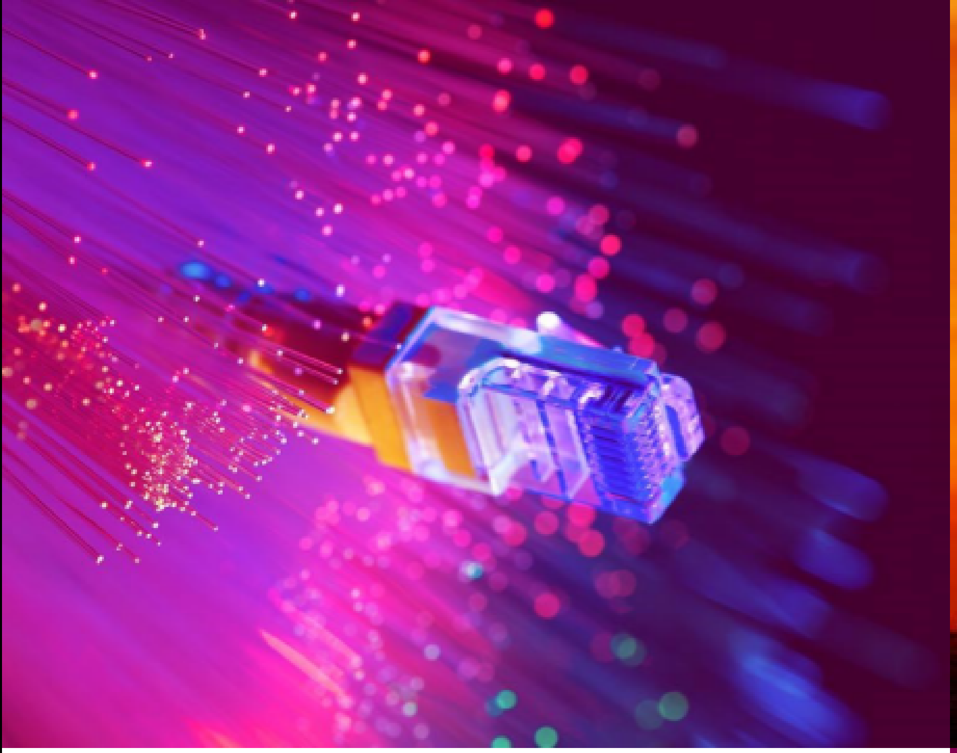
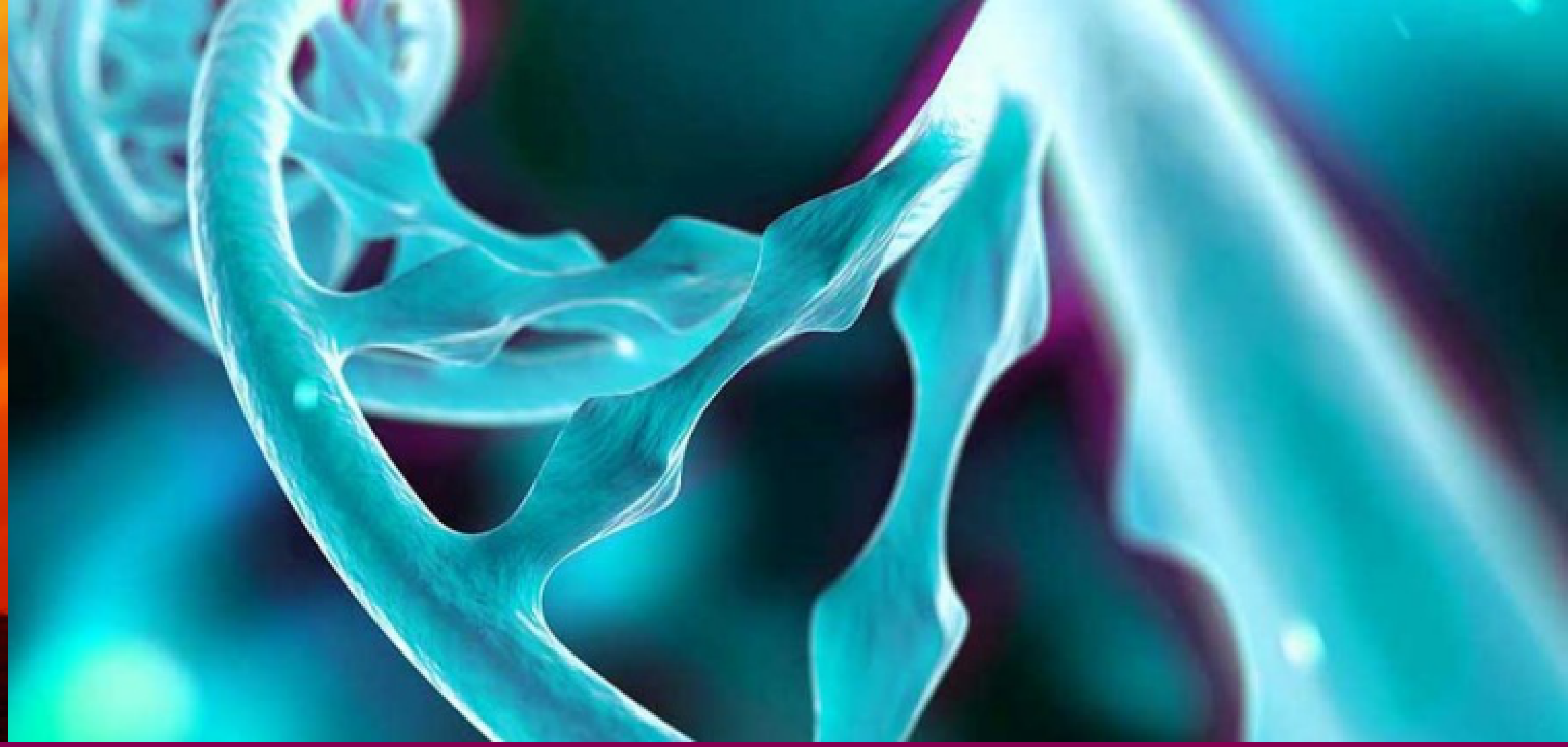




Memory Hierarchy and Memory Accesses in GPUs

High-Level GPU Programming

2024-02

CSC Training

CSC – Finnish expertise in ICT for research, education and public administration

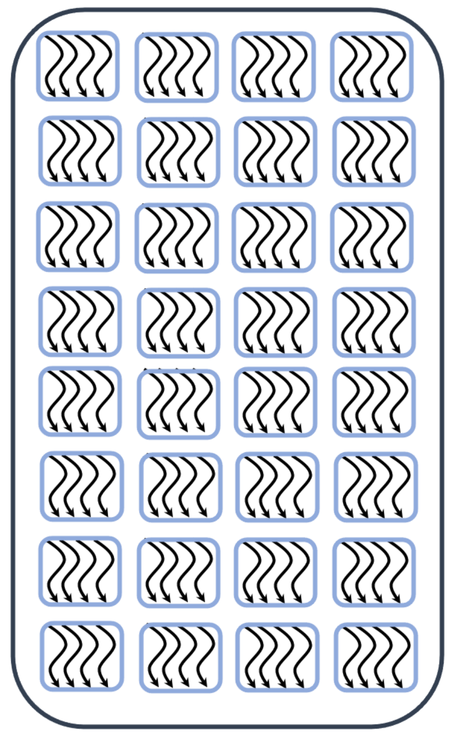
Memory Hierarchy and Memory Accesses in GPUs

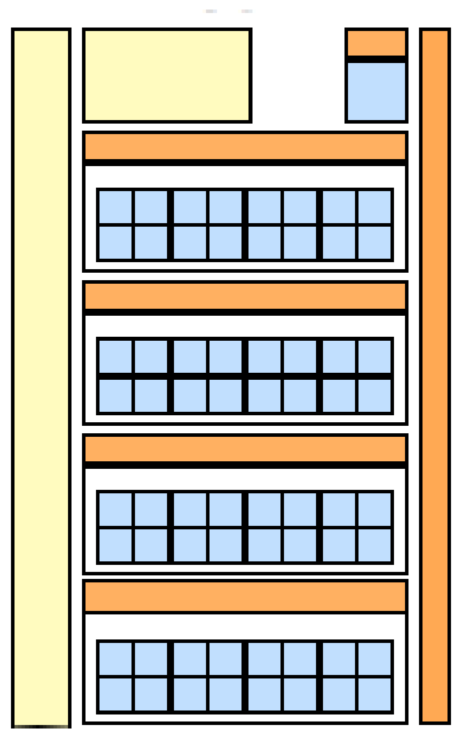
Registers

- the fastest form of memory.
- private to each thread.
- local variables and intermidiate results, `double x=sin(id);`
- no cost at reading, write & read is costlier (eg, 24-cycle latency).
- lifetime of the kernel.

Each work-item has its own registers.

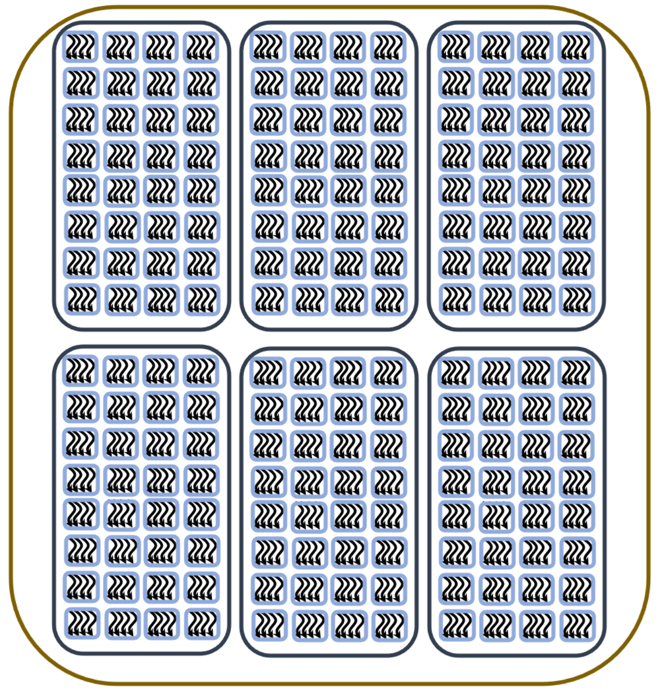
Local Shared Memory


Local shared memory accessible at work-group level.

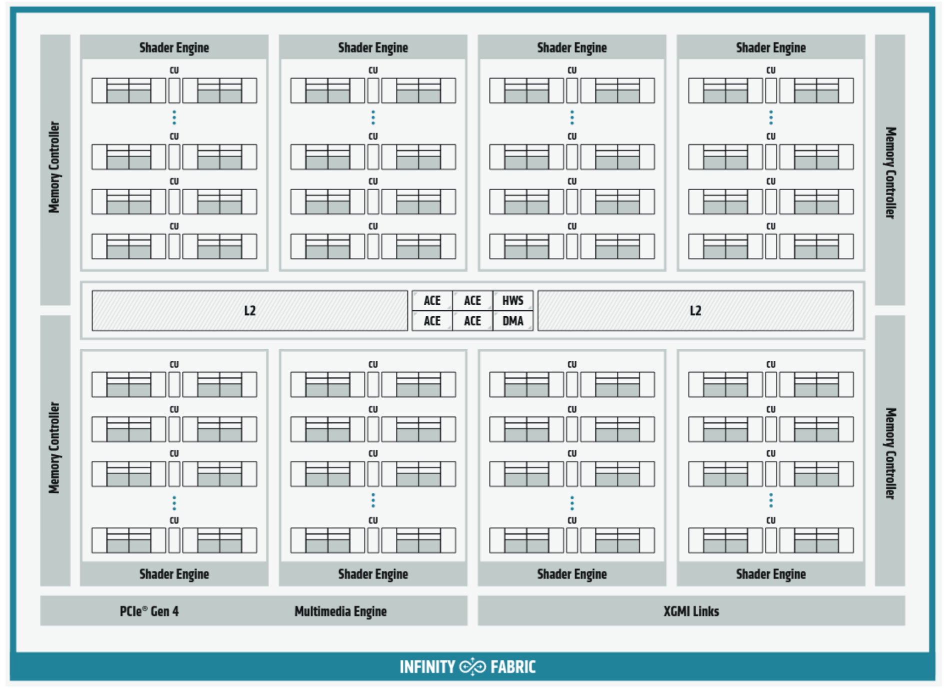

Compute Unit in an AMD GPU.

- very fast memory, latency of eg. 6 cycles.
- accessible by all work-items in a group.
- user programmable cache.
- lifetime of the work-group.

Global Memory



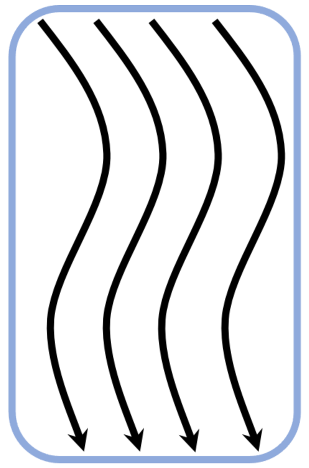
Global memory can be accessed by the whole grid.

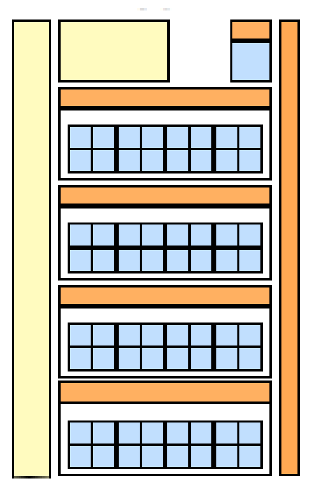


AMD Instinct MI100 architecture. (source: AMD)

- slow, latency of eg. 600-700 cycles.
- accessible by all work-items in a grid.
- can be controlled by host (via pointer operations).
- lifetime of the program.

Coalesced Memory Access


Memory access is done per sub-group.


Compute Unit in an AMD GPU.

- the work-items are physically locked in sub-groups (typically 64 or 32).
- an instruction is executed by all items in the sub-group (eg, 4 cycles).
- memory accesses are done per sub-group in contiguous blocks.
- threads in a sub-group should operate on elements close to each other.
- local shared memory can be used to improve the memory accesses.

Coalesced vs Non-coalesced Memory Access

Serial cpu code of `y=y+a*x`:

- we create instances of the same function, **kernels**, `id`

```
GPU_K void axpy_(int n, double a, double *x, double *y, int id)
{
        y[id] += a * x[id]; // Coalesced
        y[id] += a * x[id*stride]; // Strided Non-Coalesced
        y[id] += a * x[id+shift]; // Shifted Non-Coalesced
}
```

Summary

- various types of memory in GPUs.
- **registers**: the fastest, variables are local to threads, lifetime of a kernel.
- **local share**: very fast, visible by the whole work-group, lifetime of the work-group.
- **global**: slowest, visible by all threads, managed by host, lifetime of the program.
- memory accesses are done per *sub-group* in contiguous blocks of specific sizes.
- **coalesced access**: when threads in a sub-group access memory in the same block of memory.
- local share memory can be use a programmable cache to avoid some global memory operations or improve the access pattern.

NOTE! When all registers are used, there is spilling of the local variables into the global memory.