# OpenSees lefordítása a Superman-re

## Előzetes:

Az eredeti fordítási útmutató elérhető ezen a linken:
http://opensees.berkeley.edu/wiki/index.php/Compilation_Guideline_of_OpenSeeMP_on_Linux_Machines

Az alábbi írást kifejezetten a BME szuperszámítógépére, a Superman-re történő fordításhoz írtam.

*Dőlt* betűvel jelöltem a könyvtár és fájl neveket, míg a parancsokat illetve a fájlok tartalmát `így`.

Pár *nix alapparancs amiket használni fogunk:

- `cd`
- `mkdir`
- `chmod`
- `make`
- *cp*
- `nano`

Ha bármelyik paranccsal problémánk akad, akkor használjuk a hozzá tartozó útmutatót, amit a `man` utasítással érhetünk el. Pl.: `man mkdir`.

Könnyebb fájlkezeléshez használhatjuk a Midnight Commander-t is, amit az `mc` paranccsal tudunk előhívni. Egér kattintást is elfogad inputként.

A könyvtárstruktúra amit használok, így néz ki:

➔ *home*

- ➔ *bin*: futtatható bináris állományok
- ➔ *lib*: program által használt modulok
- ➔ *PARALLEL*: letöltött csomagok mappákba rendezve
- ➔ *OpenSees*: svn-ről letöltött forráskód

Hozzuk is létre őket az `mkdir` paranccsal.

Adjuk hozzá őket a környezeti változókhoz:

```
chmod 755 ~/.bash_profile
```

```
echo "export PATH=\$PATH:\$HOME/bin" >> ~/.bash_profile
```

```
echo "export PATH=\$PATH:\$HOME/lib" >> ~/.bash_profile
```

```
echo "export PARALLEL=\$HOME/PARALLEL" >> ~/.bash_profile
```

Tipp: ha a `make` parancs után használjuk a `-j` kapcsolót, akkor megadhatjuk, hogy mennyi szálon fusson a fordítás (Néha megakadhat, ilyenkor elég újból elindítani és folytatja tovább. Előfordulhat, hogy egy hosszabb fordítás során ez többször is előfordul. Ha végképp nem megy, akkor szimplán hagyjuk el.)

# 1. Csomagok letöltése a célszámítógépre.

- Windows alól használhatjuk a WinSCP-t: http://winscp.net/eng/docs/lang:hu
- *nix rendszerek alól terminálból az `scp` parancsot, pl:
  - `scp -i /Path/To/privateKey /Path/To/file username@host:~/Path/file`
- Közvetlenül a célszámítógépre is letölthetjük a fájlokat a wget paranccsal, pl:
  - `wget url_to_file`

A csomagok:

- ActiveTcl: http://downloads.activestate.com/ActiveTcl/releases/
- OracleDB: Oracle honlapjáró(regisztrálni kell)
- Lapack: http://www.netlib.org/lapack/
- Metis: http://glaros.dtc.umn.edu/gkhome/metis/metis/download
- Xblas: http://www.netlib.org/xblas/
- Scalapack: http://www.netlib.org/scalapack/
- ParMetis: http://glaros.dtc.umn.edu/gkhome/metis/parmetis/odownlad
- OpenSSL: https://www.openssl.org/source/
- MUMPS: http://ftp.mcs.anl.gov/pub/petsc/externalpackages/
- (Az eredeti honlapon a fejlesztőtől kell elkérni a kódot.)
- Mpich: http://www.mpich.org/static/downloads/
- Blacs: ftp://164.41.45.2/pub/netlib/blacs/
  - *mpiblacs.tgz*: az ebben lévő fájlokat felül kell írni az mpiblacs-patch03.tgz-ben találhatóakkal
  - *mpiblacs-patch03.tgz*
  - egyéb: http://www.netlib.org/blacs/blacs_install.ps

A legújabb csomagokat használtam(2016. január), kivéve:

- mpich2: 2.1.1, mert az újban lecserélték az mpd-t
- mumps: 4.10.0, ettől feljebb kompatibilitási problémák voltak
- parmetis: 3.2.0, ettől feljebb kompatibilitási problémák voltak
- OpenSees: 2.4.6 (rev 6123), újabbal nem teszteltem

Kicsomagoltam őket a *PARALLEL* könyvtárba és egyszerűsítettem a nevüket az alábbiak szerint:

- *blacs*
- *mpich2*
- *lapack*
- *scalapack*
- *blas*
- *xblas*
- *metis*
- *parmetis*
- *mumps*

## 2. OpenSees letöltése

Adjuk ki az alábbi parancsot a *home* könyvtárban:

```
svn co -r 5540 svn://peera.berkeley.edu/usr/local/svn/OpenSees/trunk OpenSees
```

Ha a legújabb verzióra lenne szükség, akkor ezt kell használni:

```
svn co svn://peera.berkeley.edu/usr/local/svn/OpenSees/trunk OpenSees
```

## 3. Csomagok lefordítása/telepítése

Itt megjegyezném, hogy a Superman-hez (jó esetben) nem kapunk root(rendszergazdai) hozzáférést, ezért minden amit csinálunk a saját (home) könyvtárunkon belül fog történni beleértve a telepítéseket is.

Minden csomagnál a kiindulási mappa a csomag saját mappája a *PARALLEL* könyvtáron belül.

Egyes csomagoknál vannak tesztek is. Ezekből néhány lefut automatikusan fordítás közben míg a többit manuálisan lehet elindítani. Ezekre nem fogok kitérni, mert az interneten is vannak hozzá leírások.

A szerkesztésnél figyeljünk oda, hogy a használni kívánt sor elején ne legyen kettőskereszt(#), mert azokat a sorokat nem veszi figyelembe a fordító. Értelemszerűen, ha több azonos nevű változó van a fájlban(amiknek értéket adunk, pl: CC = ….), akkor csak az egyik legyen használatban a többi előtt maradjon #.

## ActiveTcl:

Mindegy, hogy hova csomagoljuk ki a telepítésnél úgyis beállítjuk a célmappánkat.

A könyvtárban lévő *install.sh* script segítségével feltelepítettem a *~/bin/* könyvtárba. A telepítés helyét a script futtatása közben fogja kérni.

## Mpich:

A következő parancsoknál csak akkor kell a `| tee …` , ha menteni szeretnénk a parancs outputját. A configure utáni `--prefix`-el a telepítés helyét állítjuk be. Itt nem használhatjuk a ~ jelet, ezért a home mappánk teljes elérési útvonalát meg kell adnunk.

```
./configure --prefix=/path/to/home/PARALLEL/mpich2-install 2>&1 | tee c.txt
```

```
make 2>&1 | tee m.txt
```

```
make install 2>&1 | tee mi.txt
```

```
echo "export PATH=\$HOME/PARALLEL/mpich2-install/bin:\$PATH" >> ~/.bash_profile
```

Utána ellenőrzésképp lefuttathatjuk az alábbi parancsokat:

```
which mpd
```

```
which mpiexec
```

```
which mpirun
```

Ha megtalálta mindet, akkor rendben van.

Egyből hozzá is adhatjuk a környezeti változókhoz:

```
echo "export PATH=\$PATH:\$HOME/PARALLEL/mpich2-install/bin" >> ~/.bash_profile
```

```
echo "export MPIdir=\$HOME/PARALLEL/mpich2-install" >> ~/.bash_profile
```

```
echo "export MPI_BIN=\$HOME/PARALLEL/mpich2-install/bin" >> ~/.bash_profile
```

Létre kell hoznunk a *home* mappában egy *.mpd.conf* nevű fájlt, majd beleírni egy jelszót, amit csak mi ismerünk:

```
touch ~/.mpd.conf
```

```
echo "secretword=<jelszo>" > ~/.mpd.conf
```

```
chmod 600 ~/.mpd.conf
```

A legutolsó parancs azért kellett, mert így csak mi tudjuk írni és olvasni.

Ellenőrzésként:

`mpd &` → elindítja az mpich daemont

`mpdtrace` → a számítógép host nevét kell, hogy kiírja, ami esetünkben "superman"

`mpdallexit` → leállít minden mpich daemont

(`ps aux | grep mpd` paranccsal le lehet ellenőrizni, hogy biztosan leállt e)

## Blas:

*make.inc* fájl szerkesztése:

```
PLAT =

FORTRAN  = $(MPI_BIN)/mpif90

LOADER   = $(MPI_BIN)/mpif90
```

Ha kész adjuk ki a `make` parancsot.

## Xblas:

`./configure`

*make.inc*-ben írjuk át:

```
CC =$(MPI_BIN)/mpicc

EXTRA_LIBS =  -lm
```

Ha kész, akkor mehet a `make` parancs.

Ha a fordító nem találja a *libm.a* fájlt, akkor nekünk kell megadni a pontos elérési útvonalát a *make.inc*-ben az -lm helyén.

## Lapack:

*make.inc.example* file átnevezése *make.inc*-re, majd szerkeszteni:

```
FORTRAN = $(MPI_BIN)/mpif90

LOADER  = $(MPI_BIN)/mpif90

USEXBLAS        = Yes  (ha van, akkor ki kell törölni a '#' jelet a sor elejéről)

XBLASLIB        =$(PARALLEL)/xblas/libxblas.a

BLASLIB = $(PARALLEL)/blas/blas.a
```

Ha kész, mehet a `make`.

## Blacs:

Menjünk be az *INSTALL/EXE* mappába. Itt adjuk ki a `make xintface` parancsot. Ha nem sikerülne, akkor lépjünk vissza az *INSTALL*-ba majd itt próbáljuk meg `make xintface`. Sikeres `make` esetén az *EXE* mappában létre kellett jönnie egy *xintface* fájlnak. Futtassuk `./xintface`, majd jegyezzük meg amit kiír.(A Superman esetén ez INTFACE = -Dadd_ lesz valószínűleg.)

Váltsunk vissza a *blacs* mappába.

A *BMAKES* könyvtárból másoljuk ki a *Bmake.MPI-LINUX* fájlt a *blacs* könyvtárába, nevezzük át *Bmake.inc*-re, majd szerkesszük át az alábbiakat:

```
COMMLIB =

PLAT =

BTOPdir =$(PARALLEL)/blacs

MPIdir =$(PARALLEL)/mpich2-install

MPILIBdir = $(MPIdir)/mpich2-install/lib/

MPIINCdir = $(MPIdir)/include

MPILIB = $(MPIdir)/lib/libmpich.a

SYSINC = -I$(MPIINCdir)

INTFACE = -DAdd_

TRANSCOMM = -DuseMpich

WHATMPI =

SYSERRORS =

F77 = $(MPI_BIN)/mpif77

CC = $(MPI_BIN)mpicc

CCLOADER = $(MPI_BIN)/mpicc
```

Amennyiben mindent módosítottunk adjuk ki a `make MPI` parancsot.

Ha rendben lefordult, akkor menjünk be a *LIB* könyvtárba. Ha van olyan fájl amiben szerepel

"_--*0*" akkor annak hozzuk létre a postfix mentes változatát:

pl: `cp blacs_--0.a blacs.a`

Erre a mumps csomag fordításánál lesz szükségünk.

## Scalapack:

Módosítsuk az *Slmake.inc*-et:

```
CDEFS = -Dadd_  (ha mást kaptunk a xintface-től akkor azt írjuk be ide is)

FC = $(MPI_BIN)/mpif90

CC = $(MPI_BIN)/mpicc

BLASLIB = $(PARALLEL)/blas/blas.a

LAPACKLIB = $(PARALLEL)/lapack/liblapack.a
```

## Metis:

Itat a *Makefile*-t kell szerkesztenünk:

```
CC = $(MPI_BIN)/mpicc
```

Ezután `make config`, majd `make`.

## Parmetis:

Írjuk át a *Makefile.in*-ben az alábbiakat:

```
CC = $(MPI_BIN)/mpicc

INCDIR  = $(MPIdir)/include

LD = $(MPI_BIN)/mpicc
```

Majd `make config` és `make`.

Ha kész akkor adjuk hozzá a parmetis két mappáját a környezeti változókhoz:

```
echo "export PATH=\$PATH:/home/1/psha01/PARALLEL/parmetis" >> ~/.bash_profile

echo "export PATH=\$PATH:/home/1/psha01/PARALLEL/parmetis/include" >>
~/.bash_profile
```

## Mumps:

*Make.inc* mappából másoljuk ki a *Makefile.inc.generic* fájlt a mumps könyvtárába *Makefile.inc* néven, majd szerkesszük (figyeljünk a sor eleji # jelekre, amikről fentebb írtam!):

```
LMETISDIR = $(PARALLEL)/parmetis

IMETIS = -I$(LMETISDIR)

LMETIS = -L$(LMETISDIR) -lparmetis -lmetis

ORDERINGSF  = -Dmetis -Dpord -Dparmetis

PLAT    =

CC      = $(MPI_BIN)/mpicc

FC      = $(MPI_BIN)/mpif90

FL      = $(MPI_BIN)/mpif90

SCALAP  = -L$(PARALLEL)/scalapack -lscalapack \

       $(PARALLEL)/blacs/LIB/blacs.a \

       $(PARALLEL)/blacs/LIB/blacsCinit.a \

       $(PARALLEL)/blacs/LIB/blacs.a \

       $(PARALLEL)/blacs/LIB/blacsF77init.a\

       $(PARALLEL)/blacs/LIB/blacs.a \

       $(PARALLEL)/lapack/liblapack.a

INCPAR  = -I$(MPIdir)/include

LIBPAR  = $(SCALAP) \

          -L$(MPIdir)/lib

LIBBLAS = $(PARALLEL)/blas/blas.a

CDEFS = -Dadd_  (ha a blacs fordításnál a xintface más értéket adott, akkor az kell ide)
```

Miután mindent módosítottunk fordítsuk le az alábbi parancsokkal:

```
make

make d

make c

make s

make z
```

## BerkeleyDB5.3:

Lépjünk be a *build_unix* könyvtárba, majd adjuk ezeket a parancsokat:

```
mkdir ~/bin/BerkeleyDB
```

```
../dist/configure --prefix=/path/to/home/bin/BerkeleyDB
```

```
make
```

```
make install
```

## OpenSSL:

```
mkdir ~/bin/openssl
```

Itt is állítsuk be a telepítési mappát az alábbi paranccsal:

```
./config --prefix=/path/to/home/bin/openssl
```

```
make
```

```
make install
```

# 4. Az OpenSees fordítása

Hozzunk létre egy új *Makefile.def*-et a könyvtárban az alábbi tartalommal, majd szerkesszük ott, ahol írva van. Ha mindent a leírtak alapján csináltunk, akkor a Superman-en elméletileg csak a home mappa elérési útvonalán kell változtatni a fájlban, de úgy gondolom, hogy azért nem árt átnézni. Főleg ott ahol jelezeve van.

Ha bármilyen CSPARSE hibát kapnánk, akkor módosítsuk az alábbi fórum hozzászólás szerint:

http://opensees.berkeley.edu/community/viewtopic.php?f=4&t=55945#p99174

```
#############################################################################
#
#  Program:  OpenSees
#
#  Purpose:  A Top-level Makefile to create the libraries needed
#         to use the OpenSees framework. Works on Linux version 6.1
#            and below.
#
#  Written: fmk
```

```
#  Created: 10/99
#
#  Send bug reports, comments or suggestions to fmckenna@ce.berkeley.edu
#
#########################################################################
# CHANGE THIS SECTION AND HAVE A LOOK AT THE SECTION HAVING
#!!!!!!!!!!!!!USE MACHINES SPECIFIC LOCATIONS!!!!!!!!!!!!! MARK
#########################################################################
HOME = /path/to/home # SAJAT HOME MAPPA ELERESI UTVONALA
TCLdir = $(HOME)/bin/ActiveTcl-8.5 # HA MAS TCL VERZIOT HASZNALUNK FIGYELJUNK
BERKLEYDir=$(HOME)/bin/BerkeleyDB
MPIdir = $(HOME)/PARALLEL/mpich2-install


#########################################################################
PARALLELdir=$(HOME)/PARALLEL
##TCL
TCL_BIN = $(TCLdir)/bin
TCL_INC = $(TCLdir)/include
TCL_LIB = -L$(TCLdir)/lib -ltcl8.5 -ltk8.5 # FIGYELJUNK A VERZIOSZAMRA
#########################################################################
######


# %---------------------------------%
# |  SECTION 1: PROGRAM             |
# %---------------------------------%
#
# Specify the location and name of the OpenSees interpreter program
# that will be created (if this all works!)
#PROGRAMMING_MODE = SEQUENTIAL
```

```
#PROGRAMMING_MODE = PARALLEL
PROGRAMMING_MODE = PARALLEL_INTERPRETERS

OpenSees_PROGRAM = $(HOME)/bin/OpenSees

ifeq ($(PROGRAMMING_MODE), PARALLEL)
OpenSees_PROGRAM = $(HOME)/bin/OpenSeesSP
endif
ifeq ($(PROGRAMMING_MODE), PARALLEL_INTERPRETERS)
OpenSees_PROGRAM = $(HOME)/bin/OpenSeesMP
endif

# %---------------------------------%
# |  SECTION 2: MAKEFILE CONSTANTS  |
# %---------------------------------%
#
# Specify the constants the are used as control structure variables in the
Makefiles.

OPERATING_SYSTEM = LINUX
#DEBUG_MODE = DEBUG, NO_DEBUG
DEBUG_MODE = NO_DEBUG
#RELIABILITY = YES_RELIABILITY
RELIABILITY = NO_RELIABILITY
GRAPHICS = NONE

# %---------------------------------%
# |  SECTION 3: PATHS               |
# %---------------------------------%
```

```
#
# Note: if vendor supplied BLAS and LAPACK libraries or if you have
# any of the libraries already leave the directory location blank AND
# remove the directory from DIRS.



FE   = $(HOME)/OpenSees/SRC
AMDdir         = $(HOME)/OpenSees/OTHER/AMD
BLASdir      =
CBLASdir     = $(HOME)/OpenSees/OTHER/CBLAS
LAPACKdir    =
SUPERLUdir   = $(HOME)/OpenSees/OTHER/SuperLU_4.1/SRC
ARPACKdir    = $(HOME)/OpenSees/OTHER/ARPACK
UMFPACKdir   = $(HOME)/OpenSees/OTHER/UMFPACK
METISdir       =
CSPARSEdir   =$(HOME)/OpenSees/OTHER/CSPARSE
ITPACKdir = $(HOME)/OpenSees/OTHER/ITPACK
SUPERLU_DISTdir   = $(HOME)/OpenSees/OTHER/SuperLU_DIST_2.5/SRC


DIRS        = $(AMDdir) $(CBLASdir) $(ITPACKdir) $(CSPARSEdir)\
     $(SUPERLUdir) $(SUPERLU_DISTdir) $(ARPACKdir) $(UMFPACKdir) $(FE)
#DIRS        = $(AMDdir) $(BLASdir) $(CBLASdir) $(LAPACKdir)  $(ITPACKdir)\
#    $(SUPERLUdir) $(SUPERLU_DISTdir) $(ARPACKdir) $(UMFPACKdir) $(METISdir) $
(FE)
# %----------------------------------------------------%
# | SECTION 4: LIBRARIES                               |
# |                                                    |
# | The following section defines the libraries that will |
# | be created and/or linked with when the libraries are  |
```

```
# | being created or linked with.                          |
# %--------------------------------------------------------%
#
# Note: if vendor supplied BLAS and LAPACK libraries leave the
# libraries blank. You have to get your own copy of the tcl/tk
# library!!
#
# Note: For libraries that will be created (any in DIRS above)
# make sure the directory exsists where you want the library to go!
#Dir definition



FE_LIBRARY      = $(HOME)/lib/libOpenSees.a
NDARRAY_LIBRARY = $(HOME)/lib/libndarray.a # BJ_UCD jeremic@ucdavis.edu
MATMOD_LIBRARY  = $(HOME)/lib/libmatmod.a  # BJ_UCD jeremic@ucdavis.edu
BJMISC_LIBRARY  = $(HOME)/lib/libBJmisc.a  # BJ_UCD jeremic@ucdavis.edu
LAPACK_LIBRARY  = $(PARALLELdir)/lapack/liblapack.a #!!!!!!!!!!!!!!!!!!!
BLAS_LIBRARY    = $(PARALLELdir)/blas/blas.a #!!!!!!!!!!!!!!!!!!!!!!!!
SUPERLU_LIBRARY = $(HOME)/lib/libSuperLU.a
CBLAS_LIBRARY   = $(HOME)/lib/libCBlas.a
ARPACK_LIBRARY  = $(HOME)/lib/libArpack.a
AMD_LIBRARY  = $(HOME)/lib/libAMD.a
UMFPACK_LIBRARY = $(HOME)/lib/libUmfpack.a
ITPACK_LIBRARY = $(HOME)/lib/libItpack.a
METIS_LIBRARY   = $(PARALLELdir)/parmetis/libparmetis.a #PARMETIS USED
CSPARSE_LIBRARY = $(HOME)/lib/libCSparse.a
#METIS_LIBRARY   = $(HOME)/lib/libMetis.a
TCL_LIBRARY = $(TCLdir)/lib/libtcl8.5.a #TCL VERZIOSZAM!!!!!
```

```
BLITZ_LIBRARY =
# $(HOME)/blitz/lib/libblitz.a#

DISTRIBUTED_SUPERLU_LIBRARY = $(HOME)/lib/libDistributedSuperLU.a
GRAPHIC_LIBRARY     =

ifeq ($(RELIABILITY), YES_RELIABILITY)
RELIABILITY_LIBRARY = $(HOME)/lib/libReliability.a
else
RELIABILITY_LIBRARY =
endif

# WATCH OUT .. These libraries are removed when 'make wipe' is invoked.
WIPE_LIBS = $(FE_LIBRARY) \
          $(CBLAS_LIBRARY) \
          $(SUPERLU_LIBRARY) \
          $(DISTRIBUTED_SUPERLU_LIBRARY) \
          $(ARPACK_LIBRARY) \
          $(UMFPACK_LIBRARY) \
               $(NDARRAY_LIBRARY) \
          $(MATMOD_LIBRARY) \
          $(ITPACK_LIBRARY)\
          $(AMD_LIBRARY)

# %-----------------------------------------------------------%
# | SECTION 5: COMPILERS                                      |
# |                                                           |
# | The following macros specify compilers, linker/loaders, |
```

```
# | the archiver, and their options.  You need to make sure |
# | these are correct for your system.                       |
# %----------------------------------------------------------%

# ######################################################
# # Compilers
# ######################################################

MPI_BIN = $(MPIdir)/bin
MPI_INC = -I$(MPIdir)/include
MPI_LIB = -L$(MPIdir)/lib

CC++            = $(MPI_BIN)/mpicxx
CC              = $(MPI_BIN)/mpicc
FC              = $(MPI_BIN)/mpif90
FC90            = $(MPI_BIN)/mpif90
FC77      = $(MPI_BIN)/mpif90
FORTRAN         = $(FC)

LINKER          =  $(CC++)

#FORTRAN  = /usr/bin/gfortran
#CC++     = /usr/bin/g++
#CC      = /usr/bin/gcc
#FC  = /usr/bin/gfortran
#FORTRAN  = /usr/bin/gfortran
#LINKER   = /usr/bin/mpicxx
AR        = ar
```

```
ARFLAGS        = cqls
RANLIB         = ranlib
RANLIBFLAGS     =

GRAPHIC_FLAG = -D_NOGRAPHICS
PROGRAMMING_FLAG =

ifeq ($(PROGRAMMING_MODE), PARALLEL)
PROGRAMMING_FLAG = -D_PARALLEL_PROCESSING
endif

ifeq ($(PROGRAMMING_MODE), PARALLEL_INTERPRETERS)
PROGRAMMING_FLAG = -D_PARALLEL_INTERPRETERS
endif

#RELIABILITY_FLAG = -D_RELIABILITY
RELIABILITY_FLAG =
ifeq ($(RELIABILITY), YES_RELIABILITY)
RELIABILITY_FLAG = -D_RELIABILITY
else
RELIABILITY_FLAG =
endif

#DEBUG_FLAG = -D_G3DEBUG
#DEBUG_FLAG = -g -p -pg
#DEBUG_FLAG = -p -g
DEBUG_FLAG =
ifeq ($(DEBUG_MODE), DEBUG)
```

```
DEBUG_FLAG = -D_G3DEBUG
else
DEBUG_FLAG =
endif
MUMPS_FLAG =

PETSC_FLAG =

OPT_FLAG = -O2
#OPT_FLAG = -O0

#COMP_FLAG = -DMPICH_IGNORE_CXX_SEEK
COMP_FLAG =

C++FLAGS  = $(GRAPHIC_FLAG) $(RELIABILITY_FLAG) $(DEBUG_FLAG) $(OPT_FLAG) $
(COMP_FLAG) $(PROGRAMMING_FLAG)  $(PETSC_FLAG) $(MUMPS_FLAG) -D_TCL85 -D_BLAS
-D_LINUX -D_UNIX

CFLAGS          =  $(GRAPHIC_FLAG) $(RELIABILITY_FLAG) $(DEBUG_FLAG)$
(PROGRAMMING_FLAG) $(OPT_FLAG) $(COMP_FLAG) -D_TCL85 -D_BLAS
FFLAGS          =  $(OPT_FLAG) $(COMP_FLAG)
LINKFLAGS       = -Wl,-rpath

#C++FLAGS         = -D_LINUX -D_UNIX  $(GRAPHIC_FLAG) $(RELIABILITY_FLAG) $
(DEBUG_FLAG) $(OPT_FLAG) $(COMP_FLAG)\
          #$(PROGRAMMING_FLAG)  $(PETSC_FLAG) $(MUMPS_FLAG) \
          #-D_TCL85 -D_BLAS

#CFLAGS          =  $(GRAPHIC_FLAG) $(RELIABILITY_FLAG) $(DEBUG_FLAG)$
(PROGRAMMING_FLAG) $(OPT_FLAG) $(COMP_FLAG) -D_TCL85 -D_BLAS
```

```
#FFLAGS          =  $(OPT_FLAG) $(COMP_FLAG)
#LINKFLAGS       = -Wl,-rpath




# Misc
MAKE      = make
CD              = cd
ECHO            = echo
RM              = rm
RMFLAGS         = -f
SHELL           = /bin/sh


# %---------------------------------------------------------%
# | SECTION 6: COMPILATION                                  |
# |                                                         |
# | The following macros specify the macros used in         |
# | to compile the source code into object code.            |
# %---------------------------------------------------------%


.SUFFIXES:
.SUFFIXES:     .C .c .f .f77 .f90 .cpp .o .cpp


#
# %------------------%
# | Default command. |
# %------------------%
#
```

```
.DEFAULT:
     @$(ECHO) "Unknown target $@, try:  make help"
#
# %--------------------------------------------%
# |  Command to build .o files from source files. |
# %--------------------------------------------%
.cpp.o:
     @$(ECHO) Making $@ from $< $@ with $(CC++) $(C++FLAGS) $(INCLUDES) -c $<
     @$(CC++) $(C++FLAGS) $(INCLUDES) -c $<



.C.o:
     @$(ECHO) Making $@ from $<
     $(CC++) $(C++FLAGS) $(INCLUDES) -c $<


.c.o:
     @$(ECHO) Making $@ from $<
     $(CC) $(CFLAGS) -c  $<


.f.o:
     @$(ECHO) Making $@ from $<
     $(FC) $(FFLAGS) -c  $<


.f77.o:
     @$(ECHO) Making $@ from $<
     $(FC77) $(FFLAGS) -c $<


.f90.o:
```

```
	@$(ECHO) Making $@ from $<
	$(FC90) $(FFLAGS) -c $<


# %---------------------------------------------------------%
# | SECTION 7: OTHER LIBRARIES                              |
# |                                                         |
# | The following macros specify other libraries that must  |
# | be linked with when creating executables. These are     |
# | platform specific and typically order does matter!!     |
# %---------------------------------------------------------%
################### USER SPECIFIC LOCATIONS ######################
MACHINE_LINKLIBS  = $(MPI_LIB)\
-L$(BASE)/lib \
-L$(HOME)/lib \
-L/usr/lib64 \
-L/usr/lib64  -ldl -lm \
/usr/lib/gcc/x86_64-redhat-linux/4.4.4/libgfortran.a # RENDSZERENKENT MAS
LEHET!!!
# PETSC
HAVEPETSC = NO
PETSCINC =
PETSC_LIB =

ifeq ($(PROGRAMMMING_MODE), SEQUENTIAL)
HAVEPETSC = NO
endif



ifeq ($(HAVEPETSC), YES)
```

```
PETSC = YES
PETSC_FLAG = -D_PETSC
PETSC_DIR = $(HOME)/OpenSees/parlibs/petsc-3.3-p2/1#customized
PETSC_ARCH = $(HOME)/OpenSees/parlibs/petsc-3.3-p2/arch-linux2-c-
debug#customized
BOPT = O

PETSC_INC = -I$(PETSC_DIR)/include \
          -I$(PETSC_ARCH)/include \



# -D_PETSC
#

PETSC_LIB = $(FE)/system_of_eqn/linearSOE/petsc/PetscSOE.o \
     $(FE)/system_of_eqn/linearSOE/petsc/PetscSolver.o \
     $(FE)/system_of_eqn/linearSOE/petsc/PetscSparseSeqSolver.o \
     -L$(PETSC_DIR)/lib \
#-L/usr/X11/lib -lX11 -lGL
endif



HAVEMUMPS = YES
MUMPS_INCLUDE =
MUMPS_LIB =



ifeq ($(PROGRAMMMING_MODE), SEQUENTIAL)
```

```
HAVEMUMPS = NO
endif

ifeq ($(HAVEMUMPS), YES)

MUMPS = YES
MUMPS_FLAG = -D_MUMPS
#BLAS
BLASdir = $(PARALLELdir)/blas
BLAS_LIB = $(BLASdir)/blas.a
#XBLAS
XBLAS_LIB = $(PARALLELdir)/xblas/libxblas.a
#SCALAPACK
SCALAPACK_LIB = $(PARALLELdir)/scalapack/libscalapack.a
#BLACS
BLACSdir = $(PARALLELdir)/blacs
BLACS_LIB = -L$(BLACSdir)/LIB

#LAPACK
LAPACKdir = $(PARALLELdir)/lapack
LAPACK_LIB = -L$(LAPACKdir)
#MUMPS
MUMPSdir = $(PARALLELdir)/mumps
MUMPSLIB = $(MUMPSdir)/lib
MUMPS_INC = $(MUMPSdir)/include
#ParMETIS
parmetis_lib = $(HOME)/PARALLEL/parmetis/libparmetis.a
metis_lib = $(HOME)/PARALLEL/parmetis/libmetis.a
```

```
metis_inc = -I$(HOME)/PARALLEL/parmetis


SCALAP  = $(SCALAPACK_LIB) \
     $(BLACSdir)/LIB/blacsF77init.a \
     $(BLACSdir)/LIB/blacs.a \
     $(BLACSdir)/LIB/blacsCinit.a \
     $(LAPACKdir)/liblapack.a \
     $(XBLAS_LIB) \
     $(BLASdir)/blas.a


#-L/usr/lib/x86_64-linux-gnu/gcc/x86_64-linux-gnu/4.5/libgfortran.a $(BLACS_LIB)
$(BLAS_LIB)
#$(LAPACK_LIB) $(BLACS_LIB) $(BLAS_LIB) $(BLACS_LIB)


PLAT = LINUX
MUMPS_LIB = $(FE)/system_of_eqn/linearSOE/mumps/MumpsSOE.o \
     $(FE)/system_of_eqn/linearSOE/mumps/MumpsSolver.o \
     $(FE)/system_of_eqn/linearSOE/mumps/MumpsParallelSOE.o \
     $(FE)/system_of_eqn/linearSOE/mumps/MumpsParallelSolver.o \
     $(SCALAP) \
     -L$(MUMPSdir)/lib -lcmumps -ldmumps -lsmumps -lzmumps \
     -lmumps_common -lpord \
     $(parmetis_lib) $(metis_lib) \
     $(SCALAP) \
     /usr/lib/gcc/x86_64-redhat-linux/4.4.4/libgfortran.a \ # RENDSZERENKENT MAS
LEHET!!!!
     /usr/lib64/libdl.so


MUMPS_INCLUDE = -I$(MUMPS_INC)
```

```
endif


PARALLEL_LIB =
HPM_LIB =



MACHINE_NUMERICAL_LIBS  = /usr/lib/gcc/x86_64-redhat-linux/4.4.4/libgfortran.a \
# RENDSZERENKENT MAS LEHET!!!!
     -L/usr/lib64  -ldl -lm\
     $(SUPERLU_LIBRARY) \
     $(UMFPACK_LIBRARY) $(CSPARSE_LIBRARY)\
     $(ARPACK_LIBRARY) \
     $(AMD_LIBRARY) \
     $(GRAPHIC_LIBRARY)\
     $(RELIABILITY_LIBRARY) \
     $(DISTRIBUTED_SUPERLU_LIBRARY) \
     $(METIS_LIBRARY) $(PARALLEL_LIB) $(DISTRIBUTED_SUPERLU_LIBRARY) $
(PETSC_LIB) $(MUMPS_LIB) \
     $(SCALAP) $(parmetis_lib) $(metis_lib) $(SCALAP) \
      -L/usr/lib64 -lssl -lcrypto\



MACHINE_SPECIFIC_LIBS = $(FE)/tcl/tclMain.o


# %----------------------------------------------------------%
# | SECTION 8: INCLUDE FILES                                 |
# |                                                          |
# | The following macros specify include files needed for    |
```

```
# | compilation.                                                 |
# %-------------------------------------------------------------%



MACHINE_INCLUDES =  $(MPI_INC) \
                -I$(BERKLEYdir) \
                -I/usr/bin/mysql \
                -I$(UMFPACKdir) \
                -I$(SUPERLUdir) \
                -I$(SUPERLU_DISTdir) \
                 $(MUMPS_INCLUDE) \
                $(metis_inc)\
                #$(PETSC_INC)
                #-I$(OPEN_SSL_DIR)/include \
# this file contains all the OpenSees/SRC includes
include $(FE)/Makefile.incl


TCL_INCLUDES = -I$(TCLdir)/include
#-I$(FE)/tcl/include
#INCLUDES = $(TCL_INCLUDES) $(FE_INCLUDES) $(MACHINE_INCLUDES)
INCLUDES =  $(TCL_INCLUDES) $(MACHINE_INCLUDES) $(FE_INCLUDES)


##############################################################################
```