A GENERAL APPLICATION
REAL TIME PROCESSING AND DISPLAY SYSTEM

G. A. Schumacher
Telemetry Group
Sangamo Weston, Inc.-Data Systems Division
(formerly EMR Telemetry)
P.O. Box 3041
Sarasota, FL 33578

ABSTRACT

A general purpose system for the processing and display of telemetry data is described. The system uses a DEC PDP-11 mini-computer as the system processor. Systems of this type implemented in the past generally had limited data processing capabilities due to high data rates and limited CPU power.

The Real Time Data Monitor System (RTDMS) is designed to maximize the data processing capability by teaming the CPU with a pre-processor. A total hardware/software system was designed which is structured around certain commonly required classes of data. The system minimizes the processor overhead associated with each class thereby freeing time for useful processing. A complete software system was implemented which takes full advantage of the RSX-11M multi-task operating system.

Because of the substantial data base required to configure the system, particular attention was given to the operator interface. A set of programs were implemented which construct all data base files required using interactive menus. Operator input is minimized as well as the training time required to set-up the system.

INTRODUCTION

For a number of years SWI-DSD has integrated telemetry front-end equipment with minicomputers and special software in response to the needs of various users of telemetry data. These telemetry/computer data systems fall into two broad categories of applications.
• Formatting of real time data or data recorded on instrumentation tape to digital tape or disc for non-real time processing.

• Real time display for quick look, safety, or other monitoring purposes.

When real time display is needed, the usual minimum processing requirement is to convert the data to engineering unit values. The desire for additional processing beyond this can be virtually boundless. Achieving even the minimum requirement has dictated very efficient, very complex software written in low level language and highly tuned to a particular application. This factor has tended to make the software on these systems a high cost, high risk item. The resulting systems have been limited in capability and have required fairly intimate system knowledge to be properly utilized.

Real time systems have previously been approached with the same hardware and techniques used for formatting applications. One of two system configurations is used.

The first shown in Figure 1, uses conventional frame data as output from a frame synchronizer, and loads it via a DMA interface into a dual cyclic memory buffer from where it is processed. Time is merged with the data on either a frame or a buffer basis.

The second approach shown in Figure 2, is similar except that the data comes from a hardware data compressor unit. This unit accepts data from up to six frame synchronizers, merges the data with time, compresses the data, and adds appropriate identification and status tagging to each word prior to output. The data compressor unit can perform one or more of twelve algorithms on each parameter and the compression technique can be individually programmed for each parameter.

Neither approach is ideally suited to real time application.

Two constraints exist in the first configuration:

1) Because all data processing and manipulation is done in the CPU, the available processing power is quickly consumed.

2) Since the processing is event driven by frame and subframe interrupts, operating systems which provide very fast interrupt response are needed. Relatively simple single-job or foreground/background systems are preferred because of the overhead associated with multi-task systems.
The second system possesses some conflicting characteristics:

- While the data compressor is designed primarily to reduce data quantity, it performs a number of pre-processing algorithms useful for display purposes. These include cumulative summation, minimum and maximum selection, Nth sequential output, limit checks and output on bit change.

- The overhead associated with retrieving a particular parameter or algorithm result is high because of the random sequence in which data is located in the memory buffer. This means that the complete buffer must always be searched to avoid missing data.

- Processing is driven by a block-end interrupt which is generated as each buffer is filled. Since it is a dual cyclic buffer, the time available for the CPU to process buffer A depends on fill time for buffer B.

- The compression rate on a number of the algorithms depends on parameter activity. When parameter activity increases, the compressor output may increase rapidly. Thus the memory buffers fill faster and time available to process the buffers is reduced. Unfortunately, periods of peak parameter activity are frequently the times when the displayed data is most wanted, but the CPU has less time available for processing.

Because of these conflicts it was determined that a system design specifically addressing the real time problem was needed.

**DESIGN GOALS AND TECHNIQUE**

Previous experience suggested three major design goals for the system, and some techniques for achieving them.

1) A mini-computer based system with sufficient processor time available to perform substantial processing of displayed data. This was to be achieved through:
   - Maximum use of hardware pre-processing
   - Optimized methods of data entry into memory for reduced processor overhead
   - Reduced dependence on event interrupt response time.
2) A user oriented, understandable system. This was to be achieved through:

- Interactive, menu structured setup programs
- A software system written in FORTRAN
- Applications programs which could be written by the user in FORTRAN and readily installed and assigned.

3) A general purpose system closely aligned to a number of applications.

At the outset, four unique classes of data were identified.

- **Display data** - Data to refresh CRT displays on a periodic basis
- **Alarm Data** - Algorithm results from the preprocessor which identify out-limits conditions, discrete events or combinations of discrete events. This data must be precisely time tagged as it occurs.
- **Recorded Data** - Data which must be formatted to magnetic tape or disc in real time.
- **Graphic Data** - Data for output to graphic recording devices such as strip-chart recorders.

The total hardware/software system was partitioned into four parallel data paths corresponding to these data classes.

The compressor design was modified to provide four output ports, one for each data path. The data is distributed to these ports and their corresponding processor memory buffers in accordance with the type of preprocessing performed and the type of output device which will display the data. This approach allows the input buffer strategy to be matched to the processing requirements of each type of data. The objective is to minimize the software overhead associated with buffer search, data identification, and time correlation and to minimize the average data transfer rate on the PDP-11 UNIBUS. The modification provides the ability to designate any measurement to any one or more output ports and to perform different algorithms for each port. This provides the ability to perform multiple types of preprocessing, processing, and display on each measurement in parallel.
SYSTEM CAPABILITY

A typical RTDMS configuration is shown in Figure 3. The system supports data display on a mix of devices - CRT terminals, printers, and graphic recorders via DAC’s. Simultaneous formatting of data to disc or tape mass storage is also supported.

CRT Displays

The CRT’s can display measured or derived (calculated) parameter values processed by the CPU to engineering units or other desired numerical form. The source data used for CPU processing may represent the average of the measured parameter over the refresh period, the minimum and/or maximum during the refresh period, or the current value at the end of the refresh period.

Discrete event or status data may also be displayed with the discrete state represented by a descriptive word or other alphanumeric identifier selected by the user. High or low alarm limit crossing may be flagged on the CRT by reverse video or flashing.

The processed data is displayed in accordance with predefined page formats. The display formats may consist of data values, parameter names, engineering unit names, header data, time, and data arranged per a standard format or virtually any user defined format. Up to 99 pages of display may be defined and can be selected at will during real time.

Printer

A hard-copy terminal or line-printer may be used to output limit alarms, status data, and event messages. For each alarm limit crossing or discrete parameter state change a one line message is output to the terminal or printer. The message contains:

- Time of occurrence - Hours, minutes, seconds, and milliseconds.
- Parameter I.D. - Input link number and parameter number as defined in the Measurement Data File (MDF).
- Parameter Name - Alphanumeric name as defined in the MDF.
- Alarm State or Discrete State - State identifier indicating the alarm status of the parameter following the alarm limit crossing or the current state of the parameter following a status change.
A memory buffer will queue up to 32 messages during active periods. If the message buffer overflows an asterisk will print, indicating buffer overflow.

**Graphic Recorders**

Graphic chart recorders driven from DAC’s may be used to display data output from the computer. Each output parameter may be scaled in the processor to an engineering units range prior to output to the DAC’s. During setup of the graphic output, scaling or no scaling may be selected for the parameter. If scaling is selected, a zero and full scale display range must be specified.

**Mass Storage Formatting**

The RTDMS supports the formatting of merged, tagged data and time from the preprocessor to a digital magnetic tape or disc simultaneously with data output to display devices. This formatting capability is compatible with any magnetic tape unit or disc which is supported by the operating system. Data may be throughput or compressed by one of two compression algorithms on an individual parameter basis.

**Data Processing**

One of the major goals of the RTDMS design was to free CPU time for the performance of useful data processing so that more real time intelligence could be displayed. The majority of this processing is performed on the CRT display data and this data path has the most capability. The Graphic and Alarm data paths have provision for more limited real time processing and none is provided for recorded data. User programs written in FORTRAN IV PLUS or assembly language may be installed for real time data processing.

The Display data path uses two types of processes - Measured Parameter Processes and Derived Parameter Processes.

A Measured Parameter Process (MPP) is defined as a process which operates on a single input parameter and produces a corresponding single output parameter. The output of a MPP is considered to be a measured parameter and retains the same parameter identification number as the input parameter. Multiple MPP’s may be assigned to a single parameter and performed in sequence. An MPP may have any number of arguments which are assigned values during setup.

A Derived Parameter Process (DPP) is defined as a process which uses multiple input parameters to produce a new output parameter. The output parameter is called a derived parameter and a new parameter identification number, different from any of the DPP input
parameters, is assigned to it. The inputs to a DPP may be any combination of measured or derived parameters. The derived output parameter from a DPP may also be operated on sequentially by an MPP just as if it were a measured parameter. A DPP or MPP may have any number of arguments associated with it. The specification or update of process argument values is a setup function.

Because some derived parameter processes could be quite complex, there is an inherent danger that a process could hog the available CPU time. To avoid this problem, priority levels have been provided for parameter processing. Level 1 is the highest priority, and measured parameter processing as well as some derived parameter processing is done at this level. Other derived parameter processing is done at lower priority on Level 2 and Level 3 and is accomplished on a time available basis.

Level 1 processing is completed for each CRT display refresh before Level 2 processing is initiated. If a CRT refresh is required prior to completion of Level 2 processing, all Level 2 input data values will be saved. At the completion of Level 1 processing for the next CRT refresh, Level 2 will be restarted at the point of interruption.

Likewise, Level 3 processing will not be initiated unless all Level 2 processing is completed. This approach insures that the most basic, highest priority, data will be available for display each CRT refresh period, while allowing full utilization of all available CPU time for high level processing. Although the higher level results may not be updated each refresh period, they will be updated as frequently as possible. Figure 4 illustrates some possible process sequences.

Graphic and Alarm data processing are similar except that they are processed on a single priority level only.

Setup Functions

The RTDMS requires a large quantity of setup information to be defined before operation. This effort is accomplished through a set of interactive setup programs which lead and prompt the user through the sequence of variables requiring specification. If possible, the user response to each menu prompt is a single character entry, otherwise the response is alphanumeric data.

Each setup program has a CREATE, MODIFY, DELETE, and GENERATE mode. The CREATE mode of each program creates an ASCII setup file containing data which defines a particular aspect of the system. In MODIFY mode, selected records from the file may be viewed on the CRT and updated. In DELETE mode, selected records or an entire file may
be deleted. In GENERATE mode, the ASCII files are reduced to binary files from which the system load modules are built.

Four types of setup files are used to define system set-up information: the Measurement Data File, the Alpha Format File, Data Select Files, and Process Select Files. The entire collection of files is required to fully define a system operating configuration.

The Measurement Data File (MDF) defines the word position of each parameter in the input multiplex and assigns a Link No./Parameter No. (LN/PN) to it. The LN/PN is the tag used throughout the software system to identify the parameter. It is possible to assign multiple LN/PN’s to a single word position. A parameter name, engineering units name, and data type are defined in the file for each LN/PN. The allowable data types are analog with no EU conversion, analog with a polynomial conversion, analog with table look-up conversion and discrete data. For polynomial EU converted data a table of up to 23 polynomial coefficients is entered. For analog table look-ups a table of up to 23 decimal count/EU value pairs is entered. Discrete parameters consisting of one to four adjacent bits positioned anywhere within an input data word are allowable. For discrete data, a table of decimal count versus alphanumeric identifiers is entered.

The Alpha Format File (AFF) contains a definition of each CRT display page and the parameters and header data displayed on it. An AFF may define up to 99 pages of display. Each page is defined in terms of line number, field size and location, and field content. Fields may contain parameter and unit names as defined in MDF, data values, time, data, or header data defined during AFF creation.

The Data Select Files define which parameters are distributed to each of the four output ports (data paths) of the pre-processor. Four Data Select files are created: Display Data Select (DDS), Alarm Data Select (ADS), Graphic Data Select (GDS) and Record Data Select (RDS). Each file contains the LN/PN of each parameter, the algorithms assigned and the algorithm arguments. Algorithm arguments such as aperture size or limit values are input as engineering unit values.

The Process Select Files define the processing to be performed on each parameter. Three Process Select Files are created: Display Process Select (DPS), Alarm Process Select (APS), and Graphic Process Select (GPS). The DPS file contains the LN/PN of each parameter to be processed at Level 1. For each of these, the MPP’s to be performed are identified in the sequence in which they are executed, along with the applicable argument values. The identity of the DPP’s to be performed at Level 1 are identified along with the output parameter identification, the input parameter identification, and the applicable argument values. The same information is identified for Level 2 and Level 3. The GPS and APS files define the processing performed on parameters assigned to the Graphic and
Alarm paths. They are similar to the DPS file except that only Level 1 processing is defined.

**SYSTEM OPERATION**

The system pre-processor accepts 16 bit parallel input data from up to six input sources and merges them along with syllabized time from a time code generator. The unit will accept two’s complement, sign magnitude and offset binary data, provided all input sources utilize a common word length. All data is normalized to offset binary prior to algorithm processing.

The flow of data through the RTDMS as discussed below is illustrated in Figure 5.

**Display Data**

Parameters selected for the display data path may be pre-processed by any combination of the interval dependent algorithms CSU, MMA, and NSE. The CSU algorithm performs a cumulative summation over N sample periods and outputs the result as two 16 bit words. The MMA algorithm outputs the maximum and minimum parameter values which occurred during N samples. The NSE algorithm outputs the value of the Nth sequential sample.

This is the primary data which is processed and output to the CRT displays by the CPU. The data is output from the preprocessor to the CPU memory buffers at the CRT update rate via a force command issued from the CPU. The CRT update rate is a specifiable system parameter and is under software control. The value of N for each algorithm is the number of samples occurring within one CRT refresh interval.

The pre-processor Display data output port is a 32-bit wide Externally Specified Address (ESA) port which outputs a 16 bit destination address tag in parallel with each 16 bit data word. The data is output to a Buffered Data Channel (BDC) which is the DMA interface to the processor. The BDC appends address bits 17 and 18 to construct the processor memory address.

When the CPU issues a force command, the pre-processor dumps the following nine word block of data through the Display Data port for each selected parameter:

- Current Value
- N, Number of Samples
- Cumulative Sum MSW
- Cumulative Sum LSW
- Maximum Value
• Minimum Value
• Minor Time Word
• Major Time Word #1
• Major Time Word #2

A nine word memory buffer is allocated for each Display path parameter. This buffer is overwritten with current data once each CRT refresh period.

Display data processing takes place between CRT refreshes. The first processing step picks up from the input buffers all parameters required by currently selected CRT display pages and converts to engineering units in accordance with the conversion method defined in the Measurement Data File. The converted data is stored in allocated locations in the real-time data base. Levels 1, 2, and 3 process strings defined in the Display Process Select file pick-up input data from the data base and store results there for use by subsequent processing levels or for display.

The CRT display task generates the static display data using the page format and header data defined in AFF and the measurement and engineering unit names defined in MDF. Actual data values are picked up from the data base each CRT refresh and displayed as defined in AFF for the selected page.

Event and limit check data status is determined from event flags set by Alarm path software.

**Alarm Data**

Parameters selected for the Alarm data path are pre-processed by one of the four algorithms THX, BCH, BMA, and NBM. Analog parameters are limit checked by the Threshold Crossing (THX) algorithm which compares all input data samples to the upper and lower limits specified in the Alarm Data Select File. The algorithm outputs an in-limit or out-of-limit status word each time the parameter crosses a limit threshold. Data words containing packed discrete parameters may be scanned and output on any single bit change using the Bit Change (BCH) algorithm or output on the occurrence or non-occurrence of any selected bit pattern using the Bit Match (BMA) or No Bit Match (NBM) algorithms. Only a single algorithm may be performed on any parameter in the Alarm data path.

Each time a parameter passes an algorithm, a five word block is output consisting of:

• Data ID Tag
• Current Value or Limit Status
• Minor Time
The Alarm data port is a 16 bit wide port which outputs data through a BDC to a cyclic buffer. The buffer is checked for new data each CRT refresh or at buffer overflow. Only data values entering the buffer since it was previously checked are read at each interrupt.

Any parameter blocks output by the pre-processor will be picked-up by the alarm processing. Event flags are set for use by the Display path for every parameter selected for the Alarm path. Alarm messages are formatted and output to the terminal and printer only if so specified in the Alarm Data Select File.

**Graphic Data**

Graphic data is a selected set of measurements which is throughput along with time to the processor. The data is used for displays such as chart recorders where continuous data samples are required.

The pre-processor graphic data port is a 32 bit wide ESA port which outputs data and address tags through a BDC in a manner identical to the Display data port.

The data consists of parameter values and time words, all throughput at time of occurrence. The data and time words are loaded in a current value buffer which always contains the most current value of the selected parameters as well as the most current time. Data and time can be sampled from this buffer with a worst case interparameter correlation error of one frame time.

The Graphic data path DAC process picks up the data values from the current value buffer and scales the data based on the MDF EU conversion definition and the EU output range specified for the channel in the Graphic Process Select File. The DAC process appends two control words required by the DAC unit used and outputs the packet through a BDC to the DAC unit.

**Recorded Data**

Any input parameter, regardless of its use for display, may be designated for mass storage. During selection one of three compression algorithms may be specified for each parameter - ZFN, DSL, or THR. ZFN stands for Zero-order, Fixed aperture, Non-redundant point transmitted. The ZFN algorithm allows an aperture size to be selected for a parameter. The algorithm will only output the parameter if it differs from the last value output by more than the aperture value. The Delta Slope (DSL) algorithm is the same as ZFN except that
only the points which also indicate a slope change are output. The THR algorithm is simple throughput.

Minor and major times words are merged with the recorded data. Minor time with a resolution of one millisecond is compressed by outputting it only if one or more data words have been passed since the last minor time was output. The two major time words are always output once per second upon occurrence. All data and time words passed are preceded by a data identification tag. The Record data port is a 16-bit wide port which outputs data through a BDC DMA interface to conventional dual-cyclic buffers allocated in processor memory.

The formatting operation is driven by block-end interrupts generated by the BDC when each buffer is filled. The full buffer is picked up and written to mass storage using RSX-11M I/O services while the other buffer is being filled. The buffer size allocated is large enough that interrupt frequency and service time is not a problem. No parameter processing is performed in the Record data path.

If desired, a header containing parameter descriptive data for all parameters selected for Record may be written to mass storage prior to the start of real time acquisition. The header records contain most of the MDF data including position in the multiplex, name, units name, EU conversion tables, link number, and parameter number.

**Initialization and Control**

Initialization and control of the RTDMS is through a menu oriented control task which provides the operator interface to the system. Six functions can be performed:

1) **Build Real Time Tables** - The setup files applicable to the upcoming run are identified. All processor and pre-processor resident data tables and files are built from the binary set-up files.

2) **Parameter Options** - Certain system variables such as CRT refresh rate, cyclic buffer sizes, etc. are specified.

3) **Load Real Time Tables** - The processor and pre-processor resident tables are loaded.

4) **Write Data Base Record** - If the parameter data base is desired on mass storage it may be written.
5) Start Real Time - The data paths applicable to the current data run are selected and real time operation is started.

6) Stop Real Time - Real time operation is stopped.

CONCLUSIONS

The RTDMS is a flexible system with capability to meet a number of real time requirements. Because of its flexibility some consideration should be given to its limitations. As data rates, number of parameters processed, and display requirements are all increased the system throughput will ultimately become bound at one of three places:

1) Pre-processor - The pre-processor throughput is most impacted by very large numbers of CRT display requirements. The algorithms used for the Display path have the longest execution times.

2) Processor UNIBUS - The UNIBUS throughput is impacted most by high rate input data which is formatted to mass storage. Use of a PDP-11 with a MASSBUS eases this problem. Large numbers of Graphic parameters with high data rates also impact UNIBUS throughput.

3) Processor - The processor is impacted most by the number and complexity of the Display path processes and by the Graphic path data rates, parameter quantities, and scaling requirements. The speed of the processor and hardware floating point unit are critical.

The RTDMS has a great deal of capability which can be applied to real time problems. Its resources, however, have limitations which must be considered in its application.
FIGURE 1. CONVENTIONAL DATA SYSTEM

FIGURE 2. COMPRESSED DATA SYSTEM
FIGURE 3  TYPICAL RTDMS CONFIGURATION
FIGURE 4. DISPLAY DATA PROCESSING DATA FLOW EXAMPLE

MPP - MEASURED PARAMETER PROCESS
DPP - DERIVED PARAMETER PROCESS
PN - PARAMETER ID N
FIGURE 5. RTDMS REAL TIME DATA FLOW