SixTrack Overview for CWG

Speaker: R. De Maria

Special thanks to: F. Schmidt, E. McIntosh, LHC@Home volunteers.

Developers: A. Aviral, J. Barranco, D. Banfi, L. Deniau, M. Fjellstrom, P. Gargia, M. Fitterer, M. Giovannozzi, M. Javed, A. Mereghetti, A. Mituca, X. Valls Pla, Y. Papaphilippou, A. Patapenka, T. Pieloni, A. Santamaria, K. Singh, K. Sjobak.

LHC@Home support: I. Zacharov, P. Jones, A. Mereghetti, N. Oimyr, .

Collimation team: R. Assmann, C. Bracco, G. Robert-Demolaize, P. Hermes, L. Lari, A. Mereghetti, D. Mirarchi, V. Previtali, S. Redaelli, A. Rossi, C. Tambasco, T. Weiler.

Former contributors: M. Berz, E. Forest, H. Grote, F. James, K. Koelbig, K. Jeinemann, F. Neri H. Renshall, G. Ripken, M. Vaenttinen, A. Wrulich.

Legal support: N. Ziogas, M. Ayass.

What effects are included and what are not?

- SixTrack is an open source single particle tracking code used to simulate charged particle trajectories in synchrotrons for many turns with or without acceleration, with or without dynamic effects
- It contains symplectic models for drift, thick dipole and quadrupoles, thin multipoles and solenoids, accelerating cavities, (frozen) beam-beam interactions, linear and non-linear deflecting cavities and, in addition, scattering routines for absorber interactions.
- It also computes:
  - Phase space observables from tracking data: Linear and non linear invariants, Lyapunov analysis, tunes;
  - Optics functions using 4D, 5D, 6D formalisms;
  - High order transfer maps, normal forms;
- It supports Linux, Windows, MAC in a numerically portable way.
- It originates from RaceTrack and it has been developed at CERN in the last few decades mainly by F. Schmidt and E. Mcintosh.

What is used for at CERN

SixTrack is used to:

- Evaluate impact of magnetic field imperfection in the LHC, HL-LHC, FCC and specify target for field quality
- Evaluate the impact of weak-strong beam-beam effect in the LHC, HL-LHC, FCC with or without machine imperfections
- Simulate losses and collimation efficiencies in the LHC, HL-LHC, background
- Simulate failure scenarios (e.g. crab cavities in the HL-LHC)

Programming languages, style

SixTrack is made of:

- 70K lines written in Fortran 77/90 code blocks.
- Code blocks are assembled with two pre-processing steps to generate real fortran code
- It use an external (but embedded) C library to generate portable special functions
- It supports several compilers and operating systems
- Can be linked with BOINC libraries for the LHC@Home project.

The style is monolithic, procedural with very few functions/procedures, statically allocated shared stat. Very steep learning curve: dead or duplicated code, unused variables, cryptic names, unclear work flow due to extensive usage of goto statements. The resulting executable is very fast.

The code use vectorization as a form of parallelization.

The main tracking algorithm can be parallelized in the number of particles and can run asynchronously (should fit well GPU models, developments are on going SixTrackLib).

Programming languages, style

SixDesk and SixDB are used to prepare, submit, manage, collect and process jobs for LHC and FCC studies starting from MadX input and a parameter definition file.

- SixDesk and SixDB
  - About 70k lines shell scripting, Python.
  - Jobs management and physics intermixed.

The code is being reviewed by Alessio and it is heavily affected by the change of IT infrastructure.

Example of usage

Example of an LHC simulation:

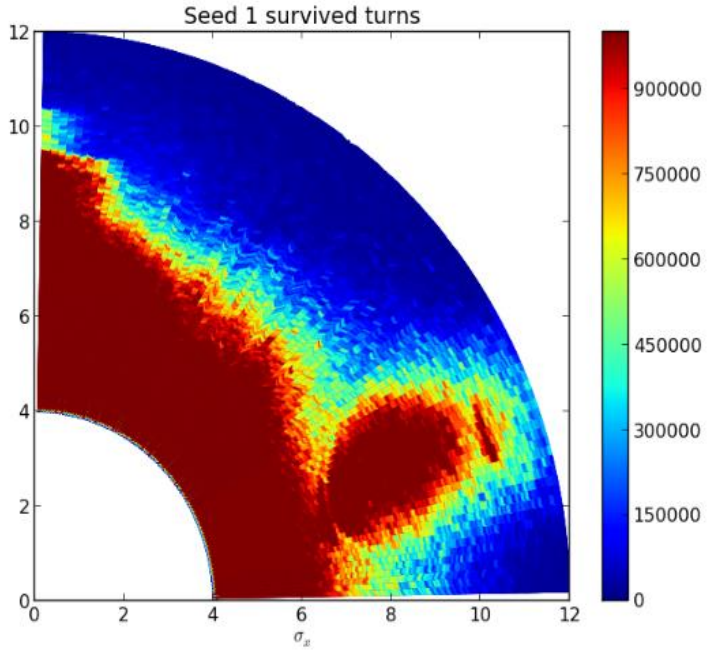
- 30k particles;
- 10^7 turns;
- 20k beam line elements.

Code speed:

- average 100 ns per particle per beam element
- 500 turns/(particle·sec) on serial code in recent hardware (LHC particles make 11245 turns/sec)

Memory footprint

- 100 MB



Documentation, License, Sources

SixTrack Website (cern.ch/sixtrack) is the single point of information:

- Access to code: GitHub repository for SixTrack and code browser.
- Documentation:
  - User manual: stable but under review
  - Physics manual: draft in progress but almost complete
  - Developer Manual: informal live wiki document
- Contacts: Support email, Mailing List.

SixTrack is licensed with LGPLv2.0 (source is open, can be used in non GPL application, modification needs to be published under LGPLv2.0)

Future Plans

- Continue the development and support of the main code:
  - Adding new physics for LHC/HLLHC/FCC studies
  - Extending physics Booster/PS/SPS and multiple ion species
  - Merging branches
  - General clean-up and improvement in the documentation
- Develop SixTrackLib
  - C Library implementing SixTrack Physics that can be embedded in other applications
  - Support GPU

Resources

Present resources for SixTrack (including SixDesk and Collimation) working on the code:

30% FTE summing all CERN staffs (2)

100% summing all CERN fellows (3),

students and several external collaborators do also contribute.

Resources are not sufficient to fix all bugs and feature request within the time scale of a year.

Typically user needs are covered by user themselves who often commit the code back to the baseline.

SixTrack Hardware

SixTrack runs one commodity PCs Linux and Windows, however studies are not practical on single machines. 300 CPUs are the needed for to study a field quality case in 3 hours.

Clusters like LXBATCH are well suited for SixTrack.

LHC@Home and volunteer computer is given an equivalent managed cluster of 5000 CPUs (or more as additional robustness will be achieved).

SixTrack can also run, almost unmodified, on supercomputers like CRAY o BlueGene using MPI.

In addition SixTrackLib would benefit from GPU with substantial FP64 Flops, e.g.:

AMD Radeon 280X (1TFlop/250$ gaming card not in production anymore),

AMD Radeon W8100 (1TFlop/1000$), AMD Radeon W9100 (2.5TFlop/3000$),

NVIDIA Tesla P100 (4.6TFlop/4600$).