King Saud University
College of Computer and Information Sciences
Department of Computer Science
CSC453 – Parallel Processing – Tutorial No 8 – Autumn 2022

Question

Let's consider the following code of a parallel reduction that calculates the sum of an array of integers.

```
__global__ void reduce(int *g_idata, int *g_odata) {
extern __shared__ int sdata[];
// each thread loads one element from global to shared mem
unsigned int tid = threadIdx.x;
unsigned int i = blockIdx.x*blockDim.x + threadIdx.x;
sdata[tid] = g_idata[i];
__syncthreads();
// do reduction in shared mem
        for(unsigned int stride=1; stride < blockDim.x; stride *= 2) {
                if (tid % (2* stride) == 0) {
                        sdata[tid] += sdata[tid + stride];
                }
                __syncthreads();
        }
        // write result for this block to global mem
        if (tid == 0)
                g_odata[blockIdx.x] = sdata[0];
}
```

1. What is reduction. reducing the set of input elements to a single value output element.
2. Why the threads are doing the reduction in shared memory. to accelerate the access to memory since shared memory is on-chip.
3. This code is suffering from what is called interleaved addressing.
    a. What is interleaved addressing. Show it with a figure. threads are not sequentially accessing sequential addresses.
    b. Why warps in this code are not efficient. because the half of the warp is used, and on every iteration the number of threads involved is reduced to half.
    c. How can we fix that?
    by unrolling the last warp.