MKL Fortran example:

- **Objectives and learning goals:** compile a program for coprocessor only execution; make use of automatic offload; make use of compiler-assisted offload; Also, learn to adjust the affinity settings for Intel OpenMP*, and experiment with large memory pages – an option that is offered by the Linux* μOS on the coprocessor.

1. **On the host, cross-compile** `00_getting_started.f90` for coprocessor execution **using:**
   ```
   ifort -openmp -mkl -mmic 00_getting_started.f90 -o 00_getting_started_native
   ```

2. **Copy the generated executable to the coprocessor:**
   ```
   scp 00_getting_started_native hostname-mic0:~/
   ```

   **Login to the coprocessor and run the program:**
   ```
   ./00_getting_started_native
   ```

   If the program reports any missing libraries, copy the necessary files from $MKL_BASE/lib/mic and $IFORT_BASE/compiler/lib/mic on the host to a directory on the coprocessor. Set the LD_LIBRARY_PATH environment variable to point to that directory, then rerun the executable. Alternately, you can use micnativeloadex utility.

3. **Next compile** `00_getting_started.f90` to use automatic offload. On the host, open `00_getting_started.f90` and add the line
   ```
   call mkl_mic_enable()
   ```
   near the beginning of the `getting_started` function before the execution proceeds to SGEMM or DGEMM. Alternatively, you can set the environment variable MKL_MIC_ENABLE=1.

   Compile and execute the program on the host. Some of the work will automatically be offloaded to the coprocessor:
   ```
   ifort -openmp -mkl 00_getting_started.f90 -o 00_getting_started
   ```

4. **Now compile** `01_offload.f90` using the Language Extensions for Offload (LEO) to offload the entire `run` function to the coprocessor. Open `01_offload.f90` and add a `!dir$ offload` directive before each call to the `run` function. Specify which data is going into the offload section and which is coming out. For example:
   ```
   !dir$ offload target(mic) in(a:length(n))
   ```
   in front of a function copies in the array `a`. Compile and run the program:
   ```
   ifort -openmp -mkl 01_offload.f90 -o 01_offload
   ```

   The Intel compiler does not require an option in order to enable compiler-assisted offload. LEO can be disabled even when an offload directive is found, using `-no-offload`.

#### Compare the execution models for Intel Xeon Phi ######