CURRENT STATUS AND RECENT DEVELOPMENT OF THE XCOLL-FLUKA INTERFACE

A. Donadon Servelle*, R. Bruce, F. Cerutti, G. Hugo, B. Lindström, S. Redaelli, L.S. Esposito, F. Van der Veken, V. Vlachoudis, CERN, Geneva, Switzerland

Abstract

The Xsuite framework offers a modern environment for high-performance accelerator physics simulations. Its Xcoll module handles particle-collimator interactions using dedicated scattering routines, both built-in and external. FLUKA, a well-established Monte Carlo code, can be linked to Xcoll, enabling detailed modelling of particle-matter interactions and support for complex geometries. This paper presents the status of the Xcoll-FLUKA interface, which provides a consistent and efficient link between deterministic beam tracking and Monte Carlo simulations. The new setup builds upon the experience gained with the previous SixTrack-FLUKA coupling, introducing a more user-friendly design and significant improvements in data exchange, modularity, and extensibility within the Xsuite architecture. Recent developments include a new filtering algorithm with memory optimisation that improves simulation speed, a versatile definition of bent crystals, support for simplified collimator geometries represented as plain blocks of material, and an enhanced beam-beam interaction routine. These advancements represent an important step toward a flexible and performant framework for collimation and background studies in present and future accelerators.

INTRODUCTION

Simulations of beam collimation processes are regularly performed in accelerator complexes to assess cleaning performance and minimise particle losses around the ring. These simulations combine accelerator physics for element-by-element symplectic tracking of high-energy particles through a lattice, and particle-matter interactions at collimators.

Historically, Sixtrack [1] with its internal K2 scattering routine [2] served as the reference tracking code [3]. Since 2010 [4], Sixtrack could be coupled to the Monte Carlo code FLUKA [5–7], enabling more accurate results and support for ion beams, which K2 did not support. Sixtrack is being phase out and replaced by Xtrack. This is one of the various packages included in the Xsuite [8, 9] ecosystem, a modern collection of Python packages for beam dynamics simulations. Among these packages, Xcoll [10] handles particle-collimator interactions via its native engine *Everest* [11], the successor of K2, or through the interface with external codes like aforementioned FLUKA and Geant4 [12–14] via BDSIM [15].

This paper announces the public release of the Xcoll-FLUKA interface, to summarise its current status and the most relevant developments available.

* andre.donadon.servelle@cern.ch

SIMULATION WORKFLOW

Using the Xcoll-FLUKA interface requires a sequence of setup and runtime steps that are largely shared between the Sixtrack and Xsuite codes. The workflow proceeds as follows:

- Insert extraction/insertion markers (defined as `FlukaCollimator/FlukaCrystal` elements) into the lattice.
- Generate the FLUKA input file using *FLUKABuilder*, a sub-branch of Linebuilder code (see next section). This file defines, among other simulation features, the detailed geometry of all collimators included in the simulation.
- Launch `flukaserver` and initialise the TCP/IP network through `FLUKAIO` [16].
- During Xtrack tracking, whenever a particle reaches a collimator marker, Xcoll redirects the particles to FLUKA for interaction with matter (Fig. 1).
- Surviving particles, including energy losses and scattering products, are returned to Xsuite.
- Tracking continues until the target number of turns is reached or all particles are lost.

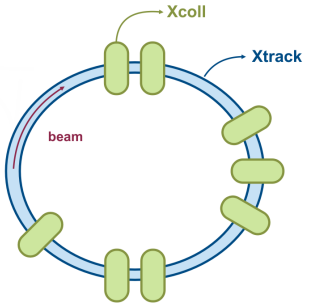

Figure 1: Schematic of Xtrack and Xcoll domains during particle tracking.

While these steps are conceptually equivalent between the two codes, Xcoll introduces a key architectural improvement: the `FlukaEngine` class encapsulates and automates the entire workflow. It launches both FLUKA and the `flukaserver` as background processes from within Xcoll, and interfaces Xsuite tracking with `FLUKAIO` through a lightweight Fortran wrapper. This design eliminates the manual, multi-step setup previously required from users and reduces the risk of configuration errors.

FLUKA INPUT GENERATION

A central task of the Xcoll-FLUKA interface is the automated generation of the FLUKA input file. The

`FlukaEngine` handles this by assembling a complete and self-consistent input from two complementary tools: the FLUKA Element DataBase (FEDB) and the LineBuilder. Together, these provide a structured library of accelerator components and the logic to compose them into beam-line geometries. The approach mirrors the one used for SixTrack, adapted here to Xsuite's data model.

FLUKA Element Database

The FEDB [17] is a compilation of FLUKA geometry models representing accelerator components used across CERN machines, including the LHC and the Future Circular Collider (FCC), among others. It encompasses collimators, magnets, masks, dumps, and other devices. Components are defined as geometry *prototypes*, which act as basic building blocks. For collimators, a prototype describes a single jaw, and combining two jaws with the surrounding tank (another prototype) forms a complete collimator, called an *assembly*. LineBuilder uses these assemblies to construct full beam lines or to use them individually.

LineBuilder

The LineBuilder [17] provides an efficient method to define complex accelerator beam-line geometries in FLUKA. It takes as input the lattice sequence, magnet strengths, and optical parameters, and produces a complete FLUKA input file with the corresponding beam-line geometry. For the Xsuite interface, LineBuilder generates isolated geometries for each collimator element in the lattice. When a particle reaches a collimator marker during Xsuite tracking, it is transported to the corresponding FLUKA geometry at the correct position.

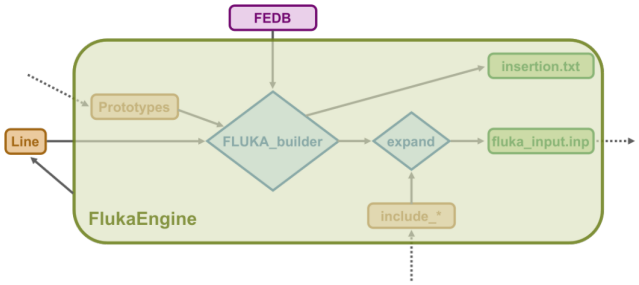

Figure 2: Schematic of the FLUKA input generation workflow.

The `FlukaEngine` coordinates this entire process (see Fig. 2) by generating the required auxiliary interface files, assembling the main FLUKA input from the set of `FlukaCollimator` elements, automatically producing include files based on the reference particles and user-defined settings, and building prototype files from `FlukaAssembly` definitions and the internal Xcoll database. The class also performs consistency checks, such as verifying agreement between the Xtrack lattice and the FLUKA input, and cross-checking particle mass values between Xsuite and FLUKA.

CURRENT STATUS AND RECENT DEVELOPMENTS

The Xcoll-FLUKA interface is now fully functional, supporting all the main features previously performed with the SixTrack-FLUKA coupling. Betatron and off-momentum loss maps for both protons and ions have been produced and benchmarked against simulations and measurements, showing excellent agreement [10, 18]. Building on this validated baseline, further developments were introduced to optimise performance and improve the user experience.

Filtering Algorithm and Memory Optimisation

Computing time was found to scale strongly with the number of particles sent to FLUKA and with the memory allocated for secondary particles, motivating the implementation of a particle-filtering algorithm. Before transferring particles to FLUKA, an optional geometrical pre-check is performed: particles predicted to miss the jaws simply drift and are not passed to FLUKA, while the remainder are forwarded to FLUKA for detailed tracking.

Because the collimator geometry contains the full length of the FLUKA assembly, some particles that pass the pre-check may still miss the material once tracked in FLUKA. Nevertheless, this filtering substantially reduces the number of particles injected into FLUKA and permits a better handling of the memory that must be reserved for secondaries. Knowing in advance how many particles will interact allows for a more appropriate choice of memory allocation in FLUKA, which showed to be the main bottleneck in the Xcoll-FLUKA interface.

Table 1 shows the runtime improvement for typical lossmap simulations with protons and ions, comparing the original Xsuite setup (without optimisation), the equivalent Sixtrack configuration, and the current Xsuite implementation with the algorithm enabled. The results demonstrate a dramatic improvement when using the algorithm, reducing Xsuite runtimes from days to hours. The new implementation achieves similar performance to SixTrack-FLUKA, confirming the effectiveness of this optimisation. A more detailed investigation into how the memory was handled in Sixtrack-FLUKA could allow to have a deeper understanding of these results, and if further optimisation is possible.

Table 1: Runtime Comparison between Xsuite and SixTrack for Different Simulation Setups

	Xsuite	**SixTrack**	**Xsuite (now)**
50k p, 200 turns	~2 days	< 6–8 h	< 2 h
1k Pb, 700 turns	~3 days	< 1 h	< 1 h

Generic Collimators

Some simulations may require collimator assemblies not yet available in the FEDB or might not require a full complex geometry. To address this, Xcoll supports *generic* collimators. These are simple rectangular material blocks that serve as jaws, defined by their active length and material alone.

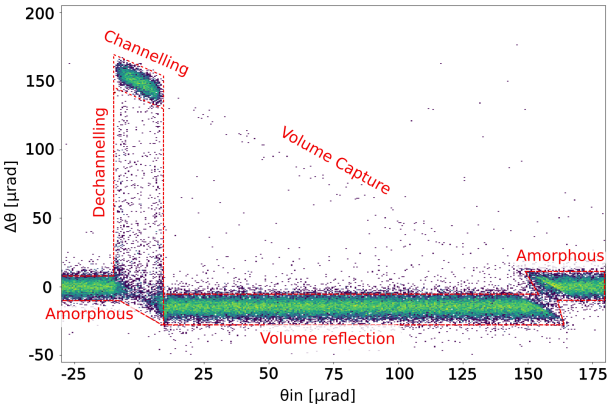

Figure 3: Proton angular kick distribution as a function of relative angle between incoming particles and crystal orientation.

```
xcoll.FlukaCollimator(length=0.6, material='cu')
```

Users can optionally adjust the height and width of the block if required.

Bent Crystals

Bent crystals are supported in Sixtrack collimation studies [3], but required hardcoded prototype parameters that imposed significant constraints on users. Xcoll introduces the `FlukaCrystal` class, which makes a crystal definition as straightforward as that of generic collimators, with full control over its length, material (silicon or germanium), bending angle or radius, width, height, and side.

```
xcoll.FlukaCrystal(length=2e-3, material='Si',
    jaw=1e-3, bending_angle=149e-6, width=1e-3,
    height=0.05, side='+')
```

This development is crucial for ion collimation studies, where bent crystals serve as the primary cleaning mechanism in the LHC. Since the implemented crystals in *Everest* are still not applicable for ions, only FLUKA and Geant4 can accurately model their scattering behaviour (see Fig. 3) . The inclusion of the `FlukaCrystal` element, therefore, completes the set of capabilities needed to fully replace SixTrack with Xsuite for crystal collimator workflows.

Beam-Beam Interaction Routine

The beam-beam interaction routine, which allows particle-particle collision with the FLUKA DPMJET-III [19] event generator, previously used with Sixtrack, suffered from hardcoded parameters requiring user intervention for each study. Xcoll's new `FlukaBeamBeamSource` class replaces this with a declarative interface, allowing users to specify the interaction point, interaction type (elastic, inelastic, electromagnetic dissociation), and beam characteristics directly.

```
xcoll.FlukaBeamBeamSource(
    line, colldb, int_type, ip,
    num_particles, pdg_id_b1,
    p0c_b1, pdg_id_b2, sig_z, sig_dpp)
```

Although this routine is still experimental and its API will most likely change, it has already been used in several studies, such as the analysis of the transmutation effect of oxygen and

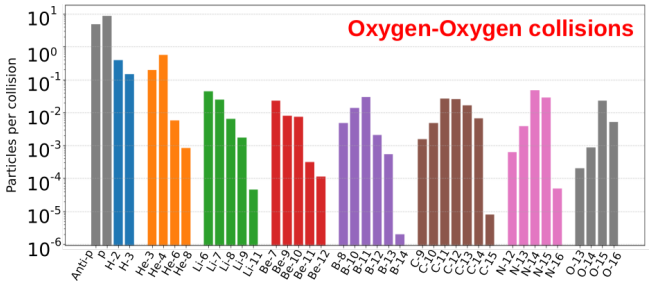

Figure 4: Particles produced from Oxygen-Oxygen beams in collision.

neon during the light ion run in the LHC in 2025 [20] (see Fig. 4), the assessment of losses affecting the background for the SND and FASER experiments [21], and the ongoing investigation of losses on detector in the context of the LHC-g study [22].

CONCLUSIONS AND OUTLOOK

The Xcoll-FLUKA interface, now publicly released, provides an integrated and automated solution for collimation simulations within the Xsuite framework, building on the experience gained with the legacy Sixtrack-FLUKA coupling. Before this development, users faced a complex, multi-step workflow requiring manual intervention at every stage: hand-crafted FLUKA input files, separate `flukaserver` launches, and hardcoded crystal parameters. Existing SixTrack-FLUKA setups already enabled complete simulations, but without the integration, modular data exchange, and Python-based interfaces now available in Xcoll.

The implementation presented in this paper combines deterministic particle tracking in Xtrack with the detailed particle-matter interaction models of FLUKA, while addressing both the runtime and usability limitations encountered in earlier Xsuite-based setups, similar performance with established SixTrack-FLUKA configurations has been achieved through a particle-filtering algorithm and memory optimisation that reduces the load on FLUKA. Usability has been improved by systematically removing hardcoded parameters and automating the workflow. The `FlukaEngine` class centralises setup and runtime management, eliminating many manual steps previously scattered across user scripts, and high-level classes such as `FlukaCollimator`, `FlukaCrystal`, and `FlukaBeamBeamSource` replace rigid configuration files with concise Python constructors.

Looking ahead, several directions are under active development. New features such as dynamic FLUKA configuration changes within Xcoll, for example, to modify collimator gaps as a function of time, and a reduction in the number of intermediate programming layers in connection to FLUKA are foreseen. These efforts are expected to further improve performance and maintainability, consolidating Xcoll-FLUKA as a central tool for collimation and background studies in current and future accelerator projects.

REFERENCES

[1] R. D. Maria *et al.*, "SixTrack Version 5: Status and New Developments", in *Proc. IPAC'19*, Melbourne, Australia, May 2019, pp. 3200–3203. doi:10.18429/JACoW-IPAC2019-WEPTS043

[2] T. Trenkler and J. Jeanneret, "K2, a software package evaluating collimation systems in circular colliders (manual)", CERN, Rep. SL/94-105 (AP), 1994.

[3] *ICFA Mini-Workshop on Tracking for Collimation in Particle Accelerators*, S. Redaelli, Ed. Geneva, Switzerland: CERN, 2018. doi:10.23732/CYRCP-2018-002

[4] E. Skordis *et al.*, "FLUKA coupling to Sixtrack", in *Proc. of the ICFA Mini-Workshop on Tracking for Collimation*, CERN, Geneva, Switzerland, Oct. 2015, pp. 17–25. doi:10.23732/CYRCP-2018-002.17

[5] G. Battistoni *et al.*, "Overview of the FLUKA code", *Ann. Nucl. Energy*, vol. 82, pp. 10–18, 2015. doi:10.1016/j.anucene.2014.11.007

[6] C. Ahdida *et al.*, "New capabilities of the FLUKA multi-purpose code", *Front. Phys.*, vol. 9, 2022. doi:10.3389/fphy.2021.788253

[7] FLUKA website, fluka.cern/

[8] G. Iadarola *et al.*, "Xsuite: An Integrated Beam Physics Simulation Framework", in *Proc. HB'23*, Geneva, Switzerland, Oct. 2023, pp. 73–80. doi:10.18429/JACoW-HB2023-TUA2I1

[9] Xsuite documentation, xsuite.web.cern.ch

[10] F. F. Van der Veken, "Introducing Xcoll: A streamlined approach to collimation and beam loss simulations using Xsuite", Presented at ICAP'24, Seeheim, Germany, Oct. 2024, indico.gsi.de/event/19249/contributions/82656/attachments/48844/70973/ICAP_Xcoll.pdf,

[11] D. Demetriadou, A. Abramov, G. Iadarola, and F. V. der Veken, "Tools for integrated simulation of collimation processes in Xsuite", in *Proc. IPAC'23*, Venice, Italy, May 2023, pp. 2801–2804. doi:10.18429/JACoW-IPAC2023-WEPA066

[12] S. Agostinelli *et al.*, "Geant4 – a simulation toolkit", *Nucl. Instrum. Methods Phys. Res. A*, vol. 506, pp. 250–303, 2003. doi:10.1016/S0168-9002(03)01368-8

[13] J. Allison *et al.*, "Recent developments in Geant4", *Nucl. Instrum. Methods Phys. Res. A*, vol. 835, pp. 186–225, 2016. doi:10.1016/j.nima.2016.06.125

[14] Geant4 website, geant4.web.cern.ch

[15] B. Lindstrom *et al.*, "Xcoll-BDSIM coupling for beam collimation", in *Proc. IPAC'25*, Taipei, Taiwan, May 2025, pp. 667–670. doi:10.18429/JACoW-IPAC2025-MOPS030

[16] P. Garcia Ortega, "Status of SixTrack-FLUKA coupling", Presentation at the LHC Collimation Working Group, Oct. 6, 2014, indico . cern . ch / event / 343152 / contributions / 1742013 / attachments / 675018 / 927588/Coupling.v2.pdf,

[17] A. Mereghetti, V. Boccone, F. Cerutti, R. Versaci, and V. Vlachoudis, "The FLUKA LineBuilder and Element DataBase: Tools for Building Complex Models of Accelerator Beam Lines", in *Proc. IPAC'12*, New Orleans, LA, USA, May 2012, paper WEPPD071, pp. 2687–2689. jacow.org/IPAC2012/papers/WEPPD071.pdf

[18] A. Donadon Servelle *et al.*, "An evaluation of collimation settings for the High Luminosity LHC baseline", in *Proc. IPAC'25*, Taipei, Taiwan, May 2025, pp. 2532–2535. doi:10.18429/JACoW-IPAC2025-THPB015

[19] S. Roesler, R. Engel, and J. Ranft, "The Monte Carlo event generator DPMJET-III", SLAC, Stanford, CA, USA, Rep. SLAC-PUB-8740, 2000.

[20] J. M. Jowett, "Beam transmutation in light-ion collisions", Presentation at the Light Ion Collisions at the LHC Workshop, Dec. 3, 2025, 2025, indico.cern.ch/event/1597414/contributions/6780513/,

[21] A. D. Servelle *et al.*, "Mitigation of muon backgrounds at LHC forward-physics experiments through orbit bumps", presented at IPAC'26, Deauville, France, May 2026, paper THP4105, this conference,

[22] A. Riart Vendrell *et al.*, "Simulation of beam-induced background in the LHC-g study", Presentation at the NDC section meeting, Mar. 25, 2026, indico . cern . ch / event / 1664372 / contributions / 7012071 / attachments / 3244971/5789261/thesis_LHCg_final.pdf,