OpenCL exercise 6: Prefix sum

Kaicong Sun

Prefix sum

- Prefix sum = all prefix sums for an input vector
- For input values $x_0, x_1, x_2, ...$ compute:

$$\begin{aligned} y_0 &= x_0 \\ y_1 &= x_0 + x_1 \\ y_2 &= x_0 + x_1 + x_2 \end{aligned}$$

- Also can be some other associative binary operation instead of $+$, e.g. min, max, ...

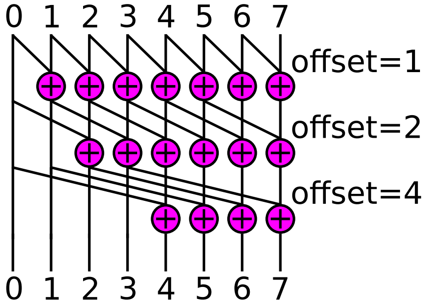
Prefix sum

Host code:

```
cl_int sum = h_input[0];
h_output[0] = sum;
for (std::size_t i = 1; i < h_input.size (); i++) {
    sum += h_input[i];
    h_output[i] = sum;
}
```

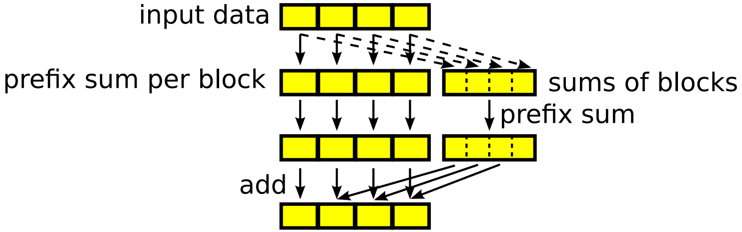
Parallel prefix sum

- Parallel prefix sum:



GPU

- Problem: Can use only one work group
- Solution: Work with blocks



- Launch Kernel1: Do prefix sum per block and write the sum of each block to a temp array until there is only one work group. (For details see another .PDF file)
- Launch Kernel2: For all blocks except the first: Add `temp2[blockIndex-1]` to all values in the current `temp1` block and then recursively add `temp1[blockIndex-1]` to all values in the current `d_output`.

GPU

- Task: Implement prefix sum on GPU
  - Plus usual code for performance measurements
- Kernel1 should calculate prefix sum blockwise:
  - Load input data to local memory
  - Loop over offsets
  - Write results to global memory
  - Use one work item per value
  - Do not forget to add `barrier` calls for synchronization
- Kernel2 should add the sum of previous blocks to each element of the current block.

Hints on Device

```
//To locate your work group:
uint grpid = get_group_id(0);

//Can not use:
if (li >= offset)
ldata[li] += ldata[li - offset];
barrier(CLK_LOCAL_MEM_FENCE);
```

Why?
Solution? ⇒Instead of updating ldata[li] directly, using temporary variable and barrier(CLK_LOCAL_MEM_FENCE).