**Fortran/C on HCC**

This quick start demonstrates how to implement a Fortran/C program on HCC supercomputers. The sample codes and submit scripts can be downloaded from `<serial_dir.zip>`.

**Login to a HCC Cluster (Tusker or Sandhills)**

Log in to a HCC cluster through PuTTY (For Windows Users) or Terminal (For Mac/Linux Users) and make a subdirectory called `serial_dir` under the `$WORK` directory.

```
$ cd $WORK
$ mkdir serial_dir
```

In the subdirectory `serial_dir`, save all the relevant Fortran/C codes. Here we include two demo programs, `demo_f_serial.f90` and `demo_c_serial.c`, that compute the sum from 1 to 20.

---

**demo_f_serial.f90**

```
Program demo_f_serial
     implicit none
     integer, parameter :: N = 20
     real*8 w
     integer i
     common/sol/ x
     real*8 x
     real*8, dimension(N) :: y

     do i = 1,N
         w = i*1d0
         call proc(w)
         y(i) = x
         write(6,*) 'i,x = ', i, y(i)
     enddo
     write(6,*) 'sum(y) =', sum(y)
Stop
End Program

Subroutine proc(w)
     real*8, intent(in) :: w
     common/sol/ x
     real*8 x

     x = w

     Return
End Subroutine
```
//demo_c_serial
#include <stdio.h>

double proc(double w){
    double x;
    x = w;
    return x;
}

int main(int argc, char* argv[]){
    int N=20;
    double w;
    int i;
    double x;
    double y[N];
    double sum;
    for (i = 1; i <= N; i++){
        w = i*1e0;
        x = proc(w);
        y[i-1] = x;
        printf("i,x= %d %lf\n", i, y[i-1]) ;
    }
    sum = 0e0;
    for (i = 1; i<= N; i++){
        sum = sum + y[i-1];
    }
    printf("sum(y)= %lf\n", sum);
    return 0;
}
Creating a Submit Script

Create a submit script to request one core (default) and 1-min run time on the supercomputer. The name of the main program enters at the last line.

<table>
<thead>
<tr>
<th>submit_f.serial</th>
</tr>
</thead>
<tbody>
<tr>
<td>#!/bin/sh</td>
</tr>
<tr>
<td>#SBATCH --mem-per-cpu=1024</td>
</tr>
<tr>
<td>#SBATCH --time=00:01:00</td>
</tr>
<tr>
<td>#SBATCH --job-name=Fortran</td>
</tr>
<tr>
<td>#SBATCH --error=Fortran.%J.err</td>
</tr>
<tr>
<td>#SBATCH --output=Fortran.%J.out</td>
</tr>
<tr>
<td>module load compiler/gcc/4.9</td>
</tr>
<tr>
<td>./demo_f_serial.x</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>submit_c.serial</th>
</tr>
</thead>
<tbody>
<tr>
<td>#!/bin/sh</td>
</tr>
<tr>
<td>#SBATCH --mem-per-cpu=1024</td>
</tr>
<tr>
<td>#SBATCH --time=00:01:00</td>
</tr>
<tr>
<td>#SBATCH --job-name=C</td>
</tr>
<tr>
<td>#SBATCH --error=C.%J.err</td>
</tr>
<tr>
<td>#SBATCH --output=C.%J.out</td>
</tr>
<tr>
<td>module load compiler/gcc/4.9</td>
</tr>
<tr>
<td>./demo_c_serial.x</td>
</tr>
</tbody>
</table>

Submit the Job

The job can be submitted through the command `sbatch`. The job status can be monitored by entering `squeue` with the `-u` option.

$ sbatch submit_f.serial
$ sbatch submit_c.serial
$ squeue -u <username>

Sample Output

The sum from 1 to 20 is computed and printed to the .out file (see below).
### Fortran.out

```
<table>
<thead>
<tr>
<th>i, x</th>
<th>1</th>
<th>1.0000000000000000</th>
</tr>
</thead>
<tbody>
<tr>
<td>i, x</td>
<td>2</td>
<td>2.0000000000000000</td>
</tr>
<tr>
<td>i, x</td>
<td>3</td>
<td>3.0000000000000000</td>
</tr>
<tr>
<td>i, x</td>
<td>4</td>
<td>4.0000000000000000</td>
</tr>
<tr>
<td>i, x</td>
<td>5</td>
<td>5.0000000000000000</td>
</tr>
<tr>
<td>i, x</td>
<td>6</td>
<td>6.0000000000000000</td>
</tr>
<tr>
<td>i, x</td>
<td>7</td>
<td>7.0000000000000000</td>
</tr>
<tr>
<td>i, x</td>
<td>8</td>
<td>8.0000000000000000</td>
</tr>
<tr>
<td>i, x</td>
<td>9</td>
<td>9.0000000000000000</td>
</tr>
<tr>
<td>i, x</td>
<td>10</td>
<td>10.000000000000000</td>
</tr>
<tr>
<td>i, x</td>
<td>11</td>
<td>11.000000000000000</td>
</tr>
<tr>
<td>i, x</td>
<td>12</td>
<td>12.000000000000000</td>
</tr>
<tr>
<td>i, x</td>
<td>13</td>
<td>13.000000000000000</td>
</tr>
<tr>
<td>i, x</td>
<td>14</td>
<td>14.000000000000000</td>
</tr>
<tr>
<td>i, x</td>
<td>15</td>
<td>15.000000000000000</td>
</tr>
<tr>
<td>i, x</td>
<td>16</td>
<td>16.000000000000000</td>
</tr>
<tr>
<td>i, x</td>
<td>17</td>
<td>17.000000000000000</td>
</tr>
<tr>
<td>i, x</td>
<td>18</td>
<td>18.000000000000000</td>
</tr>
<tr>
<td>i, x</td>
<td>19</td>
<td>19.000000000000000</td>
</tr>
<tr>
<td>i, x</td>
<td>20</td>
<td>20.000000000000000</td>
</tr>
</tbody>
</table>
```

\[ \text{sum}(y) = 210.000000000000000 \]

### C.out

```
i, x = 1  1.000000
i, x = 2  2.000000
i, x = 3  3.000000
i, x = 4  4.000000
i, x = 5  5.000000
i, x = 6  6.000000
i, x = 7  7.000000
i, x = 8  8.000000
i, x = 9  9.000000
i, x = 10 10.000000
i, x = 11 11.000000
i, x = 12 12.000000
i, x = 13 13.000000
i, x = 14 14.000000
i, x = 15 15.000000
i, x = 16 16.000000
i, x = 17 17.000000
i, x = 18 18.000000
i, x = 19 19.000000
i, x = 20 20.000000
```

\[ \text{sum}(y) = 210.000000000000000 \]