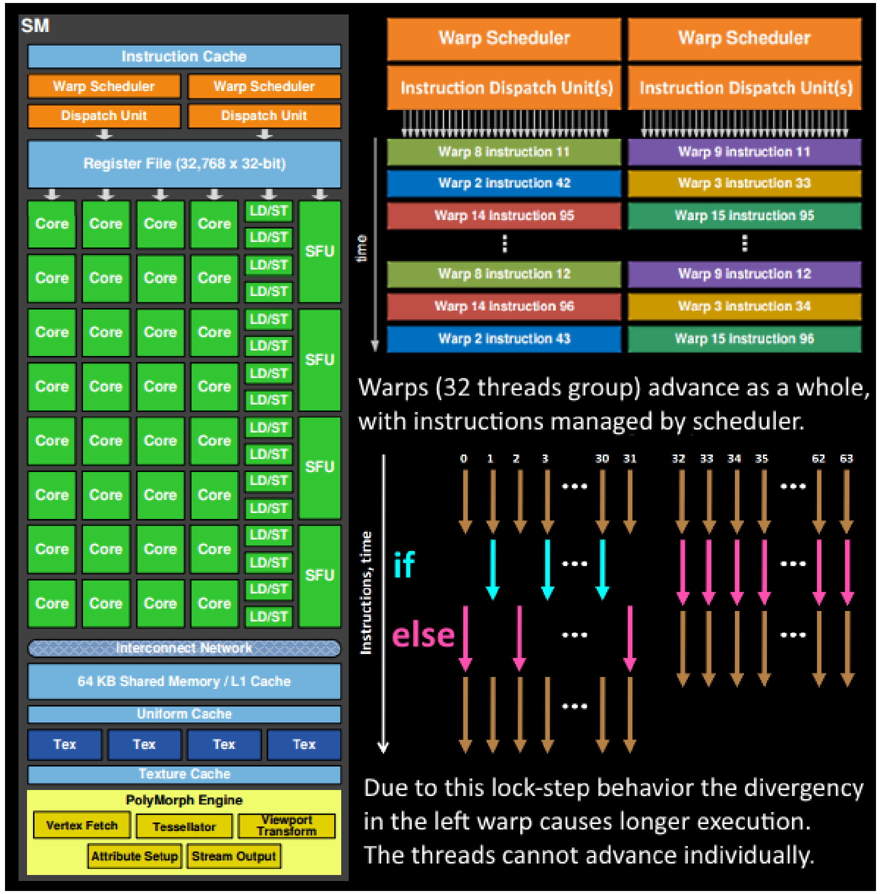
Branching and GPU Efficiency

Many-Core, but not independent

- GPUs are many core, but the structure of memory in the GPU and how threads are organized to access the memory is strictly tied together.
- Threads are bundled in “warps” which have access to the same register and have a bundled set of instructions



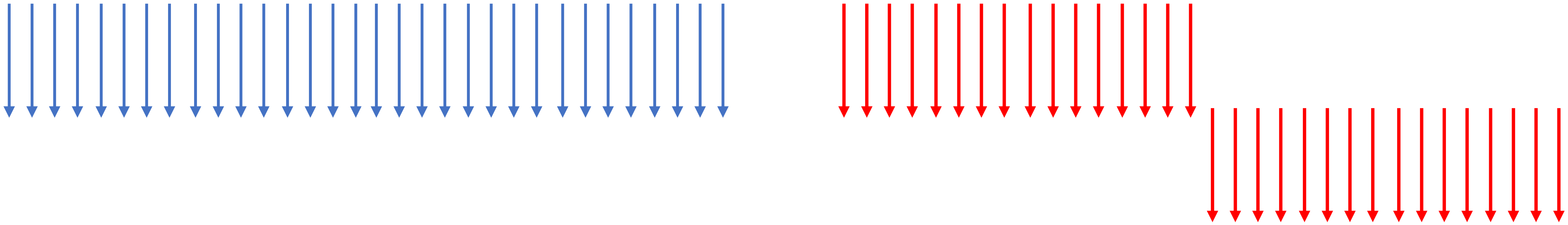
Warps (32 threads group) advance as a whole, with instructions managed by scheduler.

Due to this lock-step behavior the divergency in the left warp causes longer execution. The threads cannot advance individually.

Image From: https://developer.nvidia.com/content/life-triangle-nvidias-logical-pipeline

Branching can reduce efficiency

- Any time one of the (typically) 32 threads in a warp does something different from the others, all of the others have to wait for the thread to complete it’s branched activity



What branching looks like in your kernel

```
__global__ void branching(int n)
{

 int threadMod = threadIdx.x%threadsPerBlock;

     if ( threadMod < n) {
         ...
     } else if(threadMod<2*n){
         ...
     } else if(threadMod<3*n){
         ...
     } else if(threadMod<4*n){
         ...
     }
}
```