```
1   program chanCompr
2   !======================================================================
3   use mpi
4   use comData
5
6   implicit none
7   !======================Variable Declaration Starts=====================
8   character(len=30)     :: filename
9   logical               :: isBlown=.false.
10  integer               :: i,j,k,m,p,nstep,iter,st,nXY
11  integer,dimension(5)  :: plane =[5,30,55,90,125]
12  double precision      :: tStart, tEnd, tauxzL, tauxzU, tauyzL, tauyzU
13  double precision      :: time, Pe, flux,dAyz,Ayz
14  double precision      :: f1d2c0, f1d2c1, f1d2c2, b1d2c0, b1d2c1, b1d2c2
15  double precision      :: f2d1c0, f2d1c1, f2d1c2, b2d1c0, b2d1c1, b2d1c2
16  double precision      :: dz1,dzN,alpha,dz2,dz
17  double precision      :: f2d2c0, f2d2c1, f2d2c2, f2d2c3
18  double precision      :: b2d2c0, b2d2c1, b2d2c2, b2d2c3
19  double precision      :: df21,df31,df32,df41,df42,df43
20  double precision      :: db12,db13,db23,db14,db24,db34
21  !-------------------------Temporary Variables--------------------------
22  integer               ::tmpI1,tmpI2,tmpI3,tmpI4,tmpI5
23  double precision      ::tmpR1,tmpR2,tmpR3,tmpR4,tmpA1(5),loc(nP(1),nP(2),7)
24  double precision      ::Ut,Vt,Wt,dudx,dudy,dudz,dvdx,dvdy,dvdz,dwdx,dwdy,dwdz
25  double precision      ::dTdx,dTdy,dTdz,divg,muT,ktT,PressT,dissF,dissG,dissH
26  double precision      ::dudxdz,dvdydz,mud2wdz2,drhow2dz,lhs,dudzdx,dvdzdy,dwdzdmudz
27  double precision      ::drhouwdx,d2wdx2,drhovwdy,d2wdy2,drhowdt
28  integer,allocatable,dimension(:,:)  ::tArr,bb,ee
29
30  !-----------------------MPI Variables---------------------------------
31  integer,dimension(3)    :: bs,es,bn,en,siz,des,src,myDim,myCoord
32  integer   :: liney,yz1p, xz1p, xy1p, yz1p5v, xz1p5v, xy1p5v, yz1p7v, xz1p7v, xy1p7v
33  integer   :: lineyN, yzN1p, xzN1p, xyN1p, yzN2p, xzN2p, xyN2p, yz2p5v, xz2p5v, xy2p5v
34  integer   :: comm3d,nproc, id, ierr, sizDP, STATUS(mpi_status_size)
35  logical   :: myperiod(3), bottom, top, east, west, north, south, master
36  integer   :: linex, linex7v, liney7v
37  !----------------------------------------------------------------------
38  double precision,dimension(nP(3),7) :: stat1Spc ! In order of rho,u,v,w,e,t,p
39  double precision,dimension(nP(3),7) :: stat2Spc ! In order of rho,u,v,w,e,t,p
40  double precision,dimension(nP(3),9) :: corrSpc  ! In order of rhow,rhot,uv,uw,vw,vt,tw,pw,pt
41
42  double precision,dimension(0:nP(1)+1,0:nP(2)+1,0:nP(3)+1,5,2) :: solV
43  double precision,dimension(nP(1),nP(2),nP(3),7)               :: var ! In order of rho,U,V,W,E,T,P
```

```
44   double precision,dimension(nP(1),nP(2),nP(3),5)              :: fD,gD,hD
45   double precision,dimension(nP(1),nP(2),nP(3),5)              :: srcV
46   double precision,dimension(nP(1),nP(2),nP(3),5,2)            :: F,G,H,R
47
48   double precision,dimension(nP(1),nP(2),nP(3))                :: mu,kt,rhow
49
50   !-------------------------Debug Variables-----------------------------
51   double precision,dimension(nP(3))     :: pl
52   !---------------------------------------------------------------------
53   double precision ::Sigma=50,SC=1,Q
54   !=======================Variable Declaration Ends=====================
55   call readParam()
56   call setupMesh()
57   !---------------------------------------------------------------------
58   dz1   = x3(2)-x3(1)
59   alpha = (x3(3)-x3(2))/dz1
60   dz2   = x3(3)-x3(2)
61   dz    = (dz1 + dz2)/two
62
63   tmpR1 = dz1*alpha*(alpha + one)
64   f1d2c0  = (one - (alpha + one)**two)/tmpR1
65   f1d2c1  = ((alpha + one)**two)/tmpR1
66   f1d2c2  = -one / tmpR1
67
68   tmpR1 = haf*dz1**two*alpha*(alpha + one)
69   f2d1c0  = alpha/tmpR1
70   f2d1c1  = -(alpha + one)/tmpR1
71   f2d1c2  = one / tmpR1
72
73   df21 = x3(2)-x3(1)
74
75   df31 = x3(3)-x3(1)
76   df32 = x3(3)-x3(2)
77
78   df41 = x3(4)-x3(1)
79   df42 = x3(4)-x3(2)
80   df43 = x3(4)-x3(3)
81
82   f2d2c0 =  ( 2.0d0*(-df21-df31-df41) )/( (-df21)*(-df31)*(-df41) )
83   f2d2c1 =  ( 2.0d0*(-df31-df41)      )/(   df21 *(-df32)*(-df42) )
84   f2d2c2 =  ( 2.0d0*(-df21-df41)      )/(   df31 *  df32 *(-df43) )
85   f2d2c3 =  ( 2.0d0*(-df21-df31)      )/(   df41 *  df42 *  df43  )
86   !---------------------------------------------------------------------
```

```
 87   dzN   = x3(nP(3))-x3(nP(3)-1)
 88   alpha = (x3(nP(3)-1)-x3(nP(3)-2))/dzN
 89
 90   tmpR1 = dzN*alpha*(alpha + one)
 91   b1d2c0  = ((alpha + one)**two - one)/tmpR1
 92   b1d2c1  = -((alpha + one)**two)/tmpR1
 93   b1d2c2  = one / tmpR1
 94
 95   tmpR1 = haf*dzN**two*alpha*(alpha + one)
 96   b2d1c0  = alpha/tmpR1
 97   b2d1c1  = -(alpha + one)/tmpR1
 98   b2d1c2  = one / tmpR1
 99
100   db12 =x3(nP(3))  -x3(nP(3)-1)
101
102   db13 =x3(nP(3))  -x3(nP(3)-2)
103   db23 =x3(nP(3)-1)-x3(nP(3)-2)
104
105   db14 =x3(nP(3))  -x3(nP(3)-3)
106   db24 =x3(nP(3)-1)-x3(nP(3)-3)
107   db34 =x3(nP(3)-2)-x3(nP(3)-3)
108
109   b2d2c0 =  ( 2.0d0*(db12 + db13 + db14 ))/(   db12 *  db13 *  db14  )
110   b2d2c1 =  ( 2.0d0*(db13 + db14        ))/( (-db12)*  db23 *  db24  )
111   b2d2c2 =  ( 2.0d0*(db12 + db14        ))/( (-db13)*(-db23)*  db34  )
112   b2d2c3 =  ( 2.0d0*(db12 + db13        ))/( (-db14)*(-db24)*(-db34) )
113   !------------------------------------------------------------------------
114
115   !============================MPI====================================
116   CALL mpi_init(ierr)
117   CALL mpi_comm_rank(mpi_comm_world,id,ierr)
118   CALL mpi_comm_size(mpi_comm_world,nproc,ierr)
119
120   call mpi_cart_create(mpi_comm_world,3,topo,period,.false.,comm3d,ierr)
121   call mpi_cart_get(comm3d,3,myDim,myperiod,mycoord,ierr)
122
123   call mpi_cart_shift(comm3d,0,1,src(1),des(1),ierr)
124   call mpi_cart_shift(comm3d,1,1,src(2),des(2),ierr)
125   call mpi_cart_shift(comm3d,2,1,src(3),des(3),ierr)
126
127   allocate(tArr(0:maxval(myDim)-1,3))
128   allocate(bb(3,0:nproc-1),ee(3,0:nproc-1))
129
```

```
130   master=(id==0)
131   east  =(mycoord(1)==myDim(1)-1)
132   west  =(mycoord(1)==0)
133   north =(mycoord(2)==myDim(2)-1)
134   south =(mycoord(2)==0)
135   top   =(mycoord(3)==myDim(3)-1)
136   bottom=(mycoord(3)==0)
137
138   do k=1,3
139
140     tmpI1=nP(k)/myDim(k)
141     tmpI2=myDim(k)-mod(nP(k),myDim(k))
142
143     tArr(0,1)=1
144     do i=0,myDim(k)-1
145       if(i==tmpI2) tmpI1=tmpI1+1
146       tArr(i,2)=tArr(i,1)+tmpI1-1
147       tArr(i,3)=tArr(i,2)-tArr(i,1)+1
148       if(i==myDim(k)-1) exit
149       tArr(i+1,1)=tArr(i,2)+1
150     enddo
151
152     do i=0,myDim(k)-1
153       if(i==mycoord(k)) then
154         bs(k)=tArr(i,1)
155         es(k)=tArr(i,2)
156         siz(k)=tArr(i,3)
157       endif
158     enddo
159
160     bn(k)=bs(k)
161     en(k)=es(k)
162
163     if((west .and. k==1) .or. (south  .and. k==2) .or. (bottom .and. k==3)) then
164       bn(k)=bs(k)+1
165       siz(k)=siz(k)-1
166     endif
167
168     if((east .and. k==1) .or. (north  .and. k==2) .or. (top    .and. k==3)) then
169       en(k)=es(k)-1
170       siz(k)=siz(k)-1
171     endif
172
```

```
173   enddo
174   !------------------------------------------------------------------------
175   CALL mpi_type_extent (mpi_double_precision,sizDP,ierr)
176   CALL mpi_type_vector (siz(2), 1, nP(1), mpi_double_precision,liney ,ierr)
177   CALL mpi_type_vector (siz(1), 1,    1 , mpi_double_precision,linex ,ierr)
178
179   CALL mpi_type_hvector(siz(3), 1     , nP(1)*nP(2)*sizDP ,liney                ,yz1p,ierr)
180   CALL mpi_type_vector (siz(3), siz(1), nP(1)*nP(2)       , mpi_double_precision,xz1p,ierr)
181   CALL mpi_type_vector (siz(2), siz(1), nP(1)             , mpi_double_precision,xy1p,ierr)
182
183   CALL mpi_type_commit(yz1p,ierr)
184   CALL mpi_type_commit(xz1p,ierr)
185   CALL mpi_type_commit(xy1p,ierr)
186
187   CALL mpi_type_hvector(5, 1 , product(nP)*sizDP , yz1p , yz1p5v ,ierr)
188   CALL mpi_type_hvector(5, 1 , product(nP)*sizDP , xz1p , xz1p5v ,ierr)
189   CALL mpi_type_hvector(5, 1 , product(nP)*sizDP , xy1p , xy1p5v ,ierr)
190
191   CALL mpi_type_commit(yz1p5v,ierr)
192   CALL mpi_type_commit(xz1p5v,ierr)
193   CALL mpi_type_commit(xy1p5v,ierr)
194
195   CALL mpi_type_hvector(7, 1 , product(nP)*sizDP , yz1p , yz1p7v ,ierr)
196   CALL mpi_type_hvector(7, 1 , product(nP)*sizDP , xz1p , xz1p7v ,ierr)
197   CALL mpi_type_hvector(7, 1 , product(nP)*sizDP , xy1p , xy1p7v ,ierr)
198
199
200   CALL mpi_type_commit(yz1p7v,ierr)
201   CALL mpi_type_commit(xz1p7v,ierr)
202   CALL mpi_type_commit(xy1p7v,ierr)
203
204   CALL mpi_type_hvector(7, 1 , product(nP)*sizDP , liney , liney7v ,ierr)
205   CALL mpi_type_hvector(7, 1 , product(nP)*sizDP , linex , linex7v ,ierr)
206
207   CALL mpi_type_commit(liney7v,ierr)
208   CALL mpi_type_commit(linex7v,ierr)
209   !------------------------------------------------------------------------
210   CALL mpi_type_vector (siz(2),      1, nP(1)+2, mpi_double_precision,lineyN ,ierr)
211
212   CALL mpi_type_hvector(siz(3),      1, (nP(1)+2)*(nP(2)+2)*sizDP,lineyN,yzN1p,ierr)
213   CALL mpi_type_vector (siz(3), siz(1), (nP(1)+2)*(nP(2)+2), mpi_double_precision,xzN1p ,ierr)
214   CALL mpi_type_vector (siz(2), siz(1), nP(1)+2, mpi_double_precision,xyN1p ,ierr)
215
```

```
216   CALL mpi_type_hvector(2, 1, sizDP                    , yzN1p,yzN2p,ierr)
217   CALL mpi_type_hvector(2, 1, (nP(1)+2)*sizDP          , xzN1p,xzN2p,ierr)
218   CALL mpi_type_hvector(2, 1, (nP(1)+2)*(nP(2)+2)*sizDP , xyN1p,xyN2p,ierr)
219
220   CALL mpi_type_hvector(5, 1 , product(nP+2)*sizDP , yzN2p , yz2p5v ,ierr)
221   CALL mpi_type_hvector(5, 1 , product(nP+2)*sizDP , xzN2p , xz2p5v ,ierr)
222   CALL mpi_type_hvector(5, 1 , product(nP+2)*sizDP , xyN2p , xy2p5v ,ierr)
223
224   CALL mpi_type_commit(yz2p5v,ierr)
225   CALL mpi_type_commit(xz2p5v,ierr)
226   CALL mpi_type_commit(xy2p5v,ierr)
227
228   CALL mpi_barrier(mpi_comm_world,ierr)
229   !-----------------------------MPI----------------------------------------
230
231   if(master) then
232     write(6,'(A)')"================================================================================================="
233     write(6,'(2A)')"Compressible Channel Flow Solver started at :",dateTime()
234     write(6,'(A)')"================================================================================================="
235     write(6,'(A)')
236     write(6,'(A)')"================================================================================================="
237     write(6,'(A)')"Parameters read from param.dat"
238     write(6,'(A)')"-------------------------------------------------------------------------------------------------"
239     write(6,'(2A)')"Case Description        : ",descp
240     write(6,'(2A)')"Path to input file      : ",inputPath
241     write(6,'(A,3(I3,1X))')"Process Topology        : ",topo(1),topo(2),topo(3)
242     write(6,'(A,F10.4)'   )"Reynolds Number         : ",Re
243     write(6,'(A,F10.2)'   )"Mach Number             : ",Mac
244     write(6,'(A,F10.8)'   )"Timestep                : ",dt
245     write(6,'(A,I10)'     )"Freq of stat calculation: ",istat
246     write(6,'(A,F10.4)'   )"Peclet in x-direction   : ",PexVal
247     write(6,'(A,F10.4)'   )"Peclet in y-direction   : ",PeyVal
248     write(6,'(A,F10.4)'   )"Peclet in z-direction   : ",PezVal
249     write(6,'(A,I10)'     )"Restart Option  [1/2]   : ",resOp
250     write(6,'(A,I10)'     )"Debug Option    [0/1]   : ",debugOp
251     write(6,'(A)')"================================================================================================="
252   endif
253
254   select case(resOp)
255     case(1)
256       !======================Read Combined Input File=====================
257       open(unit=10,file=inputPath)
258
```

```
259       read (10,100)time,nstep
260       read (10,*)
261       read (10,*)
262
263       100 format(8X,F20.10,I9,1X)
264
265       do k=1,nP(3)
266         read(10,*)
267         do j=1,nP(2)
268           read(10,*)
269           do i=1,nP(1)
270             read(10,101,iostat=st) x1(i),x2(j),x3(k),var(i,j,k,1),var(i,j,k,2),var(i,j,k,3),var(i,j,k,4)  &
271                                                                    ,var(i,j,k,5),var(i,j,k,6),var(i,j,k,7)
272
273             if(st/=0) then
274               write(6,'(A,4(1X,I4))') "Error in reading combined input file at",i,j,k,st
275               stop
276             endif
277
278           end do
279         end do
280       end do
281
282       close(10)
283       101 format(3(1X,F10.6),7(1X,E22.15))
284     case(2)
285       !======================Read Component Input File====================
286       WRITE(filename,'(a,i3.3,a)')"../output/3D",id+1,".dat"
287       OPEN (UNIT=10,FILE=filename)
288
289       read(10,100) time,nstep
290       read(10,*)
291       read(10,*)
292
293       do k=bs(3),es(3)
294         read(10,*)
295         do j=bs(2),es(2)
296           read(10,*)
297           do i=bs(1),es(1)
298             read(10,101,iostat=st) x1(i),x2(j),x3(k),var(i,j,k,1),var(i,j,k,2),var(i,j,k,3),var(i,j,k,4)  &
299                                                                    ,var(i,j,k,5),var(i,j,k,6),var(i,j,k,7)
300             if(st/=0) then
301               write(6,'(A,5(1X,I4))') "Error in reading component input file at",id,i,j,k,st
```

```
302             write(6,'(A)') "Change in MPI process topology since last run could possibly be a reason"
303             stop
304           endif
305
306         end do
307       end do
308     enddo
309     close(10)
310     !---------------------------------------------------------------------
311 end select
312
313 if(master) write(6,'(A)') "Input file successfully read"
314
315 !--------------------Initial Solution Vector---------------------------
316 do k=bs(3),es(3)
317   do j=bs(2),es(2)
318     do i=bs(1),es(1)
319       solV(i,j,k,1,1)   = var(i,j,k,1)
320       solV(i,j,k,2:5,1) = var(i,j,k,1) * var(i,j,k,2:5)
321
322       mu(i,j,k)         = var(i,j,k,6)**0.7d0
323       kt(i,j,k)         = var(i,j,k,6)**0.7d0
324     enddo
325   enddo
326 enddo
327
328 !=====================================================================
329 call mpi_sendrecv(solV(en(1)-1,bn(2),bn(3),1,1),1,yz2p5v,des(1),50,solV(bn(1)-2,bn(2),bn(3),1,1),1,yz2p5v,src(1),
    50,mpi_comm_world,STATUS,ierr)
330 call mpi_sendrecv(solV(bn(1)  ,bn(2),bn(3),1,1),1,yz2p5v,src(1),50,solV(en(1)+1,bn(2),bn(3),1,1),1,yz2p5v,des(1),
    50,mpi_comm_world,STATUS,ierr)
331
332 call mpi_sendrecv(solV(bn(1),en(2)-1,bn(3),1,1),1,xz2p5v,des(2),50,solV(bn(1),bn(2)-2,bn(3),1,1),1,xz2p5v,src(2),
    50,mpi_comm_world,STATUS,ierr)
333 call mpi_sendrecv(solV(bn(1),bn(2)  ,bn(3),1,1),1,xz2p5v,src(2),50,solV(bn(1),en(2)+1,bn(3),1,1),1,xz2p5v,des(2),
    50,mpi_comm_world,STATUS,ierr)
334
335 call mpi_sendrecv(solV(bn(1),bn(2),en(3)-1,1,1),1,xy2p5v,des(3),50,solV(bn(1),bn(2),bn(3)-2,1,1),1,xy2p5v,src(3),
    50,mpi_comm_world,STATUS,ierr)
336 call mpi_sendrecv(solV(bn(1),bn(2),bn(3)  ,1,1),1,xy2p5v,src(3),50,solV(bn(1),bn(2),en(3)+1,1,1),1,xy2p5v,des(3),
    50,mpi_comm_world,STATUS,ierr)
337 !---------------------------------------------------------------------
338 call mpi_sendrecv(var(en(1),bn(2),bn(3),1),1,yz1p7v,des(1),50,var(bn(1)-1,bn(2),bn(3),1),1,yz1p7v,src(1),
```

```
     50,mpi_comm_world,STATUS,ierr)
339  call mpi_sendrecv(var(bn(1),bn(2),bn(3),1),1,yz1p7v,src(1),50,var(en(1)+1,bn(2),bn(3),1),1,yz1p7v,des(1),
     50,mpi_comm_world,STATUS,ierr)
340
341  call mpi_sendrecv(var(bn(1),en(2),bn(3),1),1,xz1p7v,des(2),50,var(bn(1),bn(2)-1,bn(3),1),1,xz1p7v,src(2),
     50,mpi_comm_world,STATUS,ierr)
342  call mpi_sendrecv(var(bn(1),bn(2),bn(3),1),1,xz1p7v,src(2),50,var(bn(1),en(2)+1,bn(3),1),1,xz1p7v,des(2),
     50,mpi_comm_world,STATUS,ierr)
343
344  call mpi_sendrecv(var(bn(1),bn(2),en(3),1),1,xy1p7v,des(3),50,var(bn(1),bn(2),bn(3)-1,1),1,xy1p7v,src(3),
     50,mpi_comm_world,STATUS,ierr)
345  call mpi_sendrecv(var(bn(1),bn(2),bn(3),1),1,xy1p7v,src(3),50,var(bn(1),bn(2),en(3)+1,1),1,xy1p7v,des(3),
     50,mpi_comm_world,STATUS,ierr)
346
347  !=======================Main time loop (starts)=======================
348  DO WHILE (nstep < maxstep)
349
350    if(master .and. mod(nstep,10)==0) call cpu_time(tStart)
351    nstep = nstep+1
352    time  = time + dt
353
354    !=======================RK2 Sub-steps (starts)=======================
355    do iter=1,maxiter
356
357      do k=bn(3),en(3)
358        do j=bn(2),en(2)
359          do i=bn(1),en(1)
360
361            Pe = Re*abs(solV(i,j,k,2,1))*dx/mu(i,j,k)
362
363            if((Pe .gt. PexVal) .or. ( i .eq. bn(1)) .or. (i .eq. en(1))) then
364              tmpR1  = ( var(i,j,k,2)+var(i+1,j,k,2) )*haf
365              tmpR2  = ( var(i,j,k,2)+var(i-1,j,k,2) )*haf
366
367              if(tmpR1 .ge. 0.0d0 .and. tmpR2 .ge. 0.0d0) then
368                F(i,j,k,1:5,1) =( tmpR1*( 6.0d0*solV (i  ,j,k,1:5,1) + 3.0d0*solV (i+1,j,k,1:5,1) - 1.0d0*solV (i-1,j,k,1:5,1) ) -  &
369                                  tmpR2*( 6.0d0*solV (i-1,j,k,1:5,1) + 3.0d0*solV (i  ,j,k,1:5,1) - 1.0d0*solV (i-2,j,k,1:5,1) )    &
370                                )*invdx/8.0d0
371              endif
372
373              if(tmpR1 .lt. 0.0d0 .and. tmpR2 .lt. 0.0d0) then
374                F(i,j,k,1:5,1) =( tmpR1*( 6.0d0*solV (i+1,j,k,1:5,1) + 3.0d0*solV (i  ,j,k,1:5,1) - 1.0d0*solV (i+2,j,k,1:5,1) ) -  &
375                                  tmpR2*( 6.0d0*solV (i  ,j,k,1:5,1) + 3.0d0*solV (i-1,j,k,1:5,1) - 1.0d0*solV (i+1,j,k,1:5,1) )    &
```

```
376                             )*invdx/8.0d0
377               endif
378
379               if(tmpR1 .lt. 0.0d0 .and. tmpR2 .ge. 0.0d0) then
380                 F(i,j,k,1:5,1) =( tmpR1*( 6.0d0*solV (i+1,j,k,1:5,1) + 3.0d0*solV (i,j,k,1:5,1) - 1.0d0*solV (i+2,j,k,1:5,1) ) -  &
381                                   tmpR2*( 6.0d0*solV (i-1,j,k,1:5,1) + 3.0d0*solV (i,j,k,1:5,1) - 1.0d0*solV (i-2,j,k,1:5,1) )    &
382                                 )*invdx/8.0d0
383               endif
384
385               if(tmpR1 .ge. 0.0d0 .and. tmpR2 .lt. 0.0d0) then
386                 F(i,j,k,1:5,1) =( tmpR1*( 6.0d0*solV (i,j,k,1:5,1) + 3.0d0*solV (i+1,j,k,1:5,1) - 1.0d0*solV (i-1,j,k,1:5,1) ) -  &
387                                   tmpR2*( 6.0d0*solV (i,j,k,1:5,1) + 3.0d0*solV (i-1,j,k,1:5,1) - 1.0d0*solV (i+1,j,k,1:5,1) )    &
388                                 )*invdx/8.0d0
389               endif
390
391             else   ! Fourth order Central Differencing
392               F(i,j,k,1:5,1) =   (-        var(i+2,j,k,2)*solV (i+2,j,k,1:5,1)  &
393                                   +  8.0d0*var(i+1,j,k,2)*solV (i+1,j,k,1:5,1)  &
394                                   -  8.0d0*var(i-1,j,k,2)*solV (i-1,j,k,1:5,1)  &
395                                   +        var(i-2,j,k,2)*solV (i-2,j,k,1:5,1)  &
396                                 )*invdx*one12th
397             endif
398             !-----------------------------------------------------------------
399             Pe = Re*abs(solV(i,j,k,3,1))*dy/mu(i,j,k)
400
401             if((Pe .gt. PeyVal) .or. (j .eq. bn(2)) .or. (j .eq. en(2))) then
402               tmpR1  = ( var(i,j,k,3)+var(i,j+1,k,3) )*haf
403               tmpR2  = ( var(i,j,k,3)+var(i,j-1,k,3) )*haf
404
405               if(tmpR1.ge.0.0d0.and.tmpR2.ge.0.0d0) then
406                 G(i,j,k,1:5,1) =( tmpR1*( 6.0d0*solV (i,j  ,k,1:5,1) + 3.0d0*solV (i,j+1,k,1:5,1) - 1.0d0*solV (i,j-1,k,1:5,1) ) -  &
407                                   tmpR2*( 6.0d0*solV (i,j-1,k,1:5,1) + 3.0d0*solV (i,j  ,k,1:5,1) - 1.0d0*solV (i,j-2,k,1:5,1) )    &
408                                 )*invdy/8.0d0
409               endif
410
411               if(tmpR1.lt.0.0d0.and.tmpR2.lt.0.0d0) then
412                 G(i,j,k,1:5,1) =( tmpR1*( 6.0d0*solV (i,j+1,k,1:5,1) + 3.0d0*solV (i,j  ,k,1:5,1) - 1.0d0*solV (i,j+2,k,1:5,1) ) -  &
413                                   tmpR2*( 6.0d0*solV (i,j  ,k,1:5,1) + 3.0d0*solV (i,j-1,k,1:5,1) - 1.0d0*solV (i,j+1,k,1:5,1) )    &
414                                 )*invdy/8.0d0
415               endif
416
417               if(tmpR1.lt.0.0d0.and.tmpR2.ge.0.0d0) then
418                 G(i,j,k,1:5,1) =( tmpR1*( 6.0d0*solV (i,j+1,k,1:5,1) + 3.0d0*solV (i,j,k,1:5,1) - 1.0d0*solV (i,j+2,k,1:5,1) ) -  &
```

```
419                                 tmpR2*( 6.0d0*solV (i,j-1,k,1:5,1) + 3.0d0*solV (i,j,k,1:5,1) - 1.0d0*solV (i,j-2,k,1:5,1) )    &
420                               )*invdy/8.0d0
421               endif
422
423               if(tmpR1.ge.0.0d0.and.tmpR2.lt.0.0d0) then
424                 G(i,j,k,1:5,1) =( tmpR1*( 6.0d0*solV (i,j,k,1:5,1) + 3.0d0*solV (i,j+1,k,1:5,1) - 1.0d0*solV (i,j-1,k,1:5,1) ) -  &
425                                   tmpR2*( 6.0d0*solV (i,j,k,1:5,1) + 3.0d0*solV (i,j-1,k,1:5,1) - 1.0d0*solV (i,j+1,k,1:5,1) )    &
426                                 )*invdy/8.0d0
427               endif
428
429             else   ! Fourth order Central Differencing
430               G(i,j,k,1:5,1) =   (-        var(i,j+2,k,3)*solV (i,j+2,k,1:5,1)  &
431                                   +  8.0d0*var(i,j+1,k,3)*solV (i,j+1,k,1:5,1)  &
432                                   -  8.0d0*var(i,j-1,k,3)*solV (i,j-1,k,1:5,1)  &
433                                   +        var(i,j-2,k,3)*solV (i,j-2,k,1:5,1)  &
434                                 )*invdy*one12th
435
436             endif
437             !-------------------------------------------------------------
438             Pe = Re*abs(solV(i,j,k,4,1))*dXi*Jac(k)/mu(i,j,k)
439
440             if (k .lt. 3 .or. k .gt. (nP(3)-2)) then
441               !H(i,j,k,1:5,1) = ( var(i,j,k+1,4)*solV (i,j,k+1,1:5,1)-var(i,j,k-1,4)*solV (i,j,k-1,1:5,1) )*invdXi*invJac(k)*haf
442               tmpR1  = ( var(i,j,k,4) + var(i,j,k+1,4) )*haf
443               tmpR2  = ( var(i,j,k,4) + var(i,j,k-1,4) )*haf
444
445               if(tmpR1.ge.0.0d0.and.tmpR2.ge.0.0d0) then
446                 H(i,j,k,1:5,1) =( tmpR1* solV (i,j,k  ,1:5,1) - tmpR2* solV (i,j,k-1,1:5,1) )/dz
447               endif
448
449               if(tmpR1 .lt. 0.0d0 .and. tmpR2 .lt. 0.0d0) then
450                 H(i,j,k,1:5,1) =( tmpR1* solV (i,j,k+1,1:5,1) - tmpR2* solV (i,j,k  ,1:5,1) )/dz
451               endif
452
453               if(tmpR1.lt.0.0d0.and.tmpR2.ge.0.0d0) then
454                 H(i,j,k,1:5,1) =( tmpR1* solV (i,j,k+1,1:5,1) - tmpR2* solV (i,j,k-1,1:5,1) )/dz
455               endif
456
457               if(tmpR1.ge.0.0d0.and.tmpR2.lt.0.0d0) then
458                 H(i,j,k,1:5,1) =( tmpR1* solV (i,j,k  ,1:5,1) - tmpR2* solV (i,j,k  ,1:5,1) )/dz
459               endif
460
461               else
```

```
462             !elseif((Pe .gt. PezVal) .or. (k .eq. bn(3)) .or. (k .eq. en(3))) then
463
464               tmpR1  = ( var(i,j,k,4) + var(i,j,k+1,4) )*haf
465               tmpR2  = ( var(i,j,k,4) + var(i,j,k-1,4) )*haf
466
467               if(tmpR1.ge.0.0d0.and.tmpR2.ge.0.0d0) then
468                 H(i,j,k,1:5,1) =( tmpR1*( 6.0d0*solV (i,j,k,1:5,1  ) + 3.0d0*solV (i,j,k+1,1:5,1) - 1.0d0*solV (i,j,k-1,1:5,1) ) -  &
469                                   tmpR2*( 6.0d0*solV (i,j,k-1,1:5,1) + 3.0d0*solV (i,j,k,1:5,1  ) - 1.0d0*solV (i,j,k-2,1:5,1) )    &
470                                 )*invdXi*invJac(k)/8.0d0
471               endif
472
473               if(tmpR1 .lt. 0.0d0 .and. tmpR2 .lt. 0.0d0) then
474                 H(i,j,k,1:5,1) =( tmpR1*( 6.0d0*solV (i,j,k+1,1:5,1) + 3.0d0*solV (i,j,k  ,1:5,1) - 1.0d0*solV (i,j,k+2,1:5,1) ) -  &
475                                   tmpR2*( 6.0d0*solV (i,j,k  ,1:5,1) + 3.0d0*solV (i,j,k-1,1:5,1) - 1.0d0*solV (i,j,k+1,1:5,1) )    &
476                                 )*invdXi*invJac(k)/8.0d0
477               endif
478
479               if(tmpR1.lt.0.0d0.and.tmpR2.ge.0.0d0) then
480                 H(i,j,k,1:5,1) =( tmpR1*( 6.0d0*solV (i,j,k+1,1:5,1) + 3.0d0*solV (i,j,k,1:5,1) - 1.0d0*solV (i,j,k+2,1:5,1) ) -  &
481                                   tmpR2*( 6.0d0*solV (i,j,k-1,1:5,1) + 3.0d0*solV (i,j,k,1:5,1) - 1.0d0*solV (i,j,k-2,1:5,1) )    &
482                                 )*invdXi*invJac(k)/8.0d0
483               endif
484
485               if(tmpR1.ge.0.0d0.and.tmpR2.lt.0.0d0) then
486                 H(i,j,k,1:5,1) =( tmpR1*( 6.0d0*solV (i,j,k,1:5,1) + 3.0d0*solV (i,j,k+1,1:5,1) - 1.0d0*solV (i,j,k-1,1:5,1) ) -  &
487                                   tmpR2*( 6.0d0*solV (i,j,k,1:5,1) + 3.0d0*solV (i,j,k-1,1:5,1) - 1.0d0*solV (i,j,k+1,1:5,1) )  &
488                                 )*invdXi*invJac(k)/8.0d0
489               endif
490
491             !else   ! Fourth order Central Differencing
492             ! H(i,j,k,1:5,1) =  ( -       var(i,j,k+2,4)*solV (i,j,k+2,1:5,1) &
493             !                   + 8.0d0*var(i,j,k+1,4)*solV (i,j,k+1,1:5,1) &
494             !                   - 8.0d0*var(i,j,k-1,4)*solV (i,j,k-1,1:5,1) &
495             !                   +       var(i,j,k-2,4)*solV (i,j,k-2,1:5,1) &
496             !                  )*invdXi*invJac(k)*one12th
497             !endif
498
499           enddo
500         enddo
501       enddo
502       !---------------------------------------------------------------------
503       !==========Discretization of Diffusive terms in Flux Form==========
504       do k=bs(3),es(3)
```

```
505         do j=bn(2),en(2)
506           do i=bn(1),en(1)
507
508             Ut  =var(i,j,k,2)
509             Vt  =var(i,j,k,3)
510             Wt  =var(i,j,k,4)
511             !-------------------------------------------------------------
512             dudx  =(var(i+1,j,k,2)-var(i-1,j,k,2))*haf*invdx
513             dudy  =(var(i,j+1,k,2)-var(i,j-1,k,2))*haf*invdy
514             !-------------------------------------------------------------
515             dvdx  =(var(i+1,j,k,3)-var(i-1,j,k,3))*haf*invdx
516             dvdy  =(var(i,j+1,k,3)-var(i,j-1,k,3))*haf*invdy
517             !-------------------------------------------------------------
518             dwdx  =(var(i+1,j,k,4)-var(i-1,j,k,4))*haf*invdx
519             dwdy  =(var(i,j+1,k,4)-var(i,j-1,k,4))*haf*invdy
520             !-------------------------------------------------------------
521             dTdx  =(var(i+1,j,k,6)-var(i-1,j,k,6))*haf*invdx
522             dTdy  =(var(i,j+1,k,6)-var(i,j-1,k,6))*haf*invdy
523             !-------------------------------------------------------------
524             if(k==1) then
525               dudz = (  var(i,j,1,2)*f1d2c0 + &
526                         var(i,j,2,2)*f1d2c1 + &
527                         var(i,j,3,2)*f1d2c2   &
528                       )
529               dvdz = (  var(i,j,1,3)*f1d2c0 + &
530                         var(i,j,2,3)*f1d2c1 + &
531                         var(i,j,3,3)*f1d2c2   &
532                       )
533               dwdz = (  var(i,j,1,4)*f1d2c0 + &
534                         var(i,j,2,4)*f1d2c1 + &
535                         var(i,j,3,4)*f1d2c2   &
536                       )
537               dTdz = (  var(i,j,1,6)*f1d2c0 + &
538                         var(i,j,2,6)*f1d2c1 + &
539                         var(i,j,3,6)*f1d2c2   &
540                       )
541             elseif(k==nP(3)) then
542               dudz = (  var(i,j,nP(3)  ,2)*b1d2c0 + &
543                         var(i,j,nP(3)-1,2)*b1d2c1 + &
544                         var(i,j,nP(3)-2,2)*b1d2c2   &
545                       )
546               dvdz = (  var(i,j,nP(3)  ,3)*b1d2c0 + &
547                         var(i,j,nP(3)-1,3)*b1d2c1 + &
```

```
548                       var(i,j,nP(3)-2,3)*b1d2c2   &
549                     )
550               dwdz = (  var(i,j,nP(3)  ,4)*b1d2c0 + &
551                       var(i,j,nP(3)-1,4)*b1d2c1 + &
552                       var(i,j,nP(3)-2,4)*b1d2c2   &
553                     )
554               dTdz = (  var(i,j,nP(3)  ,6)*b1d2c0 + &
555                       var(i,j,nP(3)-1,6)*b1d2c1 + &
556                       var(i,j,nP(3)-2,6)*b1d2c2   &
557                     )
558             else
559               dudz  =(var(i,j,k+1,2)-var(i,j,k-1,2))*haf*invdXi*invJac(k)
560               dvdz  =(var(i,j,k+1,3)-var(i,j,k-1,3))*haf*invdXi*invJac(k)
561               dwdz  =(var(i,j,k+1,4)-var(i,j,k-1,4))*haf*invdXi*invJac(k)
562               dTdz  =(var(i,j,k+1,6)-var(i,j,k-1,6))*haf*invdXi*invJac(k)
563             endif
564             !-----------------------------------------------------------
565             divg  =dudx + dvdy + dwdz
566             muT   =mu(i,j,k)
567             ktT   =kt(i,j,k)
568             PressT=var(i,j,k,7)
569
570             dissF =ggM*invRe*(  two3rd*muT*Ut*divg - two*muT*Ut*dudx - muT*Vt*(dvdx + dudy) - muT*Wt*(dwdx + dudz) )
571             dissG =ggM*invRe*( -muT*Ut*(dvdx + dudy) + two3rd*muT*Vt*divg - two*muT*Vt*dvdy - muT*Wt*(dwdy + dvdz) )
572             dissH =ggM*invRe*( -muT*Ut*(dwdx + dudz) - muT*Vt*(dwdy + dvdz) + two3rd*muT*Wt*divg - two*muT*Wt*dwdz )
573
574             !-----------------------------------------------------------
575             fD(i,j,k,1) = zer
576
577             fD(i,j,k,2) = PressT + two3rd*muT*invRe*divg - two*muT*invRe*dudx
578
579             fD(i,j,k,3) = -muT*invRe*( dvdx + dudy )
580
581             fD(i,j,k,4) = -muT*invRe*( dwdx + dudz )
582
583             fD(i,j,k,5) = -gama*ktT*invRe*invPr*dTdx + ggM*Ut*PressT + dissF
584
585             !-----------------------------------------------------------
586             gD(i,j,k,1) = zer
587
588             gD(i,j,k,2) = -muT*invRe*(dvdx + dudy)
589
590             gD(i,j,k,3) = PressT + two3rd*muT*invRe*divg - two*muT*invRe*dvdy
```

```
591
592             gD(i,j,k,4) = -muT*invRe*(dwdy + dvdz)
593
594             gD(i,j,k,5) = -gama*ktT*invRe*invPr*dTdy + ggM*Vt*PressT + dissG
595
596             !-------------------------------------------------------------
597             hD(i,j,k,1) = zer
598
599             hD(i,j,k,2) = -muT*invRe*(dwdx + dudz)
600
601             hD(i,j,k,3) = -muT*invRe*(dwdy + dvdz)
602
603             hD(i,j,k,4) = PressT + two3rd*muT*invRe*divg - two*muT*invRe*dwdz
604
605             hD(i,j,k,5) = -gama*ktT*invRe*invPr*dTdz + ggM*Wt*PressT + dissH
606             !-------------------------------------------------------------
607           enddo
608         enddo
609       enddo
610
611       !--------------------------------------------------------------------
612       call mpi_sendrecv(fD(en(1),bn(2),bn(3),1),1,yz1p5v,des(1),50,fD(bn(1)-1,bn(2),bn(3),1),1,yz1p5v,src(1),
      50,mpi_comm_world,STATUS,ierr)
613       call mpi_sendrecv(fD(bn(1),bn(2),bn(3),1),1,yz1p5v,src(1),50,fD(en(1)+1,bn(2),bn(3),1),1,yz1p5v,des(1),
      50,mpi_comm_world,STATUS,ierr)
614
615       call mpi_sendrecv(gD(bn(1),en(2),bn(3),1),1,xz1p5v,des(2),50,gD(bn(1),bn(2)-1,bn(3),1),1,xz1p5v,src(2),
      50,mpi_comm_world,STATUS,ierr)
616       call mpi_sendrecv(gD(bn(1),bn(2),bn(3),1),1,xz1p5v,src(2),50,gD(bn(1),en(2)+1,bn(3),1),1,xz1p5v,des(2),
      50,mpi_comm_world,STATUS,ierr)
617
618       call mpi_sendrecv(hD(bn(1),bn(2),en(3),1),1,xy1p5v,des(3),50,hD(bn(1),bn(2),bn(3)-1,1),1,xy1p5v,src(3),
      50,mpi_comm_world,STATUS,ierr)
619       call mpi_sendrecv(hD(bn(1),bn(2),bn(3),1),1,xy1p5v,src(3),50,hD(bn(1),bn(2),en(3)+1,1),1,xy1p5v,des(3),
      50,mpi_comm_world,STATUS,ierr)
620       !--------------------------------------------------------------------
621
622       do k=bn(3),en(3)
623         do j=bn(2),en(2)
624           do i=bn(1),en(1)
625             F(i,j,k,:,2) = ( fD(i+1,j,k,:) - fD(i-1,j,k,:) )*haf*invdx
626
627             G(i,j,k,:,2) = ( gD(i,j+1,k,:) - gD(i,j-1,k,:) )*haf*invdy
```

```
628
629             H(i,j,k,:,2) = ( hD(i,j,k+1,:) - hD(i,j,k-1,:) )*haf*invdXi*invJac(k)
630           enddo
631         enddo
632       enddo
633       !-----------------------------------------------------------------
634       do k=bn(3),en(3)
635         do j=bn(2),en(2)
636           do i=bn(1),en(1)
637             srcV(i,j,k,1)   = zer
638             srcV(i,j,k,2)   = haf*(tauxzL-tauxzU)*stat1Spc(k,1)
639             srcV(i,j,k,3:4) = zer
640             srcV(i,j,k,5)   = ggM*var(i,j,k,2)*haf*(tauxzL-tauxzU)*stat1Spc(k,1)
641             R(i,j,k,:,1) = srcV(i,j,k,:) - (F(i,j,k,:,1) + F(i,j,k,:,2) + G(i,j,k,:,1) + G(i,j,k,:,2) + H(i,j,k,:,1) + H(i,j,k,:,2))
642
643             if (iter.eq.1) then
644               solV(i,j,k,:,2) = solV(i,j,k,:,1)
645               solV(i,j,k,:,1) = solV(i,j,k,:,1) + haf*dt*R(i,j,k,:,1)
646             endif
647
648             if (iter.eq.2) then
649               solV(i,j,k,:,1) = solV(i,j,k,:,2) + dt*( R(i,j,k,:,1) )
650             endif
651
652           enddo
653         enddo
654       enddo
655
656       !-----------------------------------------------------------------------
657       call mpi_sendrecv(solV(en(1)-1,bn(2),bn(3),1,1),1,yz2p5v,des(1),50,solV(bn(1)-2,bn(2),bn(3),1,1),1,yz2p5v,src(1),
      50,mpi_comm_world,STATUS,ierr)
658       call mpi_sendrecv(solV(bn(1)  ,bn(2),bn(3),1,1),1,yz2p5v,src(1),50,solV(en(1)+1,bn(2),bn(3),1,1),1,yz2p5v,des(1),
      50,mpi_comm_world,STATUS,ierr)
659
660       call mpi_sendrecv(solV(bn(1),en(2)-1,bn(3),1,1),1,xz2p5v,des(2),50,solV(bn(1),bn(2)-2,bn(3),1,1),1,xz2p5v,src(2),
      50,mpi_comm_world,STATUS,ierr)
661       call mpi_sendrecv(solV(bn(1),bn(2)  ,bn(3),1,1),1,xz2p5v,src(2),50,solV(bn(1),en(2)+1,bn(3),1,1),1,xz2p5v,des(2),
      50,mpi_comm_world,STATUS,ierr)
662
663       call mpi_sendrecv(solV(bn(1),bn(2),en(3)-1,1,1),1,xy2p5v,des(3),50,solV(bn(1),bn(2),bn(3)-2,1,1),1,xy2p5v,src(3),
      50,mpi_comm_world,STATUS,ierr)
664       call mpi_sendrecv(solV(bn(1),bn(2),bn(3)  ,1,1),1,xy2p5v,src(3),50,solV(bn(1),bn(2),en(3)+1,1,1),1,xy2p5v,des(3),
      50,mpi_comm_world,STATUS,ierr)
```

```
665       !-------------------------------------------------------------------
666
667       ! Finding Primitive Variables
668       do k=bn(3),en(3)
669         do j=bn(2),en(2)
670           do i=bn(1),en(1)
671             var(i,j,k,1  )  = solV(i,j,k,1,1)
672             var(i,j,k,2:5)  = solV(i,j,k,2:5,1) / solV(i,j,k,1,1)
673             var(i,j,k,6)    = var(i,j,k,5) - ggM*haf*sum(var(i,j,k,2:4)**two)
674             var(i,j,k,7)    = var(i,j,k,1)*var(i,j,k,6) / (gama*Mac**two)
675           enddo
676         enddo
677       enddo
678
679       !-------------------------------------------------------------------
680       call mpi_sendrecv(var(en(1),bn(2),bn(3),1),1,yz1p7v,des(1),50,var(bn(1)-1,bn(2),bn(3),1),1,yz1p7v,src(1),
      50,mpi_comm_world,STATUS,ierr)
681       call mpi_sendrecv(var(bn(1),bn(2),bn(3),1),1,yz1p7v,src(1),50,var(en(1)+1,bn(2),bn(3),1),1,yz1p7v,des(1),
      50,mpi_comm_world,STATUS,ierr)
682
683       call mpi_sendrecv(var(bn(1),en(2),bn(3),1),1,xz1p7v,des(2),50,var(bn(1),bn(2)-1,bn(3),1),1,xz1p7v,src(2),
      50,mpi_comm_world,STATUS,ierr)
684       call mpi_sendrecv(var(bn(1),bn(2),bn(3),1),1,xz1p7v,src(2),50,var(bn(1),en(2)+1,bn(3),1),1,xz1p7v,des(2),
      50,mpi_comm_world,STATUS,ierr)
685
686       call mpi_sendrecv(var(bn(1),bn(2),en(3),1),1,xy1p7v,des(3),50,var(bn(1),bn(2),bn(3)-1,1),1,xy1p7v,src(3),
      50,mpi_comm_world,STATUS,ierr)
687       call mpi_sendrecv(var(bn(1),bn(2),bn(3),1),1,xy1p7v,src(3),50,var(bn(1),bn(2),en(3)+1,1),1,xy1p7v,des(3),
      50,mpi_comm_world,STATUS,ierr)
688
689       mu(bn(1)-1:en(1)+1,bn(2)-1:en(2)+1,bs(3):es(3)) = var(bn(1)-1:en(1)+1,bn(2)-1:en(2)+1,bs(3):es(3),6)**0.7d0
690
691       do j=bn(2),en(2)
692         do i=bn(1),en(1)
693           !------------------------Pressure BC--------------------------
694           !Simplifying z-momentum equation assuming impervious wall
695           if(bottom) then
696             dudxdz    = ( (   mu(i+1,j,1)*(   f1d2c0*var(i+1,j,1,2) &
697                                             + f1d2c1*var(i+1,j,2,2) &
698                                             + f1d2c2*var(i+1,j,3,2) &
699                                           )                         &
700                         ) -                                         &
701                         (   mu(i-1,j,1)*(   f1d2c0*var(i-1,j,1,2) &
```

```
                                        + f1d2c1*var(i-1,j,2,2) &
                                        + f1d2c2*var(i-1,j,3,2) &
                                      )                         &
                    )                                           &
                  ) *haf*invdx
dudzdx    = ( (   f1d2c0*mu(i,j,1)*var(i+1,j,1,2) &
                  + f1d2c1*mu(i,j,2)*var(i+1,j,2,2) &
                  + f1d2c2*mu(i,j,3)*var(i+1,j,3,2) &
                ) -                                 &
                (   f1d2c0*mu(i,j,1)*var(i-1,j,1,2) &
                  + f1d2c1*mu(i,j,2)*var(i-1,j,2,2) &
                  + f1d2c2*mu(i,j,3)*var(i-1,j,3,2) &
                )                                   &
                  ) *haf*invdx
drhouwdx  = (     var(i+1,j,1,1)*var(i+1,j,1,2)*var(i+1,j,1,4)  &
                  - var(i-1,j,1,1)*var(i-1,j,1,2)*var(i-1,j,1,4)  &
                )*haf*invdx
d2wdx2    = (     mu(i,j,1)*(     var(i+1,j,1,4)  &
                                +   var(i-1,j,1,4)  &
                                - 2*var(i  ,j,1,4)  &
                              )                     &
                )*invdx*invdx
dvdydz    = ( (   mu(i,j+1,1)*(   f1d2c0*var(i,j+1,1,3) &
                                    + f1d2c1*var(i,j+1,2,3) &
                                    + f1d2c2*var(i,j+1,3,3) &
                                  )                         &
                ) -                                         &
                (   mu(i,j-1,1)*(   f1d2c0*var(i,j-1,1,3) &
                                    + f1d2c1*var(i,j-1,2,3) &
                                    + f1d2c2*var(i,j-1,3,3) &
                                  )                         &
                )                                           &
                  ) *haf*invdy
dvdzdy    = ( (   f1d2c0*mu(i,j,1)*var(i,j+1,1,3) &
                  + f1d2c1*mu(i,j,2)*var(i,j+1,2,3) &
                  + f1d2c2*mu(i,j,3)*var(i,j+1,3,3) &
                ) -                                 &
                (   f1d2c0*mu(i,j,1)*var(i,j-1,1,3) &
                  + f1d2c1*mu(i,j,2)*var(i,j-1,2,3) &
                  + f1d2c2*mu(i,j,3)*var(i,j-1,3,3) &
                )                                   &
                  ) *haf*invdy
drhovwdy  = (   var(i,j+1,1,1)*var(i,j+1,1,3)*var(i,j+1,1,4)  &
```

```
745                         - var(i,j-1,1,1)*var(i,j-1,1,3)*var(i,j-1,1,4)  &
746                       )*haf*invdy
747           d2wdy2    = (     mu(i,j,1)*(     var(i,j+1,1,4)  &
748                                           +   var(i,j-1,1,4)  &
749                                           - 2*var(i,j  ,1,4)  &
750                                         )                     &
751                       )*invdy*invdy
752           mud2wdz2  = mu(i,j,1)*(   f2d2c0*var(i,j,1,4) &
753                                   + f2d2c1*var(i,j,2,4) &
754                                   + f2d2c2*var(i,j,3,4) &
755                                   + f2d2c3*var(i,j,4,4) &
756                                 )
757           dwdzdmudz = (     f1d2c0*var(i,j,1,4) &
758                           + f1d2c1*var(i,j,2,4) &
759                           + f1d2c2*var(i,j,3,4) &
760                       )*                        &
761                       (     f1d2c0*mu(i,j,1)    &
762                           + f1d2c1*mu(i,j,2)    &
763                           + f1d2c2*mu(i,j,3)    &
764                       )
765           drhow2dz  =   f1d2c0*var(i,j,1,1)*var(i,j,1,4)**two &
766                       + f1d2c1*var(i,j,2,1)*var(i,j,2,4)**two &
767                       + f1d2c2*var(i,j,3,1)*var(i,j,3,4)**two
768           drhowdt     = ((var(i,j,1,1)*var(i,j,1,4)) - rhow(i,j,1))*invdt
769
770           lhs       = invRe*(-dudxdz -dvdydz -d2wdx2 -d2wdy2 + two3rd*(dudzdx + dvdzdy - two*(mud2wdz2 + dwdzdmudz))) &
771                                  + drhow2dz + drhouwdx + drhovwdy + drhowdt
772
773           var(i,j,1,7) = -(lhs + f1d2c1*var(i,j,2,7) + f1d2c2*var(i,j,3,7) ) / f1d2c0
774         endif
775
776         if(top) then
777           dudxdz    = ( (   mu(i+1,j,nP(3))*(   b1d2c0*var(i+1,j,nP(3)  ,2) &
778                                               + b1d2c1*var(i+1,j,nP(3)-1,2) &
779                                               + b1d2c2*var(i+1,j,nP(3)-2,2) &
780                                             )                               &
781                         ) -                                                 &
782                         (   mu(i-1,j,nP(3))*(   b1d2c0*var(i-1,j,nP(3)  ,2) &
783                                               + b1d2c1*var(i-1,j,nP(3)-1,2) &
784                                               + b1d2c2*var(i-1,j,nP(3)-2,2) &
785                                             )                               &
786                         )                                                   &
787                       ) *haf*invdx
```

```
788             dudzdx    = ( (   b1d2c0*mu(i,j,nP(3)  )*var(i+1,j,nP(3)  ,2) &
789                             + b1d2c1*mu(i,j,nP(3)-1)*var(i+1,j,nP(3)-1,2) &
790                             + b1d2c2*mu(i,j,nP(3)-2)*var(i+1,j,nP(3)-2,2) &
791                           ) -                                             &
792                           (   b1d2c0*mu(i,j,nP(3)  )*var(i-1,j,nP(3)  ,2) &
793                             + b1d2c1*mu(i,j,nP(3)-1)*var(i-1,j,nP(3)-1,2) &
794                             + b1d2c2*mu(i,j,nP(3)-2)*var(i-1,j,nP(3)-2,2) &
795                           )                                               &
796                         ) *haf*invdx
797             drhouwdx  = (     var(i+1,j,nP(3),1)*var(i+1,j,nP(3),2)*var(i+1,j,nP(3),4)  &
798                             - var(i-1,j,nP(3),1)*var(i-1,j,nP(3),2)*var(i-1,j,nP(3),4)  &
799                         )*haf*invdx
800             d2wdx2    = (   mu(i,j,nP(3))*(     var(i+1,j,nP(3),4)  &
801                                               +   var(i-1,j,nP(3),4)  &
802                                               - 2*var(i  ,j,nP(3),4)  &
803                                             )                         &
804                         )*invdx*invdx
805             dvdydz    = ( (   mu(i,j+1,nP(3))*(   b1d2c0*var(i,j+1,nP(3)  ,3) &
806                                                   + b1d2c1*var(i,j+1,nP(3)-1,3) &
807                                                   + b1d2c2*var(i,j+1,nP(3)-2,3) &
808                                                 )                               &
809                           ) -                                                   &
810                           (   mu(i,j-1,nP(3))*(   b1d2c0*var(i,j-1,nP(3)  ,3) &
811                                                   + b1d2c1*var(i,j-1,nP(3)-1,3) &
812                                                   + b1d2c2*var(i,j-1,nP(3)-2,3) &
813                                                 )                               &
814                           )                                                     &
815                         ) *haf*invdy
816             dvdzdy    = ( (   b1d2c0*mu(i,j,nP(3)  )*var(i,j+1,nP(3)  ,3) &
817                             + b1d2c1*mu(i,j,nP(3)-1)*var(i,j+1,nP(3)-1,3) &
818                             + b1d2c2*mu(i,j,nP(3)-2)*var(i,j+1,nP(3)-2,3) &
819                           ) -                                             &
820                           (   b1d2c0*mu(i,j,nP(3)  )*var(i,j-1,nP(3)  ,3) &
821                             + b1d2c1*mu(i,j,nP(3)-1)*var(i,j-1,nP(3)-1,3) &
822                             + b1d2c2*mu(i,j,nP(3)-2)*var(i,j-1,nP(3)-2,3) &
823                           )                                               &
824                         ) *haf*invdy
825             drhovwdy  = (   var(i,j+1,nP(3),1)*var(i,j+1,nP(3),3)*var(i,j+1,nP(3),4)  &
826                           - var(i,j-1,nP(3),1)*var(i,j-1,nP(3),3)*var(i,j-1,nP(3),4)  &
827                         )*haf*invdy
828             d2wdy2    = (   mu(i,j,nP(3))*(     var(i,j+1,nP(3),4)  &
829                                               +   var(i,j-1,nP(3),4)  &
830                                               - 2*var(i,j  ,nP(3),4)  &
```

```
831                                       )                         &
832                     )*invdy*invdy
833           mud2wdz2  =   mu(i,j,nP(3))*(   b2d2c0*var(i,j,nP(3)  ,4) &
834                                         + b2d2c1*var(i,j,nP(3)-1,4) &
835                                         + b2d2c2*var(i,j,nP(3)-2,4) &
836                                         + b2d2c3*var(i,j,nP(3)-3,4) &
837                                       )
838           dwdzdmudz = (   b1d2c0*var(i,j,nP(3)  ,4) &
839                         + b1d2c1*var(i,j,nP(3)-1,4) &
840                         + b1d2c2*var(i,j,nP(3)-2,4) &
841                       ) *                           &
842                       (   b1d2c0*mu(i,j,nP(3)  )    &
843                         + b1d2c1*mu(i,j,nP(3)-1)    &
844                         + b1d2c2*mu(i,j,nP(3)-2)    &
845                       )
846           drhow2dz  =   b1d2c0*var(i,j,nP(3)  ,1)*var(i,j,nP(3)  ,4)**two &
847                       + b1d2c1*var(i,j,nP(3)-1,1)*var(i,j,nP(3)-1,4)**two &
848                       + b1d2c2*var(i,j,nP(3)-2,1)*var(i,j,nP(3)-2,4)**two
849           drhowdt   = ((var(i,j,nP(3),1)*var(i,j,nP(3),4)) - rhow(i,j,nP(3)))*invdt
850
851           lhs       = invRe*(-dudxdz -dvdydz -d2wdx2 -d2wdy2 + two3rd*(dudzdx + dvdzdy - two*(mud2wdz2 + dwdzdmudz))) &
852                             + drhow2dz + drhouwdx + drhovwdy + drhowdt
853
854           var(i,j,nP(3),7) = -(lhs + b1d2c1*var(i,j,nP(3)-1,7) + b1d2c2*var(i,j,nP(3)-2,7) ) / b1d2c0
855           !-------------------------------------------------------------
856         endif
857
858       enddo
859     enddo
860
861     kt(bs(1):es(1),bs(2):es(2),bs(3):es(3)) = var(bs(1):es(1),bs(2):es(2),bs(3):es(3),6)**0.7d0
862
863     if (bottom)     rhow(:,:,    1)   = var(:,:,    1,1)*var(:,:,    1,4)
864     if (top)        rhow(:,:,nP(3))   = var(:,:,nP(3),1)*var(:,:,nP(3),4)
865     !--------------------------------------------------------------------
866
867     Q = -2.0d0*Re*(tauxzL-tauxzU)/Sigma**two
868
869     ! Boundary Conditions
870     do j=bn(2),en(2)
871       do i=bn(1),en(1)
872
873         if(bottom) then
```

```
874             var(i,j,1,2)          = -(f1d2c1*var(i,j,2,2) + f1d2c2*var(i,j,3,2) + SC*Sigma*Q*haf)/(f1d2c0 - SC*Sigma*haf)
875             var(i,j,1,3)          = -(f1d2c1*var(i,j,2,3) + f1d2c2*var(i,j,3,3)                 )/(f1d2c0 - SC*Sigma*haf)
876             var(i,j,1,4)          = 0.0d0
877             var(i,j,1,6)          = 1.0d0
878             var(i,j,1,5)          = var(i,j,1,6) +  ggM*haf*sum(var(i,j,1,2:4)**two)
879             var(i,j,1,1)          = gama*Mac**two*var(i,j,1,7) / var(i,j,1,6)
880             solV(i,j,1,1,1)       = var(i,j,1,1)
881             solV(i,j,1,2:5,1)     = var(i,j,1,1)*var(i,j,1,2:5)
882           endif
883
884           if(top) then
885             var(i,j,nP(3),2)      = -(b1d2c1*var(i,j,nP(3)-1,2) + b1d2c2*var(i,j,nP(3)-2,2) + SC*Sigma*Q*haf)/(b1d2c0 - SC*Sigma*haf)
886             var(i,j,nP(3),3)      = -(b1d2c1*var(i,j,nP(3)-1,3) + b1d2c2*var(i,j,nP(3)-2,3)                 )/(b1d2c0 - SC*Sigma*haf)
887             var(i,j,nP(3),4)      = 0.0d0
888             var(i,j,nP(3),6)      = 1.0d0
889             var(i,j,nP(3),5)      = var(i,j,nP(3),6) +  ggM*haf*sum(var(i,j,nP(3),2:4)**2)
890             var(i,j,nP(3),1)      = gama*Mac**two*var(i,j,nP(3),7) / var(i,j,nP(3),6)
891             solV(i,j,nP(3),1,1)   = var(i,j,nP(3),1)
892             solV(i,j,nP(3),2:5,1) = var(i,j,nP(3),1)*var(i,j,nP(3),2:5)
893           endif
894
895         enddo
896       enddo
897
898       !-----------------------------------------------------------------
899       if (bottom) then
900         call mpi_sendrecv(var(en(1),bn(2),1,1),1,liney7v,des(1),50,var(bn(1)-1,bn(2),1,1),1,liney7v,src(1),
      50,mpi_comm_world,STATUS,ierr)
901         call mpi_sendrecv(var(bn(1),bn(2),1,1),1,liney7v,src(1),50,var(en(1)+1,bn(2),1,1),1,liney7v,des(1),
      50,mpi_comm_world,STATUS,ierr)
902
903         call mpi_sendrecv(var(bn(1),en(2),1,1),1,linex7v,des(2),50,var(bn(1),bn(2)-1,1,1),1,linex7v,src(2),
      50,mpi_comm_world,STATUS,ierr)
904         call mpi_sendrecv(var(bn(1),bn(2),1,1),1,linex7v,src(2),50,var(bn(1),en(2)+1,1,1),1,linex7v,des(2),
      50,mpi_comm_world,STATUS,ierr)
905       endif
906       if (top) then
907         call mpi_sendrecv(var(en(1),bn(2),nP(3),1),1,liney7v,des(1),50,var(bn(1)-1,bn(2),nP(3),1),1,liney7v,src(1),
      50,mpi_comm_world,STATUS,ierr)
908         call mpi_sendrecv(var(bn(1),bn(2),nP(3),1),1,liney7v,src(1),50,var(en(1)+1,bn(2),nP(3),1),1,liney7v,des(1),
      50,mpi_comm_world,STATUS,ierr)
909
910         call mpi_sendrecv(var(bn(1),en(2),nP(3),1),1,linex7v,des(2),50,var(bn(1),bn(2)-1,nP(3),1),1,linex7v,src(2),
```

```
     50,mpi_comm_world,STATUS,ierr)
911        call mpi_sendrecv(var(bn(1),bn(2),nP(3),1),1,linex7v,src(2),50,var(bn(1),en(2)+1,nP(3),1),1,linex7v,des(2),
     50,mpi_comm_world,STATUS,ierr)
912      endif
913
914    enddo
915    !=========================RK2 Sub-steps (ends)=======================
916    !=====================Check for unphysical values====================
917    do k=bs(3),es(3)
918      do j=bs(2),es(2)
919        do i=bs(1),es(1)
920
921          if (var(i,j,k,1) < 0.0d0 .or. var(i,j,k,6) < 0.0d0) then
922            write(6,102) nstep,id,i,j,k,var(i,j,k,1),var(i,j,k,6)
923            isBlown=.true.
924          endif
925
926        enddo
927      enddo
928    enddo
929
930    if(isBlown) then
931
932      WRITE(filename,'(a,i3.3,a)')"../debug/3D",id+1,".dat"
933      OPEN (UNIT=10,FILE=filename)
934
935      write(10,103) 'TITLE ="',time,nstep,'"'
936      write(10,*) 'Variables ="x","y","z","Rho","U","V","W","E","T","P"'
937      write(10,104) 'ZONE k=',es(3)-bs(3)+1,',j=',es(2)-bs(2)+1,',i=',es(1)-bs(1)+1,',DATAPACKING="POINT"'
938
939      do k=bs(3),es(3)
940        write(10,*)
941        do j=bs(2),es(2)
942          write(10,*)
943          do i=bs(1),es(1)
944            write(10,116) x1(i),x2(j),x3(k),var(i,j,k,1),var(i,j,k,2),var(i,j,k,3),var(i,j,k,4) &
945                                                    ,var(i,j,k,5),var(i,j,k,6),var(i,j,k,7)
946          end do
947        end do
948      enddo
949      close(10)
950      stop
951    endif
```

```
952
953     102 format(I10,4(I4),2(F8.4))
954     116 format(3(1X,F7.4),7(1X,E12.5))
955
956     !==================Write output files (starts)=======================
957     if(mod(nstep,1000).eq.0) then
958
959       if(mod(nstep/1000,2).eq.0) then
960         WRITE(filename,'(a,i3.3,a)')"../output/3D",id+1,".dat"
961       else
962         WRITE(filename,'(a,i3.3,a)')"../output/3D",id+1,"_.dat"
963       endif
964
965       OPEN (UNIT=10,FILE=filename)
966
967       write(10,103) 'TITLE ="',time,nstep,'"'
968       write(10,*) 'Variables ="x","y","z","Rho","U","V","W","E","T","P"'
969       write(10,104) 'ZONE k=',es(3)-bs(3)+1,',j=',es(2)-bs(2)+1,',i=',es(1)-bs(1)+1,',DATAPACKING="POINT"'
970
971       do k=bs(3),es(3)
972         write(10,*)
973         do j=bs(2),es(2)
974           write(10,*)
975           do i=bs(1),es(1)
976             write(10,101) x1(i),x2(j),x3(k),var(i,j,k,1),var(i,j,k,2),var(i,j,k,3),var(i,j,k,4) &
977                                                    ,var(i,j,k,5),var(i,j,k,6),var(i,j,k,7)
978           end do
979         end do
980       enddo
981       close(10)
982       103 format(A,F20.10,I9,A)
983       104 format(A,I3,A,I3,A,I3,A)
984
985       !---------------------------------------------------------------------
986       call mpi_gather(bs(1),3,mpi_integer,bb(1,0),3,mpi_integer,0,mpi_comm_world,ierr)
987       call mpi_gather(es(1),3,mpi_integer,ee(1,0),3,mpi_integer,0,mpi_comm_world,ierr)
988       !----------------------Write Domain Info--------------------------
989       if(master) then
990         open(unit=10,file="../output/domInfo.txt")
991         write(10,105) nP(1),nP(2),nP(3)
992         write(10,106) nproc
993         do i=0,nproc-1
994           write(10,107) bb(1,i),ee(1,i),bb(2,i),ee(2,i),bb(3,i),ee(3,i)
```

```
995         enddo
996         close(10)
997         105 format(3(i4))
998         106 format(i3)
999         107 format(6(i4,1x))
1000      endif
1001
1002    endif
1003    !==================Write output files (ends)==========================
1004
1005    !==================Write Diagnostic Files (starts)====================
1006
1007    if (mod(nstep,10).eq.0) then
1008      !-----------------------Write Kinetic Energy-----------------------
1009      tmpR1=0.0d0
1010      tmpR2 =0.0d0
1011      do k=bs(3),es(3)
1012        do j=bs(2),es(2)
1013          do i=bs(1),es(1)
1014            tmpR1 = tmpR1 + sum(var(i,j,k,2:4)**two)
1015          enddo
1016        enddo
1017      enddo
1018
1019      CALL MPI_Reduce(tmpR1,tmpR2,1,mpi_double_precision,mpi_sum,0,mpi_comm_world,ierr)
1020
1021      if(master) then
1022        tmpR2 = tmpR2/product(nP)
1023        open(10,file="../output/KE.dat",access="append")
1024        write(10,108) time,tmpR2
1025        close(10)
1026      endif
1027      108 format(2(1X,F15.7))
1028      !-----------------------Write Bulk Density-----------------------
1029      Ayz = x3(nP(3))*x2(nP(2))
1030
1031      tmpA1 =0.0d0
1032      do i=1,5
1033        tmpR1 =0.0d0
1034
1035        if(plane(i) > bn(1) .and. plane(i) < en(1)) then
1036          do k=bs(3),en(3)
1037            do j=bs(2),en(2)
```

```
1038              dAyz = dy*(x3(k+1)-x3(k))
1039              tmpR1 =tmpR1 + quar*( var(plane(i),j  ,k  ,1) + &
1040                                    var(plane(i),j+1,k  ,1) + &
1041                                    var(plane(i),j  ,k+1,1) + &
1042                                    var(plane(i),j+1,k+1,1)   &
1043                                   )*dAyz
1044            enddo
1045          enddo
1046        endif
1047
1048        CALL MPI_Reduce(tmpR1,tmpA1(i),1,mpi_double_precision,mpi_sum,0,mpi_comm_world,ierr)
1049      enddo
1050
1051      if(master) then
1052        tmpA1=tmpA1/Ayz
1053        open (unit=10,file="../output/rhobulk.dat",access="append")
1054        write(10,109)time,tmpA1(1),tmpA1(2),tmpA1(3),tmpA1(4),tmpA1(5)
1055        close(10)
1056      endif
1057      109 format(6(1X,F15.7))
1058      !------------------------Write Bulk Velocity-----------------------
1059      tmpA1 =0.0d0
1060      do i=1,5
1061        tmpR1 =0.0d0
1062
1063        if(plane(i) > bn(1) .and. plane(i) < en(1)) then
1064          do k=bs(3),en(3)
1065            do j=bs(2),en(2)
1066              dAyz = dy*(x3(k+1)-x3(k))
1067              tmpR1 =tmpR1 + quar*( var(plane(i),j  ,k  ,2) + &
1068                                    var(plane(i),j+1,k  ,2) + &
1069                                    var(plane(i),j  ,k+1,2) + &
1070                                    var(plane(i),j+1,k+1,2)   &
1071                                   )*dAyz
1072            enddo
1073          enddo
1074        endif
1075
1076        CALL MPI_Reduce(tmpR1,tmpA1(i),1,mpi_double_precision,mpi_sum,0,mpi_comm_world,ierr)
1077      enddo
1078
1079      if(master) then
1080        tmpA1=tmpA1/Ayz
```

```
1081          open (unit=10,file="../output/Ubulk.dat",access="append")
1082          write(10,109)time,tmpA1(1),tmpA1(2),tmpA1(3),tmpA1(4),tmpA1(5)
1083          close(10)
1084        endif
1085
1086        !-------------------------Write Massflow--------------------------
1087        tmpA1 =0.0d0
1088        do i=1,5
1089          tmpR1 =0.0d0
1090
1091          if(plane(i) > bn(1) .and. plane(i) < en(1)) then
1092            do k=bs(3),en(3)
1093              do j=bs(2),en(2)
1094                dAyz = dy*(x3(k+1)-x3(k))
1095                tmpR1 =tmpR1 + quar*( var(plane(i),j  ,k  ,1)*var(plane(i),j  ,k  ,2) + &
1096                                      var(plane(i),j+1,k  ,1)*var(plane(i),j+1,k  ,2) + &
1097                                      var(plane(i),j  ,k+1,1)*var(plane(i),j  ,k+1,2) + &
1098                                      var(plane(i),j+1,k+1,1)*var(plane(i),j+1,k+1,2)   &
1099                                     )*dAyz
1100              enddo
1101            enddo
1102          endif
1103
1104          CALL MPI_Reduce(tmpR1,tmpA1(i),1,mpi_double_precision,mpi_sum,0,mpi_comm_world,ierr)
1105        enddo
1106
1107        if(master) then
1108          tmpA1=tmpA1/Ayz
1109          open (unit=10,file="../output/massflow.dat",access="append")
1110          write(10,109)time,tmpA1(1),tmpA1(2),tmpA1(3),tmpA1(4),tmpA1(5)
1111          close(10)
1112        endif
1113
1114      endif
1115      !====================Write Diagnostic Files (ends)====================
1116
1117      !==================Wall shear stress (starts)==========================
1118      !==================Wall shear stress (ends)============================
1119      if(mod(nstep,istat)==0) then
1120        !---------------------------------------------------------------------
1121        tmpR1=0.0d0
1122        tmpR2=0.0d0
1123        if(bottom) then
```

```
1124
1125          do j=bn(2),en(2)
1126            do i=bn(1),en(1)
1127              dwdx = mu(i,j,1) * (var(i+1,j,1,4) - var(i-1,j,1,4)) *haf*invdx !zero for impervious bottom wall
1128
1129              dudz = mu(i,j,1)* (   f1d2c0*var(i,j,1,2) &
1130                                  + f1d2c1*var(i,j,2,2) &
1131                                  + f1d2c2*var(i,j,3,2) &
1132                                )
1133
1134              tmpR1 = tmpR1 + (dwdx + dudz)*invRe
1135
1136              dwdy = mu(i,j,1) * (var(i,j+1,1,4) - var(i,j-1,1,4)) *haf*invdy !zero for impervious bottom wall
1137
1138              dvdz = mu(i,j,1)* (   f1d2c0*var(i,j,1,3) &
1139                                  + f1d2c1*var(i,j,2,3) &
1140                                  + f1d2c2*var(i,j,3,3) &
1141                                )
1142
1143              tmpR2 = tmpR2 + (dwdy + dvdz)*invRe
1144            enddo
1145          enddo
1146
1147        endif
1148
1149        CALL MPI_Reduce(tmpR1,tauxzL,1,mpi_double_precision,mpi_sum,0,mpi_comm_world,ierr)
1150        CALL MPI_Reduce(tmpR2,tauyzL,1,mpi_double_precision,mpi_sum,0,mpi_comm_world,ierr)
1151        tauxzL= tauxzL/dble((nP(1)-2)*(nP(2)-2))
1152        tauyzL= tauyzL/dble((nP(1)-2)*(nP(2)-2))
1153
1154        call mpi_bcast(tauxzL,1,mpi_double_precision,0,mpi_comm_world,ierr)
1155        call mpi_bcast(tauyzL,1,mpi_double_precision,0,mpi_comm_world,ierr)
1156        !-----------------------------------------------------------------
1157
1158        tmpR1=0.0d0
1159        tmpR2=0.0d0
1160        if(top) then
1161
1162          do j=bn(2),en(2)
1163            do i=bn(1),en(1)
1164              dwdx = mu(i,j,nP(3)) * (var(i+1,j,nP(3),4) - var(i-1,j,nP(3),4)) *haf*invdx !zero for impervious top wall
1165
1166              dudz = mu(i,j,nP(3)) * (    b1d2c0*var(i,j,nP(3)  ,2) &
```

```
1167                                      + b1d2c1*var(i,j,nP(3)-1,2) &
1168                                      + b1d2c2*var(i,j,nP(3)-2,2) &
1169                                     )
1170
1171            tmpR1 = tmpR1 + (dwdx + dudz)*invRe
1172
1173            dwdy = mu(i,j,nP(3)) * (var(i,j+1,nP(3),4) - var(i,j-1,nP(3),4)) *haf*invdy !zero for impervious top wall
1174
1175            dvdz = mu(i,j,nP(3)) * (    b1d2c0*var(i,j,nP(3)  ,3) &
1176                                      + b1d2c1*var(i,j,nP(3)-1,3) &
1177                                      + b1d2c2*var(i,j,nP(3)-2,3) &
1178                                     )
1179
1180            tmpR2 = tmpR2 + (dwdy + dvdz)*invRe
1181          enddo
1182        enddo
1183
1184      endif
1185
1186      CALL MPI_Reduce(tmpR1,tauxzU,1,mpi_double_precision,mpi_sum,0,mpi_comm_world,ierr)
1187      CALL MPI_Reduce(tmpR2,tauyzU,1,mpi_double_precision,mpi_sum,0,mpi_comm_world,ierr)
1188      tauxzU= tauxzU/dble(((nP(1)-2)*(nP(2)-2)))
1189      tauyzU= tauyzU/dble(((nP(1)-2)*(nP(2)-2)))
1190
1191      call mpi_bcast(tauxzU,1,mpi_double_precision,0,mpi_comm_world,ierr)
1192      call mpi_bcast(tauyzU,1,mpi_double_precision,0,mpi_comm_world,ierr)
1193      !---------------------------------------------------------------------
1194
1195      if(master) then
1196        open(10,file="../output/tauWall.dat",access="append")
1197  !     write(10,110)time, tauxzL, tauxzU
1198        write(10,110)time, tauxzL+tauyzL, tauxzU+tauyzU, tauxzL, tauxzU, tauyzL, tauyzU
1199        close(10)
1200      endif
1201  !   110 format(3(1X,F15.7))
1202      110 format(7(1X,F15.7))
1203
1204    end if
1205    !==================Spatial Statistics (starts)=======================
1206
1207    if(mod(nstep,istat)==0) then
1208      nXY=nP(1)*nP(2)
1209
```

```
1210        do k=1,nP(3)
1211          loc=0.0d0
1212
1213          if(k >= bs(3) .and. k <= es(3)) then
1214            do j=bs(2),es(2)
1215              do i=bs(1),es(1)
1216                loc(i,j,1:7)  = var(i,j,k,1:7)
1217              enddo
1218            enddo
1219          endif
1220
1221          do p=1,7
1222            call mpi_reduce(sum(loc(:,:,p)   )/nXY,stat1Spc(k,p),1,mpi_double_precision,mpi_sum,0,mpi_comm_world,ierr)
1223            call mpi_reduce(sum(loc(:,:,p)**2)/nXY,stat2Spc(k,p),1,mpi_double_precision,mpi_sum,0,mpi_comm_world,ierr)
1224          enddo
1225
1226          call mpi_reduce(sum(loc(:,:,1)*loc(:,:,4))/nXY,corrSpc(k,1),1,mpi_double_precision,mpi_sum,0,mpi_comm_world,ierr)
1227          call mpi_reduce(sum(loc(:,:,1)*loc(:,:,6))/nXY,corrSpc(k,2),1,mpi_double_precision,mpi_sum,0,mpi_comm_world,ierr)
1228          call mpi_reduce(sum(loc(:,:,2)*loc(:,:,3))/nXY,corrSpc(k,3),1,mpi_double_precision,mpi_sum,0,mpi_comm_world,ierr)
1229          call mpi_reduce(sum(loc(:,:,2)*loc(:,:,4))/nXY,corrSpc(k,4),1,mpi_double_precision,mpi_sum,0,mpi_comm_world,ierr)
1230          call mpi_reduce(sum(loc(:,:,3)*loc(:,:,4))/nXY,corrSpc(k,5),1,mpi_double_precision,mpi_sum,0,mpi_comm_world,ierr)
1231          call mpi_reduce(sum(loc(:,:,3)*loc(:,:,6))/nXY,corrSpc(k,6),1,mpi_double_precision,mpi_sum,0,mpi_comm_world,ierr)
1232          call mpi_reduce(sum(loc(:,:,6)*loc(:,:,4))/nXY,corrSpc(k,7),1,mpi_double_precision,mpi_sum,0,mpi_comm_world,ierr)
1233          call mpi_reduce(sum(loc(:,:,7)*loc(:,:,4))/nXY,corrSpc(k,8),1,mpi_double_precision,mpi_sum,0,mpi_comm_world,ierr)
1234          call mpi_reduce(sum(loc(:,:,7)*loc(:,:,6))/nXY,corrSpc(k,9),1,mpi_double_precision,mpi_sum,0,mpi_comm_world,ierr)
1235        enddo
1236
1237        call mpi_bcast(stat1Spc(1,1),nP(3),mpi_double_precision,0,mpi_comm_world,ierr)
1238
1239        if(master) then
1240          !-----------------------------------------------------------------
1241          open(unit=10,file="../output/moment1st.dat",access="append")
1242          write(10,111,advance="no") time
1243          do k=1,nP(3)
1244            write(10,112,advance="no") stat1Spc(k,1),stat1Spc(k,2),stat1Spc(k,3),stat1Spc(k,4)  &
1245                                                   ,stat1Spc(k,5),stat1Spc(k,6),stat1Spc(k,7)
1246          enddo
1247          write (10, *)
1248          close(10)
1249          111 format(1(1X,F15.7))
1250          112 format(7(1X,F15.7))
1251          !-----------------------------------------------------------------
1252          open(unit=10,file="../output/moment2nd.dat",access="append")
```

```
1253          write(10,111,advance="no") time
1254          do k=1,nP(3)
1255            write(10,112,advance="no") stat2Spc(k,1),stat2Spc(k,2),stat2Spc(k,3),stat2Spc(k,4)  &
1256                                                    ,stat2Spc(k,5),stat2Spc(k,6),stat2Spc(k,7)
1257          enddo
1258          write (10, *)
1259          close(10)
1260          !---------------------------------------------------------------------
1261          open(unit=10,file="../output/momentJoint.dat",access="append")
1262          write(10,111,advance="no")time
1263          do k=1,nP(3)
1264            write(10,113,advance="no") corrSpc(k,1),corrSpc(k,2),corrSpc(k,3) &
1265                                      ,corrSpc(k,4),corrSpc(k,5),corrSpc(k,6) &
1266                                      ,corrSpc(k,7),corrSpc(k,8),corrSpc(k,9)
1267          enddo
1268          write (10,*)
1269          close(10)
1270          113 format(9(1X,F15.7))
1271          !---------------------------------------------------------------------
1272    !   if(master) then
1273          open(10,file="../output/uv_slip.dat",access="append")
1274          write(10,110)time, stat1Spc(1,2), stat1Spc(1,3)
1275          close(10)
1276    !   endif
1277        endif
1278
1279      endif
1280      !====================Spatial Statistics (ends)=======================
1281
1282      if(master .and. mod(nstep,10)==0) then
1283        call cpu_time(tEnd)
1284        write(6,114) nstep,time," | CPU Time:",tEnd-tStart,"sec for 10 iters | ", dateTime()
1285        write(6,'(A)')"-------------------------------------------------------------------------------"
1286      endif
1287      114  format(I10,2X,F10.5,A,F6.3,2(1X,A))
1288
1289    END DO
1290    !=======================Main time loop (ends)=========================
1291
1292    CALL mpi_type_free(yz1p,ierr)
1293    CALL mpi_type_free(xz1p,ierr)
1294    CALL mpi_type_free(xy1p,ierr)
1295
```

```
1296   CALL mpi_type_free(yz1p5v,ierr)
1297   CALL mpi_type_free(xz1p5v,ierr)
1298   CALL mpi_type_free(xy1p5v,ierr)
1299
1300   CALL mpi_type_free(yz2p5v,ierr)
1301   CALL mpi_type_free(xz2p5v,ierr)
1302   CALL mpi_type_free(xy2p5v,ierr)
1303
1304   CALL mpi_type_free(yz1p7v,ierr)
1305   CALL mpi_type_free(xz1p7v,ierr)
1306   CALL mpi_type_free(xy1p7v,ierr)
1307
1308   CALL mpi_finalize(ierr)
1309   !==========================================================================
1310   if(master) write(*,*) "Compressible Channel Flow Solver exited at :",dateTime()
1311   !==========================================================================
1312   end program chanCompr
```