

Add explicit group vector to acc, needed by CRAY compilers #298

Edit <> Code

Merged sfiligoi merged 3 commits into gafusion:master from sfiligoi:crusher\_238227 on Feb 27

Conversation 0 Commits 3 Checks 0 Files changed 9 +67 -35

Changes from all commits File filter Conversations

Filter changed files

- cgyro/src
- cgyro\_error\_estimate.F90
- cgyro\_field.F90
- cgyro\_math.F90
- cgyro\_nl\_comm.F90
- cgyro\_n\_fftw.gpu.F90
- cgyro\_parallel\_lib.F90
- cgyro\_rhs.gpu.F90
- cgyro\_source.F90
- cgyro\_upwind.F90

29	! NOTE: If I have multiple itor, sum them all together	29	! NOTE: If I have multiple itor, sum them all together
30	h_s=0.0	30	h_s=0.0
31	r_s=0.0	31	r_s=0.0
32	- !\$sacc parallel loop collapse(3) independent present(h_x,rhs(:, :, 1)) reduction(+:h_s,r_s) async(2)	32	+ !\$sacc parallel loop collapse(3) independent gang vector &
33		33	+ !\$sacc& present(h_x,rhs(:, :, 1)) reduction(+:h_s,r_s) async(2)
34	do itor=nt1,nt2	34	do itor=nt1,nt2
35	do iv_loc=1,nv_loc	35	do iv_loc=1,nv_loc
	do ic=1,nc	36	do ic=1,nc

169		169	! Poisson and Ampere RHS integrals of H
170	! Poisson and Ampere RHS integrals of H	170	! Poisson and Ampere RHS integrals of H
171		171	
172	- !\$sacc parallel loop collapse(3) independent copyin(start_t) &	172	+ !\$sacc parallel loop collapse(3) gang vector &
		173	+ !\$sacc& independent copyin(start_t) &
173	!\$sacc& present(nt2,nc,n_field) default(none)	174	!\$sacc& present(nt2,nc,n_field) default(none)
174	do itor=start_t,nt2	175	do itor=start_t,nt2
175	do ic=1,nc	176	do ic=1,nc
224		225	
225	call timer_lib_in('field')	226	call timer_lib_in('field')
226	! Poisson LHS factors	227	! Poisson LHS factors
227	- !\$sacc parallel loop collapse(3) independent present(fcoef) copyin(start_t) &	228	+ !\$sacc parallel loop collapse(3) gang vector &
		229	+ !\$sacc& independent present(fcoef) copyin(start_t) &
228	!\$sacc& present(nt2,nc,n_field) default(none)	230	!\$sacc& present(nt2,nc,n_field) default(none)
229	do itor=start_t,nt2	231	do itor=start_t,nt2
230	! assuming (.not.(itor == 0 .and. ae_flag == 1))	232	! assuming (.not.(itor == 0 .and. ae_flag == 1))
443		445	
444	! Poisson and Ampere RHS integrals of h	446	! Poisson and Ampere RHS integrals of h
445		447	
446	- !\$sacc parallel loop collapse(3) independent private(field_loc_l) &	448	+ !\$sacc parallel loop collapse(3) gang vector &
		449	+ !\$sacc& independent private(field_loc_l) &
447	!\$sacc& present(djvec_c) present(nt1,nt2,nc,n_field,nv1,nv2) default(none)	450	!\$sacc& present(djvec_c) present(nt1,nt2,nc,n_field,nv1,nv2) default(none)
448	do itor=nt1,nt2	451	do itor=nt1,nt2
449	do ic=1,nc	452	do ic=1,nc
484	call timer_lib_out('field_com')	487	call timer_lib_out('field_com')
485	call timer_lib_in('field')	488	call timer_lib_in('field')
486	if (n_field > 2) then	489	if (n_field > 2) then
487	- !\$sacc parallel loop collapse(2) independent present(fcoef) &	490	+ !\$sacc parallel loop collapse(2) gang vector &
		491	+ !\$sacc& independent present(fcoef) &
488	!\$sacc& present(nt1,nt2,nc) default(none)	492	!\$sacc& present(nt1,nt2,nc) default(none)
489	do itor=nt1,nt2	493	do itor=nt1,nt2
490	do ic=1,nc	494	do ic=1,nc
512		512	
509	if (itor1<=itor2) then	513	if (itor1<=itor2) then
510	if (n_field > 2) then	514	if (n_field > 2) then
511	- !\$sacc parallel loop collapse(2) independent private(tmp) present(gcoef) &	515	+ !\$sacc parallel loop collapse(2) gang vector &
		516	+ !\$sacc& independent private(tmp) present(gcoef) &
512	!\$sacc& copyin(itor1,itor2) present(nc) default(none)	517	!\$sacc& copyin(itor1,itor2) present(nc) default(none)
513	do itor=itor1,itor2	518	do itor=itor1,itor2
514	do ic=1,nc	519	do ic=1,nc
521	enddo	526	enddo
522	enddo	527	enddo
523	else	528	else
524	- !\$sacc parallel loop collapse(3) independent present(gcoef) &	529	+ !\$sacc parallel loop collapse(3) gang vector &
		530	+ !\$sacc& independent present(gcoef) &
525	!\$sacc& copyin(itor1,itor2) present(nc,n_field) default(none)	531	!\$sacc& copyin(itor1,itor2) present(nc,n_field) default(none)
526	do itor=itor1,itor2	532	do itor=itor1,itor2
527	do ic=1,nc	533	do ic=1,nc
570		576	
571	! Poisson and Ampere RHS integrals of h	577	! Poisson and Ampere RHS integrals of h
572		578	
573	- !\$sacc parallel loop collapse(3) independent private(field_loc_l) &	579	+ !\$sacc parallel loop collapse(3) gang vector &
		580	+ !\$sacc& independent private(field_loc_l) &
574	!\$sacc& present(djvec_c) present(nc,n_field,nv1,nv2) default(none)	581	!\$sacc& present(djvec_c) present(nc,n_field,nv1,nv2) default(none)
575	do itor=0,0	582	do itor=0,0
576	do ic=1,nc	583	do ic=1,nc
611	call timer_lib_out('field_com')	618	call timer_lib_out('field_com')
612	call timer_lib_in('field')	619	call timer_lib_in('field')
613	if (n_field > 2) then	620	if (n_field > 2) then
614	- !\$sacc parallel loop collapse(2) independent present(fcoef) present(nc) default(none)	621	+ !\$sacc parallel loop collapse(2) gang vector &
		622	+ !\$sacc& independent present(fcoef) present(nc) default(none)
615	do itor=0,0	623	do itor=0,0
616	do ic=1,nc	624	do ic=1,nc
617	field(3,ic,itor) = field(3,ic,itor)*fcoef(3,ic,itor)	625	field(3,ic,itor) = field(3,ic,itor)*fcoef(3,ic,itor)

26	integer :: i	26	integer :: i
27	!-----	27	!-----
28	#ifdef _OPENACC	28	#ifdef _OPENACC
29	- !\$sacc parallel loop independent present(left,r1)	29	+ !\$sacc parallel loop independent gang vector &
		30	+ !\$sacc& present(left,r1)
30		31	do i=1,sz
31	left(i) = r1(i)	32	left(i) = r1(i)
32	enddo	33	enddo
51	complex :: tmp	52	complex :: tmp
52	!-----	53	!-----
53	#ifdef _OPENACC	54	#ifdef _OPENACC
54	- !\$sacc parallel loop independent present(left1,left2,r1) private(tmp)	55	+ !\$sacc parallel loop independent gang vector &
		56	+ !\$sacc& present(left1,left2,r1) private(tmp)
55		57	do i=1,sz
56	tmp = r1(i)	58	tmp = r1(i)
57	left1(i) = tmp	59	left1(i) = tmp
89	if (present(abssum)) then	91	if (present(abssum)) then
90	s = 0.0	92	s = 0.0
91	#ifdef _OPENACC	93	#ifdef _OPENACC
92	- !\$sacc parallel loop independent present(left,r1,r2) private(tmp) reduction(+:s)	94	+ !\$sacc parallel loop independent gang vector &
		95	+ !\$sacc& present(left,r1,r2) private(tmp) reduction(+:s)
93	#else	96	#else
94	!\$omp parallel do private(tmp) reduction(+:s)	97	!\$omp parallel do private(tmp) reduction(+:s)
95	#endif	98	#endif
101	abssum = s	104	abssum = s
102	else	105	else
103	#ifdef _OPENACC	106	#ifdef _OPENACC
104	- !\$sacc parallel loop independent present(left,r1,r2)	107	+ !\$sacc parallel loop independent gang vector &
		108	+ !\$sacc& present(left,r1,r2)
105	#else	109	#else
106	!\$omp parallel do	110	!\$omp parallel do
107	#endif	111	#endif
131	if (present(abssum)) then	135	if (present(abssum)) then
132	s = 0.0	136	s = 0.0
133	#ifdef _OPENACC	137	#ifdef _OPENACC
134	- !\$sacc parallel loop independent present(left,r1,r2,r3) private(tmp) reduction(+:s)	138	+ !\$sacc parallel loop independent gang vector &
		139	+ !\$sacc& present(left,r1,r2,r3) private(tmp) reduction(+:s)
135	#else	140	#else
136	!\$omp parallel do private(tmp) reduction(+:s)	141	!\$omp parallel do private(tmp) reduction(+:s)
137	#endif	142	#endif
143	abssum = s	148	abssum = s
144	else	149	else
145	#ifdef _OPENACC	150	#ifdef _OPENACC
146	- !\$sacc parallel loop independent present(left,r1,r2,r3)	151	+ !\$sacc parallel loop independent gang vector &
		152	+ !\$sacc& present(left,r1,r2,r3)
147	#else	153	#else
148	!\$omp parallel do	154	!\$omp parallel do
149	#endif	155	#endif
175	if (present(abssum)) then	181	if (present(abssum)) then
176	s = 0.0	182	s = 0.0
177	#ifdef _OPENACC	183	#ifdef _OPENACC
178	- !\$sacc parallel loop independent present(left,r1,r2,r3,r4) private(tmp) reduction(+:s)	184	+ !\$sacc parallel loop independent gang vector &
		185	+ !\$sacc& present(left,r1,r2,r3,r4) private(tmp) reduction(+:s)
179	#else	186	#else
180	!\$omp parallel do private(tmp) reduction(+:s)	187	!\$omp parallel do private(tmp) reduction(+:s)
181	#endif	188	#endif
187	abssum = s	194	abssum = s
188	else	195	else
189	#ifdef _OPENACC	196	#ifdef _OPENACC
190	- !\$sacc parallel loop independent present(left,r1,r2,r3,r4)	197	+ !\$sacc parallel loop independent gang vector &
		198	+ !\$sacc& present(left,r1,r2,r3,r4)
191	#else	199	#else
192	!\$omp parallel do	200	!\$omp parallel do
193	#endif	201	#endif
221	if (present(abssum)) then	229	if (present(abssum)) then
222	s = 0.0	230	s = 0.0
223	#ifdef _OPENACC	231	#ifdef _OPENACC
224	- !\$sacc parallel loop independent present(left,r1,r2,r3,r4,r5) private(tmp) reduction(+:s)	232	+ !\$sacc parallel loop independent gang vector &
		233	+ !\$sacc& present(left,r1,r2,r3,r4,r5) private(tmp) reduction(+:s)
225	#else	234	#else
226	!\$omp parallel do private(tmp) reduction(+:s)	235	!\$omp parallel do private(tmp) reduction(+:s)
227	#endif	236	#endif
233	abssum = s	242	abssum = s
234	else	243	else
235	#ifdef _OPENACC	244	#ifdef _OPENACC
236	- !\$sacc parallel loop independent present(left,r1,r2,r3,r4,r5)	245	+ !\$sacc parallel loop independent gang vector &
		246	+ !\$sacc& present(left,r1,r2,r3,r4,r5)
237	#else	247	#else
238	!\$omp parallel do	248	!\$omp parallel do



239	<code>#endif</code>	249	<code>#endif</code>
269	<code>if (present(abssum)) then</code>	279	<code>if (present(abssum)) then</code>
270	<code>s = 0.0</code>	280	<code>s = 0.0</code>
271	<code>#ifdef _OPENACC</code>	281	<code>#ifdef _OPENACC</code>
272	<code>!\$acc parallel loop independent present(left,r1,r2,r3,r4,r5,r6) private(tmp) reduction(+:s)</code>	282	<code>+ !\$acc parallel loop independent gang vector &amp;</code>
		283	<code>+ !\$acc&amp; present(left,r1,r2,r3,r4,r5,r6) private(tmp) reduction(+:s)</code>
273	<code>#else</code>	284	<code>#else</code>
274	<code>!\$omp parallel do private(tmp) reduction(+:s)</code>	285	<code>!\$omp parallel do private(tmp) reduction(+:s)</code>
275	<code>#endif</code>	286	<code>#endif</code>
281	<code>abssum = s</code>	292	<code>abssum = s</code>
282	<code>else</code>	293	<code>else</code>
	<code>#ifdef _OPENACC</code>	294	<code>#ifdef _OPENACC</code>
284	<code>!\$acc parallel loop independent present(left,r1,r2,r3,r4,r5,r6)</code>	295	<code>+ !\$acc parallel loop independent gang vector &amp;</code>
		296	<code>+ !\$acc&amp; present(left,r1,r2,r3,r4,r5,r6)</code>
285	<code>#else</code>	297	<code>#else</code>
286	<code>!\$omp parallel do</code>	298	<code>!\$omp parallel do</code>
287	<code>#endif</code>	299	<code>#endif</code>
311	<code>if (present(abssum)) then</code>	323	<code>if (present(abssum)) then</code>
312	<code>s = 0.0</code>	324	<code>s = 0.0</code>
313	<code>#ifdef _OPENACC</code>	325	<code>#ifdef _OPENACC</code>
314	<code>!\$acc parallel loop independent present(left,r1,rN) copyin(cN) private(tmp,j) reduction(+:s)</code>	326	<code>+ !\$acc parallel loop independent gang vector &amp;</code>
		327	<code>+ !\$acc&amp; present(left,r1,rN) copyin(cN) private(tmp,j) reduction(+:s)</code>
315	<code>#else</code>	328	<code>#else</code>
316	<code>!\$omp parallel do private(tmp,j) reduction(+:s)</code>	329	<code>!\$omp parallel do private(tmp,j) reduction(+:s)</code>
317	<code>#endif</code>	330	<code>#endif</code>
327	<code>abssum = s</code>	340	<code>abssum = s</code>
328	<code>else</code>	341	<code>else</code>
	<code>#ifdef _OPENACC</code>	342	<code>#ifdef _OPENACC</code>
330	<code>!\$acc parallel loop independent present(left,r1,rN) copyin(cN) private(tmp,j)</code>	343	<code>+ !\$acc parallel loop independent gang vector &amp;</code>
		344	<code>+ !\$acc&amp; present(left,r1,rN) copyin(cN) private(tmp,j)</code>
331	<code>#else</code>	345	<code>#else</code>
332	<code>!\$omp parallel do private(tmp,j)</code>	346	<code>!\$omp parallel do private(tmp,j)</code>
333	<code>#endif</code>	347	<code>#endif</code>
371	<code>s1 = 0.0</code>	385	<code>s1 = 0.0</code>
372	<code>sm = 0.0</code>	386	<code>sm = 0.0</code>
373	<code>#ifdef _OPENACC</code>	387	<code>#ifdef _OPENACC</code>
374	<code>!\$acc parallel loop independent present(left,r0,m1,rN) copyin(cN,ecN) private(tmp,tmp1,tmpm,j) reduction(+:s1,sm)</code>	388	<code>+ !\$acc parallel loop independent gang vector &amp;</code>
		389	<code>+ !\$acc&amp; present(left,r0,m1,rN) copyin(cN,ecN) private(tmp,tmp1,tmpm,j) &amp;</code>
		390	<code>+ !\$acc&amp; reduction(+:s1,sm)</code>
375	<code>#else</code>	391	<code>#else</code>
376	<code>!\$omp parallel do private(tmp,tmp1,tmpm,j) reduction(+:s1,sm)</code>	392	<code>!\$omp parallel do private(tmp,tmp1,tmpm,j) reduction(+:s1,sm)</code>
377	<code>#endif</code>	393	<code>#endif</code>

3 cgYRO/src/cgyro_n1_comm.F90			
53	<code>enddo</code>	53	<code>enddo</code>
54		54	
55	<code>#ifdef _OPENACC</code>	55	<code>#ifdef _OPENACC</code>
56	<code>!\$acc parallel loop independent present(fpack) if ( (nv_loc*n_theta) &lt; (nsplit*n_toroidal_procs) )</code>	56	<code>+ !\$acc parallel loop independent gang vector &amp;</code>
		57	<code>+ !\$acc&amp; present(fpack) if ( (nv_loc*n_theta) &lt; (nsplit*n_toroidal_procs) )</code>
57	<code>#endif</code>	58	<code>#endif</code>
58	<code>do iexch=nv_loc*n_theta+1,nsplit*n_toroidal_procs</code>	59	<code>do iexch=nv_loc*n_theta+1,nsplit*n_toroidal_procs</code>
59	<code>fpack(1:n_radial,1:nt_loc,iexch) = (0.0,0.0)</code>	60	<code>fpack(1:n_radial,1:nt_loc,iexch) = (0.0,0.0)</code>

9 cgYRO/src/cgyro_n1_fftw.gpu.F90			
18	<code>integer :: i</code>	18	<code>integer :: i</code>
19		19	
20		20	
21	<code>!\$acc parallel loop independent present(v1,v2,v3,v4) private(i)</code>	21	<code>+ !\$acc parallel loop independent gang vector &amp;</code>
		22	<code>+ !\$acc&amp; present(v1,v2,v3,v4) private(i)</code>
22	<code>do i=1,sz</code>	23	<code>do i=1,sz</code>
23	<code>v1(i) = 0.0</code>	24	<code>v1(i) = 0.0</code>
24	<code>v2(i) = 0.0</code>	25	<code>v2(i) = 0.0</code>
38		39	
39	<code>integer :: i</code>	40	<code>integer :: i</code>
40		41	
41	<code>!\$acc parallel loop independent present(uvm,uxm,vym,uyv,vxm) private(i)</code>	42	<code>+ !\$acc parallel loop independent gang vector &amp;</code>
		43	<code>+ !\$acc&amp; present(uvm,uxm,vym,uyv,vxm) private(i)</code>
42	<code>do i=1,sz</code>	44	<code>do i=1,sz</code>
43	<code>uvm(i) = (uxm(i)+vym(i)-uym(i)+vxm(i))*inv_nxny</code>	45	<code>uvm(i) = (uxm(i)+vym(i)-uym(i)+vxm(i))*inv_nxny</code>
44	<code>enddo</code>	46	<code>enddo</code>
285	<code>! NOTE: The FFT will generate an unwanted n=0,p=-nr/2 component</code>	287	<code>! NOTE: The FFT will generate an unwanted n=0,p=-nr/2 component</code>
286	<code>! that will be filtered in the main time-stepping loop</code>	288	<code>! that will be filtered in the main time-stepping loop</code>
287		289	
288	<code>!\$acc parallel loop independent collapse(4) private(itor,ix,iy) present(f_n1,fxmany)</code>	290	<code>+ !\$acc parallel loop independent collapse(4) gang vector &amp;</code>
		291	<code>+ !\$acc&amp; private(itor,ix,iy) present(f_n1,fxmany)</code>
289	<code>do itm=1,n_toroidal_procs</code>	292	<code>do itm=1,n_toroidal_procs</code>
290	<code>do itl=1,nt_loc</code>	293	<code>do itl=1,nt_loc</code>
291	<code>do j=1,nsplit</code>	294	<code>do j=1,nsplit</code>

5 cgYRO/src/cgyro_parallel_lib.F90			
283		283	
284	<code>j1 = 1+iprocnj_loc</code>	284	<code>j1 = 1+iprocnj_loc</code>
285	<code>j2 = (1+iprocnj_loc</code>	285	<code>j2 = (1+iprocnj_loc</code>
286	<code>!\$acc parallel loop collapse(4) independent private(j_loc) &amp;</code>	286	<code>+ !\$acc parallel loop collapse(4) gang vector independent private(j_loc) &amp;</code>
287	<code>!\$acc&amp; present(fsendr,fin) present(nproc,nk1,nk2,ni_loc) copyin(j1,j2) default(none)</code>	287	<code>+ !\$acc&amp; present(fsendr,fin) present(nproc,nk1,nk2,ni_loc) &amp;</code>
		288	<code>+ !\$acc&amp; copyin(j1,j2) default(none)</code>
288	<code>do k=1,nproc</code>	289	<code>do k=1,nproc</code>
289	<code>do itor=nk1,nk2</code>	290	<code>do itor=nk1,nk2</code>
290	<code>do j=j1,j2</code>	291	<code>do j=j1,j2</code>

6 cgYRO/src/cgyro_rhs.gpu.F90			
63	<code>if (n_field &gt; 1) then</code>	63	<code>if (n_field &gt; 1) then</code>
64	<code>call timer_lib_in('str')</code>	64	<code>call timer_lib_in('str')</code>
65		65	
66	<code>!\$acc parallel loop collapse(3) independent private(iv_loc,is) &amp;</code>	66	<code>+ !\$acc parallel loop collapse(3) independent gang vector &amp;</code>
		67	<code>+ !\$acc&amp; private(iv_loc,is) &amp;</code>
67	<code>!\$acc&amp; present(is_v,z,temp,jvec_c) &amp;</code>	68	<code>!\$acc&amp; present(is_v,z,temp,jvec_c) &amp;</code>
68	<code>!\$acc&amp; present(nt1,nt2,nv1,nv2,nc) &amp;</code>	69	<code>!\$acc&amp; present(nt1,nt2,nv1,nv2,nc) &amp;</code>
69	<code>!\$acc&amp; default(none) async(1)</code>	70	<code>!\$acc&amp; default(none) async(1)</code>
87	<code>else</code>	88	<code>else</code>
88	<code>call timer_lib_in('str_mem')</code>	89	<code>call timer_lib_in('str_mem')</code>
89		90	
90	<code>!\$acc parallel loop collapse(3) independent private(iv_loc) &amp;</code>	91	<code>+ !\$acc parallel loop collapse(3) gang vector &amp;</code>
		92	<code>+ !\$acc&amp; independent private(iv_loc) &amp;</code>
91	<code>!\$acc&amp; present(nt1,nt2,nv1,nv2,nc) &amp;</code>	93	<code>!\$acc&amp; present(nt1,nt2,nv1,nv2,nc) &amp;</code>
92	<code>!\$acc&amp; default(none) async(1)</code>	94	<code>!\$acc&amp; default(none) async(1)</code>
93	<code>do itor=nt1,nt2</code>	95	<code>do itor=nt1,nt2</code>

2 cgYRO/src/cgyro_source.F90			
32	<code>icp = (ir-1+1)*n_theta</code>	32	<code>icp = (ir-1+1)*n_theta</code>
33		33	
34	<code>#ifdef _OPENACC</code>	34	<code>#ifdef _OPENACC</code>
35	<code>!\$acc parallel loop private(j) present(h_x,source)</code>	35	<code>+ !\$acc parallel loop gang_vector private(j) present(h_x,source)</code>
36	<code>#else</code>	36	<code>#else</code>
37	<code>!\$omp parallel do private(j)</code>	37	<code>!\$omp parallel do private(j)</code>
38	<code>#endif</code>	38	<code>#endif</code>

4 cgYRO/src/cgyro_upwind.F90			
98	<code>call timer_lib_in('str')</code>	98	<code>call timer_lib_in('str')</code>
99		99	
100	<code>#ifdef _OPENACC</code>	100	<code>#ifdef _OPENACC</code>
101	<code>!\$acc parallel loop collapse(3) independent &amp;</code>	101	<code>+ !\$acc parallel loop collapse(3) independent gang_vector &amp;</code>
102	<code>!\$acc&amp; present(is_v,ix_v,ie_v,xi,vel,upfac2,g_x,upwind_res) &amp;</code>	102	<code>!\$acc&amp; present(is_v,ix_v,ie_v,xi,vel,upfac2,g_x,upwind_res) &amp;</code>
103	<code>!\$acc&amp; private(iv_loc,is,ix,ie) present(nt1,nt2,nv1,nv2,nc) default(none)</code>	103	<code>!\$acc&amp; private(iv_loc,is,ix,ie) present(nt1,nt2,nv1,nv2,nc) default(none)</code>
104	<code>#else</code>	104	<code>#else</code>
105	<code>call timer_lib_in('str')</code>	105	<code>call timer_lib_in('str')</code>
106		106	
107	<code>#ifdef _OPENACC</code>	107	<code>#ifdef _OPENACC</code>
108	<code>!\$acc parallel loop collapse(3) independent &amp;</code>	108	<code>+ !\$acc parallel loop collapse(3) independent gang_vector &amp;</code>
109	<code>!\$acc&amp; present(is_v,ix_v,ie_v,xi,vel,upfac2,g_x,upwind32_res) &amp;</code>	109	<code>!\$acc&amp; present(is_v,ix_v,ie_v,xi,vel,upfac2,g_x,upwind32_res) &amp;</code>
110	<code>!\$acc&amp; private(iv_loc,is,ix,ie) &amp;</code>	110	<code>!\$acc&amp; private(iv_loc,is,ix,ie) &amp;</code>
111	<code>!\$acc&amp; present(nt1,nt2,nv1,nv2,nc) default(none)</code>	111	<code>!\$acc&amp; present(nt1,nt2,nv1,nv2,nc) default(none)</code>