Server specific setting

Use ISSP System B: Ohtaka

- Module settings
  OneAPI(Intel) + OpenMPI is recommended

```
module purge
module load openmpi/4.1.5-oneapi-2023.0.0-classic
```

> [!TIP]
> OneAPI_MPI is slightly faster but less stable.

```
#!/bin/sh
#SBATCH -p i8cpu
#SBATCH -N 1
#SBATCH -n 8
#SBATCH --exclusive

module purge
module load openmpi/4.1.5-oneapi-2023.0.0-classic
ulimit -s unlimited

FC=ifort ./InstallAll --clean
```

> [!TIP]
> Although `mpirun` is used internally in `InstallAll`, TestInstall passes based on experience (perhaps because it uses a single node).

PROF

Additional information on parallelism

- Specifying the number of parallel processes
  OpenMP parallelism is less effective, so basically use MPI parallelism.
  However, if you run out of memory, reduce the number of cores per node to ensure enough memory per core.

```
#SBATCH -p F16cpu
#SBATCH -N 16
#SBATCH -n 512
#SBATCH -c 4
#SBATCH --ntasks-per-node=32
#SBATCH -t 24:00:00
```

> [!TIP]
> If you encounter MKL (Intel Math Kernel Library) errors during execution, set `-c 1`.

special modification of ecalj code for kugui (just for developer)

The ecalj MPI execution command assumes `mpirun`. (It is specified in the script)
Before installation, change `mpirun -np` to `srun -n` in the following files to match the ISSP specifications. (It will work with `mpirun` but may give warnings)
Replace the `ecalj` directory path with your own.

```
~/ecalj/SRC/exec/gwutil.py
~/ecalj/SRC/exec/run_arg.py
~/ecalj/SRC/exec/job_tdos
```

```
def run_program(commandline, ncore=0,x0=0):
    import subprocess,datetime
    xdate=datetime.datetime.now() #today().isoformat()
    mpirun='mpirun -np %d '%ncore if ncore!=0 else ''
```

```
def run_arg(argin, mpi_size, nfpgw, command, output, *target):
    echo_run = True  # standard
    mpi_run = f"mpirun -np {mpi_size}"  # standard
```

```
def run_arg(argin, mpi_size, nfpgw, command, output, *target):
    echo_run = ""  # standard
    serial_run = ""  # standard
    mpi_run = f"mpirun -np {mpi_size}"  # standard
```

PROF

> [!TIP]
> You can also modify the files in `~/bin/` after installation, but they will be overwritten if you reinstall.

Use ISSP System C: Kugui

- Assuming the use of GPU
- Module settings
  nvfortran + OpenMPI

```
module purge
module load nvhpc-nompi/24.7 openmpi_nvhpc compiler-rt tbb mkl
```

> [!TIP]
> The Intel MKL math library is also used for CPU calculations. It can be used from NVIDIA or GNU Fortran.

- To use MPS (multiple MPI processes sharing a single GPU), add the following to `~/.bashrc`

```
if which nvidia-cuda-mps-control > /dev/null 2>&1 ; then
  export CUDA_MPS_PIPE_DIRECTORY=$(pwd)/nvidia-mps-$(hostname)
  export CUDA_MPS_LOG_DIRECTORY=$(pwd)/nvidia-log-$(hostname)
  echo "start nvidia-cuda-mps-control at" $(hostname)
  nvidia-cuda-mps-control -d
fi
```

- Job script

```
#!/bin/sh
#PBS -q F1accs
#PBS -l select=1:ncpus=64:mpiprocs=64:ompthreads=1
#
ulimit -s unlimited
id=inas6gasb6
gwsc -np 64 -np2 4 --gpu 5 $id > lgwsc
```

- Specify the number of parallel processes for CPU calculations with `-np`.
- Specify the number of parallel processes for GPU calculations with `-np2`. Usually, this is the number of available GPUs.
- `--gpu` uses the GPU version of the executable.

The GPU version may not speed up small systems, but it can handle up to about 40 atoms with 4 GPUs (depending on the calculation conditions. Note that the calculation cost increases with the square of the number of k-points).

PROF

Script modification: Optional (only for developer)

When using OpenMP parallelism for CPU calculations, change `mpirun -np` to `mpiexec --bind-to none --map-by node -np`.
`--map-by node` is nessesary to to specify the different number of MPI processes in the single jobs.

`--bind-to none` and `--map-by node` are the options in OpenMPI. It may not work in other MPI library.

Replace the `ecalj` directory path with your own.

```
~/ecalj/SRC/exec/gwutil.py
```

```
def run_program(commandline, ncore=0,x0=0):
    import subprocess,datetime
    xdate=datetime.datetime.now() #today().isoformat()
    mpirun='mpiexec --bind-to none --map-by node -np %d '%ncore if
ncore!=0 else ''
```

Also, change `mpirun` in other scripts as needed.