THE GROUND TEST ACCELERATOR CONTROL SYSTEM DATABASE: CONFIGURATION, RUN-TIME OPERATION, AND ACCESS*

L. R. Dalesio
MS H820, Los Alamos National Laboratory, Los Alamos, NM 87545

Abstract

A database is used to implement the interface between the control system and the accelerator and to provide flexibility in configuring the I/O. This flexibility is necessary to allow the control system to keep pace with the changing requirements that are inherent in an experimental environment. This is not achieved without cost. Problems often associated with using databases are painful data entry, poor performance, and embedded knowledge of the database structure in code throughout the control system. This report describes how the database configuration, access, conversion, and execution in the Ground Test Accelerator (GTA) Control System overcome these problems.

Database Configuration

In the GTA Control System, the database configuration task provides an easy-to-learn, multiuse tool. This task uses a menu-oriented operator interface. Menus are used extensively to reduce the amount of knowledge needed to configure the database. Figure 1 shows the screen used to define the process points connected to an analog input card in the I/O controller. At the top level, the user can create, report, modify, or delete one of the portions of the distributed database. A directory of all channels throughout the distributed control system by I/O processor, record type, and alphabetized signal names is available at the top level. An ASCII database save file can be created with the report command, edited with any text editor, and reentered through the restore command. At the lower level, individual database records are created, reported, modified, and deleted. Records can be created, reported, and modified individually or from a signal list form. For a single channel, a list of the individual fields is presented to the application engineer. Range checking is provided for each entry. A future modification will allow the priviledged user to modify the run-time database and save run-time modifications to the disk-based system. The database configuration task produces the process variable directory, which is a listing of all channels and their target processor, record type, and relative address. This process variable directory is kept as a hash table with linked collision lists. Access time into this table is critical as it is used during operation for access to the database records.

Database Access

Access to the distributed databases is made using the record name along with the field name. The remote processors convert the record name to a record type and memory address with the information in the process variable directory. The record name look-up is done at run time so that it can be converted directly to a memory address. This conversion time is minimized through the use of a hash table directory. The hash table is maintained through the database configuration task. Hash table access is fast as long as the number of collisions is low. The remote processors convert the field name to a field type, field size, and field location with the information in a field convert file. The field convert file is built from an ASCII record description. It is in alphabetical order and searched with a binary search routine. Both the process variable directory and field convert file are kept in memory to ensure the fastest conversion. Average conversions are accomplished in approximately 550 μs. Any network entity resolving a database field specification to a database access address need only call this conversion routine once. The structure returned from this routine is used for all subsequent requests for that field's data. Figure 2 shows how the database access routines provide the control network with an interface into the database. To further eliminate the need for database knowledge from the requester, database access calls include the format in which the data are desired. If the data are desired as they exists in the database, the time to access them is merely the time to access the network along with the time to copy memory to memory. For data that are desired in a different format, conversions and field manipulations are performed.

Database fields are one of the following: floating-point, 16-bit integer, enumerated item, string, and no access. Enumeration fields contain an index into a set of string values. The menu is either derived from within the database record (e.g., the value field of a binary input has a 'zero' string and a 'one' string in the database) or from a set of system menus

SYSTEM: KLYRF — Base Address 0xff3100
ANALOG INPUT CARD TYPE: VMIVME-3100 SE — I/O Parameters
CARD NUMBER: 0

Name	LINR	10	0	Volts
CATHODE_AMP	LINEAR	45.00	0.00	Amps
CATHODE_VLT	LINEAR	120.00	0.00	kV
MOD_ANODE_VLT	LINEAR	75.00	0.00	kV
COLLECT_AMP	LINEAR	45.00	0.00	Amps
MOD_ANODE_AMP	LINEAR	20.00	0.00	mAmps
HV_BIAS_KV	LINEAR	20.00	0.00	kV
ANODE_BIAS_KV	LINEAR	2.000	0.000	kV
BODY_AMP	LINEAR	250.00	0.00	mAmps
FIL_VLT	LINEAR	40.00	0.00	VAC
FIL_AMP	LINEAR	20.00	0.00	Amps
CAV_TEMP	K_DEGC	1000	0	Degc
BODY_TEMP	K_DEGC	1000	0	Degc
NOT_ASSIGNED				
NOT_ASSIGNED				
NOT_ASSIGNED				
NOT_ASSIGNED				

Modify Copy Delete NextCard PreviousCard ChangeScreen

Fig. 1. A sample database configuration page.

*Work supported and funded under the Department of Defense, US Army Strategic Defense Command, under the auspices of the Department of Energy.

(e.g., the card type in an analog input field). These fields can be requested as a single entity or an array of floating-point values, 16-bit integers, enumerated items, or strings. Enumerated items that are requested as strings, return the menu item to which their value refers. In addition to the conversion of the field, data can be requested that reflect the alarm condition of the record, parameters from the record that are used to graphically represent the value, and parameters that are used to modify the value (e.g., for an analog output value, the control parameters include the database control limits: an enumeration field will return the menu from which the new value is selected). The time cost of the conversion and collection of additional parameters is used to purchase the complete separation of the operator interface task, the sequencer task, and any other data acquisition or control task from the structure of the database. All that these tasks need to know is the basic data types contained in the database. This cost is kept at a minimum for adding the parameters for alarm status, graphic display, and control by having the database structure compiled into the code that accesses the database. The database record C-structures are created along with the field conversion file.

Run-time Operation

The database library routines use the C-structures to provide the most efficient processing of the process I/O and closed loop control. The database library routines' connection to the database is shown in Fig. 2. The beginning address of a record is placed in a C-pointer to a structure. All subsequent manipulation of the record is accomplished in the compiler. Careful use of the C-register variables results in the most efficient processing. The C-structure was used in place of assembly language offsets to provide an easier path for making changes to the structure of a database record. When a record type's structure is changed, a new C-structure is produced automatically, all database routines are recompiled and the database is reported and reentered through the new structure using a database configuration utility.

New technology employed in the GTA Control System has also enhanced the overall performance of the I/O interface. The GTA I/O controller is based on the 68020 and contains a 68881 floating-point processor. The VME I/O is memory mapped into the VME memory space for quick access. The software contribution to the performance improvement is the VxWorks operating system, which services interrupts in 4 μs and performs subroutine calls in 10 μs.

The changes in technology, along with careful attention to timing costs, have resulted in an implementation that is both flexible and able to handle change without undue pain and performance degradation. Adding channels and changes to parameters within channels is available during operation. Setpoints, outputs, computer/operator control, alarm limits, scan rates, archiving limits, and conversions can be changed during operation. I/O channels and continuous control loops, containing alarm checking and archiving, can be configured or modified without programming. A database channel's structure can be changed with little pain and no programming. Preliminary time tests using an I/O controller with a 68020 and

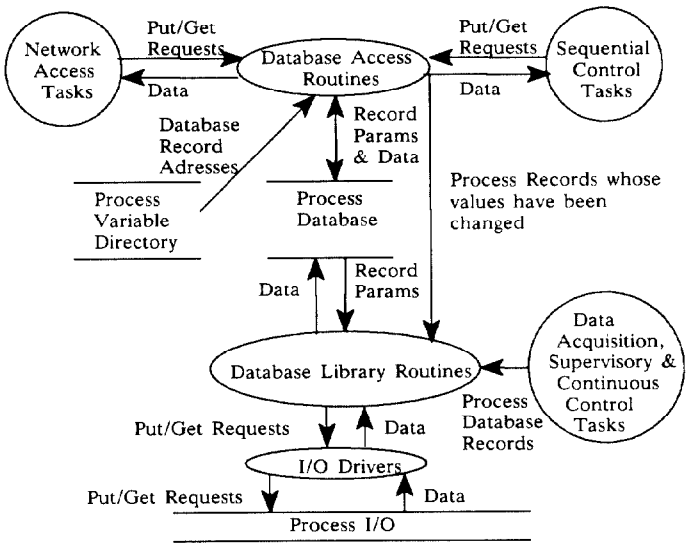

Fig. 2. A data flow diagram of the I/O controller.

the 68881 floating-point processor during a 1-second interval are as follows:

4413	analog inputs read, converted, checked for alarms
13333	binary inputs read, converted, checked for alarms
12500	database accesses with no conversion
4139	database accesses to a field with control parameters
1591	analog input record and field name conversions
1806	binary input record and field name conversions

During these tests, the code was running at the highest priority. The database access timing includes the time to execute in the I/O controller and does not include the network overhead. Further benchmarks need to be made on data monitors, closed-loop control, and sequential control CPU usage. This does show that 400 analog inputs and 1300 binary inputs, scanned at a rate of once a second, use only 20% of the CPU.

Conclusion

Configuration of the database is accomplished using an interactive tool that requires minimal training. The configuration tool provides a variety of modification and reporting facilities to aid the application engineers. Database access routines provide a run-time binding to the database and remove the need for code to know the structure of the database in order to access the data. They provide this interface with a minimum amount of overhead. Database scan routines are compiled with knowledge of the database structure to efficiently process the I/O. Attention to the potential problems of using a database along with increased processor bandwidth have resulted in flexibility with excellent performance.

Future enhancements to this package will include a graphic configuration tool that takes advantage of a point and click selection mechanism with a graphic representation of the I/O connections. This will provide the application engineer a more intuitive way to configure the instrumentation. It will further reduce the training time and the configuration time.