

Compiling and executing an MPI program on *Kraken: Draft version 2011.12.28*

We shall assume that you have your program ready to compile. The first and most important thing to realize is that to compile an MPI program quickly and to not have to worry about linking to libraries, a *makefile* is required. A makefile has all of the important information about what libraries to use, where to find them, which executable to create using which programs and much more. For the most part this semester, we will be compiling single programs into single executables.

You can download the barebones makefile from the website. A copy of this file should be in your working directory. Please be sure to create a working directory for EACH program you are working on.

The only two lines you need change are:

```
PROGRAM    = PROGRAM_NAME    # name of the binary
SRCS       = PROGRAM_NAME.c
```

The `PROGRAM_NAME` on the `PROGRAM` line, and `PROGRAM_NAME.c` should be changed to the output executable and name of c file you want compiled, respectively. They do not have to be the same, but it is suggested that you do name them the same. For instance, if we have a C program called `HelloWorld.c`, and wish to create an executable called `HelloWorld`, we would change the two lines of the makefile given above to

```
PROGRAM    = HelloWorld      # name of the binary
SRCS       = HelloWorld.c
```

and that is it. Once you have updated this file and are certain that it is in your working directory, you are ready to compile. We will assume you are once again in your working directory, and the first command you will enter is: `make clean`

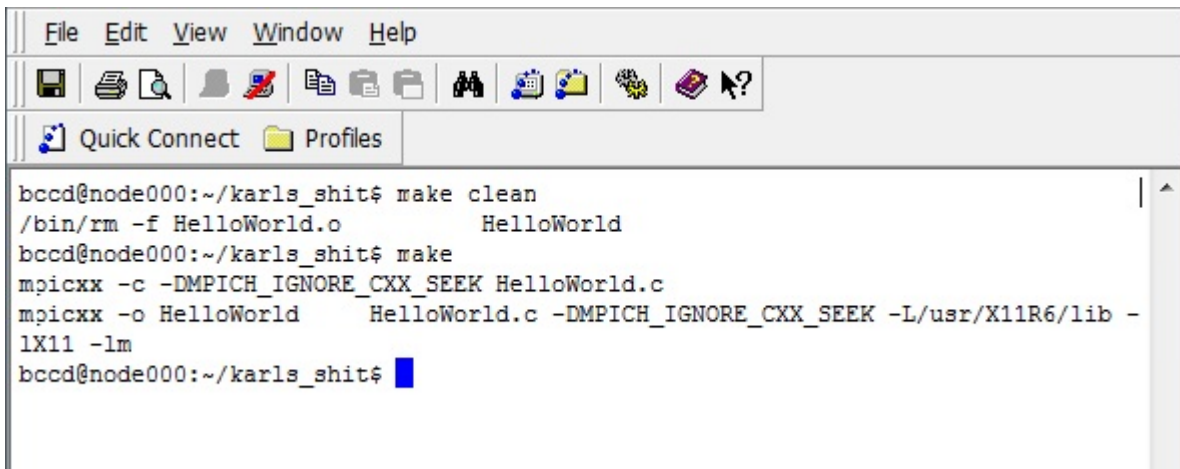
A screenshot of a terminal window with a standard Linux desktop environment. The window title is "File Edit View Window Help". The terminal shows the following text:

```
bccd@node000:~/karls_shit$ make clean
/bin/rm -f HelloWorld.o      HelloWorld
bccd@node000:~/karls_shit$
```

The terminal prompt is blue, and the output is black text on a white background.

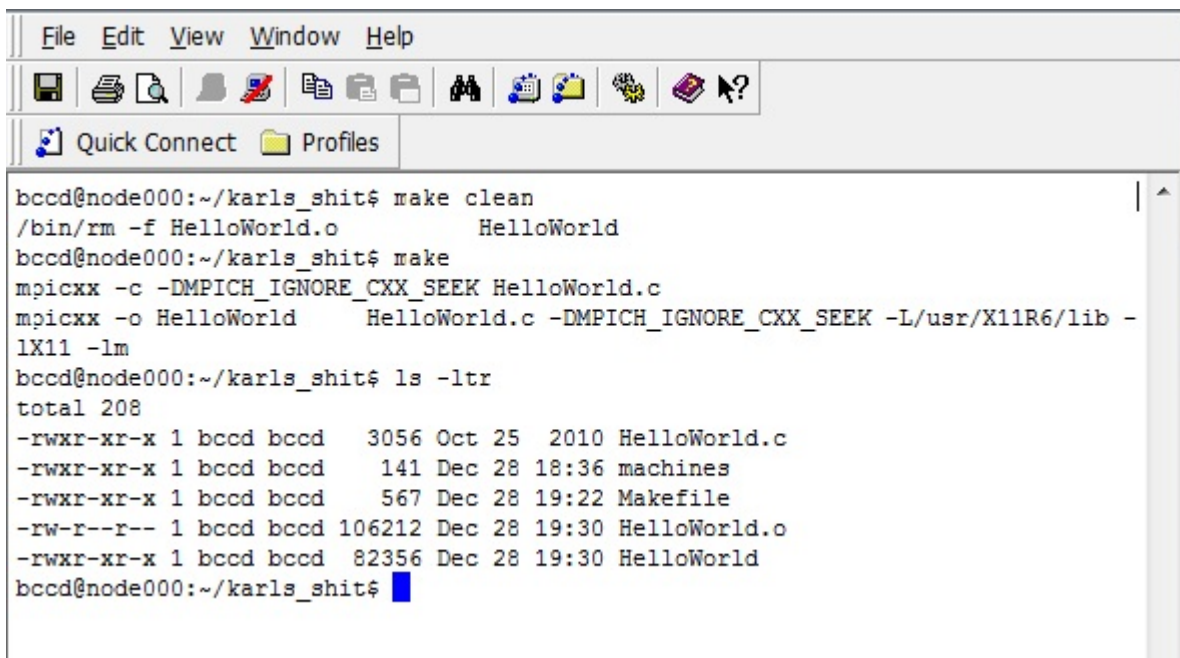
Notice that the command has some output. All it did was remove any old executables so that when you compile there will be no overwrite permission issues.

Now that everything is cleaned up, we simply type in the command: make



```
File Edit View Window Help
Quick Connect Profiles
bccd@node000:~/karls_shit$ make clean
/bin/rm -f HelloWorld.o      HelloWorld
bccd@node000:~/karls_shit$ make
mpicxx -c -DMPICH_IGNORE_CXX_SEEK HelloWorld.c
mpicxx -o HelloWorld      HelloWorld.c -DMPICH_IGNORE_CXX_SEEK -L/usr/X11R6/lib -lX11 -lm
bccd@node000:~/karls_shit$
```

If your program compiled correctly, there will be no errors and you will have an executable now in your directory. To verify, list the contents of the directory in table format in time order with the command: `ls -ltr`



```
File Edit View Window Help
Quick Connect Profiles
bccd@node000:~/karls_shit$ make clean
/bin/rm -f HelloWorld.o      HelloWorld
bccd@node000:~/karls_shit$ make
mpicxx -c -DMPICH_IGNORE_CXX_SEEK HelloWorld.c
mpicxx -o HelloWorld      HelloWorld.c -DMPICH_IGNORE_CXX_SEEK -L/usr/X11R6/lib -lX11 -lm
bccd@node000:~/karls_shit$ ls -ltr
total 208
-rwxr-xr-x 1 bccd bccd  3056 Oct 25  2010 HelloWorld.c
-rwxr-xr-x 1 bccd bccd   141 Dec 28 18:36 machines
-rwxr-xr-x 1 bccd bccd   567 Dec 28 19:22 Makefile
-rw-r--r-- 1 bccd bccd 106212 Dec 28 19:30 HelloWorld.o
-rwxr-xr-x 1 bccd bccd 82356 Dec 28 19:30 HelloWorld
bccd@node000:~/karls_shit$
```

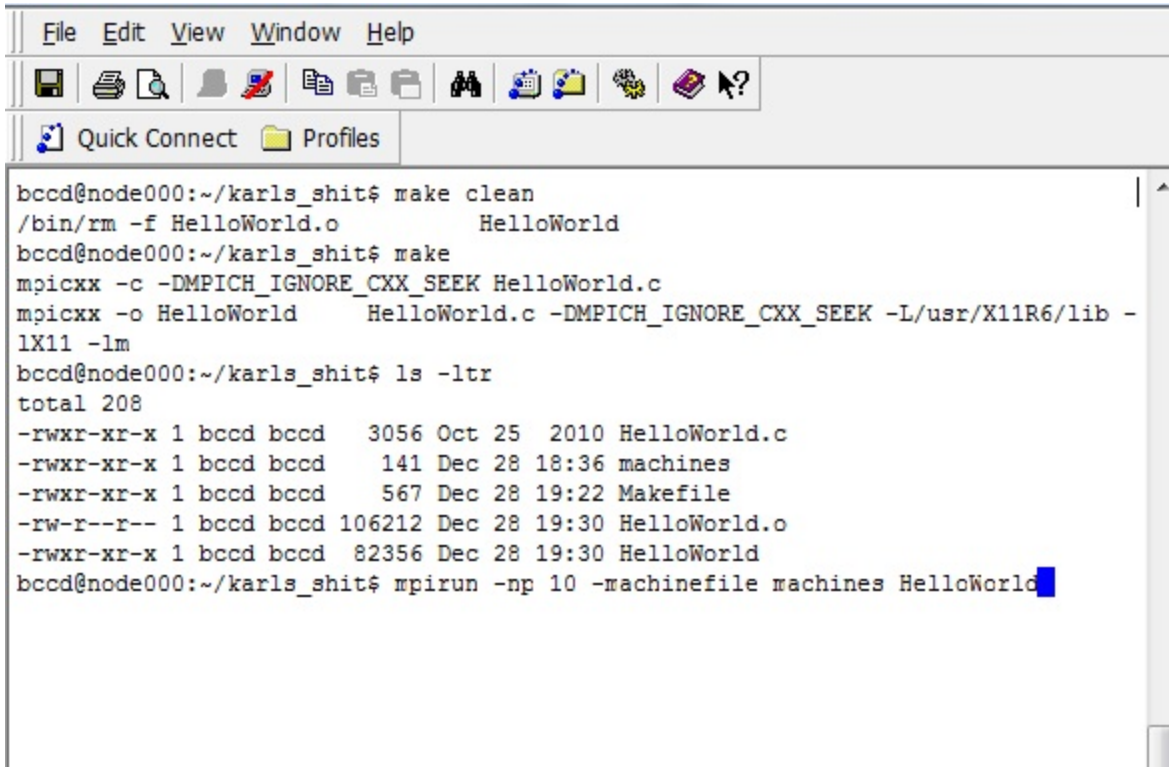
As you can see, the last file created is HelloWorld. This is the executable, which we will run next.

Now that we have our executable, which was compiled with MPI instructions in it, we need to put these instructions to good use, which means making sure that more than just two processors run it! For *kraken* to know where to send the program to at execute time, it needs a *machine file*. You will notice in the directory list above that there is a file called *machines*. This can be copied from the website, or you can create a new one by typing in the command: `bccd-snarhosts` and then copying the machine file from the home directory into your directory. Either way, you need that machine file there! You can also create a copy of the file to keep in your directory and then make a copy called *machines* each time you wish to run an mpi program.

We will next run our program HelloWorld with the mpi syntax in mind. The general syntax is as follows: `mpirun -np NP -machinefile MACHINEFILE Program_Name OPTS`

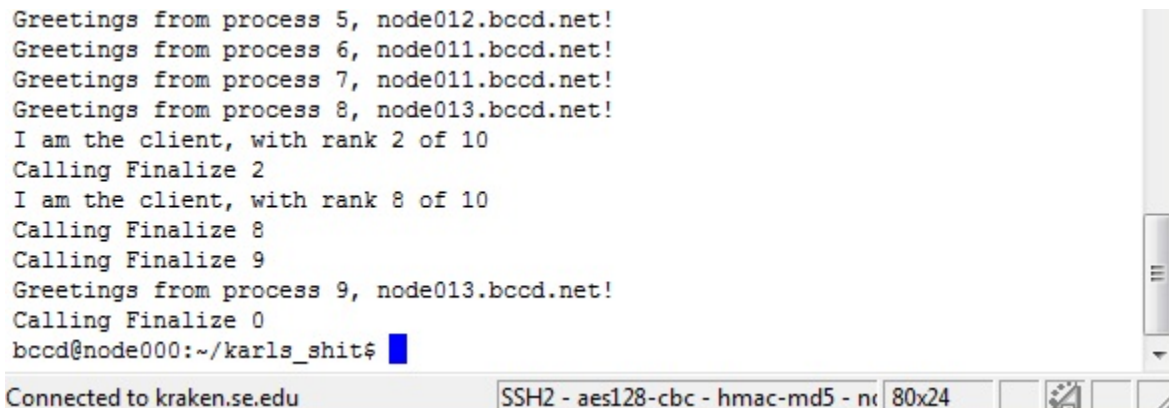
Here NP (number of processes) is a number between 1 and 12 for our system, MACHINEFILE is the name of the machine file, which in our case, should usually be machines. Finally, Program_Name is the name of your program that you wish to execute and OPTS are optional arguments based on the program itself (many programs do not require the input of additional information, but some more sophisticated ones do).

For our example, a simple execution command would look as follows:



```
File Edit View Window Help
Quick Connect Profiles
bccd@node000:~/karls_shit$ make clean
/bin/rm -f HelloWorld.o      HelloWorld
bccd@node000:~/karls_shit$ make
mpicxx -c -DMPICH_IGNORE_CXX_SEEK HelloWorld.c
mpicxx -o HelloWorld      HelloWorld.c -DMPICH_IGNORE_CXX_SEEK -L/usr/X11R6/lib -lX11 -lm
bccd@node000:~/karls_shit$ ls -ltr
total 208
-rwxr-xr-x 1 bccd bccd  3056 Oct 25  2010 HelloWorld.c
-rwxr-xr-x 1 bccd bccd   141 Dec 28  18:36 machines
-rwxr-xr-x 1 bccd bccd   567 Dec 28  19:22 Makefile
-rw-r--r-- 1 bccd bccd 106212 Dec 28  19:30 HelloWorld.o
-rwxr-xr-x 1 bccd bccd  82356 Dec 28  19:30 HelloWorld
bccd@node000:~/karls_shit$ mpirun -np 10 -machinefile machines HelloWorld
```

Your program will commence running! Once your program is done running, you should get back to your command line:



```
Greetings from process 5, node012.bccd.net!
Greetings from process 6, node011.bccd.net!
Greetings from process 7, node011.bccd.net!
Greetings from process 8, node013.bccd.net!
I am the client, with rank 2 of 10
Calling Finalize 2
I am the client, with rank 8 of 10
Calling Finalize 8
Calling Finalize 9
Greetings from process 9, node013.bccd.net!
Calling Finalize 0
bccd@node000:~/karls_shit$
```

Connected to kraken.se.edu SSH2 - aes128-cbc - hmac-md5 - nc 80x24

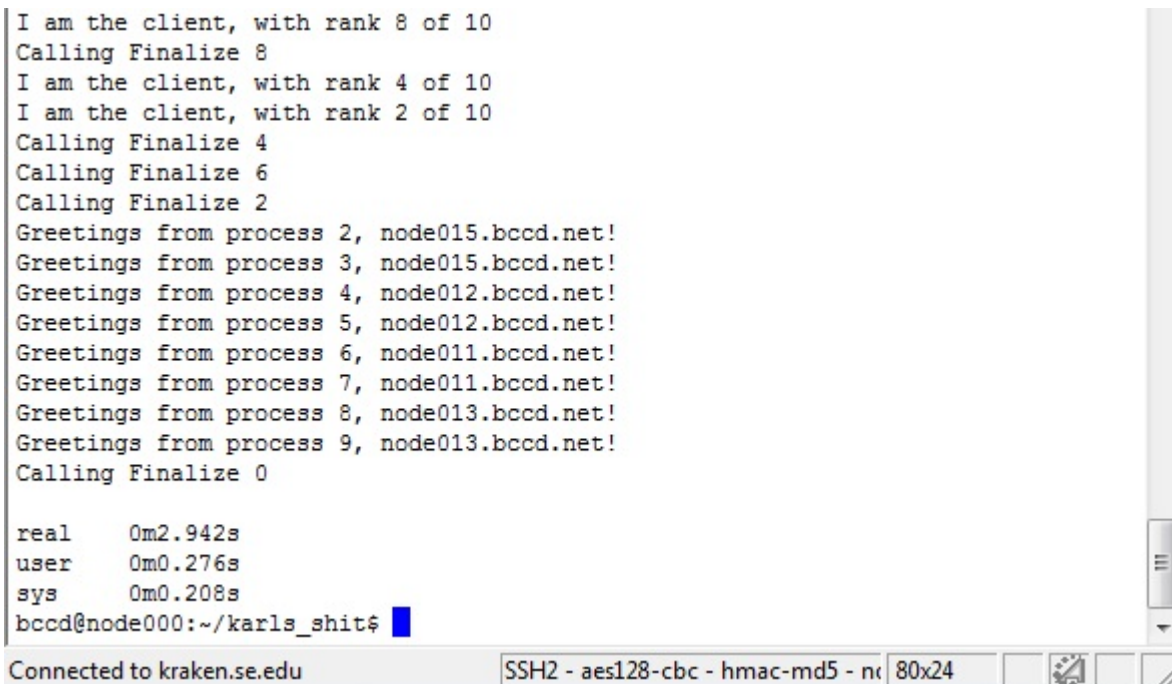
If you need to stop your program before it is done running (for instance infinite loops), just enter ctrl+c, this will cancel the currently executing command (which in this case is your program).

Sometimes, it is important to determine how long it took for your program to run. The mpirun command can be preceded by a *time* option, as depicted in the following screenshot:



```
File Edit View Window Help
[Icons]
Quick Connect Profiles
bccd@node000:~/karls_shit$ time mpirun -np 10 -machinefile machines HelloWorld
```

Note the output at the end of the execution cycle right above the current command line:



```
I am the client, with rank 8 of 10
Calling Finalize 8
I am the client, with rank 4 of 10
I am the client, with rank 2 of 10
Calling Finalize 4
Calling Finalize 6
Calling Finalize 2
Greetings from process 2, node015.bccd.net!
Greetings from process 3, node015.bccd.net!
Greetings from process 4, node012.bccd.net!
Greetings from process 5, node012.bccd.net!
Greetings from process 6, node011.bccd.net!
Greetings from process 7, node011.bccd.net!
Greetings from process 8, node013.bccd.net!
Greetings from process 9, node013.bccd.net!
Calling Finalize 0

real    0m2.942s
user    0m0.276s
sys     0m0.208s
bccd@node000:~/karls_shit$
```

Be sure to keep a copy of the unmodified makefile and the machine file handy, as they will both be needed for each program you wish to compile and execute. The machine file (machines) should not be modified, but the makefile will have to be modified for each program you wish to compile, in the manner described at the beginning of this document.