Lesson 21: 1D Work Division
Lesson 21: 1D Work Division

Printing the Block index

• Open helloWorld in your favourite editor
• Change the Kernel as shown on the right side
• Switch to your build directory and rebuild:
  
  cmake --build . 
  --config Release

• Execute the example
Lesson 21: 1D Work Division

Printing the Thread index

• Open helloWorld again

• Change the Kernel as shown below:

```c
// Add these lines:
uint32_t blockThreadIdx = idx::getIdx<Block, Threads>(acc)[0];
printf("Hello, World from alpaka thread %u in block %u!\n", blockThreadIdx, gridBlockIdx);
```

• Switch to your build directory and rebuild:
  ```
cmake --build . --config Release
  ```

• Execute the example again
Lesson 21: 1D Work Division

Computing the index

- Some algorithms may need to (re-)compute the indices
- Example: Computing the gridThreadIdx

```cpp
uint32_t gridBlockIdx = idx::getIdx<Grid, Blocks>(acc)[0];
uint32_t blockThreadExtent = workdiv::GetWorkDiv<Block, Threads>(acc)[0];
uint32_t blockThreadId = idx::getIdx<Block, Threads>(acc)[0];

uint32_t gridThreadId = gridBlockIdx * blockThreadExtent + blockThreadId;
```
Printing the index

- Open `helloWorld` again
- Change the kernel as shown below:
  ```c
  // Add these lines:
  uint32_t blockThreadExtent = workdiv::getWorkDiv<Block, Threads>(acc)[0];
  uint32_t gridThreadId = gridBlockId * blockThreadExtent + blockThreadId;
  printf("Hello, World from alpaka thread %u!\n", gridThreadId);
  ```
- Switch to your build directory and rebuild:
  ```bash
  cmake --build . --config Release
  ```
- Execute the example again
Lesson 21: 1D Work Division

Summary

- Indices are obtained by the `alpaka::idx::getIdx` functions
- Extents are obtained by the `alpaka:workdiv::getWorkDiv` functions
- Indices can be calculated if required