PSFC/JA-07-22

MDSplus Extensions for Long Pulse Experiments

Fredian, T.W., Stillerman, J.A., Manduchi, G*

* Consorzio RFX, Euratom-ENEA Association, Padova, Italy

Plasma Science and Fusion Center
Massachusetts Institute of Technology
Cambridge MA 02139 USA

This work was supported by the U.S. Department of Energy, Grant No. DE-FC02-99ER54512.

MDSplus Extensions for Long Pulse Experiments

T. Fredian [1)*], J. Stillerman [1)], G. Manduchi [2)]

1) Massachusetts Institute of Technology, 175 Albany Street, Cambridge, MA 02139, USA
2) Consorzio RFX, Euratom-ENEA Association, Corso Stati Uniti 4, Padova 35127, Italy

Abstract

MDSplus is a data acquisition and analysis software package used widely throughout the international fusion research community. It was originally designed for use on pulsed experiments where experiment measurements are typically acquired by external measurement devices with local memory and then gathered by the data system after the pulse has completed. Today, more and more fusion research devices are being constructed which have very long pulse lengths. It is no longer acceptable to wait until the pulse completes before acquiring the measurements and making them available to the researchers. To explore the possibility of adapting MDSplus for use on experiments with long pulses, we have designed and implemented some prototype extensions to make MDSplus more suitable for those types of experiments. These extensions may be applicable in other areas such as data handling for fusion modeling codes. To implement these extensions, an additional API was developed to enable applications to store data incrementally in a node. The data is stored in a linked list of data segments in the MDSplus data file. An index of these data segments is maintained such that retrieval of subsets of the data can be performed efficiently without reading in the node's entire data set. Prior to accessing this type of appended data, an application simply has to establish a time context consisting of a start time, end time and optionally a delta time using a simple function call. Once the time context is established, all the existing data retrieval functions in MDSplus behave as normally except that data from segmented records will only include those samples falling within the time window. MDSplus functionality provided by the expression evaluator is retained without modification.

This work was supported by the U.S. Department of Energy, Cooperative Grant No. DE-FC02-99ER54512 and by the European Communities under the contract or Association with EURATOM/ENEA.

Keywords: MDSplus, Data Acquisition, Data Analysis, Long Pulse

[*] Corresponding author. Tel: (617) 253-7623; fax: (617) 253-0627
E-mail address: twf@psfc.mit.edu.

1. Introduction

MDSplus[1,2,3] is a data acquisition, data handling and analysis system used widely in the fusion community. It provides numerous powerful capabilities to simplify the process of storing measurements or analysis results in archives and to enable scientists to retrieve these data using a wide variety of programming languages and data manipulation tools. It provides a powerful expression evaluator which can be used to manipulate the data on retrieval and expressions can be stored in the archive to define new data items which are computed when retrieved. MDSplus also provides a remote data access capability enabling scientists to access data from anywhere on the Internet. The MDSplus system runs on many computing platforms including Windows, MacOS, Linux and several other variants of UNIX. It is used at over 30 fusion experiment or modeling sites worldwide.

MDSplus was originally designed and implemented in the late 1980's by a team comprised of the people from the Massachusetts Institute of Technology (USA), Los Alamos National Laboratory (USA) and Istituto Gas Ionizzati (Italy). The system was designed to support three short pulse fusion experiments under construction, C-Mod, RFX and ZTH. At that time, fusion experiments typically experienced pulse lengths ranging from fractions of seconds up to a few seconds and generated less than ten megabytes of data per pulse. Fortunately the advances of computer hardware have kept pace with the growth of data handling demands of fusion devices over the last 20 years. Although fusion experiments now often generate many gigabytes of data per pulse, MDSplus still proves to be a powerful tool for handling data in fusion experiments.

2. New Era of Fusion Experiments and Modeling Codes

Many fusion experiments recently built, under construction or planned such as EAST[4], KSTAR[5] and ITER[6] are designed to produce much longer duration pulses and in some cases approaching continuous operation. This greatly alters the data handling requirements compared to short pulse experiments. In short pulse experiments, the bulk of the data acquisition is done in a batch mode and occurs after pulse has completed. Since the pulses were short, scientists have been content to wait a minute or two before viewing or analyzing the data. With long pulse experiments, where the pulses may endure for many minutes or more, the quantity of data increases dramatically and the data must be acquired and accessed during the pulse.

A similar phenomenon is occurring in the modeling community. Codes are generating large quantities of data over long periods of time and much of the data must be monitored during the course of the code run.

Some sites, LHD[7] for example, have dealt with long pulses by approximating them as a series of adjacent short pulses. Other planned experiments such as W7X[8] and perhaps ITER are developing completely new data systems to handle quasi-continuous data flows.

3. The Challenge

Since MDSplus is currently in use at numerous experiments worldwide and many of the scientists in the fusion community are familiar with its tools and data access capabilities, a significant investment in learning time of both the scientist and the support staff could be avoided if the MDSplus system could be enhanced to provide a mechanism for handling long pulse data streams while continuing to provide all of the powerful features found in the original MDSplus. In addition, utilizing MDSplus on these long pulse experiments would enable scientists worldwide to access the data using the same tools they currently use for accessing existing short pulse experiments.

The additional functionality needed for long pulse experiments or modeling codes should be added in such a way that minimizes the changes required in the MDSplus interfaces and retain compatibility with existing data archives. While other scientific storage methods such as hdf5[9] provide very little backward compatibility with data sets constructed with earlier versions, MDSplus today can still access the original data files that were written dating back to the early 1990's. Any new enhancements for long pulse should not alter this policy of retaining compatibility with existing data or applications.

To provide most of the additional functionality required by long pulse experiments, MDSplus must be enhanced to permit the ability to efficiently append data to a signal stored in a node of an MDSplus tree and to provide a mechanism for efficiently retrieving a subset of the data. While MDSplus currently provides the ability to retrieve sub-sampled data using the subscripting capability in the expression evaluator, MDSplus subscripting retrieves the entire signal from the data file and extracts the desired samples. The existing capability is therefore too inefficient for long pulse data.

The challenge is to provide this ability to append data and retrieve subsets of the data efficiently in MDSplus while maintaining compatibility with existing capabilities and archived data.

4. Design Meeting

A one week design meeting was held during the summer of 2006 in the US to discuss and potentially design long pulse enhancements for MDSplus. The meeting was attended by the authors of this paper, three of the original designers and developers of the MDSplus system. Each of the authors has thought about ways to accommodate quasi-continuous data flows in MDSplus over the past few years and this hope of this meeting was to combine those thoughts in a brain storming session and hopefully arrive at a preliminary design to prototype.

The meeting was extremely productive and by the end of the week a design was roughed out and the tasks for developing a prototype implementation were defined and assigned. Subsequent correspondence via emails solidified the design and the resulting implementation will now be discussed.

5. Segmented Records

The main design concept arrived at for supporting quasi-continuous data in MDSplus is the use of "Segmented Records". As data is acquired, the data is stored in segments in the MDSplus data file. A segmented record in MDSplus consists of an index of the data segments and the segments themselves. As new data arrives it is appended to the current data segment if space permits. If not, a new segment in the data file is allocated and added to the index and the new data is added to the new segment. The index of segments includes the start and end times of the segment represented as a 64-bit integer. The index provides an efficient mechanism for identifying and locating the segments of interest when retrieving the data. The size of each segment is specified in the API so the segment size can be optimized based on the data flow characteristics of each measurement. A data segment consists of rows of data where each row is associated with a single timestamp. A row in a segment can be a scalar value or a multidimensional array. Segment indexes are allocated in fixed size tables. When an index becomes full, an additional index is allocated in the data file with pointers back to the previous index.

Two different types of segment records have been implemented to handle two distinct data flow models expected to occur in long pulse experiments. On type of segmented record which was called "normal" segmented records for lack of a better name, is likely to be used when storing data from transient recorders which buffer up samples occurring at multiple time stamps. When storing data of this type, the application must provide an MDSplus dimension description to be stored with the segment. The other segment record type, called "timestamped" segmented records, is used when storing measurements which more typically arrive one timestamp at a time. With this type of segment, when a new segment is allocated additional space is allocated to accommodate one timestamp per row. Data in a timestamped segmented record are added one row at a time.

Time is represented in the segmented records by 64-bit unsigned integers. The only stipulation with time values is that the times associated with rows of a segmented record must be increasing in value. MDSplus provides some built in functions for obtaining 64-bit time stamps using the OpenVMS[10] time representation with is the number of 100 nanosecond ticks since a base time (November 17, 1858). However, any time stamp representation can be used depending on the needs of the application. In some cases it may be convenient for the time stamps to represent the number of clock ticks since a base time such as 10 seconds before beginning of pulse.

Before retrieving data from segment records, the user can specify a "time context" which can include one or more of the following parameters; the start time, the end time and the delta time. Once a time context has been specified, future retrievals from segmented records will return a decimated signal created by selecting the segments lying within the start and end time boundaries, resampling these segments based on the delta time and then combining the reduced segments. MDSplus provides default functions for resampling the segments and assembling them into the resulting signal. However, these functions can be overridden on a per node basis by specifying the names of user provided expression evaluator functions as extended node attributes described below. This capability makes it possible to accommodate any site or node specific representation of time with the only restriction being that the time stamps fit into a 64-bit field.

6. Extended Node Attributes

During the design meeting it was decided that it may be convenient to add some additional MDSplus functionality while making the major modifications of the internal file access routines required to support the segmented records. One such enhancement was to add the ability to define arbitrary attributes associated with nodes in the tree. Extended node attributes are simply data items associated with an MDSplus tree node which consist of a name and a data value. The data value can be any supported MDSplus data structure. Applications can utilize these extended attributes to alter the way they treat each node. For example, extended node attributes were used in the implementation of the segment records retrieval. Users can

alter the resampling behavior of MDSplus by specify the names of MDSplus expression evaluator functions to use for resampling the nodes segmented records instead of using the default resampling algorithms.

7. Data Versions

Similar to the addition of extend node attributes, the authors took the opportunity afforded by the major alteration of the data file handling routines to add the ability to turn on data versions in an MDSplus tree. With data versions enabled on an MDSplus tree, old data is retained when it is overwritten by new data. To access older versions of data an application can set a "view date" which will cause MDSplus to simulate the behavior that the application would have experienced if run at the "view date" date and time. Retrieval of all data from nodes will return the same values that they would have returned if performed on the "view date".

8. New Expression Evaluator Functions

New MDSplus expression evaluator functions and TCL commands have been added to support these new MDSplus features. This enables existing tools to utilize these features without code modification of the tools themselves. Table 1 lists some of these new functions.

Table 1

Segmented Record Support Functions	
num=GetNumSegments(node)	Return number of data segments
limits = GetSegmentLimits(*node, segment-idx*)	Return start and end limits for a segment
signal = GetSegment(*node, segment-idx*)	Return segment as a signal
status = SetTimeContext(*start, end, inc*)	Set time context for retrieval of nodes containing segmented records
status = PutRow(*node, rows-per-segment, timestamp, data*)	Store timestamped data row
status = BeginSegment(*node, start, end, dim, initial-data [,segment-idx]*)	Allocate new segment
status =PutSegment(*node, row-idx, data*)	Store data in segment
Extended Attribute Support Functions	
Tcl('show attribute *node-name'*)	List names of defined extended attributes
Tcl('show attribute *node-name*/name=*attr-name')*	Show value of attribute
Tcl('set attribute/name=attr-name node-name value')	Set value of extended attribute
status = SetExtendedAttribute(*node, attr-name, value*)	Set value of extended attribute
value = GetExtendedAttribute(*node, attr-name*)	Get value of extended attribute
Data Versions Support	
Tcl('dir/full *node-name'*)	List information of data stored in node including all version records
Tcl('set view *view-datetime'*)	Establish viewing timeframe
Tcl('set versions [/[no]model] [/[no]shot]')	Turn on or off versioning in model or shot

9. Simple Access to Segmented Records

The segmented record extensions have been designed to be compatible with regular MDSplus data records. One only needs to set the time context prior to referencing segmented record nodes and then everything works identically to normal MDSplus expressions. The time context defines how segmented records will be sub-sampled.

.

10. jScope Enhancements

The jScope application in MDSplus used for plotting signals has been enhanced to efficiently pan and subset segmented records. The application retrieves only the segments necessary to draw the plot. As the user pans through the time axis, jScope retrieves more segment data as needed. Likewise when jScope plots an extended range of data, it automatically sub-samples the segments at the highest resolution that can be displayed on the device. At no time does jScope need to retain the entire signal at full resolution. The application was also enhanced to show date and time labels and day boundaries on the X axis.

11. C-Mod Trial Usage

The segmented records enhancements have been utilized on the Alcator C-Mod[11] experiment at MIT for several months. Trending data from the control system are now stored using this feature. Each minute approximately 160 measurements are appended to timestamped segments of 160 MDSplus trees nodes, or one sample per timestamp. In addition to these trending measurements, data from an RGA mass spectrometer are stored in a segmented segment. In this

case the data is stored in a single node with 800 values per timestamp. This RGA data is also acquired approximately once per minute.

In addition, segmented records are now being used to store video data from a high speed camera. During each pulse of the experiment, 30,000 frames of video (64x64x16-bit) are stored in a node in the tree using segmented records.

Several visualization tools have been configured for viewing this data.

12. Get It Now

The enhancements described in this paper are available now in the MDSplus cvs repository. A new build of MDSplus generated from sources found in the repository will contain all of the enhancements. It is anticipated that new installation kits will be available with these features included by the end of summer 2007. These installation kits will appear on the download pages of the MDSplus web site. These enhancements will not be available on the OpenVMS platform as they use file record access which is incompatible with the OpenVMS record management system.

13. Future Possibilities

It is anticipated that additional desired features will be identified as users obtain experience with the use of these enhancements. Once we learn more about how users interact with high volume streaming data it will be possible to add new features to various applications such as the standard visualization tools provided in MDSplus to utilize these features to their full potential.

A few potential improvements already identified include segment manipulation and repair tools, compression support with segmented records, backup and restore utilities and version management utilities.

14. Summary

Several enhancements have been added to the MDSplus data handling capabilities including segmented records for handling quasi-continuous data streams, extended node attributes and data versioning. Initial testing of these enhancements in production use at the Alcator C-Mod experiment has returned very positive results. The addition of data handling of quasi-continuous data flows in the MDSplus environment will provide a valuable learning experience and perhaps provide insight on how to handle this type of data while still providing the powerful features of MDSplus including simple programming interface, remote data access, powerful expression evaluator, hierarchical data organization, visualization tools, and a common access to many fusion research experiments and modeling code results. It is hoped that these enhancements will provide solutions for the handling of large and continuous data flows and will be beneficial to those site already using or planning to use MDSplus on experiments with long pulse.

References

[1] T.W. Fredian, J.A. Stillerman, M. Greenwald, “Data acquisition system for Alcator C-Mod”, *Rev. Sci. Inst.*, January 1997, **68**(1), pp 935-938.

[2] T. W. Fredian and J. A. Stillerman, "MDSplus: Current Developments and Future Directions", *Fus. Eng. Des.*, **60**, (2002), 229

[3] http://www.mdsplus.org/

[4] Yantai Shu, Jiarong Luo, Fenng Zhao, Huazhong Wang, and Di Wang,”Performance Evaluation of the HT-7U Data System”,*IEEE Transactions on Nuclear Science*, April 2002, VOL 49,No.2, pp 428-431

[5] M.Kwon, I. S. Choi, J. W. Choi, J. S. Hong, M. C. Keum, K. H. Kim, M. G. Kim, M. K. Park, S. H. Seo, S. Baek, H. G. Jhang, J. Y. Kim and KSTAR Team,”The control system of KSTAR”, *Fus. Eng. Des.*,June 2004, **71**, pp 17-21

[6] http://www.iter.org/

[7]Nakanishi Hideya, Kojima Mamoru, Ohsuna Masaki, Komada Seiji, Nonomura Miki, Yoshida Masanobu, Imazu Setsuo and Sudo Shigeru,”Steady-state data acquisition method for LHD diagnostics”, *Fus. Eng. Des.*,September 2003, 66-68, pp 827-832

[8] P. Heimann, K. Engelhardt, S. Heinzel, C. Hennig, H. Kroiss, G. Kühner, J. Maier, J. Reetz, M. Zilker, The data acquisition system of W7X, Fut. Eng. Des. (2002) Vol. 60.

[9] http://hdf.ncsa.uiuc.edu/HDF5/

[10] http://h71000.www7.hp.com/

[11] http://www.psfc.mit.edu/research/alcator/