EUROPEAN ORGANIZATION FOR NUCLEAR RESEARCH

CERN – SL DIVISION

Werner.Herr@cern.ch **AB-Note-2006-043 (ABP)**

Using SDDS data sets with MAD-X

W. Herr and F. Schmidt
AB Department, CERN, 1211-Geneva 23

Abstract

The MAD-X program, which is used for the design of the LHC optics and other accelerators, is presently modified to be used as an on-line simulation tool. In this note we describe additional features developed for MAD-X to read and write the data structures used in the LHC Application Software project.

Geneva, Switzerland

6 November 2006

1 SDDS data sets

1.1 Why SDDS in MAD-X ?

The Self Describing Data Set (SDDS) [1] was developed at the Advanced Photon Source (APS) at Argonne National Laboratory for use in the simulation and operation of accelerators. It was adopted as a data structure in the LHC Application Software project [2].

The MAD-X program [3], which is used for the design of the LHC optics and other accelerators, is presently modified to be used as an on-line simulation tool within the LHC control system. The main purpose is to have an on-line tool available for modeling and simulation of the LHC and to evaluate foreseen manipulations of the beams. However, it is not foreseen that it is used to directly control the beams, i.e. change settings of the hardware.
This MAD-XO (MAD-X On-line) will not be an integral part of the standard MAD-X program, but rather an extension and therefore a customized version to account for the needs of the control and modeling of the LHC.
A major issue is therefore an efficient communication between MAD-X and the control system (application !) software [4]. The standard format for input and output for MAD-X are files in the TFS (Table Filing System) style. To that purpose it was decided to provide a data interface between TFS and SDDS. The following tools are now available:

- Stand alone conversion between TFS and SDDS files.
- Stand alone template to access SDDS files.
- Read SDDS files and internal conversion to MAD-X tables.
- Conversion from MAD-X tables and write SDDS files.

Since MAD-X can write and read TFS files, it can be used as a converter.

1.2 SDDS protocol

A more complete description of the SDDS protocol and file structure is given in [5]. To make it self describing, all data elements are preceded by a definition of the data, including type, size and description. This information is stored in a header.
The header also includes so-called $parameters$ which can be used to store additional information such as time stamps, versions, energy etc.
The data elements are stored in form of tabulated data (columns) or arrays. Both, binary and ASCII files can be handled.

1.3 MAD-X and SDDS data sets

The MAD-X program [3] uses its own self describing data structure, the so-called TABLE structure. All information is stored internally in this $table$ data set and can be written to file or read from file using standard MAD-X commands. The main features and use of this data sets are:

- Dynamically defined at execution time.
- Simple format for archiving in files (tables).
- Communication of data from MAD-X.
- Communication of data to MAD-X.

The format of the TFS files allow easy processing or manipulation of the input and output data. The graphical presentation of data can be provided by generic programs like $gnuplot$ which can read and interpret these files. TFS files are always in ASCII format, a binary version is not foreseen. Both, TFS

	MAD-X	SDDS
one dimensional data	column	1-D array
two dimensional data	n/a	2-D array
tabular data	column	column
additional information	header	parameter

Table 1: *MAD-X TABLE and SDDS comparison.*

and SDDS formats are not relocatable.
For the more complex structure of SDDS a powerful toolkit is available [5] to manipulate the files or to analyze the data.

However, there are a few differences between TABLE and SDDS and in Tab.1 below we show a short comparison which also defines the different naming conventions. From Tab. 1 it can be seen that not all data structures have an equivalent counterpart in both representations. Therefore only data elements which can be handled in the TABLE structure are treated.

2 Implementation in MAD-X

For handling SDDS data sets, a new module has been written for MAD-X. To allow the input and output of SDDS files in MAD-X, the available SDDS source was used [6]. This ensures full compatibility and allows the use of the SDDS toolkit. The files (binary or ASCII) are read and an internal MAD-X TABLE structure is created to hold the data and other information. This table can be used internally like any other MAD-X table.
Data elements from a SDDS data set which cannot be implemented in a MAD-X table are ignored on input. The SDDS parameters are stored as "headers" in the MAD-X table. If the table is dumped to a file with $write, table$, the headers are conserved.

2.1 Input and output commands

The read command has the following form:

SDDSIN, FILE=$sdds\_filename$,TABLE=$tfs\_tablename$;

The internal table has the name $tfs\_tablename$, the default name is $sdds$.
The command for writing a SDDS table has the form:

SDDSOUT, FILE=$sdds\_filename$,TABLE=$tfs\_tablename$;

Any internal table can be written as a SDDS data set.

2.2 Selective input and output

The input command can be used together with a SELECT command which allows to read and store only the selected arrays. The command acts on the array names and therefore only $PATTERN$ can be used for the selection.

```
SELECT,FLAG=SDDS,CLEAR;
SELECT,FLAG=SDDS,PATTERN= ....;
SDDSIN, FILE=sdds_filename,TABLE=tfs_tablename;
```

This can be used to avoid problems with incompatible data elements. Regular expressions are allowed as a pattern.

It is possible to select only certain columns of an internal table for output on a SDDS data set. This is done with a command of the form:

```
SELECT,FLAG=tfs_tablename,CLEAR;
SELECT,FLAG=tfs_tablename,COLUMN= ....;
SDDSOUT, FILE=sdds_filename,TABLE=tfs_tablename;
```

Please note that the keyword $COLUMN$ is used for the selection since it refers to a TFS table. A selection of rows of the table is not foreseen.

3 Comments and pitfalls

3.1 Available data types

The possible data types are those which can be used in MAD-X tables, i.e. *double*, *long* and *string* data. All single precision floating point data or short integers are converted[1].
The implementation required a few changes to the MAD-X core, but are invisible to the user. These modifications will be propagated into the standard MAD-X program.

3.2 Naming conventions

The names of SDDS arrays and parameters are used for the TFS columns and headers respectively. This is the case for conversion in both directions. The selection with regular expressions using the $PATTERN$ option is case sensitive. However, internally all names of headers and columns are used as lower case and converted during the input of SDDS or TFS data sets.
When TFS tables are written, they are converted to upper case.
Therefore one should note that:

- Input files are accepted with both, lower and upper case names (take care with $PATTERN$ selection).
- After conversion all names on output files (SDDS or TFS) are uppercase.
- Lower and upper case must not be used to distinguish arrays (columns) with otherwise identical names.
- Parameters (headers) can have names identical to arrays (columns) since they are treated separately.

[1]Internally long integers are also stored as double precision floating point number.

4 Examples for the use of SDDS files

4.1 Reading SDDS files

```
TITLE, s='MAD-X test';
option,-echo;
sddsin,file="cngs-test.sdds",table="sdds_tfs";
write, table="sdds_tfs",file="tfs1";
```

4.2 Reading SDDS files with array selection

```
TITLE, s='MAD-X test';
option,-echo;
select,flag=sdds,clear;
select,flag=sdds,pattern="^b.*";
select,flag=sdds,pattern="^d.*";
! convert: read SDDS and write TFS files
sddsin,file="cngs-test.sdds",table="sdds_tfs";
write, table="sdds_tfs",file="tfs1";
```

4.3 Writing SDDS files

```
TITLE, s='MAD-X test';
option,-echo;
write, table="newtab",file="tfs3";
sddsout,table="newtab",file="myOUTfile";
```

4.4 Writing SDDS files from Twiss table

```
TITLE, s='MAD-X test';
option,-echo;
call file="TT41_010606_9.seq";
option,-echo,-info;
BEAM ENERGY=400.0,particle=proton;
use, period=tt41;
Select,flag=twiss,column=name,s,x,px,y,py,betx,bety,dx;
TWISS,file=tt41_06_9.out;
! write Twiss data as SDDS, use default table name "twiss"
sddsout,file="myOUTfile",table="twiss";
```

The command sequence below is equivalent: an existing TFS file is read into memory via the $readtable$ command and stored in the named table $newtab$. This table is dumped on a SDDS file again as above.

```
TITLE, s='MAD-X test';
option,-echo;
readtable,file="tt41_06_9.out",table="newtab";
sddsout,file="myOUTfile2",table="newtab";
```

References

[1] *Operations and Analysis Software Documentation*,
http://aps.anl.gov/Accelerator_Systems_Division/Operations_Analysis/oagSoftware.shtml

[2] http://slwww.cern.ch/pcrops/releaseinfo/pcropsdist/accsoft/sdds /accsoft-sdds-core/PRO/build/docs/api/

[3] *The MAD-X Home Page, version April 2005*, http://cern.ch/frank.schmidt/Xdoc/mad-X.html.

[4] http://ab-project-lsa.web.cern.ch/ab-project-lsa/

[5] M. Borland; *Getting Started with SDDS*, http://aps.anl.gov/Accelerator_Systems_Division/Operations_Analysis /manuals/GettingStartedWithSDDS/HTML/GettingStartedWithSDDS.html

[6] M. Borland and R. Soliday; *Applications Programmer's Guide for SDDS Version 1.5*, Advanced Photon Source, Argonne National Laboratory, U.S.A., May 2006.