content visible to all threads but it has long access latency.

**OpenCL Device Memory Allocation:**

clCreateBuffer() – It allocates objects in the device Global Memory. It returns a pointer to the object and requires five parameters: OpenCL context pointer, Flags for access type by device, Size of allocated object, Host memory pointer, if used in copy-from-host mode, & Error variable.

**OpenCL Host-to-Device Data Transfer:**

clEnqueueWriteBuffer() - It does memory data transfer to device. It requires nine parameters: OpenCL command queue pointer, Destination OpenCL memory buffer, Blocking flag, Offset in bytes, Sizeof bytes of written data, Host memory pointer, List of events to be completed before execution of this command, & Event object tied to this command.

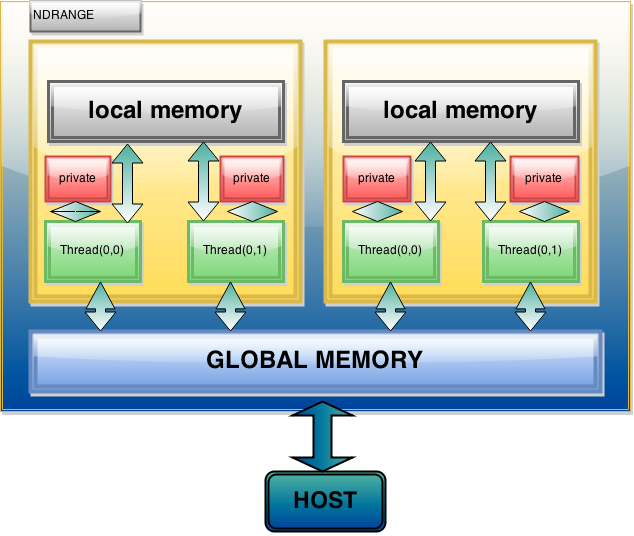
**OpenCL Device-to-Host Data Transfer:**

clEnqueueReadBuffer() – It does memory data transfer to host. It requires nine parameters: OpenCL command queue pointer, Source OpenCL memory buffer, Blocking flag, Offset in bytes, Sizeof bytes of read data, Destination host memory pointer, List of events to be completed before execution of this command, & Event object tied to this command.

* A Kernel is invoked once for each **work item**. Each work item has **private** memory.
* Work items are grouped into a **work group**. Each work group shares **local** memory
* The total number of all work items is specified by the **global work size**. **global** and **constants** memory is shared across all work work items of all work groups

**OpenCL** Memory Systems-

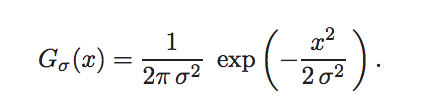
* \_\_global – large, long latency.
* \_\_private – on-chip device registers
* \_\_local – memory accessible from multiple PEs or work items. It can  be SRAM or DRAM, must query.
* \_\_constant – read-only constant cache.



**Filtering:**

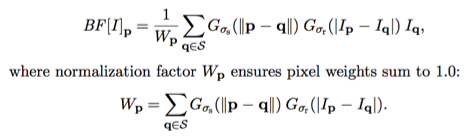
The idea of spatial image filtering is to have a mask or kernel of a certain size that applies a desired operation to the image pixels under the kernel. By moving the kernel around the image, so that all pixels are visited by the center of the kernel, the image is filtered. The operation applied to the underlying image may be either linear or nonlinear. Gaussian filter is a linear filter while bilateral filter is a nonlinear filter. This implies that the filter kernel will be a matrix of coefficients that will be multiplied with the corresponding underlying image pixels. The sum of all products at each location will yield the pixel values of the filtered image.

*Linear Filtering with Gaussian Blur (GB):* Convolution by a positive kernel is the basic operation in linear image filtering. It amounts to estimating at each position a local average of intensities and corresponds to low-pass filtering.



So, *Gaussian filtering* is a weighted average of the intensity of the adjacent positions with a weight decreasing with the spatial distance to the center position p. This distance is defined by Gσ(||p − q||), where σ is a parameter defining the extension of the neighborhood. As a result, image edges are blurred.

*Nonlinear Filtering with Bilateral Filter (BF):*  Similarly to the Gaussian convolution, the bilateral filter is also defined as a weighted average of pixels. The difference is that the bilateral filter takes into account the variation of intensities to preserve edges. The rationale of bilateral filtering is that two pixels are close to each other not only if they occupy nearby spatial locations but also if they have some similarity in the photometric range. The bilateral filter, denoted by BF[\*], is defined by



Bilateral filtering is non-iterative, nonlinear and combines domain (spatial closeness) and range (color similarity) filtering. The coefficients of the kernel are computed locally and depend on both variables, thus achieving a good filtering behavior where the pixels’ similarity is high e.g. inside regions. It also preserves areas where similarity is low, e.g. on edges. The bilateral filtering is less content dependent and provides exceptional results with a fixed set of parameters, thus giving highest quality at the lowest computational cost compared to other filters of kind e.g anisotropic diffusion filtering.

Parameters σs and σr will control the amount of filtering for the image. The equation is a normalized weighted average where Gσs is a spatial Gaussian that decreases the influence of distant pixels. Gσr is a range Gaussian that decreases the influence of pixels q with an intensity value different from p. Note that the term range addresses quantities related to pixel values, unlike the term spatial which refers to pixel location.

*Parameters*- The bilateral filter is controlled by two parameters: σs and σr.

• As the range parameter σr increases, the bilateral filter becomes closer to Gaussian blur because the range Gaussian is flatter i.e., almost a constant over the intensity interval covered by the image.

• Increasing the spatial parameter σs smooths larger features. An important characteristic of bilateral filtering is that the weights are multiplied, which implies that as soon as one of the weight is close to 0, no smoothing occurs. As an example, a large spatial Gaussian coupled with narrow range Gaussian achieves a limited smoothing although the filter has large spatial extent. The range weight enforces a strict preservation of the contours.