

Hands-on: SIMD in loops with if statements

- Choose either C/C++ (`cpp`) or Fortran (`fortran`) samples. Both of them are fine, as well.

C/C++

How to compile and how to execute

1. Run a script of creating working space

- See `objst` in `create_project.sh`.
 - It indicates a list of the settings of compile options.
 - Before running `create_project.sh`, please modify `[resource_group]` to the adequate resource group in the `task.sh`.
- Running `create_project.sh`, the executable file (`run.x`) is generated in `Obj_***` directory.
 - This directory is also automatically generated.
 - When you desire to use the own compiler, set `MAKE_DIR` variable in `create_project.sh` from `config` to `config.own`.
- Example:

```
$ cd cpp
$ bash create_project.sh
$ ls
Obj_clang  Obj_clang.novec
```

2. Run program

- A job script to execute the program is located in `Obj_***/results`.
- You can run the program either:

```
## To run as a batch job
$ cd Obj_clang/results
$ pjsub task.sh
## Or, to run in an interactive job
$ cd Obj_clang/results
$ bash task.sh
```

- Each of the samples in the Exercises will be completed within 1 minutes.
 - For safety, we set the upper limit of job elapsed time as 3 minutes.

Exercises A

- E1: Check the compiler reports (`*.lst`) about `Makefile.clang` or `Makefile.clang.novec`. Are the loops of the kernels (`calc1`, `calc2`, and `calc2_mod`) successfully vectorized?

- E2: Compare the performance (Elapsed time) in `Obj_clang.novec` to that in `Obj_clang`, to consider whether vectorization is effective even in the presence of if statements inside a loop.

Exercises B (advanced)

- E3: Set the second argument of `run.` as `0` (Thus, the if statements in `mykernel.cpp` are always false). Then, check the elapsed time.
- E4: Test the case of `cpp_mix-trad`, in which the main part is compiled with Clang mode but a part of `mykernel.cpp` (renamed by `optkernel.cpp`) with Trad mode.

Note

- We summarize the main compiler options of each Makefile.

```
Makefile.clang      #Clang mode. Auto-vectorization with scalable vector length
                    and interleaving are allowed.
Makefile.clang.512  #Clang mode. Auto-vectorization with 512-bit is allowed.
Makefile.clang.novec #Clang mode, without vectorization.
Makefile.trad.nosimd #Trad mode, without vectorization.
Makefile.trad.simd   #Trad mode. Auto-vectorization with -Ksimd=auto is allowed.
Makefile.trad.simd2  #Trad mode. Auto-vectorization with -Ksimd=2 is allowed.
```

Fortran

1. Run a script of creating working space

- See `objst` in `create_project.sh`.
 - It indicates a list of the settings of compile options.
 - Before running `create_project.sh`, please modify `[resource_group]` to the adequate resource group in the `task.sh`.
- Running `create_project.sh`, the executable file (`run.x`) is generated in `Obj_***` directory.
 - This directory is also automatically generated.
 - When you desire to use the own compiler, set `MAKE_DIR` variable in `create_project.sh` from `config` to `config.own`.
- Example:

```
$ cd fortran
$ bash create_project.sh
$ ls
Obj_nosimd  Obj_simd  Obj_simd2
```

2. Run program

- A job script to execute the program is located in `Obj_***/results`.
- You can run the program either:

```
## To run as a batch job
$ cd Obj_simd/results
$ pjsub task.sh
## Or, to run in an interactive job
$ cd Obj_simd/results
$ bash task.sh
```

- Each of the samples in the Exercises will be completed within 1 minutes.
 - For safety, we set the upper limit of job elapsed time as 3 minutes.

Exercises A

- E1: Check the compiler reports (*.lst) about `Makefile.simd` and `Makefile.simd2`. Are the loops of the kernels (`calc1`, `calc2`, and `calc2_mod`) successfully vectorized?
- E2: Compare the performance (Elapsed time) in `Obj_nosimd` to that in `Obj_simd` or `Obj_simd2`, to consider whether vectorization is effective even in the presence of if statements inside a loop.

Exercises B (advanced)

- E3: Set the second argument of `run.` as `0` (Thus, the if statements in `mykernel.cpp` are always false). Then, check the elapsed time.

Note

- We summarize the main compiler options of each Makefile.

```
Makefile.nosimd  #Auto-vectorization is not allowed.
Makefile.simd    #Auto-vectorization with -Ksimd=auto is allowed.
Makefile.simd2   #Auto-vectorization with -Ksimd=2 is allowed.
```