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, ...

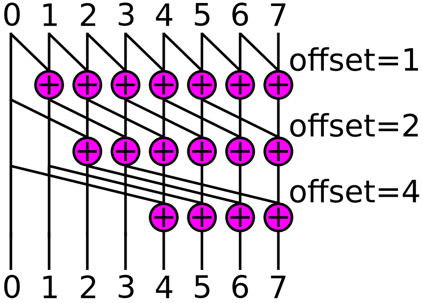
Prefix sum

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:

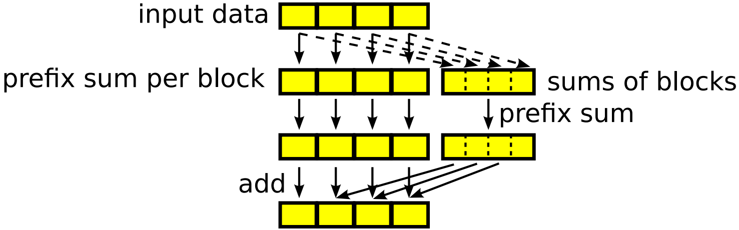


GPU

- Task: Implement prefix sum on GPU
    - Plus usual code for performance measurements
- Kernel should:
    - 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

GPU

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



- Do prefix sum per block
  - Also write sum of block to another array, `temp1`
- Do prefix sum for `temp1` (recursivly, using `temp2` as temp array)
- 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`.(write second kernel for this step)