

Hands-on: Measuring memory latency by LMBench

- Both of C/C++ and Fortran users will perform the common benchmarks, although LMBench is written by C.

How to execute

1. Edit a job script

- Before trying this hands-on, you need to do `00_lmbench` to compile LMBench.
- A job script to execute the program is located in `results/`.
- Edit `BINDIR` variable in `results/task.sh` before the execution.
 - You need to write your installed location of LMBench there. In this exercises, we use `lat_mem_rd`.

2. Run program

- You can run the program either:

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

- The job in the Exercise will be completed within 5 minutes.
 - For safety, we set the elapsed time of the job script as 7 minutes.

Exercises A

- E1: Examine the hardware specification of cache in A64FX, focusing on the following points.
 - How many layers in cache are there?
 - What about the size of each cache?
 - Whether is a cache shared with cores or not?
- E2: Check the command-line option in `lat_mem_rd`.
 - You do not need to take care of the first option (-P 1).
 - The second option (256) means the maximum memory size, in unit of MiB.
 - This value should be larger than the size of the last-level cache at least.
 - The remaining numbers indicate strides in unit of bytes to access memory address.
- E3: Check the output of `lat_mem_rd`, depending on the values of stride. In particular, find the points of memory size in which the latency abruptly changes.
 - The first column indicates memory size (MiB),
 - The second one indicates the measured latency (ns).