Lesson 26: Computing π – Part IV

Recap

- Introduced parameter passing
- Introduced mathematical functions
- Introduced memory management
- Now: compute π
Lesson 26: Computing \( \pi \) – Part IV

**Approach**

- We will use the formula for the area of a circle quarter:
  \[
  A = \frac{\pi \cdot r^2}{4}
  \]

- The number of points inside the circle \( (P) \) can be used to approximate \( A \):
  \[
  \frac{P}{n} \approx \frac{A}{r^2} = \frac{\pi}{4} \rightarrow \pi \approx \frac{4P}{n}
  \]

- The `PixelFinderKernel` does the counting on the Device, integration is done by the Host.
Kernel execution and memory transfer

- We will measure the execution time:
  ```cpp
  auto start = std::chrono::steady_clock::now();
  ```

- Execute the kernel using `alpaka::kernel::exec()`:
  ```cpp
  PixelFinderKernel pixelFinderKernel;
  auto taskRunKernel = kernel::createTaskKernel<Acc>(workDiv, pixelFinderKernel, pointsAcc, r);
  queue::enqueue(queue, taskRunKernel);
  ```

- Copy back the results and synchronize:
  ```cpp
  mem::view::copy(devQueue, insideBufferHost, insideBufferAcc, extents);
  alpaka::wait::wait(queue);
  ```
Integration

- First, determine $P$:

```cpp
uint64_t P = 0;
for (std::size_t i = 0; i < n; ++i)
{
    if (pointsHost.inside[i])
        ++P;
}
```

- Then, divide by the radius to approximate $\pi$:

```cpp
float pi = (4.f * P) / n;
```

- Measure the execution time:

```cpp
auto end = std::chrono::steady_clock::now();
```
Lesson 26: Computing π – Part IV

Aftermath

- Print out π and execution time:

  ```cpp
  std::chrono::duration<double, std::milli> duration = end - start;
  std::cout << "Computed pi is " << pi << "\n";
  std::cout << "Execution time: " << duration.count() << "ms" << std::endl;
  ```

- Homework #1: Play around with \( n \). How does this affect the precision of \( \pi \) and the execution time?

- Homework #2: Implement the kernel in a more generic way, so that it works for any number of threads, blocks and grids.
  - The workload has to be distributed between all threads in the grid.
  - It requires to have a loop over points inside the kernel. A sample is given in a Q&A answer from Tuesday.