tclSDDS TCL/TK Extension

Claude Saunders
Operations Analysis Group
Accelerator Systems Division
Argonne National Laboratory
Feb. 17, 1997

1. Introduction

This document serves as a User's Manual and Reference for the tclSDDS extension for tcl/tk. This extension provides commands to a tcl/tk interpreter which permit reading and writing SDDS files. All of the standard SDDS data types are supported, and all of the data models (parameter, array, column).

All tclSDDS commands begin with "sdds", and are followed by a space and then the desired sub-command. All commands return catchable errors. You can get a usage statement for every comand by typing "sdds <sub-command>".

2. Sample Application

The examples in the following sections omit the use of the catch statement for brevity, however it should always be used, as any and all of the commands return catchable errors. Here is an example of how a catch statement should be used.

```
# Example
if [catch {sdds open foo.sdds r} result] {
    puts stderr "unable to open foo.sdds: $result"
    exit
} else {
    set fileDescriptor $result
}
```

2.1. Reading

Assume a file called foo.sdds exists with parameter p1, 2-d array a1, and column c1. There are 2 data pages.

```
set fd [sdds open foo.sdds r]
```

```
# You can retrieve the list of column, parameter, and array names in a file
sdds getNames $fd column
sdds getNames $fd parameter
sdds getNames $fd array

# You can retrieve meta data about any given column, parameter, or array.
# These commands set a tcl array, with one element for each piece of information.
sdds getColumnInformation $fd c1 c1Info
puts "$c1Info(name) $c1Info(units) $c1Info(description) etc.."
# The command parray will print out all the contents of an array
parray c1Info

# The commands for parameter and array are identical
sdds getParameterInformation $fd p1 p1Info
sdds getArrayInformation $fd a1 a1Info

# This returns a list of two parameters, one from each page
set parameterList [sdds getParameter $fd p1]
# This returns a single list of all pages of column data concatenated together
set colData [sdds getColumn $fd c1]

# Alternately, you may cycle through the pages manually until EOF is returned
set moreData 1
set n 1
while {$moreData} {
    if [catch {sdds getColumn $fd c1 -page $n} res] {
        if {![string compare $res EOF]} {
            set moreData 0
        } else {
            puts stderr "error: $res"
            exit
        }
    } else {
        puts "page $n data: $res"
    }
    incr n
}

# Arrays must be retrieved one page at a time. The getArray command does not
# return the data, but rather sets an array variable in your current scope.
sdds getArray $fd tclArrayName a1 -page 1
parray tclArrayName

# Don't forget to close the file
sdds close $fd
```

2.2. Writing

Writing SDDS files from tcl/tk closely models the actual SDDS C functions.

```
# You can also do "sdds open new.sdds w SDDS_BINARY"
set fd [sdds open new.sdds w]

# Define all parameters, columns, and arrays
sdds defineParameter $fd par1 -type SDDS_STRING -fixedValue "any thing"
sdds defineParameter $fd par2 -type SDDS_DOUBLE -description "par" -units Amps
sdds defineColumn $fd col1 -type SDDS_LONG
sdds defineArray $fd sddsArray1 -type SDDS_DOUBLE -dimensions 2

# Write the header out
sdds writeLayout $fd
# Start a data page with 200 row max
sdds startPage $fd 200

# Set the page data
sdds setParameter $fd par2 123.234
```

```
sdds setColumn $fd col1 23 43 55 22 0 23 83 220 4401 32

# To set an array, first define the tcl array (a 2x3 array in this case)
set tclArr(0,0) 23.3
set tclArr(0,1) 44.3
set tclArr(0,2) 1.0032
set tclArr(1,0) 2.0093
set tclArr(1,1) 483.0
set tclArr(1,2) 9

# Set the array, giving the range of each dimension as arguments
sdds setArray $fd tclArr sddsArray1 2 3

# Write the data page out
sdds writePage $fd

# Can start additional data pages if desired
sdds startPage $fd 200
...
sdds writePage $fd

# Make sure to close the file
sdds close $fd
```

3. Command Reference

sdds open

sdds open <fileName> [<mode>] [SDDS_BINARY]

Open an sdds file for read or write. Opening for write will replace any existing file contents. The SDDS_BINARY option is available when writing a file.

- **<fileName>** - any legal unix file name
- **<mode>** - ‘r’ or ‘w’.

Returns:

- **<fd>** - a unique string for use in all other sdds calls

sdds close

sdds close <fd>

Closes sdds file and frees up resources.

sdds getColumn

sdds getColumn <fd> <columnName> [-page <n>]

Read given column from file and return as a list with one element per row. If -page option is omitted or 0, all pages are read and columns are concatenated. Pages start at number 1. You may proceed to read pages one at a time by incrementing the -page option until the command returns a catchable error with the return string “EOF”. This indicates all pages have been read.

- **<fd>** - from sdds open call
- **<columnName>** - name of sdds column
- **-page** - desired page number, or 0 for all

Returns:

- list with one element per row

sdds getParameter

sdds getParameter <fd> <parameterName> [-page <n>]

Read given parameter from file and return its value in a list. If -page option is omitted or 0, all pages are read and all parameter values are concatenated to one list. Pages start at number 1. You may proceed to read pages one at a time by incrementing the -page option until the command returns a catchable error with the return string “EOF”. This indicates all pages have been read.

- **<fd>** - from sdds open call
- **<parameterName>** - name of sdds parameter
- **-page** - desired page number, or 0 for all

Returns:

- list with one or more parameters

sdds getArray

sdds getArray <fd> <tclArrayName> <sddsArrayName> -page <n>

Read array from sdds file and create a tcl array with same dimensions. Nothing is returned with this call other than a catchable return.

- **<fd>** - from sdds open call
- **<tclArrayName>** - name of tcl array to create (in current scope)
- **<sddsArrayName>** - name of sdds array to read in
- **-page** - you must specify which page

Returns:

- nothing, other than catchable error

sdds getColumnInformation

sdds getColumnInformation <fd> <columnName> <tclArrayName>

Get all information associated with given sdds column name, and put it in given tcl array, one element per piece of information. Ie. array(units), array(symbol), array(type).

- **<fd>** - from sdds open call
- **<columnName>** - name of sdds column
- **<tclArrayName>** - name of tcl array to create

sdds getParameterInformation

sdds getParameterInformation <fd> <parameterName> <tclArrayName>

Get all information associated with given sdds parameter name, and put it in given tcl array, one element per piece of information. Ie. array(units), array(symbol), array(type).

- **<fd>** - from sdds open call
- **<parameterName>** - name of sdds parameter
- **<tclArrayName>** - name of tcl array to create

sdds getArrayInformation

sdds getArrayInformation <fd> <arrayName> <tclArrayName>

Get all information associated with given sdds array name, and put it in given tcl array, one element per piece of information. Ie. array(units), array(symbol), array(type).

- **<fd>** - from sdds open call
- **<arrayName>** - name of sdds array
- **<tclArrayName>** - name of tcl array to create

sdds getNames

sdds getNames <fd> [<class>]

Find out what columns, parameters, or arrays are defined in an open file.

- **<fd>** - from sdds open call
- **<class>** - “column” (default), “parameter”, or “array”

Returns:

- List of names

sdds defineParameter

sdds defineParameter <fd> <parameterName> [-type <string>] [-symbol <string>] [-units <string>] [-description <string>] [-formatString <string>] [-fixedValue <string>]

For writing files. Defines a new parameter. Defaults to type SDDS_STRING.

- **<fd>** - from sdds open call
- **<parameterName>** - name of new sdds parameter
- **-type** - SDDS_DOUBLE, SDDS_FLOAT, SDDS_LONG, SDDS_SHORT, SDDS_STRING, or SDDS_CHARACTER

Returns:

- nothing, other than catchable error

sdds defineColumn

sdds defineColumn <fd> <columnName> [-type <string>] [-symbol <string>] [-units <string>] [-description <string>] [-formatString <string>] [-fieldLength <string>]

For writing files. Defines a new column. Defaults to type SDDS_STRING.

- **<fd>** - from sdds open call
- **<columnName>** - name of new sdds column
- **-type** - SDDS_DOUBLE, SDDS_FLOAT, SDDS_LONG, SDDS_SHORT, SDDS_STRING, or SDDS_CHARACTER

Returns:

- nothing, other than catchable error

sdds defineArray

sdds defineArray <fd> <arrayName> [-type <string>] [-symbol <string>] [-units <string>] [-description <string>] [-formatString <string>] [-fieldLength <int>] [-dimensions <int>] [-groupName <string>]

For writing files. Defines a new array. Defaults to type SDDS_STRING.

- **<fd>** - from sdds open call
- **<arrayName>** - name of new sdds array
- **-type** - SDDS_DOUBLE, SDDS_FLOAT, SDDS_LONG, SDDS_SHORT, SDDS_STRING, or SDDS_CHARACTER
- **-dimensions** - number of dimensions (ie 2 or 3). Only 1, 2, or 3 is supported.

Returns:

- nothing, other than catchable error

sdds writeLayout

sdds writeLayout <fd>

For writing files. Writes SDDS header out after all sdds defines are complete. This must precede sdds startPage.

- **<fd>** - from sdds open call

Returns:

- nothing, other than catchable error

sdds startPage

sdds startPage <fd> <expectedRows>

For writing files. Begins a new SDDS data page. User supplies a maximum expected row count. This command must precede any sdds set commands.

- **<fd>** - from sdds open call
- **<expectedRows>** - maximum number of rows you expect to set

Returns:

- nothing, other than catchable error

sdds setParameter

sdds setParameter <fd> <parameterName> <parameterValue>

For writing files. Sets given parameter in current data page. Note that given value must convert to native SDDS type given in corresponding sdds defineParameter command. Ie. don't give a string "abc" when the parameter was defined as SDDS_LONG.

- **<fd>** - from sdds open call
- **<parameterName>** - name of defined sdds parameter
- **<parameterValue>** - a string containing parameter value

Returns:

- nothing, other than catchable error

sdds setColumn

sdds setColumn <fd> <columnName> <columnValue> [columnValue ...]

For writing files. Sets given column in current data page. Note that given value(s) must convert to native SDDS type given in corresponding sdds defineColumn command. Ie. don't give a string "abc" when the column was defined as SDDS_LONG.

- **<fd>** - from sdds open call
- **<columnName>** - name of defined sdds column
- **<columnValue>** - a string containing a column value

Returns:

- nothing, other than catchable error

sdds setRowValues

sdds setRowValues <fd> <rowNumber> <columnName> <columnValue> [<columnName> <columnValue> ...]

For writing files. Sets given row number in current data page. Note that given value(s) must convert to native SDDS type given in corresponding sdds defineColumn commands. Ie. don't give a string "abc" when the column was defined as SDDS_LONG.

- **<fd>** - from sdds open call
- **<rowNumber>** - row in data page to set
- **<columnName>** - name of defined sdds column
- **<columnValue>** - a string containing column value

Returns:

- nothing, other than catchable error

sdds setArray

sdds setArray <fd> <tclArrayName> <sddsArrayName> <dim0Range> ... <dimnRange>

For writing files. Sets given sdds array in current data page using data from tcl array. Tcl array is indexed as follows: array(1), or array(2,4), or array(1,3,0). Only one, two, and three-dimensional arrays are supported.

The arity of the tcl array must match that of the sdds array. Ie. tcl array must be 2d if sdds array was defined as 2d. In addition, the elements of the tcl array must convert to the native data type of the sdds array. You must supply the range of each dimension. Ie. if your tcl array is 2x3x5, you must indicated 2 3 5 as the last three arguments.

- **<fd>** - from sdds open call
- **<tclArrayName>** - tcl array containing data
- **<sddsArrayName>** - name of defined sdds array
- **<dimXRange>** - range of dimensions (if 2x3 array, then put 2 3)

Returns:

- nothing, other than catchable error

sdds writePage

sdds writePage <fd>

For writing files. Writes SDDS data page out to file. You must do another sdds startPage before setting any additional values.

- **<fd>** - from sdds open call

Returns:

- nothing, other than catchable error

4. Linking Against the tclSDDS Library

The “sdds” command set is available as a tcl extension library called “libtclSDDS.a”. The library is located in the epics/extensions source tree in the following locations:

```
epics/extensions/src/tcl/extensions/sdds
```

You must also link against the SDDS library which is in epics/extensions:

```
epics/extensions/src/SDDS
```

The xxAppInit.c file of your interpreter should include the following lines in order to initialize the tclSDDS extension:

```
if (TclSDDS_Init(interp) == TCL_ERROR) {
    return TCL_ERROR;
}
```

The link command for your interpreter must include this library along with several SDDS libraries:

-ltclSDDS -lSDDS1 -lrpnlib -lnamelist -lmdblib