# atmmain.F

\* package atmmain !! AGCM main routine

\*

\* [his] 96/06/14(numaguti) agcm5.4f

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

program agcm !! agcm main routine

\*

\* [var]

real \* 8 time

real \* 8 tend

\*

\* [work]

logical oquit

\*

write ( 6,\* ) ' @@@ linear baroclinic model @@@ 99/01/01'

call yprep !! sys.dep. initialization

call calndr !! initilize calendar

\*

\* < initialize >

\*

call atmini

o ( time, tend )

\*

8000 continue

if ( time .lt. tend ) then

\*

\* < atmospheric step >

\*

call atmos

m ( time, tend )

\*

call quitck( oquit, time )

if ( .not. oquit ) goto 8000

endif

\*

call atmfin( time )

\*

call settim( time )

call histou( .true. ) !! output data

call clcout !! output of cpu time

call yfine !! sys.dep. final treatment

\*

stop

end

# cplmain.F

\* package gmain !! GCM main routine

\*

\* [his] 96/06/14(numaguti) agcm5.4f

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

program gcm !! gcm main routine

\*

\* [var]

real \* 8 time

real \* 8 tend

real \* 8 tstp

real \* 8 timea

real \* 8 timeo

\*

\* [work]

integer ifpar, jfpar

logical oquit

\*

\* [parm]

real \* 8 tstep

real\*8 step

character tunit\*4

namelist /nmmain/ step, tunit

data step / 1 /

data tunit / 'day' /

\*

write ( 6,\* ) ' @@@ agcm5.4 coupled @@@ 96/06/17'

call yprep !! sys.dep. initialization

call calndr !! initilize calendar

\*

call rewnml ( ifpar , jfpar )

read ( ifpar, nmmain, end=190 )

190 write ( jfpar, nmmain )

call cxx2ss( tstep, step, tunit, 0.d0 )

\*

\* < initialize >

\*

call atmini

o ( time , tend )

call ocnini

m ( time , tend )

\*

timea = time

timeo = time

tstp = time

8000 continue

if ( timea .lt. tend .or. timeo .lt. tend ) then

tstp = min( anint( tstp + tstep ), tend )

\*

\* < atmospheric step >

\*

8100 continue

if ( timea .lt. tstp ) then

call atmos

m ( timea, tstp )

goto 8100

endif

\*

\* < oceanic step >

\*

8200 continue

if ( timeo .lt. tstp ) then

call ocean

m ( timeo, tstp )

goto 8200

endif

\*

time = tstp

call quitck( oquit, time )

if ( .not. oquit ) goto 8000

endif

\*

call atmfin( timea )

call ocnfin( timeo )

\*

call settim( tend )

call histou( .true. ) !! output data

call clcout !! output of cpu time

call yfine !! sys.dep. final treatment

\*

stop

end

# zeq1main.F

\* [hiro] modified on 2004/04/14

\* [hiro] incorpolated to lbm package on 2004/06/14

c\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*zeq1main.f\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

c driver routine for coupled model

c note time convention in months; 0.5 corresponds to mid january 1960.

c\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

#include "zeq\_common.f"

\*

\*

\* [var]

real \* 8 time

real \* 8 tend

\*

\* [work]

logical oquit

\*

\* [internal work]

real\*8 td

integer ifpar, jfpar

integer it1, nt

integer i, j

integer ntimes !! undefined?

\*

\* [internal param]

real\*8 tfind

namelist /nmtitl/ header

namelist /nmstat/ nstart, tfind

data header / ' ' /

data tfind / 1.d20 /

\*

write ( 6,\* ) ' @@@ mlbm-cz coupled model @@@ 04/07/01'

call yprep !! sys.dep. initialization

call calndr !! initilize calendar

call rewnml ( ifpar , jfpar )

read ( ifpar, nmtitl, end=190 )

190 write ( jfpar, nmtitl )

call rewnml ( ifpar , jfpar )

read ( ifpar, nmstat, end=290 )

290 write ( jfpar, nmstat )

call openfl

write( jfpar,\* ) ' @@@ model initial setup @@@ '

\* read control paramters

natm = 2

if( nstart .lt. 0 ) go to 999

\* if history input is being used for startup, call rdhist...............

if( nstart .gt. 0 ) call rdhist(0,tfind)

natmr = natm

\* if nstart=1 or nstart=2, time is gotten from the history file.........

\* otherwise (nstart=3) time is set to tzero from input data file........

if( nstart .eq. 3 .or. nstart .eq. 0 ) then

td = tzero

else

tzero = td

endif

nt = 0

it1 = td + 120.501

it = mod( it1,12 )

mp = it + 1

if( it .eq. 0) it = 12

ty = td + 120.501 - it1

call setup

call rdbase !! read mean fields

call constc !! initialize ocean dynamics model

istart=0

call stress(td) !! initial stress field

\* if one wanted to overwrite any fields with data, it would be done here

\* based on the paramter ic2... routine insert would be created to do it.

\* if( ic2 .gt. 0 ) call insert

\* initial call to ocean force routine and dynamics update routine mloop

\* are special if starting up from rest - ie, if not restarting..........

call cforce

write( jfpar,1080 ) nt,td,it

if( nstart .gt. 0 ) then

istart=1

go to 400

else

go to 350

endif

\*

\* this is the top of main loop for all but the first cycle

\*

300 call ssta(td) !! update the ssta field

call zatmc(td) !! update the winds

call stress(td) !! compute the wind stress anomalies

call cforce !! stress forcing into ocean model

350 call mloop !! update the ocean dynamics

400 continue

\*

write( jfpar,1090 ) nt,td,it

\*

call wrgrd(nt) !! history output

call wrnino(nt,td) !! nino3 ssta diagnosis

\*

if( nt .ge. ntimes ) go to 900 !! if it's time to stop

\* if not, loop again

nt = nt + 1

td = tzero + float( nt ) \* dtd

it1 = td + 120.501

it = mod( it1,12 )

mp = it + 1

if( it .eq. 0 ) it = 12

ty = td + 120.501 - it1

go to 300

900 call wrhist !! time to finish up

\*

999 call clcout !! output of cpu time

call yfine !! sys.dep. final treatment

1080 format( ' timestep (initial)', i6, ' time =',f8.3,

$ ' months = ', i5, ' integer month' )

1090 format( ' timestep', i6, ' time =',f8.3, ' months = ',

$ i5, ' integer month' )

stop

end

# Makefile

#

# makefile for $(lbmdir)/src

#

include ./mkinclude

dir =

libsubdirs = admin dynamics physics io util sysdep

libsubdirsnp = admin dynamics io util sysdep

subdirs = $(libsubdirs) include $(specialdir)

fflags = $(stdfflags) -i$(incdir)

cflags = $(stdcflags) -i$(incdir)

cppflags = $(stdcppflags) $(cppopt)-i$(incdir)

ldflags = $(stdldflags)

objs = $(main)

srcs = $(main:.o=.f)

srcs\_all = atmmain.f cplmain.f

nonstds = $(special) $(nonstd)

exthdrs = $(incdir)/zccom.f \

$(incdir)/zcdim.f \

$(incdir)/zhdim.f

hdrs =

makefile = makefile

all: lbm

world: dirs lib lbm

lbm: $(lbm)

$(lbm): includes speciallib $(main) $(sysliby)

$(ld) $(ldflags) $(objs) $(nonstds) $(library) $(sysliby) $(syslib) -o $@

$(rm) $(main) $(nonstds)

lib: includes

@echo "making $(library)"

@for i in $(libsubdirs) ; do \

cd $$i ; \

echo "making $$i/" ; \

$(make) ; \

cd .. ; \

done

libnp: includes

@echo "making $(library)"

@for i in $(libsubdirsnp) ; do \

cd $$i ; \

echo "making $$i/" ; \

$(make) ; \

cd .. ; \

done

includes:

@cd include; $(make) "system=$(system)"

adminlib:

@cd admin; $(make) "system=$(system)"

dynamicslib:

@cd dynamics; $(make) "system=$(system)"

iolib:

@cd io; $(make) "system=$(system)"

physicslib:

@cd physics; $(make) "system=$(system)"

sysdeplib:

@cd sysdep; $(make) "system=$(system)"

utillib:

@cd util; $(make) "system=$(system)"

speciallib:

@cd $(specialdir); $(make) "system=$(system)"

dirs:

test -d $(libdir) || mkdir -p $(libdir)

test -d $(bindir) || mkdir -p $(bindir)

clean: clean.local clean.dir

clean.local:

$(rm) \*.o \*~ core \*.trace \*.f $(ftrfile) \*.html all.f \

speciallist nonstdlist

clean.dir:

@for i in $(subdirs) ; do \

cd $$i ; \

echo "cleaning $$i/" ; \

$(make) "system=$(system)" clean ; \

cd .. ; \

done

clean.special:

@for i in $(specialdir); do \

cd $$i ; \

echo "cleaning $$i/" ; \

$(make) "system=$(system)" clean ; \

cd .. ; \

done

tags: tags.local tags.dir

tags.local:

@$(tags) $(srcs)

@for i in $(libsubdirs) ; do \

$(tags) -a $$i/\*.f ; \

done

tags.dir:

@for i in $(subdirs) ; do \

cd $$i ; \

$(make) "system=$(system)" tags ; \

cd .. ; \

done

index: index.local index.dir

index.local:

@$(index) $(hdrs) $(srcs)

index.dir:

@for i in $(subdirs) ; do \

cd $$i ; \

$(make) "system=$(system)" index ; \

cd .. ; \

done

index.head:

cut -c1-10,22-30,45- index > head

pack: rmftr includes pack.local pack.dir

ftr: includes ftr.local ftr.dir

ftr.local: $(srcs\_all:.f=.ftr)

pack.local: $(srcs:.f=.ftr)

ftr.dir:

@for i in $(libsubdirs) ; do \

cd $$i ; \

echo "ftring $$i/" ; \

$(make) "system=$(system)" ftr; \

cd .. ; \

done

pack.dir:

@for i in $(libsubdirs) ; do \

cd $$i ; \

echo "packing $$i/" ; \

$(make) "system=$(system)" pack; \

cd .. ; \

done

rmftr:

$(rm) $(ftrfile)

srcall.f: includes pack

$(mv) $(ftrfile) srcall.f

srcall.f: srcall.f

$(cpp) $(cppflags) srcall.f -i$(incdir) > srcall.f

size: srcall.f

../sh/util/fmsize -d -c -l srcall.f | ../sh/util/fmsort > size.$(resolution)$(memorysize)

ftnchk: srcall.f

ftnchek -nopretty -notruncation -array=0 srcall.f > ftnchk

tree: tree.all

tree.all: srcall.f ../sh/util/fcgx

fcg\_maxlay=7 fcg\_trunc='dynmcs,physcs,rddat,histin,histif,gdread,g2w,w2g,fft99x' ../sh/util/fcgx < srcall.f > tree.all

../sh/util/fcgx:

$(fc) -o ../sh/util/fcgx ../sh/util/fcgx.f

sublist: sublist.local sublist.dir

@for i in . $(specialdir) nonstd $(libsubdirs) ; do \

cat $$i/sublist ; \

done | \

sort +0 -2 | sort +0 -1 -u > sublist.all

../sh/util/e2j sublist.all > sublist.jpn

sublist.local:

egrep '( program| subroutine | entry )' $(srcs) /dev/null | $(awk) -f ../sh/util/sublist | sort > sublist

sublist.dir:

@for i in $(libsubdirs) nonstd $(specialdir); do \

cd $$i ; \

$(make) "system=$(system)" sublist; \

cd .. ; \

done

html: clean.html sublist html.local html.dir html.list html.tree

clean.html:

$(rm) \*.html \*/\*.html

html.local: $(srcs\_all:.f=.html)

html.dir:

@for i in $(subdirs) ; do \

cd $$i ; \

$(make) "system=$(system)" html; \

cd .. ; \

done

html.list:

$(lbmdir)/sh/util/makehtmllist sublist.jpn > sublist.html

html.tree: tree.all

$(lbmdir)/sh/util/makehtmltree tree.all > tree.html

install.html:

tar cf - \*.html \*/\*.html | ( cd $(html\_lbm)/src; tar xvf - )

###

atmmain.o: $(incdir)/zcdim.f $(incdir)/zpdim.f $(incdir)/zwdim.f \

$(incdir)/zhdim.f $(incdir)/zccom.f $(incdir)/zcord.f

cplmain.o: $(incdir)/zcdim.f $(incdir)/zpdim.f $(incdir)/zwdim.f \

$(incdir)/zhdim.f $(incdir)/zccom.f $(incdir)/zcord.f

# Mkinclude

# include file for makefile for $(lbmdir)

#

# environmental valiable lbmdir should be specified

# % setenv lbmdir ~hiro/ln\_solver/model

#

include $(lnhome)/lmake.inc

lbmdir = $(lnhome)/model

############## system specification #######################

system = $(arc)

project\_ext =

################ directory specification ###############

######## root of lbm

###lbmdir # please set by environment variable

######## lbm libraries

libdir = $(lbmdir)/lib/$(system)

###libdir = $(lbmdir)/lib

######## lbm excutables

bindir = $(lbmdir)/bin/$(system)

###bindir = $(lbmdir)/bin

############ other definitions (do not edit) ###########

stdfflags = $(sysfflags) $(debug) $(check)

stdcflags = $(syscflags) $(debug)

stdldflags = $(sysldflags) $(debug)

stdcppflags = $(syscppflags) $(cppopt)$(options)

stdlibs =

srcdir = $(lbmdir)/src

incdir = $(lbmdir)/src/include

sysmakedir = $(lbmdir)/src/sysdep

projdir = $(lbmdir)/src/proj

include $(sysmakedir)/makedef.$(system)

include $(projdir)/$(project)/mkinclude$(project\_ext)

resolution = $(hresolution)$(memorysize)$(vresolution)

lbm = $(bindir)/lbm2.$(resolution)$(modelname)

libnames = lbm2$(resolution)

lib = lib$(libnames).a

library = $(libdir)/$(lib)

libs = $(linkopt)-l$(libdir) \

$(linkopt)-l$(libnames)

nonstd = $(atmos) $(user) \

$(sphert) $(cumulus) $(lsccld) \

$(radiate) $(dryadj) $(igwd) $(fft)

mkinclude = $(srcdir)/mkinclude

ftrdir = $(srcdir)/..

ftrfile = $(ftrdir)/src.ftr