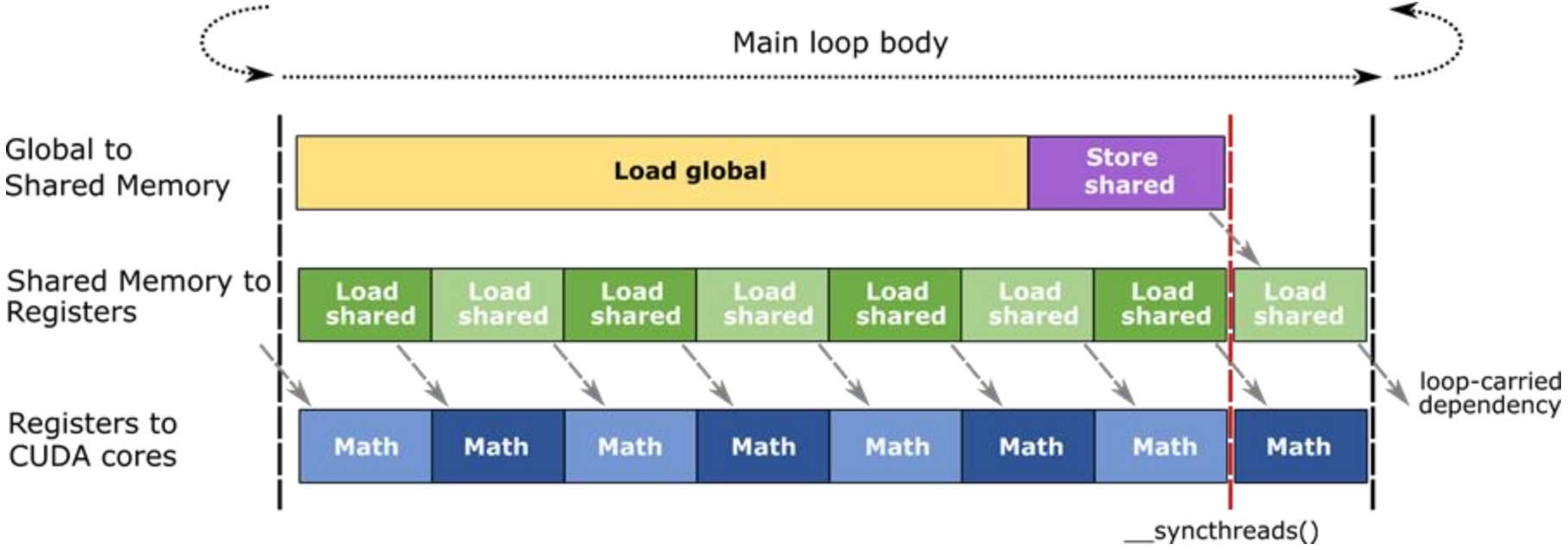
Software Pipelining in H100

Ganesh Bikshandi

Software Pipelining



Software Pipelining in CUDA : H100

- Supported hardware accelerated barrier
  - Mbarrier (from H100 onwards)
- Warp specialization
  - Producer-consumer style
    - One warp is producer
    - Other warp(s) is consumer
  - Better register allocation
    - Producer warps can release unused registers to consumer warps.
- Clusters
  - Grouping of more than one CTA (Thread blocks)

Software Pipelining in CUDA : Producer Warp Groups

```
if (warp_idx_in_warpgroup == 0 && lane_predicate) {
    int tma_k_prologue = min(Stages, kernel_params.num_iterations);
    PipelineState smem_pipe_write = make_producer_start_state<MainloopPipeline>();
    CUTLASS_PRAGMA_UNROLL
    for(int i = 0; i < tma_k_iterations ++i) {
      pipeline.producer_acquire(smem_pipe_write);
      // Perform TMA Loads using CP.ASYNC.BULK
      ++smem_pipe_write;
    }
}
```

Software Pipelining in CUDA : Consumer Warp Groups

```
else if(warp_group_idx == 1) {
    PipelineState smem_pipe_read;
    PipelineState smem_pipe_release;
    for ( ; gemm_k_iterations > 0; --gemm_k_iterations) {
      pipeline.consumer_wait(smem_pipe_read);
      warpgroup_arrive();
      // Perform GMMA using WGMMA
    }
      pipeline.consumer_release(smem_pipe_release);
      ++smem_pipe_read;
      ++smem_pipe_release;
  }
```

Conclusion

- High performing kernels need to be re-structured to follow *producer-consumer* style.
  - **Must-have feature.**
  - Loads in Producer
  - Math (GMMA) in Consumer
- Increases Shared memory pressure
  - Eases register pressure