



Source routines

An introduction to a new approach to source routines

Why user routines?

- FLUKA offers plenty of built-in tools to define primary beams and estimate quantities
- Sometime these are not enough
- There is the need to write some dedicated code: a “User Routine”
- URs are beyond the scope of this course because of intrinsic difficulties
- Nevertheless, we have started an effort to make URs more user-friendly
- We want to introduce here the first effort in this direction:
 - a modernized version of the **source routine**
- Why the source routine first? Built-in options allow to sample from a limited number of distribution and not from histograms. This is an effort to overcome this limitation

The “old” source routine

- Scary for beginners, limited documentation
- Use of **IMPLICIT** and **FORTRAN77** naming convention (see later)

```
1 *
2 *==== source =====
3 *
4 SUBROUTINE SOURCE ( NOMORE )
5
6 INCLUDE 'dblprc.inc'
7 INCLUDE 'dimpar.inc'
8 INCLUDE 'iounit.inc'
9
10 -----
11 *
12 * Copyright (C) 2003-2019: CERN & INFN
13 * All Rights Reserved.
14 *
15 * New source for FLUKA9x-FLUKA20xy:
16 *
17 * Created on 07 January 1990 by Alfredo Ferrari & Paola Sala
18 *                               Infn - Milan
19 *
20 * This is just an example of a possible user written source routine.
21 * note that the beam card still has some meaning - in the scoring the
22 * maximum momentum used in deciding the binning is taken from the
23 * beam momentum. Other beam card parameters are obsolete.
24 *
25 * Output variables:
26 *
27 *     Nomore = if > 0 the run will be terminated
28 *
29 *-----
30 *
31 * INCLUDE 'beamcm.inc'
32 * INCLUDE 'fheavy.inc'
33 * INCLUDE 'fkstk.inc'
34 * INCLUDE 'ioiocm.inc'
35 * INCLUDE 'ltclcm.inc'
36 * INCLUDE 'paprop.inc'
37 * INCLUDE 'sourcm.inc'
38 * INCLUDE 'sumcou.inc'
39 *
40 * LOGICAL LFIRST, LISNUT
41 *
42 * SAVE LFIRST
43 * DATA LFIRST / .TRUE. /
44 *
45 * Statement function:
46 LISNUT (I) = INDEX ( PRNAME (I), 'NEUTRI' ) .GT. 0
47 *-----
48 *
49 *
50 *-----
51 *
52 * NOMORE = 0
53 *
54 * First call initializations:
55 IF ( LFIRST ) THEN
56 ** The following 3 cards are mandatory **
57 TKESUM = ZERZER
58 LFIRST = .FALSE.
59 LUSSRC = .TRUE.
60 ** User initialization **
61 END IF
62 *-----
63 * Push one source particle to the stack. Note that you could as well
64 * push many but this way we reserve a maximum amount of space in the
65 * stack for the secondaries to be generated
66 * Npflka is the stack counter: of course any time source is called it
67 * must be = 0
68
69 * NPFPLKA = NPFPLKA + 1
70 * Wt is the weight of the particle
71 * WTFLK (NPFPLKA) = ONEONE
72 * WEIPRI = WEIPRI + WTFLK (NPFPLKA)
73 * Particle type (=proton.....). Ijbeam is the type set by the BEAM
74 * card
75 *-----
76 * (Radioactive) isotope:
77 IF ( IJBEAM .EQ. -2 .AND. LRDBEA ) THEN
78 IARES = IPROA
79 IZRES = IPROZ
80 IISRES = IPRON
81 CALL STISB ( IARES, IZRES, IISRES )
82 IJHION = IPRON * 100000 + MOD ( IPROZ, 100 ) * 1000 + IPROA
83 IJHION = IJHION * 100 + KXHEAV
84 IONID = IJHION
85 CALL DCION ( IONID )
86 CALL SETION ( IONID )
87 LFRPHN (NPFPLKA) = .FALSE.
88 *-----
89 * Heavy ion:
90 ELSE IF ( IJBEAM .EQ. -2 ) THEN
91 IJHION = IPRON * 100000 + MOD ( IPROZ, 100 ) * 1000 + IPROA
92 IJHION = IJHION * 100 + KXHEAV
93 IONID = IJHION
94 CALL DCION ( IONID )
95 CALL SETION ( IONID )
96 ILOFLK (NPFPLKA) = IJHION
97 * Flag this is prompt radiation
98 LRADD (NPFPLKA) = .FALSE.
99 * Group number for "low" energy neutrons, set to 0 anyway
100 IGROUP (NPFPLKA) = 0
101 * Parent radioactive isotope:
102 IRDAZM (NPFPLKA) = 0
103 * Particle age (s)
104 AGESTK (NPFPLKA) = +ZERZER
105 * Kinetic energy of the particle (GeV)
106 TKEFLK (NPFPLKA) = SQRT ( PBEAM**2 + AM (IONID)**2 )
107 * AM (IONID)
108 * Particle momentum
109 PMOFLK (NPFPLKA) = PBEAM
110 * PMOFLK (NPFPLKA) = SQRT ( TKEFLK (NPFPLKA) * ( TKEFLK (NPFPLKA)
111 * + TWOTWO * AM (IONID) ) )
112 *
113 * LFRPHN (NPFPLKA) = .FALSE.
114 *-----
115 * Normal hadron:
116 ELSE
117 IONID = IJBEAM
118 ILOFLK (NPFPLKA) = IJBEAM
119 * Flag this is prompt radiation
120 LRADD (NPFPLKA) = .FALSE.
121 * Group number for "low" energy neutrons, set to 0 anyway
122 IGROUP (NPFPLKA) = 0
123 * Parent radioactive isotope:
124 IRDAZM (NPFPLKA) = 0
125 * Particle age (s)
126
127 * AGESTK (NPFPLKA) = +ZERZER
128 * Kinetic energy of the particle (GeV)
129 TKEFLK (NPFPLKA) = SQRT ( PBEAM**2 + AM (IONID)**2 )
130 * AM (IONID)
131 * Particle momentum
132 PMOFLK (NPFPLKA) = PBEAM
133 * PMOFLK (NPFPLKA) = SQRT ( TKEFLK (NPFPLKA) * ( TKEFLK (NPFPLKA)
134 * + TWOTWO * AM (IONID) ) )
135 *
136 * Check if it is a neutrino, if so force the interaction
137 * (unless the relevant flag has been disabled)
138 IF ( LISNUT (IJBEAM) .AND. LNUFIN ) THEN
139 LFRPHN (NPFPLKA) = .TRUE.
140 *-----
141 * Not a neutrino
142 ELSE
143 LFRPHN (NPFPLKA) = .FALSE.
144 *
145 * END IF
146 *-----
147 *
148 * END IF
149 *-----
150 * From this point .....
151 * Particle generation (1 for primaries)
152 LOFLK (NPFPLKA) = 1
153 * User dependent flag:
154 LOUSE (NPFPLKA) = 0
155 * No channeling:
156 KCHFLK (NPFPLKA) = 0
157 ECRFLK (NPFPLKA) = ZERZER
158 * Extra infos:
159 INFSTK (NPFPLKA) = 0
160 LNFSTK (NPFPLKA) = 0
161 ANFSTK (NPFPLKA) = ZERZER
162 * Parent variables:
163 IPRSTK (NPFPLKA) = 0
164 EKPSK (NPFPLKA) = ZERZER
165 * User dependent spare variables:
166 DO 100 ISPR = 1, MKBMX1
167 SPAREK (ISPR,NPFPLKA) = ZERZER
168 100 CONTINUE
169 * User dependent spare flags:
170 DO 200 ISPR = 1, MKBMX2
171 ISPAK (ISPR,NPFPLKA) = 0
172 200 CONTINUE
173 * Save the track number of the stack particle:
174 ISPAK (MKBMX2,NPFPLKA) = NPFPLKA
175 NPARMA = NPARMA + 1
176 NUNPAR (NPFPLKA) = NPARMA
177 NEVENT (NPFPLKA) = 0
178 DFNEAR (NPFPLKA) = +ZERZER
179 * ... to this point: don't change anything
180 AKNSHR (NPFPLKA) = -TWOTWO
181 * Cosines (tx,ty,tz)
182 TXFLK (NPFPLKA) = UBEAM
183 TYFLK (NPFPLKA) = VBEAM
184 TZFLK (NPFPLKA) = WBEAM
185 * TZFLK (NPFPLKA) = WBEAM ( ONEONE - TXFLK (NPFPLKA)**2
186 * + TYFLK (NPFPLKA)**2 )
187 * Polarization cosines:
188 TXPOL (NPFPLKA) = -TWOTWO
189
190 TYPOL (NPFPLKA) = +ZERZER
191 TZPOL (NPFPLKA) = +ZERZER
192 * Particle coordinates
193 XFLK (NPFPLKA) = XBEAM
194 YFLK (NPFPLKA) = YBEAM
195 ZFLK (NPFPLKA) = ZBEAM
196 * Calculate the total kinetic energy of the primaries: don't change
197 *-----
198 * (Radioactive) isotope:
199 IF ( IJBEAM .EQ. -2 .AND. LRDBEA ) THEN
200 *-----
201 * Heavy ion:
202 ELSE IF ( ILOFLK (NPFPLKA) .EQ. -2 .OR.
203 * ILOFLK (NPFPLKA) .GT. 100000 ) THEN
204 * TKESUM = TKESUM + TKEFLK (NPFPLKA) * WTFLK (NPFPLKA)
205 *-----
206 *
207 * Standard particle:
208 ELSE IF ( ILOFLK (NPFPLKA) .NE. 0 ) THEN
209 * TKESUM = TKESUM + ( TKEFLK (NPFPLKA) * AMDISC (ILOFLK(NPFPLKA)) )
210 * * WTFLK (NPFPLKA)
211 *-----
212 *
213 *
214 *
215 * ELSE
216 * TKESUM = TKESUM + TKEFLK (NPFPLKA) * WTFLK (NPFPLKA)
217 * END IF
218 *-----
219 *
220 * RADDLY (NPFPLKA) = ZERZER
221 * Here we ask for the region number of the hitting point.
222 * NREG (NPFPLKA) = ...
223 * The following line makes the starting region search much more
224 * robust if particles are starting very close to a boundary:
225 CALL GEOCRS ( TXFLK (NPFPLKA), TYFLK (NPFPLKA), TZFLK (NPFPLKA) )
226 CALL GEOREG ( XFLK (NPFPLKA), YFLK (NPFPLKA), ZFLK (NPFPLKA),
227 * NRGFLK(NPFPLKA), IDISC )
228 * Do not change these cards:
229 CALL GEOSH ( NHPNT (NPFPLKA), 1, -11, MLATT )
230 NLATT (NPFPLKA) = MLATT
231 CMPATH (NPFPLKA) = ZERZER
232 CALL SOEVSV
233 RETURN
234 *==== End of subroutine Source =====
235 END
236
```

The “new” source routine

- Distributed since FLUKA4-1.0 release
- Simplified appearance
- Long & meaningful names for variables and routines
- Use of **implicit none** (see later)
- Abundant comments and examples
- Advanced sampling routines
- Variables for user’s usage clearly indicated
- Lines not to be edited are “hidden” in routines
in the **source_library.inc** library file

} Removed from
snapshot

- Old source routines can still be used

```
! =====  
! BEGINNING of customizable code ←  
! =====  
  
* particle_code = ...  
  
* heavyion_atomic_number = ...  
* heavyion_mass_number = ...  
* heavyion_isomer = ...  
  
* radioactive_isotope = .true.  
  
* momentum_energy = ...  
  
* energy_logical_flag = .true.  
  
* particle_weight = ...  
  
* divergence_x = ...  
* divergence_y = ...  
  
* gaussian_divergence_logical_flag = .true.  
  
* coordinate_x = ...  
* coordinate_y = ...  
* coordinate_z = ...  
  
* direction_cosx = ...  
* direction_cosy = ...  
* direction_cosz = ...  
  
* direction_flag = ...  
  
* polarization_cosx = ...  
* polarization_cosy = ...  
* polarization_cosz = ...  
  
* particle_age = ...  
* kshort_component = ...  
* delayed_radioactive_decay = ...  
  
! =====  
! END of customizable code - Do not change below ←  
! =====  
  
if ( nomore .eq. 0 ) then  
    call set_primary()  
    if ( debug_logical_flag ) call print_primary( debug_lines )  
end if  
  
return  
*==== End of subroutine Source =====*  
end
```

The “new” source routine

- Without removing comments, examples and advanced features (notice the ratio of code and comment lines)
- Note: the snapshot is not meant to be read – Detailed view will follow

```
1 *
2 *
3 * Copyright (C) 2020 - CSN
4 * All Rights Reserved.
5 *
6 * Source routine or FLUKA.s
7 *
8 * Created on 24 September 2020 by David Marchisio & Roberto Versari
9 *
10 *
11 * Modified on 10 November 2020 by David Marchisio & Roberto Versari
12 *
13 *
14 * This is a simplified user written source routine utilizing a
15 * separate source routine library.
16 *
17 * It is intended as an alternative non-source-friendly version of the
18 * legacy source routine. Existing FLUKA source routines remain
19 * compatible.
20 *
21 * Note that the beam card still has some meaning - in the scoring the
22 * routine momentum card in deciding the kinetic energy to take from the
23 * beam momentum. Other beam card parameters are obsolete.
24 *
25 *
26 * Output variables:
27 *
28 * =====
29 *
30 * =====
31 *
32 * =====
33 *
34 * =====
35 *
36 * =====
37 *
38 * =====
39 *
40 * =====
41 *
42 * =====
43 *
44 * =====
45 *
46 * =====
47 *
48 * =====
49 *
50 * =====
51 *
52 * =====
53 *
54 * =====
55 *
56 * =====
57 *
58 * =====
59 *
60 * =====
61 *
62 * =====
63 *
64 * =====
65 *
66 * =====
67 *
68 * =====
69 *
70 * =====
71 *
72 * =====
73 *
74 * =====
75 *
76 * =====
77 *
78 * =====
79 *
80 * =====
81 *
82 * =====
83 *
84 * =====
85 *
86 * =====
87 *
88 * =====
89 *
90 * =====
91 *
92 * =====
93 *
94 * =====
95 *
96 * =====
97 *
98 * =====
99 *
100 * =====
101 *
102 * =====
103 *
104 * =====
105 *
106 * =====
107 *
108 * =====
109 *
110 * =====
111 *
112 * =====
113 *
114 * =====
115 *
116 * =====
117 *
118 * =====
119 *
120 * =====
121 *
122 * =====
123 *
124 * =====
125 *
126 * =====
127 *
128 * =====
129 *
130 * =====
131 *
132 * =====
133 *
134 * =====
135 *
136 * =====
137 *
138 * =====
139 *
140 * =====
141 *
142 * =====
143 *
144 * =====
145 *
146 * =====
147 *
148 * =====
149 *
150 * =====
151 *
152 * =====
153 *
154 * =====
155 *
156 * =====
157 *
158 * =====
159 *
160 * =====
161 *
162 * =====
163 *
164 * =====
165 *
166 * =====
167 *
168 * =====
169 *
170 * =====
171 *
172 * =====
173 *
174 * =====
175 *
176 * =====
177 *
178 * =====
179 *
180 * =====
181 *
182 * =====
183 *
184 * =====
185 *
186 * =====
187 *
188 * =====
189 *
190 * =====
191 *
192 * =====
193 *
194 * =====
195 *
196 * =====
197 *
198 * =====
199 *
200 * =====
201 *
202 * =====
203 *
204 * =====
205 *
206 * =====
207 *
208 * =====
209 *
210 * =====
211 *
212 * =====
213 *
214 * =====
215 *
216 * =====
217 *
218 * =====
219 *
220 * =====
221 *
222 * =====
223 *
224 * =====
225 *
226 * =====
227 *
228 * =====
229 *
230 * =====
231 *
232 * =====
233 *
234 * =====
235 *
236 * =====
237 *
238 * =====
239 *
240 * =====
241 *
242 * =====
243 *
244 * =====
245 *
246 * =====
247 *
248 * =====
249 *
250 * =====
251 *
252 * =====
253 *
254 * =====
255 *
256 * =====
257 *
258 * =====
259 *
260 * =====
261 *
262 * =====
263 *
264 * =====
265 *
266 * =====
267 *
268 * =====
269 *
270 * =====
271 *
272 * =====
273 *
274 * =====
275 *
276 * =====
277 *
278 * =====
279 *
280 * =====
281 *
282 * =====
283 *
284 * =====
285 *
286 * =====
287 *
288 * =====
289 *
290 * =====
291 *
292 * =====
293 *
294 * =====
295 *
296 * =====
297 *
298 * =====
299 *
300 * =====
301 *
302 * =====
303 *
304 * =====
305 *
306 * =====
307 *
308 * =====
309 *
310 * =====
311 *
312 * =====
313 *
314 * =====
315 *
316 * =====
317 *
318 * =====
319 *
320 * =====
321 *
322 * =====
323 *
324 * =====
325 *
326 * =====
327 *
328 * =====
329 *
330 * =====
331 *
332 * =====
333 *
334 * =====
335 *
336 * =====
337 *
338 * =====
339 *
340 * =====
341 *
342 * =====
343 *
344 * =====
345 *
346 * =====
347 *
348 * =====
349 *
350 * =====
351 *
352 * =====
353 *
354 * =====
355 *
356 * =====
357 *
358 * =====
359 *
360 * =====
361 *
362 * =====
363 *
364 * =====
365 *
366 * =====
367 *
368 * =====
369 *
370 * =====
371 *
372 * =====
373 *
374 * =====
375 *
376 * =====
377 *
378 * =====
379 *
380 * =====
381 *
382 * =====
383 *
384 * =====
385 *
386 * =====
387 *
388 * =====
389 *
390 * =====
391 *
392 * =====
393 *
394 * =====
395 *
396 * =====
397 *
398 * =====
399 *
400 * =====
401 *
402 * =====
403 *
404 * =====
405 *
406 * =====
407 *
408 * =====
409 *
410 * =====
411 *
412 * =====
413 *
414 * =====
415 *
416 * =====
417 *
418 * =====
419 *
420 * =====
421 *
422 * =====
423 *
424 * =====
425 *
426 * =====
427 *
428 * =====
429 *
430 * =====
431 *
432 * =====
433 *
434 * =====
435 *
436 * =====
437 *
438 * =====
439 *
440 * =====
441 *
442 * =====
443 *
444 * =====
445 *
446 * =====
447 *
448 * =====
449 *
450 * =====
451 *
452 * =====
453 *
454 * =====
455 *
456 * =====
457 *
458 * =====
459 *
460 * =====
461 *
462 * =====
463 *
464 * =====
465 *
466 * =====
467 *
468 * =====
469 *
470 * =====
471 *
472 * =====
473 *
474 * =====
475 *
476 * =====
477 *
478 * =====
479 *
480 * =====
481 *
482 * =====
483 *
484 * =====
485 *
486 * =====
487 *
488 * =====
489 *
490 * =====
491 *
492 * =====
493 *
494 * =====
495 *
496 * =====
497 *
498 * =====
499 *
500 * =====
501 *
502 * =====
503 *
504 * =====
505 *
506 * =====
507 *
508 * =====
509 *
510 * =====
511 *
512 * =====
513 *
514 * =====
515 *
516 * =====
517 *
518 * =====
519 *
520 * =====
521 *
522 * =====
523 *
524 * =====
525 *
526 * =====
527 *
528 * =====
529 *
530 * =====
531 *
532 * =====
533 *
534 * =====
535 *
536 * =====
537 *
538 * =====
539 *
540 * =====
541 *
542 * =====
543 *
544 * =====
545 *
546 * =====
547 *
548 * =====
549 *
550 * =====
551 *
552 * =====
553 *
554 * =====
555 *
556 * =====
557 *
558 * =====
559 *
560 * =====
561 *
562 * =====
563 *
564 * =====
565 *
566 * =====
567 *
568 * =====
569 *
570 * =====
571 *
572 * =====
573 *
574 * =====
575 *
576 * =====
577 *
578 * =====
579 *
580 * =====
581 *
582 * =====
583 *
584 * =====
585 *
586 * =====
587 *
588 * =====
589 *
590 * =====
591 *
592 * =====
593 *
594 * =====
595 *
596 * =====
597 *
598 * =====
599 *
600 * =====
601 *
602 * =====
603 *
604 * =====
605 *
606 * =====
607 *
608 * =====
609 *
610 * =====
611 *
612 * =====
613 *
614 * =====
615 *
616 * =====
617 *
618 * =====
619 *
620 * =====
621 *
622 * =====
623 *
624 * =====
625 *
626 * =====
627 *
628 * =====
629 *
630 * =====
631 *
632 * =====
633 *
634 * =====
635 *
636 * =====
637 *
638 * =====
639 *
640 * =====
641 *
642 * =====
643 *
644 * =====
645 *
646 * =====
647 *
648 * =====
649 *
650 * =====
651 *
652 * =====
653 *
654 * =====
655 *
656 * =====
657 *
658 * =====
659 *
660 * =====
661 *
662 * =====
663 *
664 * =====
665 *
666 * =====
667 *
668 * =====
669 *
670 * =====
671 *
672 * =====
673 *
674 * =====
675 *
676 * =====
677 *
678 * =====
679 *
680 * =====
681 *
682 * =====
683 *
684 * =====
685 *
686 * =====
687 *
688 * =====
689 *
690 * =====
691 *
692 * =====
693 *
694 * =====
695 *
696 * =====
697 *
698 * =====
699 *
700 * =====
701 *
702 * =====
703 *
704 * =====
705 *
706 * =====
707 *
708 * =====
709 *
710 * =====
711 *
712 * =====
713 *
714 * =====
715 *
716 * =====
717 *
718 * =====
719 *
720 * =====
721 *
722 * =====
723 *
724 * =====
725 *
726 * =====
727 *
728 * =====
729 *
730 * =====
731 *
732 * =====
733 *
734 * =====
735 *
736 * =====
737 *
738 * =====
739 *
740 * =====
741 *
742 * =====
743 *
744 * =====
745 *
746 * =====
747 *
748 * =====
749 *
750 * =====
751 *
752 * =====
753 *
754 * =====
755 *
756 * =====
757 *
758 * =====
759 *
760 * =====
761 *
762 * =====
763 *
764 * =====
765 *
766 * =====
767 *
768 * =====
769 *
770 * =====
771 *
772 * =====
773 *
774 * =====
775 *
776 * =====
777 *
778 * =====
779 *
780 * =====
781 *
782 * =====
783 *
784 * =====
785 *
786 * =====
787 *
788 * =====
789 *
790 * =====
791 *
792 * =====
793 *
794 * =====
795 *
796 * =====
797 *
798 * =====
799 *
800 * =====
801 *
802 * =====
803 *
804 * =====
805 *
806 * =====
807 *
808 * =====
809 *
810 * =====
811 *
812 * =====
813 *
814 * =====
815 *
816 * =====
817 *
818 * =====
819 *
820 * =====
821 *
822 * =====
823 *
824 * =====
825 *
826 * =====
827 *
828 * =====
829 *
830 * =====
831 *
832 * =====
833 *
834 * =====
835 *
836 * =====
837 *
838 * =====
839 *
840 * =====
841 *
842 * =====
843 *
844 * =====
845 *
846 * =====
847 *
848 * =====
849 *
850 * =====
851 *
852 * =====
853 *
854 * =====
855 *
856 * =====
857 *
858 * =====
859 *
860 * =====
861 *
862 * =====
863 *
864 * =====
865 *
866 * =====
867 *
868 * =====
869 *
870 * =====
871 *
872 * =====
873 *
874 * =====
875 *
876 * =====
877 *
878 * =====
879 *
880 * =====
881 *
882 * =====
883 *
884 * =====
885 *
886 * =====
887 *
888 * =====
889 *
890 * =====
891 *
892 * =====
893 *
894 * =====
895 *
896 * =====
897 *
898 * =====
899 *
900 * =====
901 *
902 * =====
903 *
904 * =====
905 *
906 * =====
907 *
908 * =====
909 *
910 * =====
911 *
912 * =====
913 *
914 * =====
915 *
916 * =====
917 *
918 * =====
919 *
920 * =====
921 *
922 * =====
923 *
924 * =====
925 *
926 * =====
927 *
928 * =====
929 *
930 * =====
931 *
932 * =====
933 *
934 * =====
935 *
936 * =====
937 *
938 * =====
939 *
940 * =====
941 *
942 * =====
943 *
944 * =====
945 *
946 * =====
947 *
948 * =====
949 *
950 * =====
951 *
952 * =====
953 *
954 * =====
955 *
956 * =====
957 *
958 * =====
959 *
960 * =====
961 *
962 * =====
963 *
964 * =====
965 *
966 * =====
967 *
968 * =====
969 *
970 * =====
971 *
972 * =====
973 *
974 * =====
975 *
976 * =====
977 *
978 * =====
979 *
980 * =====
981 *
982 * =====
983 *
984 * =====
985 *
986 * =====
987 *
988 * =====
989 *
990 * =====
991 *
992 * =====
993 *
994 * =====
995 *
996 * =====
997 *
998 * =====
999 *
1000 * =====
```



History of Fortran

- Fortran born in the early 1950s, and the first compiler was released in 1957

Standards:

- Fortran 66 – The first standard
- Fortran 77 – Extension on Fortran 66
- Fortran 90 – Dynamic memory allocation / introduction of the *Free* format
- Fortran 95 – High performance Fortran specification
- Fortran 2003 – Object oriented programming
- Fortran 2008 / 2018 – Extensions of Fortran 2003

FLUKA is still mostly (if not fully) compatible Fortran 77

This doesn't mean that we can't use newer things in our user routines

(Unexpected) Features and limitations of Fortran (77)

- Source file format
 - Fixed
 - Free
- Naming convention
- Subprograms
 - Functions
 - Subroutines
- Variable declaration
 - Implicit
 - Explicit

Source file format

- Fortran 77 uses the *Fixed* file format (extensions: **.f** or **.for**):
 - Maximum 78 characters in one line
 - First 6 are reserved for special function:
 - If the first character is 'c' or '*', then the line is a comment
 - If the 6th position is not empty, then the line is treated as a continuation of the previous one (Often the '&' character is used)
 - With the gfortran compiler it is possible to increase the maximum line length
 - In FLUKA 4 it is extended to 132 characters
- Fortran 90 introduced the *Free* format (extensions: **.f90**, [**.f95**, etc.]):
 - Code can start at the 1st position
- *Note*: It is not possible to mix both in the same source file.
Gfortran compiler expects the “correct” format based on the file extension.

Naming convention

- Fortran 77 variable and (subprogram) names:
 - Limited to 6 alphanumerical characters
 - Have to start with a letter
 - Case insensitive
 - Starting with Fortran 90 the variable names 
 - Can be up to 31 character long
 - Can contain letters, numbers and underscore ('_')
 - Have to start with a letter
 - Case insensitive
 - *Note:* Try to use descriptive names, to make code readable
- Feature exploited in the new source routine

Subprograms

- Two types:
 - Function
 - Has a return value
 - Used in assignment: `variable = function(input_variable_1, ...)`
 - Subroutine
 - Doesn't have a return value
 - Accessible with the CALL statement: `call subroutine(input_variable_1, ...)`
- Passing variables
 - In Fortran you pass the variable, not the value of the variable (Like passing a pointer in C)
 - This means the subprograms may irreversibly modify the value of the input variables
 - Desired behavior if you want to return multiple variables
 - Can lead to side effects

Variable declaration

- Fortran by default uses *implicit declaration*, which means the type of the variable (integer, real, etc.) is determined by a preset rule.
- The default rule is:
 - If the variable starts with the letter I, J, K, L, M, or N it is an integer
 - Otherwise, it is a real (single precision float)
- In FLUKA however:
 - Variables with the 1st letter I, J, K, L, M, and N are still integers
 - But the others are double precision (floats)
- It is possible (and necessary) to overwrite this with *explicit declaration*, where you manually specify the type of the variable, like:

```
double precision my_intensity  
logical my_flag
```

Variable declaration

- Biggest issue is that typos remain hidden:
 - If you have a typo in a variable name, the compiler won't raise an error
 - It is a different, but valid variable without a value
 - Using it in calculations will lead to unexpected results
- Other issue is the unexpected type conversion:
 - For example: Information is lost if you want to assign a double precision number to INTEGER
- Solution in the “new” source routine: **implicit none**
 - This statement disables the implicit declaration, and every variable has to be manually declared
 - Exception: FLUKAs built in variables don't need to be declared in the source routine
 - (they will remain implicitly declared)
- Convention in the “new” source routine:
 - Variables with uppercase names: FLUKA variables
 - Variables with lowercase names: explicitly declared variables

Numbers and Constants in User routines

- To keep the high accuracy of the calculation
 - Every variable containing a floating-point number should have the type *double precision*
 - The assigned numbers should also be double precision:
For example: `radius = 2.0D0`, or in a function: `variable = function(1.0D0)`
The 'D' character indicated, that this is number should be treated as double precision.
If it is 'E' or missing, then the number will be single precision
- To simplify writing numbers FLUKA already defined many numbers as variables:
 - `ONEONE = 1.0D0`
 - `TWOTWO = 2.0D0`
 - `HLFHLF = 0.5D0`
 - `PIPIPI = π = 3.141592...`
 - `TWOPIPI = 2π = 6.283185...`Full list available in the `dblprc.inc` include file

Source routine – Initialization

```
154  ! =====  
155  ! BEGINNING of user declared variables  
156  ! =====  
157  
158  
159  
160  ! =====  
161  ! END of user declared variables  
162  ! =====
```

- Dedicated space for the declaration of user variables (and functions)

Source routine – Initialization

```
166  if ( lfirst ) then
167      call initialization()
168      lfirst = .false.
169  end if
```

- Initialization of internal variables
- Only performed the first time the routine is called
- To overwrite the default values the relevant lines needs to be uncommented, by removing the ‘*’ at the beginning of the line.
(See next slides)

Source routine – Primary particle

```
196 *      particle_code = ...
```

- By default, the particle type given in the **BEAM** card is taken
- Particle codes explained in FLUKA manual section 5.1
- Possible application: beam made of more than one type particles

```
206 *      heavyion_atomic_number = ...  
207 *      heavyion_mass_number = ...  
208 *      heavyion_isomer = ...
```

- Only used if primary particle is set to HEAVYION or ISOTOPE
- Default values are set on the **HI-PROPE** card, or for ^{12}C if the card is missing

Source routine – Energy / momentum

```
236 *      momentum_energy = ...
```

- By default, the particle momentum is expected
- The default value is based on the **BEAM** card
(Automatically converted into momentum if energy is given on the **BEAM** card)
- If energy is specified in the source routine, the following logical value must be set *.true.*

```
248 *      energy_logical_flag = .true.
```

Source routine – Energy / momentum

- The momentum divergence set on the **BEAM** card is not retained
- It is necessary to specify in the source routine
- It is easy with the supplied functions / subroutine

Flat spectrum: 267 * momentum_energy = sample_flat_momentum_energy([min], [max]

Gaussian spectrum: 273 * momentum_energy = sample_gaussian_momentum_energy([mean], [

Maxwell-Boltzmann spectrum: 280 * momentum_energy = sample_maxwell_boltzmann_energy([temperat

Spectrum from histogram: 292 * momentum_energy = sample_histogram_momentum_energy([filenam

Exponential spectrum:
(biased sampling) 305 * call sample_exponential_energy_weight([e_min], [e_max], [in

Source routine – Particle weight

```
257 *      particle_weight = ...
```

- Monte Carlo concept for biased sources
- The default value (`particle_weight = 1.0`) is usually sufficient
- Not for a beginners' use, mentioned here for completeness
- Note: The exponential spectrum sampling subroutine, uses variable particle weight

Source routine – Beam divergence

```
319 *      divergence_x = ...  
320 *      divergence_y = ...
```

- By default:
 - values are taken from the **BEAM** card
 - It is assumed to be a flat angular distribution
- For Gaussian divergence the following logical value must be set *.true.*

```
332 *      gaussian_divergence_logical_flag = .true.
```

Source routine – Beam starting position

```
345 *      coordinate_x = ...  
346 *      coordinate_y = ...  
347 *      coordinate_z = ...
```

- By default, values are taken from the **BEAMPOS** card
- Beam shape set on the **BEAM** card, and
- Extended sources specified on additional **BEAMPOS** cards are not implemented

Source routine – Beam starting position

- Some predefined routines (2 functions and 1 subroutine) are already available:

Flat distribution:

```
358 *      coordinate_[a] = sample_flat_distribution( [min], [max] )
```

Gaussian distribution:

```
365 *      coordinate_[a] = sample_gaussian_distribution( [mean], [fwhm] )
```

Annular distribution:

```
379 *      call sample_annular_distribution( [rmin], [rmax], coordinate_[a],
```

Remember the values must be in double precision (1.0D0).

Note: If annular sampling is used, the coordinates has to be set manually as well.

Source routine – Beam direction

```
392 *      direction_cosx = ...  
393 *      direction_cosy = ...  
394 *      direction_cosz = ...
```

- By default, values are taken from the **BEAMPOS** card

- If the **direction_flag** is set to:

```
409 *      direction_flag = ...
```

- 0 : All three values are considered and they are normalized automatically (Default)
 - 1 : The manually set value of the z direction is disregarded. Instead, it is calculated from the x and y direction cosines with a positive sign.
 - 2 : As with option 1, but negative sign is used.
- A predefined subroutine is already available for isotropic direction sampling

```
422 *      call sample_isotropic_direction( direction_cosx, direction_cosy, direction_cosz )
```

Source routine – Debugging

- To help debug the source routine, the major particle parameters can be printed
- To enable this feature, set

```
549 *      debug_logical_flag = .true.
```

- The printed parameters:
 - Energy / momentum
 - Coordinates
 - Direction
 - Weight
- The number of primaries printed can be set with:

```
558 *      debug_lines = 100
```

Some predefined FLUKA random sampling routines

- FLUKA offers some useful, predefined routines for random sampling
- `my_variable = FLRNDM(XDUMMY)`
Assigns a 64-bit random number in [0,1)
- `call FLNRRN(gauss1)`
Returns a Gaussian distributed random number
- `call FLNRR2(gauss1, gauss2)`
Returns two uncorrelated Gaussian distributed random numbers
- `call SFECFE(sint, cost)`
Returns sine and cosine of a random azimuthal angle

SOURCE card and passing parameters

- To invoke a source routine, it is necessary to add a **SOURCE** card
- A **SOURCE** card can be empty or can be used to pass parameters to the routine
- Max. 18 numerical values (**WHASOU (ii)**) and 1 string (max. 8 characters) (**SDUSOU**) can be

```
† SOURCE          #1: 7.          #2: 250.         #3: 12.5
                    sdum: linksour  #4: 3.75        #5:              #6:
                    #7:              #8:              #9:
                    #10:             #11:             #12:
                    #13:             #14:             #15:
                    #16:             #17:             #18:
```

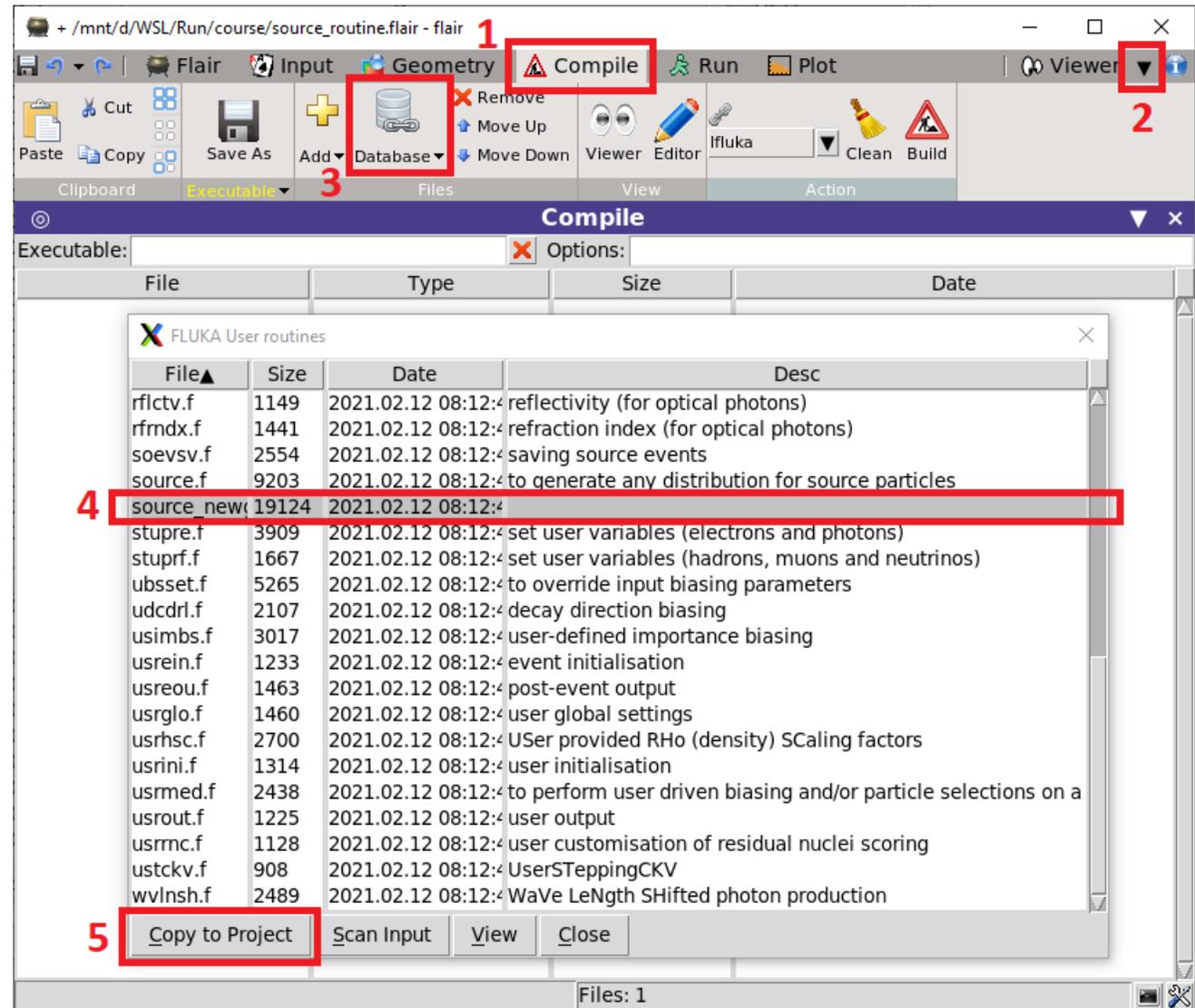
- Good practice advice:

Even if the beam energy / momentum is defined in the source routine, specify it in the **BEAM** card as it is used for internal initialization. Set a momentum value higher than the maximum possible one.

Adding the user routine to the project folder

1. Open [Compile] tab
2. It is maybe hidden in the dropdown menu
3. Click the [Database] button (Use [Add] for an existing file)
4. Select the user routine you want to use
5. Click [Copy to Project]

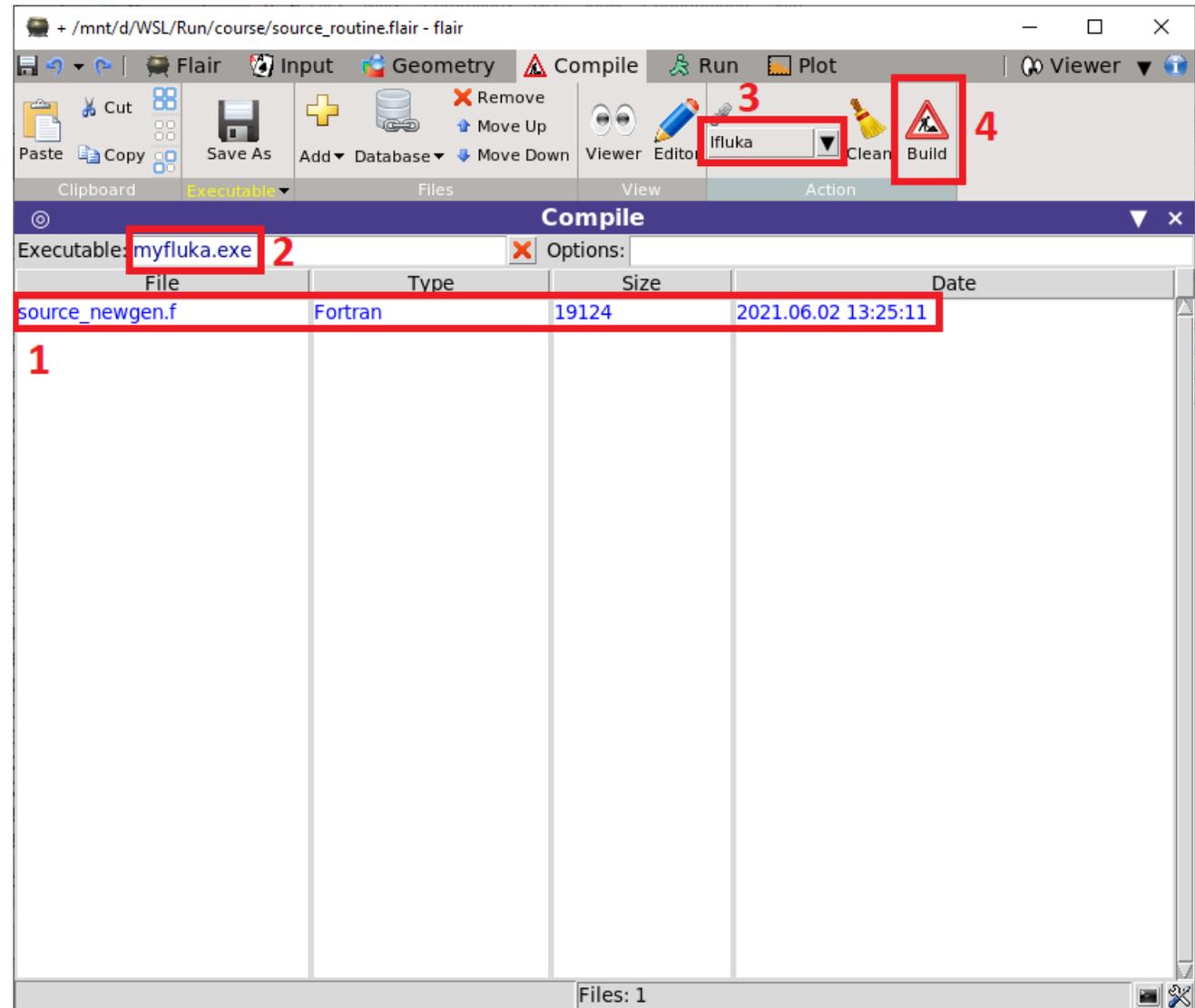
The copied user routine will be in the Flair projects directory



Compiling a custom FLUKA executable

1. Verify that the user routine is in the list
2. Name your custom executable
3. Select the appropriate linker:
 - a. Use *lfluka* by default
 - b. Use *ldpmqmd* if DPMJET or RQMD models are needed
4. Compile the executable

The custom executable should be set default on the [Run] tab automatically



Time to do some hands-on practice!

- We will now see together a few small examples of “new” source routine



xkcd.com/303

