

```

1  module user_defined_types
2      use, intrinsic :: iso_c_binding
3      use fdps_vector
4      use fdps_super_particle
5      implicit none
6
7      type, public, bind(c) :: full_ptcl !$fdps FP,EPI,EPJ,Force
8          !$fdps copyFromForce full_ptcl (pot,pot) (acc,acc)
9          !$fdps copyFromFP full_ptcl (mass,mass) (eps,eps) (pos,pos)
10         !$fdps clear id=keep, mass=keep, eps=keep, pos=keep, vel=keep
11         integer(kind=c_long_long) :: id
12         real(kind=c_double) mass !$fdps charge
13         real(kind=c_double) :: eps
14         type(fdps_f64vec) :: pos !$fdps position
15         type(fdps_f64vec) :: vel
16         real(kind=c_double) :: pot
17         type(fdps_f64vec) :: acc
18     end type full_ptcl
19
20     contains
21
22     subroutine calc_gravity_pp(ep_i,n_ip,ep_j,n_jp,f) bind(c)
23         integer(c_int), intent(in), value :: n_ip,n_jp
24         type(full_ptcl), dimension(n_ip), intent(in) :: ep_i
25         type(full_ptcl), dimension(n_jp), intent(in) :: ep_j
26         type(full_ptcl), dimension(n_ip), intent(inout) :: f
27         integer(c_int) :: i,j
28         real(c_double) :: eps2,poti,r3_inv,r_inv
29         type(fdps_f64vec) :: xi,ai,rij
30
31         do i=1,n_ip
32             eps2 = ep_i(i)%eps * ep_i(i)%eps
33             xi = ep_i(i)%pos
34             ai = 0.0d0; poti = 0.0d0
35             do j=1,n_jp
36                 rij%x = xi%x - ep_j(j)%pos%x
37                 rij%y = xi%y - ep_j(j)%pos%y
38                 rij%z = xi%z - ep_j(j)%pos%z
39                 r3_inv = rij%x*rij%x + rij%y*rij%y + rij%z*rij%z &
40                     + eps2
41                 r_inv = 1.0d0/sqrt(r3_inv)
42                 r3_inv = r_inv * r_inv
43                 r_inv = r_inv * ep_j(j)%mass
44                 r3_inv = r3_inv * r_inv
45                 ai%x = ai%x - r3_inv * rij%x
46                 ai%y = ai%y - r3_inv * rij%y
47                 ai%z = ai%z - r3_inv * rij%z
48                 poti = poti - r_inv
49             end do
50             f(i)%pot = f(i)%pot + poti
51             f(i)%acc = f(i)%acc + ai
52         end do
53
54     end subroutine calc_gravity_pp
55
56     subroutine calc_gravity_psp(ep_i,n_ip,ep_j,n_jp,f) bind(c)
57         integer(c_int), intent(in), value :: n_ip,n_jp
58         type(full_ptcl), dimension(n_ip), intent(in) :: ep_i
59         type(fdps_ep_j_monopole), dimension(n_jp), intent(in) :: ep_j
60         type(full_ptcl), dimension(n_ip), intent(inout) :: f
61         integer(c_int) :: i,j
62         real(c_double) :: eps2,poti,r3_inv,r_inv
63         type(fdps_f64vec) :: xi,ai,rij
64
65         do i=1,n_ip
66             eps2 = ep_i(i)%eps * ep_i(i)%eps
67             xi = ep_i(i)%pos
68             ai = 0.0d0; poti = 0.0d0
69             do j=1,n_jp
70                 rij%x = xi%x - ep_j(j)%pos%x
71                 rij%y = xi%y - ep_j(j)%pos%y
72                 rij%z = xi%z - ep_j(j)%pos%z
73                 r3_inv = rij%x*rij%x + rij%y*rij%y + rij%z*rij%z &
74                     + eps2
75                 r_inv = 1.0d0/sqrt(r3_inv)
76                 r3_inv = r_inv * r_inv
77                 r_inv = r_inv * ep_j(j)%mass
78                 r3_inv = r3_inv * r_inv
79                 ai%x = ai%x - r3_inv * rij%x
80                 ai%y = ai%y - r3_inv * rij%y
81                 ai%z = ai%z - r3_inv * rij%z
82                 poti = poti - r_inv
83             end do
84             f(i)%pot = f(i)%pot + poti
85             f(i)%acc = f(i)%acc + ai
86         end do
87
88     end subroutine calc_gravity_psp
89
90     end module user_defined_types
91
92     subroutine f_main()
93         use fdps_module
94         use user_defined_types
95         implicit none
96         double precision, parameter :: time_end=10.0d0
97         double precision, parameter :: dt=1.0d0/128.0d0
98         integer :: i,j,k,ierr
99         integer :: psys_num,dinfo_num,tree_num
100        character(len=64) :: tree_type
101        double precision :: time_sys=0.0d0
102        type(fdps_controller) :: fdps_ctrl
103
104        call fdps_ctrl%PS_Initialize()
105        call fdps_ctrl%create_dinfo(dinfo_num)
106        call fdps_ctrl%init_dinfo(dinfo_num)
107        call fdps_ctrl%create_psys(psys_num,'full_ptcl')
108        call fdps_ctrl%init_psys(psys_num)
109        tree_type="Long,full_ptcl,full_ptcl,full_ptcl,Monopole"
110        call fdps_ctrl%create_tree(tree_num,tree_type)
111        call fdps_ctrl%init_tree(tree_num,0)
112        call read_IC(fdps_ctrl,psys_num)
113        call calc_gravity(fdps_ctrl,psys_num,dinfo_num,tree_num)
114        do
115            call kick(fdps_ctrl,psys_num,0.5d0*dt)
116            time_sys=time_sys+dt
117            call drift(fdps_ctrl,psys_num,dt)
118            call calc_gravity(fdps_ctrl,psys_num,dinfo_num,tree_num)
119            call kick(fdps_ctrl,psys_num,0.5d0*dt)
120            if(time_sys >= time_end) exit
121        end do
122        call fdps_ctrl%PS_Finalize()
123    end subroutine f_main
124
125     subroutine calc_gravity(fdps_ctrl,psys_num,dinfo_num,tree_num)
126         use fdps_module
127         use user_defined_types
128         implicit none
129         type(fdps_controller), intent(IN) :: fdps_ctrl
130         integer, intent(IN) :: psys_num,dinfo_num,tree_num
131         type(c_funptr) :: pfunc_ep_ep,pfunc_ep_sp
132         call fdps_ctrl%decompose_domain_all(dinfo_num,psys_num)
133         call fdps_ctrl%exchange_particle(psys_num,dinfo_num)
134         pfunc_ep_ep=c_funloc(calc_gravity_pp)
135         pfunc_ep_sp=c_funloc(calc_gravity_psp)
136         call fdps_ctrl%calc_force_all_and_write_back(tree_num,&
137             pfunc_ep_ep,&
138             pfunc_ep_sp,&
139             psys_num,&
140             dinfo_num)
141     end subroutine calc_gravity
142
143     subroutine kick(fdps_ctrl,psys_num,dt)
144         use fdps_vector
145         use fdps_module
146         use user_defined_types
147         implicit none
148         type(fdps_controller), intent(IN) :: fdps_ctrl
149         integer, intent(IN) :: psys_num
150         double precision, intent(IN) :: dt
151         integer :: i,nptcl_loc
152         type(full_ptcl), dimension(:), pointer :: ptcl
153         nptcl_loc = fdps_ctrl%get_nptcl_loc(psys_num)
154         call fdps_ctrl%get_psys_fptr(psys_num,ptcl)
155         do i=1,nptcl_loc
156             ptcl(i)%vel = ptcl(i)%vel + ptcl(i)%acc*dt
157         end do
158         nullify(ptcl)
159     end subroutine kick
160
161     subroutine drift(fdps_ctrl,psys_num,dt)
162         use fdps_vector
163         use fdps_module
164         use user_defined_types
165         implicit none
166         type(fdps_controller), intent(IN) :: fdps_ctrl
167         integer, intent(IN) :: psys_num
168         double precision, intent(IN) :: dt
169         integer :: i,nptcl_loc
170         type(full_ptcl), dimension(:), pointer :: ptcl
171         nptcl_loc = fdps_ctrl%get_nptcl_loc(psys_num)
172         call fdps_ctrl%get_psys_fptr(psys_num,ptcl)
173         do i=1,nptcl_loc
174             ptcl(i)%pos = ptcl(i)%pos + ptcl(i)%vel*dt
175         end do
176         nullify(ptcl)
177     end subroutine drift
178
179     subroutine read_IC(fdps_ctrl,psys_num)
180         use fdps_module
181         use user_defined_types
182         implicit none
183         type(fdps_controller), intent(IN) :: fdps_ctrl
184         integer, intent(IN) :: psys_num
185         character(len=16), parameter :: root_dir="input_data"
186         character(len=16), parameter :: file_prefix="proc"
187         integer :: i,myrank,nptcl_loc
188         character(len=64) :: fname,proc_num
189         type(full_ptcl), dimension(:), pointer::ptcl
190         myrank = fdps_ctrl%get_rank()
191         write(proc_num,"(i5.5)")myrank
192         fname = trim(root_dir)//"/" &
193             //trim(file_prefix)//proc_num//".dat"
194         open(unit=9,file=trim(fname),action='read',form='unformatted',&
195             access='stream',status='old')
196         read(9)nptcl_loc
197         call fdps_ctrl%set_nptcl_loc(psys_num,nptcl_loc)
198         call fdps_ctrl%get_psys_fptr(psys_num,ptcl)
199         do i=1,nptcl_loc
200             read(9)ptcl(i)%id,ptcl(i)%mass,ptcl(i)%eps, &
201                 ptcl(i)%posx,ptcl(i)%posy,ptcl(i)%posz, &
202                 ptcl(i)%velx,ptcl(i)%vely,ptcl(i)%velz
203         end do
204         close(unit=9)
205         nullify(ptcl)
206     end subroutine read_IC

```