

06\_llio/01\_localtmp

# Hands-on: File output using Local Temporary Region

---

- Choose either C++ (`cpp`) or Fortran (`fortran`) sample. Both of the are fine, as well.
- In this hands-on, we use Local Temporary Area. When we run the scripts in an interactive job, we need to specify `--llio localtmp-size=10Gi` option of pjsub option in submitting the interactive job.
- In addition, when we use spack and sort\_libp, we need to specify `-x PJM_LLIO_GFSCACHE=/vol0004` in submitting the interactive job.

```
## example:
$ pjsub --interact -g [group_id] -L "node=1, elapse=6:00:00" --sparam "wait-time=unlimited" --llio localtmp-size=10Gi -x PJM_LLIO_GFSCACHE=/vol0004 --mpi
proc=4
```

## C++

How to compile and how to execute

### 1. Compile a program.

- The executable file (`run.x`) is generated in `cpp/`.

```
$ cd cpp
$ make # make -f Makefile.own # if using own compiler
$ ls
main.cpp main.lst main.o run.x
```

## Fortran

How to compile and how to execute

### 1. Compile a program.

- The executable file (`run.x`) is generated in `fortran/`.

```
$ cd fortran
$ make # make -f Makefile.own # if using own compiler
$ ls
main.f90 main.lst main.o run.x
```

### 2. Run Program

- There are two directories: **2ndLayerCache** and **LocalTmp**.
  - **LocalTmp**: The directory is for tasks in which files are output to Local Temporary Area.
  - **2ndLayerCache**: The directory is for tasks in which files are output to 2nd Layer Storage (FEFS) via Cache Area for 2nd Layer Storage.

```
## To run as a batch job
$ cd LocalTmp
$ pjsub task.sh
$ cd ../2ndLayerCache
$ pjsub task.sh
## or , to run in an interactive job
$ cd LocalTmp
$ bash task.sh
$ cd ../2ndLayerCache
$ bash task.sh
```

- The array size to be output, the number of calculation steps, the frequency of file output, and the output file name should be specified in **input.txt**.
- CAUTION: When you run the scripts, large-size files may be output in your data area of 2nd Layer Storage. For example, when the array size is 5242880, the size of each output file is 400 MiB. If you run the job using 24 processes, the total output file size is 9.6 GiB.

## Exercises A

- E1: Check the jobscript **LocalTmp/task.sh**, where the path of the Local Temporary Area is given by the environmental variable **\$PJM\_LOCALTMP** and the size of the area we use in this job is specified by the pjsub option **--llo localtmp-size**.
- E2: Run the jobscripts in **LocalTmp** and **2ndLayerCache** directories, and compare the times taken to file output by checking the standard output file (**result/(jobid).log.1.0**)

## Exercises B (optional)

- E3: Using darshan-parser, check the input and output files in this program and examine the times taken to I/O of each file.
- E4: examine the change of the time taken to file output when you change the number of MPI processes to 1, 2, 4, 8, 12, and so on.
- E5: examine the change of the time taken to file output when you change the array size to 1024, 65536, 786432, 16777216, 52428800 and so on.