NIU COSY User Guide

Herman Schaumburg
hschaumburg2@niu.edu

Revised on Tuesday 12th September, 2017

Content

1 Building COSY on GAEA

1.1 Recommended procedure

It is recommended that you use the COSY files from the BEAM repository to build the MPI version of COSY. These can be found in /BEAM/MPI_COSY. Some details on how these files were created from the MSU version of COSY are in the Appendices.
To begin, we will add some paths to your .bashrc file. Type the command

$ vim .bashrc
Type i to activated insert mode. At the bottom of the file, add the lines:

```
source /opt/intel/bin/ifortvars.sh intel64
source /opt/intel/bin/compilervars.sh intel64
source /opt/intel/bin/mpivars.sh
export PATH="/opt/intel/impi/5.0.1.035/intel64/bin:$PATH"
export PATH="opt/intel/bin/:$PATH"
```

This sets environmental variables for the future. For the current terminal session, you should run the commands:
$ **source** /opt/intel/bin/compilervars.sh intel64
$ **source** /opt/intel/bin/ifortvars.sh intel64

1.2 Building using makefiles

Large and small memory model options are in the makefile. To use gnu tools and the use
$ make fort=gnu mcmodel=large
or
$ make fort=gnu mcmodel=small
To use intel tools use
$ make fort=intel mcmodel=large
or
$ make fort=intel mcmodel=small
There are also makefile.gnu and makefile.intel files which can be renamed to makefile and built using the same flag for memeory model (mcmodel=). The paths in these makefiles may need to be updated as mpich is updated and moved on GAEA.

For details on executing COSY see B.3.9 Generate COSY.bin and B.3.10 Adding COSY to your path.

A Using the cosyresize utility

The advantage of using the cosyresize utility is that it bypasses using find and replace to edit the various fox files to adjust LMEM and LVAR. Tips on pages 5 and 5 may also be of interest.

A.1 Resize LMEM LVAR

The cosyresize utility can be copied from the location /BEAM/MPI_COSY/util/cosyresize/ to your home folder on GAEA. Change directories to where you copied the utility then run the make file using the command below.
$ make
Next, copy the mpi version of COSY created using the version utility (see appendix B.3.5) or another method to a folder with subfolder mpi in your home directory (For example ~/-COSY_BUILD/mpi). You may need to rename Makefile.txt or Makefile to makefile (though the utility does not seem to edit this makefile).
Use the command

$ **export** COSY_PATH=$HOME/COSY_BUILD
to set the correct path for cosyresize. In the case that we want to change LMEM from 140000000 to 100000000. We want to change LMEM by the factor

$$\text{lmem\_factor} = \frac{\text{New LMEM}}{\text{Old LMEM}} = \frac{100000000}{140000000} = 0.714285714286$$

Inside the same folder as cosyresize utility run
$ ./ cosyresize −m 0.714285714286 −−mpi
After this is done, the resized files will be in a build subdirectory (COSY_BUILD/build)
The -v flag can be use to scale LVAR by a factor.

B Building COSY from scratch

B.1 Windows 10 Users

It is possible to install a bash terminal in Windows 10 for builds later than 14393. You can check your version of windows by right clicking on the start button, clicking run, then typing winver and pressing enter. The steps to install bash are as follows

1. Install Developer Mode feature in windows.
2. Enable Developer Mode.
3. Enable the feature Subsystem-Linux.
4. Install Bash.

Will provide details at a future date.

B.2 Other Windows Users

Other windows users can use cygwin https://www.cygwin.com/.

B.3 Compiling MPI Version of COSY on Gaea

The steps for compiling COSY on Gaea are as follows:

1. Copy downloaded COSY to Gaea.
2. Create or Edit makefile (Makefile.txt).
3. Build the version utility to convert “normal” source to “MPI” source.
4. Run version utility on *.f files to be built.
5. Edit mpi-dafoxy.f file.
6. Build the code.
7. Add COSY to your PATH on Gaea.

The first thing to do is to download the files below to a folder named COSY on your local machine:

foxy.f - COSY INFINITY compiler and executer
dafox.f - COSY INFINITY object operations library
foxfit.f - COSY INFINITY optimizer package
foxgraf.f - COSY INFINITY graphics package
version.f - Program to modify FORTRAN source for various platforms
cosy.fox - Beam Physics library
SYSCA.DAT - Data file for fast symplectic fringe fields

from http://cosyinfinity.org/ You may also download Makefile.txt

B.3.1 Create makefile

It is **much better** to use the GAEA specific makefile avaiable on GAEA than to use the one here.

From within the COSY directory create a text file called makefile or edit Makefile.txt from MSU with the following contents:

```
# Definition of FC, FFLAGS, LINK and LIBS depends on each local system.
# We appreciate receiving information on your local makefile.
#
#
# Default Makefile Notes:
# If compiling with PGPlot, the corresponding LIBS line needs to be uncommented
#
# FFlags are primarily made for verification and optimization
#  -m32 is required on 64 bit systems
#  -fp-model strict is for verified numerics
#  -f77rtl is probably not required
#        (and will likely be deprecated in future versions of ifort)

FC = mpif77
FFLAGS=
LINK = mpif77
LIBS =
#LIBS = -L/usr/local/pgplot -lpgplot -L/usr/X11R6/lib -lX11

OBJ = mpi-foxy.o mpi-dafox.o mpi-foxfit.o mpi-foxgraf.o

cosy: $(OBJ)
        $(LINK) -o mpi-cosy $(OBJ) $(LIBS)
```

B.3.2 Copying local files to Gaea

Open a terminal and enter the command on a linux machine (Ctl + Alt + t).
$ scp −r /**local**.machine.folder/COSY user.name@gaea.niu.edu:/home/username/COSY
to copy the downloaded COSY folder to Gaea.

Tip: You can instead copy the files using the linux command sshfs to mount directories on Gaea on your local machine. As an example:
$ sshfs user.name@gaea.niu.edu:/ /MountPointEmptyLocalFolder
mounts the root directory / on Gaea.
You can also create a bash script to do this on your local machine. You can create a file named "gaeasshfs" with contents below in your local /home/user.name/bin directory:

```
#!/bin/bash
# gaea sshfs
sshfs user.name@gaea.niu.edu:/FolderOnGaea /MountPointEmptyLocalFolder
```

You can also make a bash script to unmount the folder on Gaea. You can create a file named "ungaeasshfs" with contents below in your local /home/user.name/bin directory:

```
#!/bin/bash
# gaea unsshfs
fusermount -u /WhateverMountPointEmptyLocalFolderYouChose
```

B.3.3 Adding compilers to your PATH in Gaea

In a bash terminal, login to gaea using the command
$ ssh user.name@gaea.niu.edu

Tip: You can create a bash script that can shorten logging on to Gaea on your local machine. You can create a file named "gaea" with contents below in your local /home-/user.name/bin directory:

```
#!/bin/bash
# gaea ssh
ssh user.name@gaea.niu.edu
```

Type the command
$ vim .bashrc
Type i to activated insert mode. At the bottom of the file, add the lines:

```
export PATH="/opt/intel/impi/5.0.1.035/intel64/bin:$PATH"
export PATH="opt/intel/bin/:$PATH"
```

Type ESC to exit insert mode. Then type : and then w and enter to write the file then use : q and enter to quit vim.

B.3.4 Building version utility

In a terminal, login to gaea using the command
$ ssh user.name@gaea.niu.edu
In the Gaea terminal. Use the command to change the active directory to the COSY folder
$ **cd** COSY
Use the command
$ ifort version.f −o version.o
to build the version utility. This allows you to alter the *.f files so that they can be compiled for MPI.

B.3.5 Using version utility

Run the command $ ./version.o to execute the version utility. You should see the following on the screen:

```
*************************************************
*                                               *
*             UTILITY PROGRAM  VERSION          *
*                                               *
* This program changes the type of machine/system. *
* The current COSY INFINITY system supports     *
* NORM MPI FACE RND and PGP GRW AQT.            *
* See the User's Guide and Reference Manual.    *
*                                               *
*************************************************

 GIVE OLD FILENAME:
```

For old file name type in "foxy.f" without quotes. When prompted for a new file name type "mpi_foxy.f". Next type *NORM for the ID of current version. Type *MPI for the ID of new version. Repeat the procedure starting from $ ./version for the files dafox.f, foxfit.f, and foxgraf.f appending each file name with "mpi_".

B.3.6 Edit mpi-foxy.f

We need to edit mpi-foxy.f. This can be done via vim by typing
$ vim mpi−foxy.f
First type : to bring up the command line in vim. Use the command below to do a "Find and replace".

```
:%s/PARAMETER(LMEM=140000000,LVAR=10000000,LDIM=1000)/PARAMETER(LMEM=100000000,LVAR=10000000,LDIM=1000)/g
```

Type : followed by / CALL MPI_ALLGATHER this should take you to the following snippet of the code:

```
 6026     IF(KCOMM.NE.MPICN.AND.NPEN.GT.1) THEN
             CALL MPI_ALLGATHER(NTYP(JPVA),JCOUNT,MPI_INTEGER,NTYP(IPLV), *MPI
```

```
     *         JCOUNT,MPI_INTEGER,KCOMM,MPERR)                              *MPI
              CALL MPI_ALLGATHER(NBEG(JPVA),JCOUNT,MPI_INTEGER,NBEG(IPLV), *MPI
     *         JCOUNT,MPI_INTEGER,KCOMM,MPERR)                              *MPI
              CALL MPI_ALLGATHER(NEND(JPVA),JCOUNT,MPI_INTEGER,NEND(IPLV), *MPI
     *         JCOUNT,MPI_INTEGER,KCOMM,MPERR)                              *MPI
*
```

Press i to enter insert mode. Add asterisks (*s) to comment this portion out and add new lines:

```
*THE FOLLOWING BLOCK IS RELATED TO PLOOP OPTIONS 1 AND 2
 6026    IF(KCOMM.NE.MPICN.AND.NPEN.GT.1) THEN
*             CALL MPI_ALLGATHER(NTYP(JPVA),JCOUNT,MPI_INTEGER,NTYP(IPLV), *MPI
*     *          JCOUNT,MPI_INTEGER,KCOMM,MPERR)                            *MPI
*             CALL MPI_ALLGATHER(NBEG(JPVA),JCOUNT,MPI_INTEGER,NBEG(IPLV), *MPI
*     *          JCOUNT,MPI_INTEGER,KCOMM,MPERR)                            *MPI
*             CALL MPI_ALLGATHER(NEND(JPVA),JCOUNT,MPI_INTEGER,NEND(IPLV), *MPI
*     *          JCOUNT,MPI_INTEGER,KCOMM,MPERR)                            *MPI
*
          CALL MPI_ALLGATHER(MPI_IN_PLACE,0,MPI_DATATYPE_NULL,NTYP(IPLV), *MPI
     *       JCOUNT,MPI_INTEGER,KCOMM,MPERR)                               *MPI
          CALL MPI_ALLGATHER(MPI_IN_PLACE,0,MPI_DATATYPE_NULL,NBEG(IPLV), *MPI
     *       JCOUNT,MPI_INTEGER,KCOMM,MPERR)                               *MPI
          CALL MPI_ALLGATHER(MPI_IN_PLACE,0,MPI_DATATYPE_NULL,NEND(IPLV), *MPI
     *       JCOUNT,MPI_INTEGER,KCOMM,MPERR)                               *MPI
```

Use the arrow keys to reach the snippet of code:

```
8026           CONTINUE
                IPSB = IPVA + JDISP
                KPVA = JPVA + JJ
                KCOUNT = NEND(KPVA) - NBEG(KPVA) + 1
          CALL MPI_ALLGATHERV(MPI_IN_PLACE,0,MPI_DATATYPE_NULL,           *MPI
     *        CC(IPBE),NRECV,NDISP,MPI_DOUBLE_PRECISION,KCOMM,MPERR)     *MPI
          CALL MPI_ALLGATHERV(MPI_IN_PLACE,0,MPI_DATATYPE_NULL,           *MPI
     *        NC(IPBE),NRECV,NDISP,MPI_INTEGER,KCOMM,MPERR)              *MPI
```

and comment out the lines below and add new lines:

```
8026           CONTINUE
                IPSB = IPVA + JDISP
                KPVA = JPVA + JJ
                KCOUNT = NEND(KPVA) - NBEG(KPVA) + 1
*               CALL MPI_ALLGATHERV(CC(IPSB),KCOUNT,MPI_DOUBLE_PRECISION, *MPI
*     *            CC(IPBE),NRECV,NDISP,MPI_DOUBLE_PRECISION,KCOMM,MPERR) *MPI
*               CALL MPI_ALLGATHERV(NC(IPSB),KCOUNT,MPI_INTEGER,          *MPI
*     *            NC(IPBE),NRECV,NDISP,MPI_INTEGER,KCOMM,MPERR)          *MPI
```

```
          CALL MPI_ALLGATHERV(MPI_IN_PLACE,0,MPI_DATATYPE_NULL,        *MPI
     *        CC(IPBE),NRECV,NDISP,MPI_DOUBLE_PRECISION,KCOMM,MPERR)    *MPI
          CALL MPI_ALLGATHERV(MPI_IN_PLACE,0,MPI_DATATYPE_NULL,        *MPI
     *        NC(IPBE),NRECV,NDISP,MPI_INTEGER,KCOMM,MPERR)             *MPI
```

Scroll down to the lines:

```
*
 1426     IF(KCOMM.NE.MPICN.AND.NPEN.GT.1) THEN
             CALL MPI_GATHER(NTYP(JPVA),JCOUNT,MPI_INTEGER,NTYP(IPLV),    *MPI
     *          JCOUNT,MPI_INTEGER,MROOT,KCOMM,MPERR)                     *MPI
             CALL MPI_GATHER(NBEG(JPVA),JCOUNT,MPI_INTEGER,NBEG(IPLV),    *MPI
     *          JCOUNT,MPI_INTEGER,MROOT,KCOMM,MPERR)                     *MPI
             CALL MPI_GATHER(NEND(JPVA),JCOUNT,MPI_INTEGER,NEND(IPLV),    *MPI
     *          JCOUNT,MPI_INTEGER,MROOT,KCOMM,MPERR)                     *MPI
```

and comment the lines below and add new lines:

```
*THE FOLLOWING BLOCK IS RELATED TO PLOOP OPTIONS 3 AND 4
 1426     IF(KCOMM.NE.MPICN.AND.NPEN.GT.1) THEN
*             CALL MPI_GATHER(NTYP(JPVA),JCOUNT,MPI_INTEGER,NTYP(IPLV),    *MPI
*     *          JCOUNT,MPI_INTEGER,MROOT,KCOMM,MPERR)                     *MPI
*             CALL MPI_GATHER(NBEG(JPVA),JCOUNT,MPI_INTEGER,NBEG(IPLV),    *MPI
*     *          JCOUNT,MPI_INTEGER,MROOT,KCOMM,MPERR)                     *MPI
*             CALL MPI_GATHER(NEND(JPVA),JCOUNT,MPI_INTEGER,NEND(IPLV),    *MPI
*     *          JCOUNT,MPI_INTEGER,MROOT,KCOMM,MPERR)                     *MPI
*
             IF(MRANK.EQ.MROOT) THEN
                CALL MPI_GATHER(MPI_IN_PLACE,0,MPI_DATATYPE_NULL,         *MPI
     *             NTYP(IPLV),JCOUNT,MPI_INTEGER,MROOT,KCOMM,MPERR)       *MPI
                CALL MPI_GATHER(MPI_IN_PLACE,0,MPI_DATATYPE_NULL,         *MPI
     *             NBEG(IPLV),JCOUNT,MPI_INTEGER,MROOT,KCOMM,MPERR)       *MPI
                CALL MPI_GATHER(MPI_IN_PLACE,0,MPI_DATATYPE_NULL,         *MPI
     *             NEND(IPLV),JCOUNT,MPI_INTEGER,MROOT,KCOMM,MPERR)       *MPI
             ELSE
                CALL MPI_GATHER(NTYP(JPVA),JCOUNT,MPI_INTEGER,            *MPI
     *             NTYP(IPLV),JCOUNT,MPI_INTEGER,MROOT,KCOMM,MPERR)       *MPI
                CALL MPI_GATHER(NBEG(JPVA),JCOUNT,MPI_INTEGER,            *MPI
     *             NBEG(IPLV),JCOUNT,MPI_INTEGER,MROOT,KCOMM,MPERR)       *MPI
                CALL MPI_GATHER(NEND(JPVA),JCOUNT,MPI_INTEGER,            *MPI
     *             NEND(IPLV),JCOUNT,MPI_INTEGER,MROOT,KCOMM,MPERR)       *MPI
             ENDIF
```

Scroll down to the lines

```
 3426           CONTINUE
                IPSB = IPVA + JDISP
```

```
                KPVA = JPVA + JJ
                KCOUNT = NEND(KPVA) - NBEG(KPVA) + 1
                CALL MPI_GATHERV(CC(IPSB),KCOUNT,MPI_DOUBLE_PRECISION,    *MPI
     *             CC(IPBE),NRECV,NDISP,MPI_DOUBLE_PRECISION,             *MPI
     *             MROOT,KCOMM,MPERR)                                     *MPI
                CALL MPI_GATHERV(NC(IPSB),KCOUNT,MPI_INTEGER,             *MPI
     *             NC(IPBE),NRECV,NDISP,MPI_INTEGER,                      *MPI
     *             MROOT,KCOMM,MPERR)                                     *MPI
```

and comment out the lines and add new lines to as below:

```
 3426           CONTINUE
                IPSB = IPVA + JDISP
                KPVA = JPVA + JJ
                KCOUNT = NEND(KPVA) - NBEG(KPVA) + 1
*                CALL MPI_GATHERV(CC(IPSB),KCOUNT,MPI_DOUBLE_PRECISION,    *MPI
*     *             CC(IPBE),NRECV,NDISP,MPI_DOUBLE_PRECISION,             *MPI
*     *             MROOT,KCOMM,MPERR)                                     *MPI
*                CALL MPI_GATHERV(NC(IPSB),KCOUNT,MPI_INTEGER,             *MPI
*     *             NC(IPBE),NRECV,NDISP,MPI_INTEGER,                      *MPI
*     *             MROOT,KCOMM,MPERR)                                     *MPI
*
                IF (MRANK.EQ.MROOT) THEN
                   CALL MPI_GATHERV(MPI_IN_PLACE,0,MPI_DATATYPE_NULL,       *MPI
     *                CC(IPBE),NRECV,NDISP,MPI_DOUBLE_PRECISION,             *MPI
     *                MROOT,KCOMM,MPERR)                                     *MPI
                   CALL MPI_GATHERV(MPI_IN_PLACE,0,MPI_DATATYPE_NULL,       *MPI
     *                NC(IPBE),NRECV,NDISP,MPI_INTEGER,                      *MPI
     *                MROOT,KCOMM,MPERR)                                     *MPI
                ELSE
                   CALL MPI_GATHERV(CC(IPSB),KCOUNT,MPI_DOUBLE_PRECISION,    *MPI
     *                CC(IPBE),NRECV,NDISP,MPI_DOUBLE_PRECISION,             *MPI
     *                MROOT,KCOMM,MPERR)                                     *MPI
                   CALL MPI_GATHERV(NC(IPSB),KCOUNT,MPI_INTEGER,             *MPI
     *                NC(IPBE),NRECV,NDISP,MPI_INTEGER,                      *MPI
     *                MROOT,KCOMM,MPERR)                                     *MPI
                ENDIF
```

Scroll down to the lines

```
 5426    IF(KCOMM.NE.MPICN.AND.NPEN.GT.1) THEN
            CALL MPI_SCATTER(NTYP(IPLV),JCOUNT,MPI_INTEGER,NTYP(JPVA),   *MPI
     *         JCOUNT,MPI_INTEGER,MROOT,KCOMM,MPERR)                      *MPI
            CALL MPI_SCATTER(NBEG(IPLV),JCOUNT,MPI_INTEGER,NBEG(JPVA),   *MPI
     *         JCOUNT,MPI_INTEGER,MROOT,KCOMM,MPERR)                      *MPI
            CALL MPI_SCATTER(NEND(IPLV),JCOUNT,MPI_INTEGER,NEND(JPVA),   *MPI
```

```
     *         JCOUNT,MPI_INTEGER,MROOT,KCOMM,MPERR)                    *MPI
*
```

and comment out the lines and add new lines as below:

```
*THE FOLLOWING BLOCK IS RELATED TO PLOOP OPTIONS 5 AND 6
 5426    IF(KCOMM.NE.MPICN.AND.NPEN.GT.1) THEN
*            CALL MPI_SCATTER(NTYP(IPLV),JCOUNT,MPI_INTEGER,NTYP(JPVA),   *MPI
*     *         JCOUNT,MPI_INTEGER,MROOT,KCOMM,MPERR)                    *MPI
*            CALL MPI_SCATTER(NBEG(IPLV),JCOUNT,MPI_INTEGER,NBEG(JPVA),   *MPI
*     *         JCOUNT,MPI_INTEGER,MROOT,KCOMM,MPERR)                    *MPI
*            CALL MPI_SCATTER(NEND(IPLV),JCOUNT,MPI_INTEGER,NEND(JPVA),   *MPI
*     *         JCOUNT,MPI_INTEGER,MROOT,KCOMM,MPERR)                    *MPI
*
           CALL MPI_SCATTER(NTYP(IPLV),JCOUNT,MPI_INTEGER,               *MPI
     *         MPI_IN_PLACE,0,MPI_DATATYPE_NULL,MROOT,KCOMM,MPERR)       *MPI
           CALL MPI_SCATTER(NBEG(IPLV),JCOUNT,MPI_INTEGER,               *MPI
     *         MPI_IN_PLACE,0,MPI_DATATYPE_NULL,MROOT,KCOMM,MPERR)       *MPI
           CALL MPI_SCATTER(NEND(IPLV),JCOUNT,MPI_INTEGER,               *MPI
     *         MPI_IN_PLACE,0,MPI_DATATYPE_NULL,MROOT,KCOMM,MPERR)       *MPI
*
```

Scroll down to the lines

```
 7426          CONTINUE
               IPSB = IPVA + JDISP
               KPVA = JPVA + JJ
               KCOUNT = NEND(KPVA) - NBEG(KPVA) + 1
               CALL MPI_SCATTERV(CC(IPBE),NRECV,NDISP,                   *MPI
     *            MPI_DOUBLE_PRECISION,CC(IPSB),KCOUNT,                  *MPI
     *            MPI_DOUBLE_PRECISION,MROOT,KCOMM,MPERR)                *MPI
               CALL MPI_SCATTERV(NC(IPBE),NRECV,NDISP,                   *MPI
     *            MPI_INTEGER,NC(IPSB),KCOUNT,                           *MPI
     *            MPI_INTEGER,MROOT,KCOMM,MPERR)                         *MPI
```

and comment out the lines and add new lines as below:

```
 7426          CONTINUE
               IPSB = IPVA + JDISP
               KPVA = JPVA + JJ
               KCOUNT = NEND(KPVA) - NBEG(KPVA) + 1
*               CALL MPI_SCATTERV(CC(IPBE),NRECV,NDISP,                   *MPI
*     *            MPI_DOUBLE_PRECISION,CC(IPSB),KCOUNT,                  *MPI
*     *            MPI_DOUBLE_PRECISION,MROOT,KCOMM,MPERR)                *MPI
*               CALL MPI_SCATTERV(NC(IPBE),NRECV,NDISP,                   *MPI
*     *            MPI_INTEGER,NC(IPSB),KCOUNT,                           *MPI
*     *            MPI_INTEGER,MROOT,KCOMM,MPERR)                         *MPI
```

```
*
               CALL MPI_SCATTERV(CC(IPBE),NRECV,NDISP,                    *MPI
     *            MPI_DOUBLE_PRECISION,MPI_IN_PLACE,0,                    *MPI
     *            MPI_DATATYPE_NULL,MROOT,KCOMM,MPERR)                    *MPI
               CALL MPI_SCATTERV(NC(IPBE),NRECV,NDISP,                    *MPI
     *            MPI_INTEGER,MPI_IN_PLACE,0,                             *MPI
     *            MPI_DATATYPE_NULL,MROOT,KCOMM,MPERR)                    *MPI
*
```

Scroll down to

```
*
 9426    IF(KCOMM.NE.MPICN.AND.NPEN.GT.1) THEN
```

and add the comment above

```
*THE FOLLOWING BLOCK IS RELATED TO PLOOP OPTIONS 7 AND 8
 9426    IF(KCOMM.NE.MPICN.AND.NPEN.GT.1) THEN
```

Press escape to exit insert mode. Next type : then /SUBROUTINE INPUT. Press i to enter insert mode and edit the snippet

```
      CHARACTER A*160
```

to

```
      CHARACTER A*160, SR
      INTEGER IERR, IRANK
      INCLUDE 'mpif.h'
```

Next, press escape to exit insert mode and type : then /ILAST(A,1,79). Find the second occurrence. Press i to enter insert mode and change

```
      IL = ILAST(A,1,79)
      OPEN(1,FILE=A(1:IL)//'.fox',STATUS='OLD',ERR=5)
      OPEN(2,FILE=A(1:IL)//'.lis',STATUS='UNKNOWN',ERR=5)
```

to

```
      IL = ILAST(A,1,79)
      CALL MPI_COMM_GROUP(MPI_COMM_WORLD,MGRO,IERR)                       *MPI
      CALL MPI_GROUP_RANK(MGRO,IRANK,IERR)                                *MPI
      WRITE(SR,'(I1)') IRANK
      OPEN(1,FILE=A(1:IL)//'.fox',STATUS='OLD',ERR=5)
      OPEN(2,FILE=A(1:IL)//SR//'.lis',STATUS='UNKNOWN',ERR=5)
```

Now press ESC. Then : w to save and :q to quit.

B.3.7 Edit mpi-foxgraf.f

We need to edit mpi-foxgraf.f. This can be done via vim by typing
$ vim mpi−foxy.f
First type : to bring up the command line in vim. Use the command below to do a "Find and replace".

```
:%s/PARAMETER(LMEM=140000000,LVAR=10000000,LDIM=1000)/PARAMETER(LMEM=100000000,LVAR=10000000,LDIM=1000)/g
```

Type : w to save and :q to quit.

B.3.8 Edit mpi-dafox.f and build COSY

Now we need to edit mpi-dafox.f by running $ vim mpi−dafox.f Use the command below to do a "Find and replace".

```
:%s/PARAMETER(LMEM=140000000,LVAR=10000000,LDIM=1000)/PARAMETER(LMEM=100000000,LVAR=10000000,LDIM=1000)/g
```

Type : followed by / SLEEPQQ this should take to the following snippet of code.

```
*     SLEEPQQ is an Intel Fortran specific, platform independent call.
*
      CALL SLEEPQQ(NINT(CC(NBEG(IMSEC))))
*
*     SLEEP is not compiler specific and platform independent,
*     but only does full seconds, not milliseconds
*
C     CALL SLEEP(NINT(1.D-3*CC(NBEG(IMSEC))))
*
```

Now, press i to enter insert mode and edit the file so that the C is in front of `CALL SLEEPQQ(NINT(CC(NBEG(IMSEC))))` and the C before `CALL SLEEP(NINT(1.D-3*CC(NBEG(IMSEC))))` is removed:

```
*     SLEEPQQ is an Intel Fortran specific, platform independent call.
*
C     CALL SLEEPQQ(NINT(CC(NBEG(IMSEC))))
*
*     SLEEP is not compiler specific and platform independent,
*     but only does full seconds, not milliseconds
*
      CALL SLEEP(NINT(1.D-3*CC(NBEG(IMSEC))))
*
```

Be sure that the spacing matches. Type ESC to exit insert mode followed by : w and : q after you have made this change.
Now run
$ make
to build COSY.

B.3.9 Generate COSY.bin

Some programs require the COSY.bin bin file. To create one, run
$ cosy cosy.fox
When a fox file contains the code `INCLUDE COSY` you need to have a copy of COSY.bin in that folder.

B.3.10 Adding COSY to your path

In this section, we describe how to allow COSY to run in directories outside of the COSY directory. First, use the command
$ **cd** ..
to return to your home directory. Type the command
$ ls
to see if you have a directory called bin. If not, use the command
$ mkdir bin
to create it. Then copy the cosy binary we compiled to this directory with the command
$ cp /home/username/COSY/cosy /home/username/bin/
Last, edit the file .bashrc with the command
$ vim .bashrc
add the line

```
export PATH="/home/username/bin/:$PATH"
```

at the bottom of the file (Use insert mode, : w to write, : q to quit).

B.4 Running a fox script on Gaea

Use the command
$ mpirun −n <number of processors> cosy file.fox
where <number of processors>is the number of nodes you want to use on Gaea.