


Reduce

CUDA

1. Divergence Block

Parallel Reduction: Sequential Addressing



Values (shared memory) 10 1 8 -1 0 -2 3 5 -2 -3 2 7 0 11 0 2

Step 1
Stride 8

Thread IDs 0 1 2 3 4 5 6 7

Values 8 -2 10 6 0 9 3 7 -2 -3 2 7 0 11 0 2

Step 2
Stride 4

Thread IDs 0 1 2 3

Values 8 7 13 13 0 9 3 7 -2 -3 2 7 0 11 0 2

Step 3
Stride 2

Thread IDs 0 1

Values 21 20 13 13 0 9 3 7 -2 -3 2 7 0 11 0 2

Step 4
Stride 1


Thread IDs 0

Values 41 20 13 13 0 9 3 7 -2 -3 2 7 0 11 0 2

Sequential addressing is conflict free

14

Reduction #3: Sequential Addressing



Just replace strided indexing in inner loop:

```
for (unsigned int s=1; s < blockDim.x; s *= 2) {  
    int index = 2 * s * tid;  
  
    if (index < blockDim.x) {  
        sdata[index] += sdata[index + s];  
    }  
    __syncthreads();  
}
```

With reversed loop and threadID-based indexing:

```
for (unsigned int s=blockDim.x/2; s>0; s>>=1) {  
    if (tid < s) {  
        sdata[tid] += sdata[tid + s];  
    }  
    __syncthreads();  
}
```

15

2. <https://developer.download.nvidia.com/assets/cuda/files/reduction.pdf>

DSA/ASIC

