BNL-102269-2014-TECH
RHIC/AP/165;BNL-102269-2013-IR

An SXF Extension for Alignment

W. Ficsher

November 1998

Collider Accelerator Department
Brookhaven National Laboratory

U.S. Department of Energy
USDOE Office of Science (SC)

DISCLAIMER

An SXF Extension for Alignment

W. Fischer and F. Pilat

November 13, 1998

1 Introduction

The accelerator description in the standard exchange format SXF is now used for RHIC and the LHC. Parsers exist to a growing number of simulation programs [1].

SXF, however, is a *flat* ordered list of elements and does not support the simultaneous misalignment of adjacent elements. The introduction of an element *hierarchy* is needed to align several elements by the same amount. This is of interest for two reasons: First, magnets are often modeled by more than one element and all these elements should be misaligned by the same amount. For example, a magnet model may consist of a body and two end kicks; and the body itself may be modeled by several kicks. Second, sometimes there are several magnets assembled in one cryostat. It is then desirable to misalign the whole cryostat while the individual magnets within are misaligned relative to the cryostat.

The proper description of alignment errors is of great importance to the US-LHC collaboration in which the US will deliver assembled cryostat-s of interaction region magnets to CERN. In the design phase an estimate of tolerable alignment errors is needed. Once magnets are assembled in a cryostat, the best position of the cryostat must be found in order to minimize the harmful effects of field errors.

We developed a description that allows the simultaneous misalignment of adjacent magnets. The description is closely related to the SXF format. A filter program can be used to process and merge the misalignment information into a flat SXF file.

2 An Example

We start with an example that shows how a file that describes the alignment errors may look like.

```
// description of alignment errors

// a single element with no attribute
 q1t1.r1
```

```
// a single element with an intentional non-zero alignment
 q6t1.r5 = { align = { al = [ 0.001 -0.002 0.003 0.005 -0.006 0.007 ] } }

// an element with intentional non-zero alignment and misalignment
 d1t1.r3 = { align = { al = [ 0.001 -0.002 0.003 0.005 -0.006 0.007 ] }
             align.dev = { al = [0.01 0.003 -.00012 0.0 0.1 -0.0001] }
           }

// an element that is contains of other elements
q1con = { components = { q1t1.r1 q1t2.r1 q1t3.r1 } }

// a misaligned element that contains misaligned elements
q2con = {align = { al = [ 0.002 -0.01 0.003 0.05 -0.05 0.07] }
         components = {
           q1t2.r1 = { align.dev = {al = [ .0 -0.002 0. 0. -0. 0.007]}}
           q1t3.r1 = { align.dev = {al = [ 0.001 0. 0. 0. -0.006 0.]}
                       components = {
                        x1.r1 = {align.dev = {al=[ 0. 0.2 0. 0. 0. 0.]}}
                        x2.r1 = {align.dev = {al=[ .2 0. 0. 0. 0. 0.1]}}
                       }
                     }
           q1t3.r1 = { align.dev = {al = [ 0. .2 0. 0. -0.006 0.]}}
          }
         }
```

3 Rules

The grammatical rules are the same as for SXF [1]. We repeat here the relevant rules for convenience while emphasizing specifics for our case. **Reserved words** rw are lower case. There are only four reserved words: *align*, *align.dev*, *al* and *components*. The allowed data types are **names** (which are case sensitive, with alphanumerics, upper and lower case, periods, dashes, colons and underscores all allowed), scalar **numbers** v (integer or decimal, E or e, D or d, exponential notation, or decimal point alone, all allowed), **arrays** a delimited as $[v1\ v2]$ or **associative arrays** delimited as $\{rw1 = v1\ rw2 = v2\ rw3 = v3\}$. The separation characters are blanks or end-of-line characters. Everything is case sensitive. C++ style comments are allowed; also, lines beginning with # are regarded as comments. Blank lines are ignored. Unrecognized words in locations where a reserved word is expected lead to a fatal error.

The entries of *align* describe alignments which are intentionally non-zero while the the entries of *align.dev* give alignment errors. The components of the array $al[0, 1, 2, 3, 4, 5]$ are $(\Delta x, \Delta y, \Delta s, \Delta\alpha_x, \Delta\alpha_y, \Delta\alpha_s)$ which are the (horizontal displacement, vertical displacement, longitudinal displacement, small rotations about the local x-axis, small rotations

about the local y-axis, small rotations about the local s-axis). The unit for $\Delta x, \Delta y, \Delta s$ is meter, the unit for $\Delta\alpha_x, \Delta\alpha_y, \Delta\alpha_s$ is rad.

The entries of *components* are the elements within the element. They too can be composed of sub-elements. The alignment of components within an element is always relative to the parent element.

References

[1] H. Grote et al., "SXF (Standard eXchange Format): definitions, syntax, examples", RHIC/AP/155 (1998).

[2] F. Pilat et al., "A Model of RHIC Using the Unified Accelerator Libraries", RHIC/AP/146 (1998).