

```

!-----
! Skeleton 2-1/2D Electromagnetic GPU PIC code
! written by Viktor K. Decyk, UCLA
    program gpubpic2
    use gpubpush2
    use gpulib2
    use gpufft2
    implicit none
    integer, parameter :: indx = 9, indy = 9
    integer, parameter :: npx = 3072, npy = 3072
    integer, parameter :: ndim = 3
    real, parameter :: tend = 10.0, dt = 0.04, qme = -1.0
    real, parameter :: vtx = 1.0, vty = 1.0, vx0 = 0.0, vy0 = 0.0
    real, parameter :: vtz = 1.0, vz0 = 0.0
    real :: ax = .912871, ay = .912871, ci = 0.1
! idimp = dimension of phase space = 5
! relativity = (no,yes) = (0,1) = relativity is used
    integer :: idimp = 5, ipbc = 1, relativity = 1
    real :: wke = 0.0, we = 0.0, wf = 0.0, wm = 0.0, wt = 0.0
! sorting tiles
    integer :: mx = 16, my = 16
! fraction of extra particles needed for particle management
    real :: xtras = 0.2
! declare scalars for standard code
    integer :: np, nx, ny, nxh, nyh, nxh1, nxe, nye, nxeh, nxyh, nxhy
    integer :: mx1, my1, mxy1, ntime, nloop, isign
    real :: qbme, affp, dth
    real, dimension(1) :: sum
!
! declare scalars for GPU code
    integer :: nblock = 128
! nscache = (0,1,2) = (no,small,big) cache size
    integer :: nscache = 1
    integer :: mmcc, nppmx, nppmx0, ntmax, npbm
    integer :: nxhd
    integer, dimension(1) :: irc
!
! declare arrays for standard code
    real, dimension(:,:), pointer :: part
    complex, dimension(:,:), pointer :: ffct
    integer, dimension(:), pointer :: mixup
    complex, dimension(:), pointer :: sct
!
! declare arrays for GPU code
    integer, dimension(2) :: g_ge = 0.0
    integer, dimension(2) :: g_cue = 0.0, g_fxyze = 0.0, g_bxyze = 0.0
    integer, dimension(2) :: g_ffct = 0.0
    integer, dimension(2) :: g_mixup = 0.0, g_sct = 0.0
    integer, dimension(2) :: g_q = 0.0, g_cu = 0.0
    integer, dimension(2) :: g_qt = 0.0, g_cut = 0.0
    integer, dimension(2) :: g_fxyz = 0.0, g_hxyz = 0.0
    integer, dimension(2) :: g_fxyzt = 0.0, g_hxyzt = 0.0
    integer, dimension(2) :: g_exyzt = 0.0, g_bxyzt = 0.0
    integer, dimension(2) :: g_wke = 0.0, g_we = 0.0

```

```

integer, dimension(2) :: g_wf = 0.0, g_wm = 0.0
integer, dimension(2) :: g_ppart = 0.0, g_ppbuff = 0.0
integer, dimension(2) :: g_kpic = 0.0
integer, dimension(2) :: g_ncl = 0.0, g_ihole = 0.0
integer, dimension(2) :: g_sum = 0.0
integer, dimension(2) :: g_irc = 0.0
complex, dimension(:,:), pointer :: qt
complex, dimension(:,:,:), pointer :: fxyz
real, dimension(:,:,:), pointer :: ppart
integer, dimension(:), pointer :: kpic

!
! declare and initialize timing data
real :: time
integer, dimension(4) :: itime
double precision :: dtime
real :: tdpost = 0.0, tguard = 0.0, tfft = 0.0, tfield = 0.0
real :: tdjpost = 0.0, tpush = 0.0, tsort = 0.0

!
! initialize scalars for standard code
np = npx*nty; nx = 2*indx; ny = 2*indy; nxh = nx/2; nyh = ny/2
nxh1 = nxh + 1; nxe = nx + 2; nye = ny + 1; nxeh = nxe/2
nxyh = max(nx,ny)/2; nxhy = max(nxh,ny)
mx1 = (nx - 1)/mx + 1; my1 = (ny - 1)/my + 1; mxy1 = mx1*my1
nloop = tend/dt + .0001; ntime = 0
qbme = qme
affp = real(nx*ny)/real(np)
dth = 0.0
! set size for FFT arrays
nxhd = nxh1

!
! allocate and initialize data for standard code
allocate(part(idimp,np))
allocate(ffct(nyh,nxh))
allocate(mixup(nxhy),sct(nxyh))
allocate(kpic(mxy1))
allocate(qt(ny,nxh1),fxyz(ny,ndim,nxh1))

!
! set up GPU
irc = 0
call gpu_setgbsize(nblock)
call init_cu(0,irc(1))
if (irc(1) /= 0) then
    write (*,*) 'CUDA initialization error!'
    stop
endif

! obtain compute capability
mmcc = getmmcc()
if (mmcc < 20) then
    write (*,*) 'compute capability 2.x or higher required'
    stop
endif

! set cache size
call gpu_set_cache_size(nscache)

!

```

```

! allocate data for GPU code
  call gpu_fallocate(g_ge,nxe*nye,irc(1))
  call gpu_fallocate(g_cue,ndim*nxe*nye,irc(1))
  call gpu_fallocate(g_fxyze,ndim*nxe*nye,irc(1))
  call gpu_fallocate(g_bxyze,ndim*nxe*nye,irc(1))
  call gpu_callocate(g_ffct,nyh*nxh,irc(1))
  call gpu_iallocate(g_mixup,nxhy,irc(1))
  call gpu_callocate(g_sct,nxyh,irc(1))
  call gpu_callocate(g_q,nxhd*ny,irc(1))
  call gpu_callocate(g_cu,nxhd*ndim*ny,irc(1))
  call gpu_callocate(g_qt,ny*nxh1,irc(1))
  call gpu_callocate(g_cut,ny*ndim*nxh1,irc(1))
  call gpu_callocate(g_fxzyz,nxhd*ndim*ny,irc(1))
  call gpu_callocate(g_hxyzyz,nxhd*ndim*ny,irc(1))
  call gpu_callocate(g_fxzyzt,ny*ndim*nxh1,irc(1))
  call gpu_callocate(g_hxyzyzt,ny*ndim*nxh1,irc(1))
  call gpu_callocate(g_exzyzt,ny*ndim*nxh1,irc(1))
  call gpu_callocate(g_bxyzyzt,ny*ndim*nxh1,irc(1))
  call gpu_fallocate(g_wke,mxy1,irc(1))
  call gpu_fallocate(g_we,nxh1,irc(1))
  call gpu_fallocate(g_wf,nxh1,irc(1))
  call gpu_fallocate(g_wm,nxh1,irc(1))
  call gpu_fallocate(g_sum,1,irc(1))
  if (irc(1) /= 0) then
    write (*,*) 'GPU allocate error!'
    stop
  endif
!
! prepare fft tables
  call WFFT2RINIT(mixup,sct,indx,indy,nxhy,nxyh)
! prepare NVIDIA ffts
  call gpuffft2rrcuinit(nx,ny,ndim)
  call gpuffft2cuinit(nx,ny,ndim)
! calculate form factors
  isign = 0
  call POIS23T(qt,fxzyzt,isign,ffct,ax,ay,affp,we,nx,ny,nxh1,ny,nxh, &
    &nyh)
! copy in solver arrays to GPU
  call gpu_icopyin(mixup,g_mixup,nxhy)
  call gpu_ccopyin(sct,g_sct,nxyh)
  call gpu_ccopyin(ffct,g_ffct,nyh*nxh)
! initialize electrons
  call DISTR2H(part,vtx,vty,vtz,vx0,vy0,vz0,npx,npj,idimp,np,nx,ny, &
    &ipbc)
!
! initialize transverse electromagnetic fields
  call gpu_zcmem(g_exzyzt,ny*ndim*nxh1)
  call gpu_zcmem(g_bxyzyzt,ny*ndim*nxh1)
!
! find number of particles in each of mx, my tiles: updates kplic, nppmx
  call DBLKP2L(part,kplic,nppmx,idimp,np,mx,my,mx1,mxy1,irc)
  if (irc(1) /= 0) then
    write (*,*) 'DBLKP2L error, irc=', irc
    stop
  endif

```

```

        endif
! allocate vector particle data
        nppmx0 = (1.0 + xtras)*nppmx
        ntmax = 0.5*xtras*nppmx
        npbm = 0.5*xtras*nppmx
! align data to warp size
        nppmx0 = 32*((nppmx0 - 1)/32 + 1)
        ntmax = 32*(ntmax/32 + 1)
        npbm = 32*((npbm - 1)/32 + 1)
        call gpu_fallocate(g_ppart,nppmx0*idimp*mxy1,irc(1))
        call gpu_fallocate(g_ppbuff,npbm*idimp*mxy1,irc(1))
        call gpu_iallocate(g_kpic,mxy1,irc(1))
        call gpu_iallocate(g_ncl,8*mxy1,irc(1))
        call gpu_iallocate(g_ihole,2*(ntmax+1)*mxy1,irc(1))
        call gpu_iallocate(g_irc,1,irc(1))
        if (irc(1) /= 0) then
            write (*,*) 'GPU allocate error!'
            stop
        endif
        allocate(ppart(nppmx0,idimp,mxy1))
!
! copy ordered particle data for GPU code: updates ppart and kplic
        call PPMOVIN2LT(part,ppart,kpic,nppmx0,idimp,np,mx,my,mx1,mxy1,irc&
&)
        if (irc(1) /= 0) then
            write (*,*) 'PPMOVIN2LT overflow error, irc=', irc
            stop
        endif
! sanity check
        call PPCHECK2LT(ppart,kpic,idimp,nppmx0,nx,ny,mx,my,mx1,my1,irc)
        if (irc(1) /= 0) then
            write (*,*) 'PPCHECK2LT error: irc=', irc
            stop
        endif
! copy to GPU
        call gpu_icopyin(irc,g_irc,1)
        call gpu_fcopyin(ppart,g_ppart,nppmx0*idimp*mxy1)
        call gpu_icopyin(kpic,g_kpic,mxy1)
!
        if (dt > 0.45*ci) then
            write (*,*) 'Warning: Courant condition may be exceeded!'
        endif
!
! * * * start main iteration loop * * *
!
500 if (nloop <= ntime) go to 2000
!     write (*,*) 'ntime = ', ntime
!
! deposit current with GPU code:
        call dtimer(dtime,itime,-1)
        call gpu_zfmem(g_cue,ndim*nxe*nye)
        if (relativity==1) then
! updates g_ppart, g_cue
            call cgpu2rjppost2l(g_ppart,g_cue,g_kpic,qme,dth,ci,nppmx0,
&

```

```

        &idimp,nx,ny,mx,my,nxe,nye,mx1,mxy1,ipbc)
! updates g_ppart, g_cue, g_ncl, g_ihole, g_irc
!     call cgpu2rjppostf2l(g_ppart,g_cue,g_kpic,g_ncl,g_ihole,qme,dth&
!     &,ci,nppmx0,idimp,nx,ny,mx,my,nxe,nye,mx1,mxy1,ntmax,g_irc)
!     else
! updates g_ppart, g_cue
!     call cgpu2jppost2l(g_ppart,g_cue,g_kpic,qme,dth,nppmx0,idimp,nx&
!     &,ny,mx,my,nxe,nye,mx1,mxy1,ipbc)
! updates g_ppart, g_cue, g_ncl, g_ihole, g_irc
!     call cgpu2jppostf2l(g_ppart,g_cue,g_kpic,g_ncl,g_ihole,qme,dth,&
!     &nppmx0,idimp,nx,ny,mx,my,nxe,nye,mx1,mxy1,ntmax,g_irc)
!     endif
!     call dtimer(dtime,itime,1)
!     time = real(dtime)
!     tdjpost = tdjpost + time
!
! reorder particles by tile with GPU code:
!     call dtimer(dtime,itime,-1)
! updates g_ppart, g_ppbuff, g_kpic, g_ncl, g_ihole, and g_irc
!     call cgpu2ppord2l(g_ppart,g_ppbuff,g_kpic,g_ncl,g_ihole,idimp,      &
!     &nppmx0,nx,ny,mx,my,mx1,my1,npbmx,ntmax,g_irc)
! updates g_ppart, g_ppbuff, g_kpic, g_ncl, and g_irc
!     call cgpu2ppordf2l(g_ppart,g_ppbuff,g_kpic,g_ncl,g_ihole,idimp,      &
!     &nppmx0,mx1,my1,npbmx,ntmax,g_irc)
!     call dtimer(dtime,itime,1)
!     time = real(dtime)
!     tsort = tsort + time
!
! deposit charge with GPU code: updates g_qe
!     call dtimer(dtime,itime,-1)
!     call gpu_zfmem(g_qe,nxe*nye)
!     call cgpu2ppost2l(g_ppart,g_qe,g_kpic,qme,nppmx0,idimp,mx,my,nxe, &
!     &nye,mx1,mxy1)
!     call dtimer(dtime,itime,1)
!     time = real(dtime)
!     tdpost = tdpost + time
!
! add and copy guard cells with GPU code: updates g_q and g_cu
!     call dtimer(dtime,itime,-1)
!     call cgpu2cacguard2l(g_cu,g_cue,nx,ny,nxe,nye,nxhd,ny)
!     call cgpu2caguard2l(g_q,g_qe,nx,ny,nxe,nye,nxhd,ny)
!     call dtimer(dtime,itime,1)
!     time = real(dtime)
!     tguard = tguard + time
!
! transform charge to fourier space with GPU code: updates g_q, g_qt
!     call dtimer(dtime,itime,-1)
!     isign = -1
!     call cgpu2wfft2rcs(g_q,g_qt,isign,g_mixup,g_sct,indx,indy,nxhd,ny, &
!     &nxhy,nxyh)
! NVIDIA fft
!     call gpufft2rrcu(g_q,g_qt,isign,indx,indy,nxhd,ny)
!     call dtimer(dtime,itime,1)
!     time = real(dtime)

```

```

        tfft = tfft + time
!
! transform current to fourier space with GPU code: updates g_cu, g_cut
    call dtimer(dtime,itime,-1)
    isign = -1
    call cgpwfft2rcsn(g_cu,g_cut,isign,g_mixup,g_sct,indx,indy,ndim, &
        &nxhd,ny,nxhy,nxyh)
! NVIDIA fft
!    call gpufft2rrcun(g_cu,g_cut,isign,indx,indy,ndim,nxhd,ny)
    call dtimer(dtime,itime,1)
    time = real(dtime)
    tfft = tfft + time
!
! take transverse part of current with GPU code: updates g_cut
    call dtimer(dtime,itime,-1)
    call cgpucuperp2t(g_cut,nx,ny,nxhd,ny)
    call dtimer(dtime,itime,1)
    time = real(dtime)
    tfield = tfield + time
!
! calculate electromagnetic fields in fourier space with GPU code:
! updates g_exyzt, g_bxyzt, g_wf, g_wm
    call dtimer(dtime,itime,-1)
    if (ntime==0) then
        call cgpuibpois23t(g_cut,g_bxyzt,g_ffct,ci,g_wm,nx,ny,nxh1,ny, &
            &nxh,nyh)
        call gpu_zfmem(g_wf,nxh1)
        dth = 0.5*dt
    else
        call cgpumaxwel2t(g_exyzt,g_bxyzt,g_cut,g_ffct,ci,dt,g_wf,g_wm,&
            &nx,ny,nxh1,ny,nxh,nyh)
    endif
    call dtimer(dtime,itime,1)
    time = real(dtime)
    tfield = tfield + time
!
! calculate force/charge in fourier space with GPU code:
! updates g_fxyzt, g_we
    call dtimer(dtime,itime,-1)
    call cgpuipois23t(g_qt,g_fxyzt,g_ffct,g_we,nx,ny,nxh1,ny,nxh,nyh)
    call dtimer(dtime,itime,1)
    time = real(dtime)
    tfield = tfield + time
!
! add longitudinal and transverse electric fields with with GPU code:
! updates g_fxyzt
    call dtimer(dtime,itime,-1)
    isign = 1
    call cgpuemfield2t(g_fxyzt,g_exyzt,g_ffct,isign,nx,ny,nxh1,ny,nxh,&
        &nyh)
! copy magnetic field with GPU code: updates g_hxyzt
    isign = -1
    call cgpuemfield2t(g_hxyzt,g_bxyzt,g_ffct,isign,nx,ny,nxh1,ny,nxh,&
        &nyh)

```

```

        call dtimer(dtime,itime,1)
        time = real(dtime)
        tfield = tfield + time
!
! transform electric force to real space with GPU code:
! updates g_fxyz, g_fxyz
        call dtimer(dtime,itime,-1)
        isign = 1
        call cgpwfft2rcsn(g_fxyz,g_fxyzt,isign,g_mixup,g_sct,indx,indy, &
            &ndim,nxhd,ny,nxhy,nxyh)
! NVIDIA fft
!     call gpufft2rrcun(g_fxyz,g_fxyzt,isign,indx,indy,ndim,nxhd,ny)
        call dtimer(dtime,itime,1)
        time = real(dtime)
        tfft = tfft + time
!
! transform magnetic force to real space with GPU code:
! updates g_hxyz, g_hxyz
        call dtimer(dtime,itime,-1)
        isign = 1
        call cgpwfft2rcsn(g_hxyz,g_hxyzt,isign,g_mixup,g_sct,indx,indy, &
            &ndim,nxhd,ny,nxhy,nxyh)
! NVIDIA fft
!     call gpufft2rrcun(g_hxyz,g_hxyzt,isign,indx,indy,ndim,nxhd,ny)
        call dtimer(dtime,itime,1)
        time = real(dtime)
        tfft = tfft + time
!
! copy guard cells with GPU code: updates g_fxyze, g_bxyze
        call dtimer(dtime,itime,-1)
        call cgpucbguard2l(g_fxyz,g_fxyze,nx,ny,nxe,nye,nxhd,ny)
        call cgpucbguard2l(g_hxyz,g_bxyze,nx,ny,nxe,nye,nxhd,ny)
        call dtimer(dtime,itime,1)
        time = real(dtime)
        tguard = tguard + time
!
! push particles with GPU code:
        call dtimer(dtime,itime,-1)
        if (relativity==1) then
! updates g_ppart, g_wke
            call cgpurbppush23l(g_ppart,g_fxyze,g_bxyze,g_kpic,qbme,dt,dth,&
                &ci,g_wke,idimp,nppmx0,nx,ny,mx,my,nxe,nye,mx1,mxy1,ipbc)
! updates g_ppart, g_ncl, g_ihole, g_wke, g_irc
!     call cgpurbppushf23l(g_ppart,g_fxyze,g_bxyze,g_kpic,g_ncl, &
! &g_ihole,qbme,dt,dth,ci,g_wke,idimp,nppmx0,nx,ny,mx,my,nxe,nye,mx1,&
! &mxy1,ntmax,g_irc)
        else
! updates g_ppart, g_wke
            call cgpurbppush23l(g_ppart,g_fxyze,g_bxyze,g_kpic,qbme,dt,dth, &
                &g_wke,idimp,nppmx0,nx,ny,mx,my,nxe,nye,mx1,mxy1,ipbc)
! updates g_ppart, g_ncl, g_ihole, g_wke, g_irc
!     call cgpurbppushf23l(g_ppart,g_fxyze,g_bxyze,g_kpic,g_ncl, &
! &g_ihole,qbme,dt,dth,g_wke,idimp,nppmx0,nx,ny,mx,my,nxe,nye,mx1, &
! &mxy1,ntmax,g_irc)

```

```

endif
call dtimer(dtime,itime,1)
time = real(dtime)
tpush = tpush + time
!
! reorder particles by tile with GPU code:
call dtimer(dtime,itime,-1)
! updates g_ppart, g_ppbuff, g_kpic, g_ncl, g_ihole, and g_irc
call cgpuppord2l(g_ppart,g_ppbuff,g_kpic,g_ncl,g_ihole,idimp,      &
&nppmx0,nx,ny,mx,my,mx1,my1,npbm,ntmax,g_irc)
! updates g_ppart, g_ppbuff, g_kpic, g_ncl, and g_irc
! call cgpuppordf2l(g_ppart,g_ppbuff,g_kpic,g_ncl,g_ihole,idimp,      &
! &nppmx0,mx1,my1,npbm,ntmax,g_irc)
call dtimer(dtime,itime,1)
time = real(dtime)
tsort = tsort + time
!
! sanity check
call gpu_icopyout(irc,g_irc,1)
if (irc(1) /= 0) then
write (*,*) 'deposit/push/reorder error: ntmax, irc=',ntmax,irc
stop
endif
!
! energy diagnostic
if (ntime==0) then
call gpu_zfmem(g_sum,1)
call cgpsum2(g_we,g_sum,nxh1)
call gpu_fcopyout(sum,g_sum,1); we = sum(1)
call gpu_zfmem(g_sum,1)
call cgpsum2(g_wf,g_sum,nxh1)
call gpu_fcopyout(sum,g_sum,1); wf = sum(1)
call gpu_zfmem(g_sum,1)
call cgpsum2(g_wm,g_sum,nxh1)
call gpu_fcopyout(sum,g_sum,1); wm = sum(1)
call gpu_zfmem(g_sum,1)
call cgpsum2(g_wke,g_sum,mxy1)
call gpu_fcopyout(sum,g_sum,1); wke = sum(1)
wt = we + wf + wm
write (*,*) 'Initial Total Field, Kinetic and Total Energies:'
write (*, '(3e14.7)') wt, wke, wke + wt
write (*,*) 'Initial Electrostatic, Transverse Electric and Mag&
&netic Field Energies:'
write (*, '(3e14.7)') we, wf, wm
endif
ntime = ntime + 1
go to 500
2000 continue
!
! * * * end main iteration loop * * *
!
write (*,*) 'ntime = ', ntime
write (*,*) 'relativity = ', relativity
! energy diagnostic

```

```

call gpu_zfmem(g_sum,1)
call cgpusum2(g_we,g_sum,nxh1)
call gpu_fcopyout(sum,g_sum,1); we = sum(1)
call gpu_zfmem(g_sum,1)
call cgpusum2(g_wf,g_sum,nxh1)
call gpu_fcopyout(sum,g_sum,1); wf = sum(1)
call gpu_zfmem(g_sum,1)
call cgpusum2(g_wm,g_sum,nxh1)
call gpu_fcopyout(sum,g_sum,1); wm = sum(1)
call gpu_zfmem(g_sum,1)
call cgpusum2(g_wke,g_sum,mxy1)
call gpu_fcopyout(sum,g_sum,1); wke = sum(1)
wt = we + wf + wm
write (*,*) 'Final Total Field, Kinetic and Total Energies:'
write (*, '(3e14.7)') wt, wke, wke + wt
write (*,*) 'Final Electrostatic, Transverse Electric and Magnetic&
& Field Energies:'
write (*, '(3e14.7)') we, wf, wm
!

write (*,*)
write (*,*) 'deposit time = ', tdpost
write (*,*) 'current deposit time = ', tdjpost
tdpost = tdpost + tdjpost
write (*,*) 'total deposit time = ', tdpost
write (*,*) 'guard time = ', tguard
write (*,*) 'solver time = ', tfield
write (*,*) 'fft time = ', tfft
write (*,*) 'push time = ', tpush
write (*,*) 'sort time = ', tsort
tfield = tfield + tguard + tfft
write (*,*) 'total solver time = ', tfield
time = tdpost + tpush + tsort
write (*,*) 'total particle time = ', time
wt = time + tfield
write (*,*) 'total time = ', wt
write (*,*)
!

wt = 1.0e+09/(real(nloop)*real(np))
write (*,*) 'Push Time (nsec) = ', tpush*wt
write (*,*) 'Deposit Time (nsec) = ', tdpost*wt
write (*,*) 'Sort Time (nsec) = ', tsort*wt
write (*,*) 'Total Particle Time (nsec) = ', time*wt
write (*,*)
!
! close down NVIDIA fft
call gpufft2cudel()
call gpufft2rrcudel()
! deallocate memory on GPU
call gpu_deallocate(g_irc,irc(1))
call gpu_deallocate(g_ihole,irc(1))
call gpu_deallocate(g_ncl,irc(1))
call gpu_deallocate(g_kpic,irc(1))
call gpu_deallocate(g_sum,irc(1))
call gpu_deallocate(g_wm,irc(1))

```

```

call gpu_deallocate(g_wf,irc(1))
call gpu_deallocate(g_we,irc(1))
call gpu_deallocate(g_wke,irc(1))
call gpu_deallocate(g_bxyzt,irc(1))
call gpu_deallocate(g_exyzt,irc(1))
call gpu_deallocate(g_hxyzt,irc(1))
call gpu_deallocate(g_fxyzt,irc(1))
call gpu_deallocate(g_hxyz,irc(1))
call gpu_deallocate(g_fxyz,irc(1))
call gpu_deallocate(g_cut,irc(1))
call gpu_deallocate(g_qt,irc(1))
call gpu_deallocate(g_cu,irc(1))
call gpu_deallocate(g_q,irc(1))
call gpu_deallocate(g_sct,irc(1))
call gpu_deallocate(g_mixup,irc(1))
call gpu_deallocate(g_ffct,irc(1))
call gpu_deallocate(g_ppbuff,irc(1))
call gpu_deallocate(g_ppart,irc(1))
call gpu_deallocate(g_bxyze,irc(1))
call gpu_deallocate(g_fxyze,irc(1))
call gpu_deallocate(g_cue,irc(1))
call gpu_deallocate(g_ge,irc(1))
! close down GPU
call end_cu()
!
stop
end program

```