DEVELOPMENT AND USE OF A REAL-TIME ACQUISITION, MONITORING, AND ANALYSIS SYSTEM FOR A PROCESS CONTROL ENVIRONMENT

by

Stephen M. Price, P.E.
Project Engineer
Engineering Dynamics Incorporated

ABSTRACT

In an industrial continuous processing environment, failures of system components can lead to unsafe operation, environmental damage, and significant loss of revenue. When specific components of the system are critically designed, instrumentation can be installed (e.g., pressure transducers, accelerometers, strain gages, etc.) to provide data to assess current conditions. From these data, operational parameters or even system design can be altered to maintain safe and reliable operation. However, the amount of data generated can quickly overwhelm operations/engineering personnel. Therefore, a system to continuously acquire and evaluate the data, presenting only the most useful information is vital. Such a system is discussed in this paper.

The rapid development (less than 2 months) and successful use (currently over 1 year) of a low-cost PC based system providing on-line continuous monitoring and evaluation capabilities for 35–40 channels will be discussed. The system was developed using off-the-shelf hardware, (including the *National Instruments* AT-MIO-16F5 A/D board) with custom software, written with the the *National Instruments* LabWindows product. Extensions to the *National Instruments* "double buffered" data acquisition technique will be detailed. Use of an off-line post processor (also developed with LabWindows) that can provide further diagnostic evaluation of the data acquired will also be illustrated. Finally, the specific safety, performance, and engineering design benefits derived from the data will be discussed.

1. Introduction

Structural failures of vessel components in an oil gathering system were detected by facility operations personnel. Failures of piping supports were also noticed. Although cracks were detected prior to catastrophic failure, the failures resulted in significant loss of revenue and could have posed safety and environmental hazards. After repairs were made and normal operation was resumed, data were acquired to quantify the cause(s) of the failures. The data indicated that abnormally large forces were being generated in the piping due to a two-phase flow phenomenon sometimes called "slug-flow". These forces resulted in fatigue failures at vessel nozzle penetrations. Additionally, when the forces acted on unrestrained portions of the piping, excessive pipe movement and high stress occurred.

The gathering system conveys oil and gas from individual wells to central processing facilities, where the oil is separated from the gas. Under normal conditions, the oil and gas flow in a well-behaved manner with oil in the lower portion of the pipe (at velocities of 3–6 ft/sec) and gas in the upper portion of the pipe (at velocities of 20–60 ft/sec). However, elevation changes and flow disruptions can cause a "plug" of liquid to form. Additionally, as gas velocity increases, "waves" can begin to form in the pipe, also potentially generating a liquid plug in the pipe. The liquid-plugged regions, which can grow to hundreds or even a thousand feet in length, are accelerated by the higher gas velocities, resulting in a "slug" of liquid (otherwise known as slug-flow). At locations where flow changes direction (i.e., elbows, nozzles, tees, etc.), slug-flow may produce high amplitude reaction forces, resulting in fatigue failures and high dynamic pipe stresses.

To develop both long-term and short-term solutions to the problem, several key issues had to be addressed, including:

- the amplitude of the slug forces,
- the frequency at which slug events occurred,
- slugging trends (to determine if force amplitudes were increasing),
- strain amplitudes in critical piping components, and
- what process variables could be altered (if any) to immediately reduce slug forces without compromising process throughput.

Additionally, it was sought to develop statistical databases over an extended time period. These data could help provide an estimation of the amount of damage already experienced by pipe components, as well as estimates of forces and damage that could be experienced in the future.

Because of the complexity of the problem and a short time horizon for problem resolution, an approach was developed that integrated analytical models with data acquisition to provide the necessary information to design appropriate system modifications. It became apparent that the amount of data that would be required would rapidly outpace the collection/interpretation methods used in the initial problem-identification phase of the project.

Therefore, it was undertaken to develop a data acquisition system that would provide continuous acquisition and monitoring of a large number of data channels. Capabilities for real-time analysis of the data would be required to limit the information to a manageable amount. The system would need to be able to be used by personnel with a minimal amount of training. Furthermore, the system had to be fully installed and operational in less than three months.

The data collection and monitoring system (or as it became to be known, the Slug Monitor) was to be installed at three facilities, and required four primary components.

Instrumentation: including welded strain gages and dynamic pressure transducers

Signal Conditioning: to provide bridge excitation, low-noise amplification, etc., with the ability to maintain the transducer in a Class I, Div. I (Intrinsically Safe) area

Data Acquisition: for 30–40 channels of data, acquired at approximately 100 Hz per channel

Data Analysis, Manipulation, and Storage: possessing the necessary CPU performance and storage requirements to meet system demands

A detailed discussion of the instrumentation and signal conditioning techniques that were used is beyond the scope of this text. Of particular interest, is the development and implementation of the data acquisition and monitoring system.

2. Instrumentation and Signal Conditioning

Strain gages were installed at key locations in the inlet system. These instruments measure static and dynamic strain at the location to which they are attached. Pressure transducers that could measure both static and dynamic

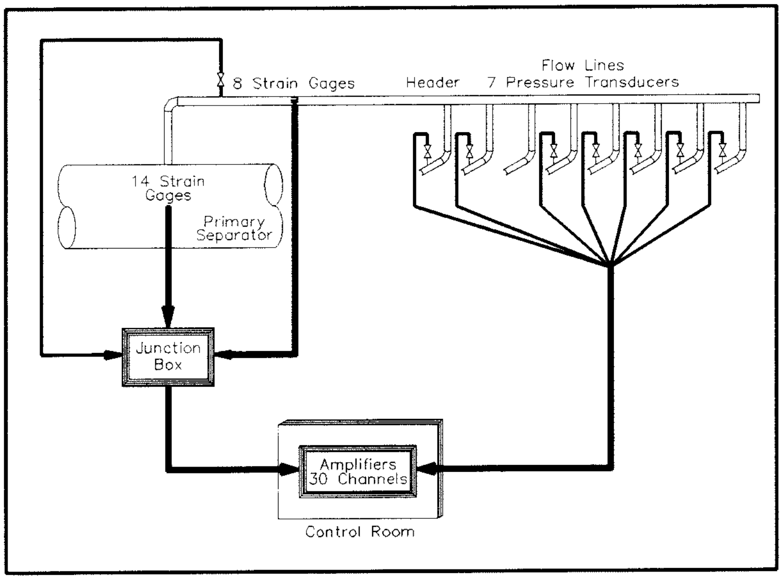


Figure 1

pressure were also installed in the inlet flow lines, and in the header immediately before the primary separator vessel. A typical installation is shown in Figure 1.

To produce a usable signal from the transducers, Wheatstone-Bridge amplifiers were required. These amplifiers were installed in a rack in the process control room. Zener diode isolation barriers were used for transducers in hazardous areas. These amplifiers, along with a **National Instruments** *AMUX-64* board and BNC connection panels were installed into a rack positioned in the process control room. Computer hardware (including the data acquisition board) were also included in the rack. A typical installation is shown in Figure 2. A diagram of the computer and acquisition hardware/software is shown in Figure 3.

3. Software Specifications

The primary function of the Slug Monitor was to capture a slug event, but ignore the data when an event was not occurring. The captured event would be ranked by severity criteria and stored in one form or another. "Interesting" events were required to be saved in their entirety (all digitized points for all channels). Less interesting events could be reduced to a few key criteria for each channel. Previous data showed that slug events would generally occur at intervals of one to 30 minutes. However, some slug events occurred in groups separated by as little as a few seconds. Therefore, data acquisition was required to continue uninterrupted while the ranking and storage functions occurred.

Previous data showed that each major slug event (an "interesting" event) had a duration of approximately 30–40 seconds from the time the slug front arrived until the end of the liquid plugged region. At typical slug propagation speeds, another 10–20 seconds was required for the slug to travel through the inlet and header piping. Therefore, a total event time of 50–60 seconds was required, including several seconds of data immediately prior to the arrival of a slug.

The preliminary data also showed that to ensure that the peak amplitudes were measured, a sampling rate of not less than 100 Hz was required. For a maximum of 40 channels, an event time of 60 seconds, and a sampling

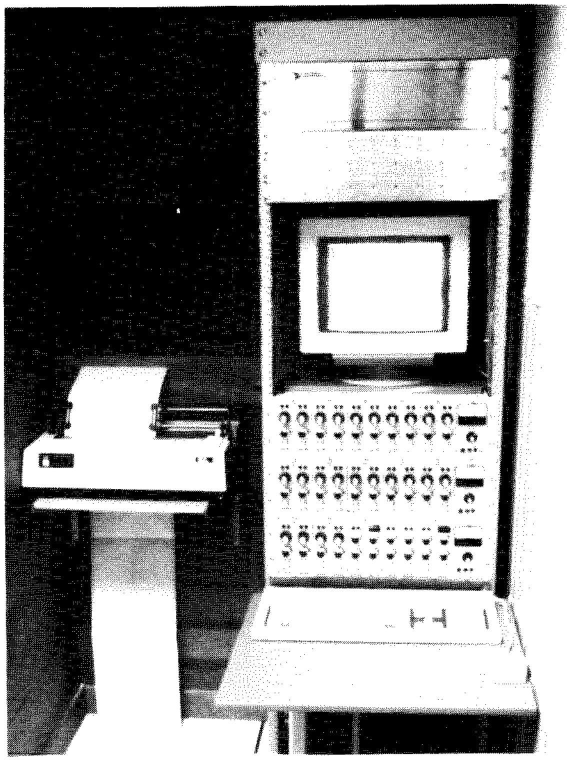
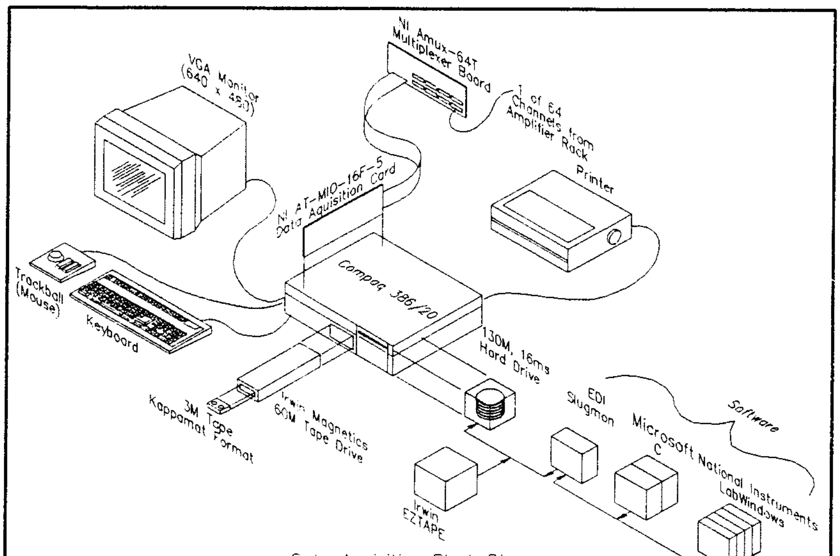


Figure 2



Data Aquisition Block Diagram

Figure 3

rate of 100 Hz, the software would be required to manipulate 240,000 digitized data points per event.

To capture a slug event, some method of determining its onset was needed. Evaluation of previous data showed that immediately prior to slug front arrival, static pressure in the line would decrease for a few seconds, followed by a rapid increase in pressure for the duration of the slug event. The data also showed that at certain strain gage locations, fluctuations in measured strain would increase momentarily to high levels. These data suggested the use of two trigger types: a "level-dwell" and a "peak–peak" trigger.

The level-dwell trigger requires an amplitude level to be exceeded for a specified amount of time. (Figure 4).

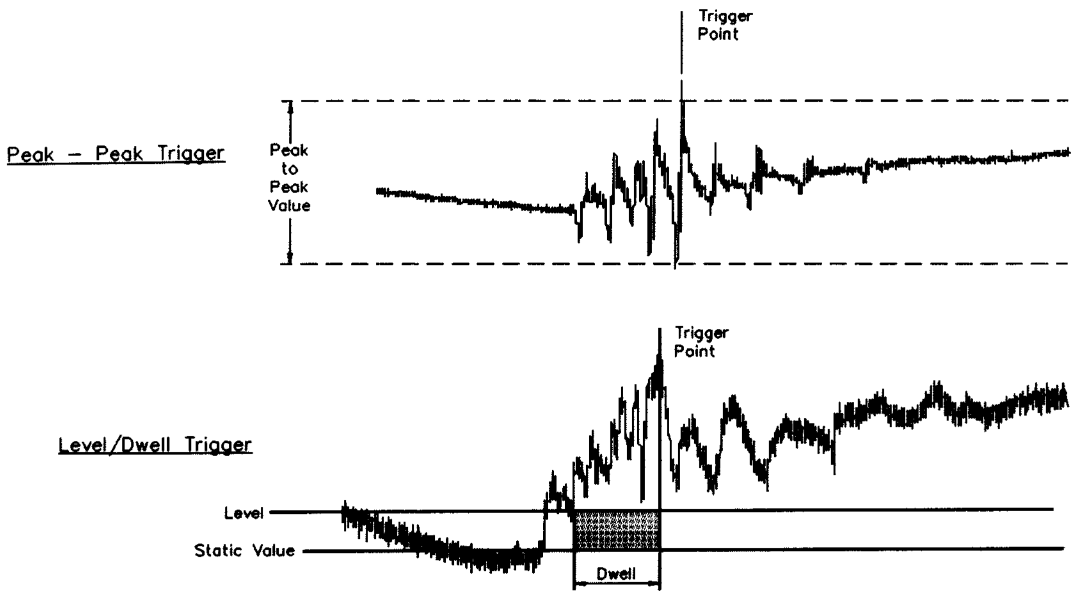


Figure 4

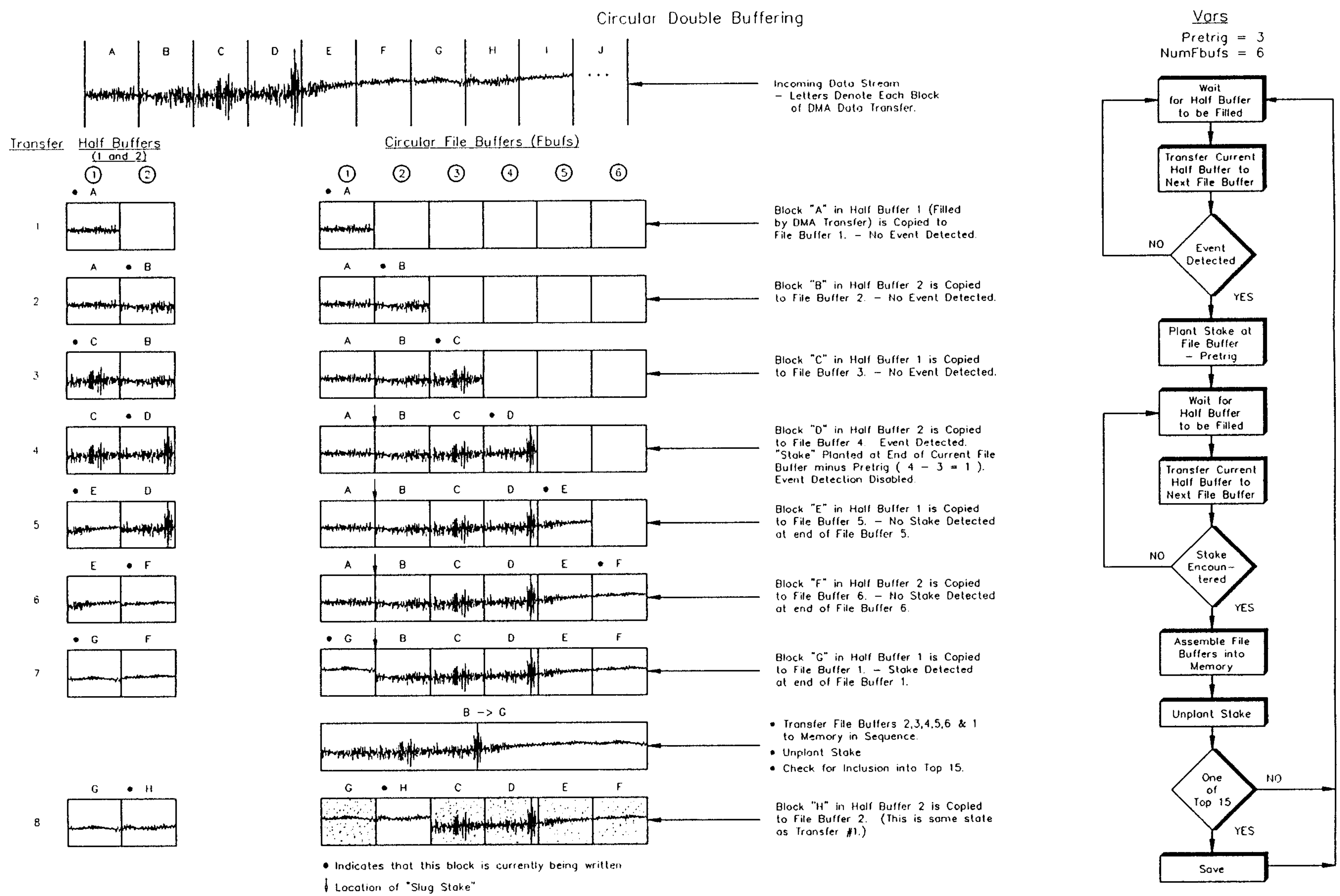
A peak–peak trigger (Figure 4) computes the maximum peak-to-peak value that occurred during some pre-defined time interval. For both trigger types, parameters can be adjusted so that only an "interesting" event should produce a trigger. The choice of which trigger type to use was made after the Slug Monitor was installed, and depended upon specific conditions at each facility.

As events were handled, several other functions were required, including: real-time data display from any channel; data trending for 4-hour, 12-hour, and 24-hour periods (to assist with process control); and statistical information from the beginning of the monitoring period (the "epoch"), for the most recent 10–15 events, and for the largest 10–15 events. A graphical user interface (GUI) was the best choice for providing this functionality.

Finally, since the CPU hardware available were 80386-based PC's, the entire software package was required to exist in a DOS environment. This rather severe restriction was to cause the greatest difficulties for completion of the project.

After evaluating off-the-shelf options available at the time, it was decided that the LabWindows product would provide the best chance for successful completion of this project in the specified amount of time. Because of the amount of data involved and the complexity of the data structures that would be required, the C programming language was selected.

Figure 5


Circular Double Buffering
A B C D E F G H I J ...
Incoming Data Stream
– Letters Denote Each Block
of DMA Data Transfer.
Transfer
Half Buffers
(1 and 2)
Circular File Buffers (Fbufs)
1 2 3 4 5 6
Block "A" in Half Buffer 1 (Filled by DMA Transfer) is Copied to File Buffer 1. – No Event Detected.
Block "B" in Half Buffer 2 is Copied to File Buffer 2. – No Event Detected.
Block "C" in Half Buffer 1 is Copied to File Buffer 3. – No Event Detected.
Block "D" in Half Buffer 2 is Copied to File Buffer 4. Event Detected. "Stake" Planted at End of Current File Buffer minus Pretrig (4 – 3 = 1). Event Detection Disabled.
Block "E" in Half Buffer 1 is Copied to File Buffer 5. – No Stake Detected at end of File Buffer 5.
Block "F" in Half Buffer 2 is Copied to File Buffer 6. – No Stake Detected at end of File Buffer 6.
Block "G" in Half Buffer 1 is Copied to File Buffer 1. – Stake Detected at end of File Buffer 1.
B -> G
• Transfer File Buffers 2,3,4,5,6 & 1 to Memory in Sequence.
• Unplant Stake
• Check for Inclusion into Top 15.
Block "H" in Half Buffer 2 is Copied to File Buffer 2. (This is same state as Transfer #1.)
• Indicates that this block is currently being written
Location of "Slug Stake"
Vars
Pretrig = 3
NumFbufs = 6
Wait for Half Buffer to be Filled
Transfer Current Half Buffer to Next File Buffer
Event Detected
NO
YES
Plant Stake at File Buffer – Pretrig
Wait for Half Buffer to be Filled
Transfer Current Half Buffer to Next File Buffer
Stake Encoun-tered
Assemble File Buffers into Memory
Unplant Stake
One of Top 15
Save

4. Acquisition Techniques

A primary requirement of the data acquisition portion of the application was that the system should continuously acquire data, regardless of other computations (GUI operation, statistical analysis, etc.) that might be occurring. The double-buffered acquisition technique [1] appeared to be the most likely method for accomplishing this task. This technique utilizes the DMA (direct memory access) controller in the PC to transfer data from the acquisition hardware directly into memory, without requiring action from the CPU. Therefore, even though data are being acquired, the CPU is free to accomplish other tasks.

4.1 Double Buffered Acquisition

Double buffering requires the allocation of a memory buffer to accept DMA transfers from the board. At any given moment, one half of the memory buffer (a "half-buffer"), is available to the CPU, while the other half is being filled with data through DMA. The half-buffer containing the acquired data may be examined, transferred to another location, etc.

Although the data are being acquired in "real time", the data becomes available for analysis after a time lag defined by the size of the half-buffer and the sampling rate. This delay can be minimized by limiting the size of the half-buffer. However, there must be enough time between half-buffer transfers to perform overhead calculations (GUI mangement, file manipulation, user-input, etc.). Since larger half-buffers will allow more processing time, half-buffer size can be increased to accommodate overhead processing requirements. Note, however, that the maximum size of a half buffer is 32K because of limitations incurred in the segmented memory architecture of the Intel processor. The half-buffer size chosen is therefore dependent upon the application requirements.

4.2 "Circular" Double Buffering

Double buffering proved adequate for acquisition of data and event detection. However, as discussed in Section 3., a method for saving data acquired *before* event detection was needed, as well as the ability to store data for a 50–60 second event. The "Circular Double Buffering" (CDB) technique was developed to accomplish this task. This process involves transferring the half-buffers to a larger section of memory, arranged in a circular fashion (i.e., the end of the last block of memory is attached to the beginning of the first block). In this manner, data before an event occurs may be recalled, but only a finite amount of memory is required.

A flow diagram of the CDB process is shown at the right of Figure 5. A data input stream for one channel is shown at the top of the figure. The data stream is broken into segments denoted by letters (A, B, C. ...). Each segment identifies a half-buffer of data. Each step in the data acquisition process is shown as a single row across the page. The two blocks along the left indicate the DMA buffer (2 half-buffers). The six blocks in the center denote a block of circular memory (represented in a linear fashion here for layout purposes)[1]. Completed DMA transfers of data to a half-buffer are shown with labels 1–8 down the left side of the plot. The symbol "•" above a block indicates which block is being read from or written to by the CPU.

Transfers 1–3 show program operation where data are being acquired by DMA into half-buffers, transferred into circular memory, and evaluated for the occurrence of an event. At transfer 4, an event is detected by peak-peak trigger. A "stake" is "planted" in memory at the end of the previous file buffer. To obtain pre-triggered data (data

[1]Note that the size of the circular memory required to store an event was 300-400 KB. Since the large memory model in DOS only allows 64K arrays, a disk-file was required. Disk transfer speeds proved to be too slow, however. Therefore, a RAM-disk was used for the circular memory block. These blocks of memory were given the name *circular file buffer*, or individually, an **Fbuf**.

before the event occurred), the stake can be planted backwards in memory a number of file buffers specified by the **Pretrig**) variable. After the stake is set, event detection is disabled. Subsequent transfers now include a search for the stake planted in memory, instead of event detection. When the stake is encountered the circular buffer contains a complete event with pre-triggered data (data before the event trigger).

Transfers 5–7 show operation after event detection and before the memory transfers wrap around to the stake. Physically, this corresponds to acquisition of the event. At transfer 7, the memory stake is detected, indicating that a complete event has been saved. The completed event must now be "unwrapped" from circular memory and assembled in a contiguous fashion. Next, the event is evaluated and ranked. Statistical information is stored for the event. If the event is "interesting", it is saved to a file in its entirety. To reset the system for another event, the memory stake is pulled, event detection is enabled, and the process begins anew (transfer 8). Note that for uninterrupted operation, the unwrapping, evaluation, storage, and reset of the system must occur faster than the time required for one half-buffer to be acquired. Half-buffer length, sampling rate, event length, trigger parameters, and hardware speed must be considered carefully to maximize performance.

5. User Interface Design

To provide user interface to the Slug Monitor application, a GUI was designed. Several different screens were required to accommodate the functionality desired. Each screen is described briefly below.

5.1 Main Screen

The start-up user interface is shown in Figure 6. This screen consists of the company logo, and buttons to select

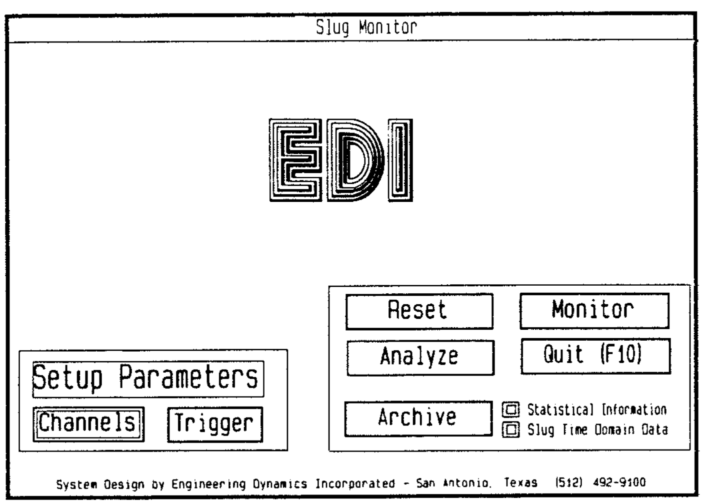


Figure 6

other actions. Channel setup, acquisition/trigger control, data reset, monitoring, analysis, archival, or program end can be selected. The data reset control provides a way to initialize the system to a new epoch. The archive control calls a series of routines to copy data to a DC2000 tape cartridge. Once archived, the data may be taken to other computers for further analysis. In practice, this function was selected at approximately 1 month intervals, after which, the system epoch was reset.

5.2 Channel Setup Screen

In any data acquisition system, definition of input channel parameters is vital for obtaining proper data calibration. The channel setup screen (Figure 7) was designed to facilitate proper entry (e.g., transducer calibration is

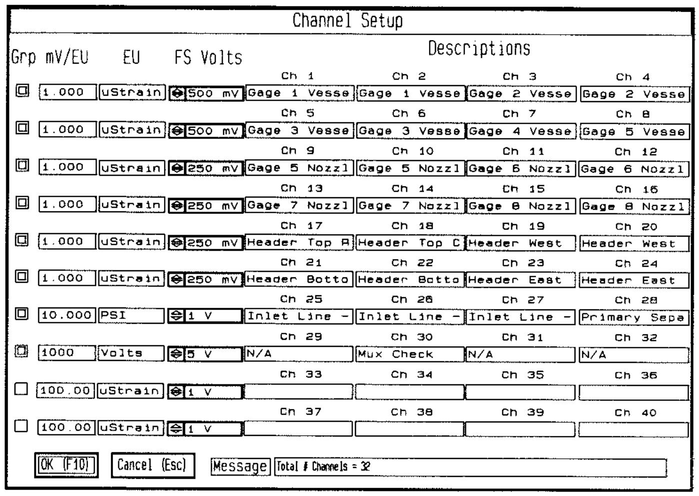


Figure 7

provided in engineering units). Because the *AMUX-64* hardware multiplexes four input channels to a single channel on the A/D board, channels must be set up in groups of four. The setup screen allows these groups to be selected or ignored. After a group is selected, input sensitivities, A/D board gain values, and channel descriptions can be selected.

5.3 Acquisition/Trigger Setup Screen

The trigger screen (Figure 8) allows selection of the type of trigger to be used, the input values for the selected trigger, the sampling rate, and half buffer size. A check feature is incorporated that ensures all user input is mutually compatible (e.g., sampling rates do not exceed maximum rates, buffer sizes do not overflow memory, etc.).

5.4 Monitor Screen

During the monitor phase of program operation, the application was required to display real-time data for any channel, display 4, 12, and 24 hour trends, plot histograms and other statistics, and tabulate previous and maximum events (Figure 9). A scheme was developed in which the screen was divided into four sections. Various displays could be chosen to provide information needed at the time. For the upper left section, the possible selections included the following.

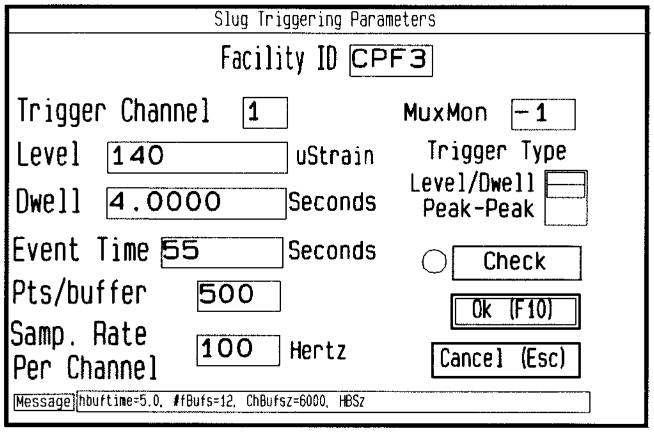


Figure 8

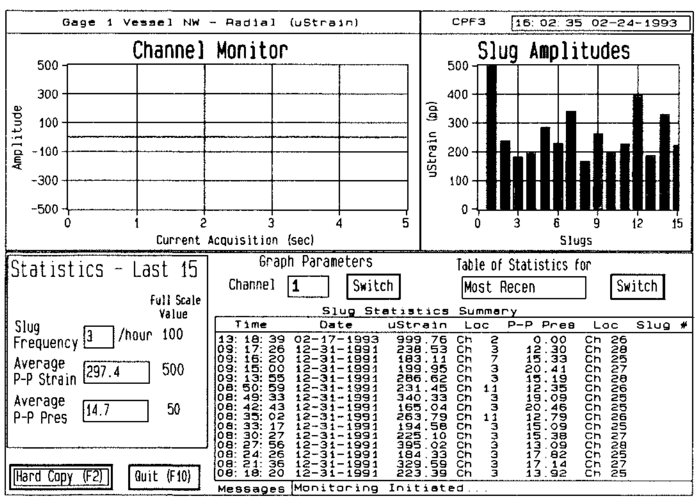


Figure 9

Data Monitor: a real-time display of data as it is being acquired — Any channel can be selected at any time through user input. An identification (input through the channel setup interface) is displayed at the top of the graph.

Data Trends: 4, 12, and 24 hour strip charts of maximum event amplitudes and frequency — These strip charts provide operations personnel with current event characteristics. Such information is useful to plant operations personnel in adjusting the process to minimize slug forces and frequency.

Histogram: a bar chart showing the number of events in given amplitude bands — Data maintained since the epoch is displayed. Histograms per channel or overall can be selected. Time since the epoch is also displayed. These data provide Operations and Engineering personnel with an at-a-glance estimate of the potential for damage caused by slug events.

The upper right and lower right sections can display statistics for either the most recent 15 events, or for the largest 15 events. Information displayed includes the event number (since the epoch), the channels that had the greatest amplitudes for a given event, and the time and day of the event. A message box displays information concerning acquisition (transfer errors, noise spike detection, etc.).

5.5 Analysis Screens

To evaluate stored data, an off-line analysis interface was provided. This functionality was split into two components, event analysis, and statistical information.

5.5.1 Event Analysis

The event analysis screen (Figure 10) provides a method for event selection and display of two channels of data

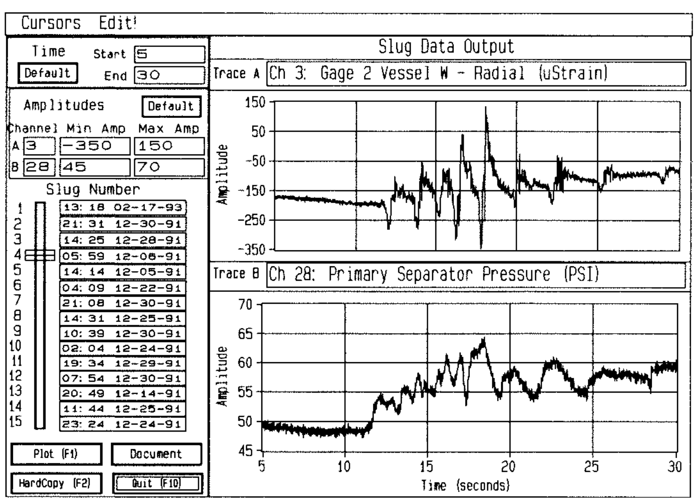


Figure 10

for a stored event. Each of the 15 events can be selected using the slide bar at the left. The data are displayed in two time-domain graphs. Over each graph is a channel description (from the Channel Input interface). Data in each graph can be displayed with independently scaled Y-axes. Graphical cursors help provide this scaling ability.

When the user is ready to create a hard-copy of the display, a documentation interface replaces the event selection interface at the left side of the screen (Figure 11). Information about the event is displayed in a fixed documentation

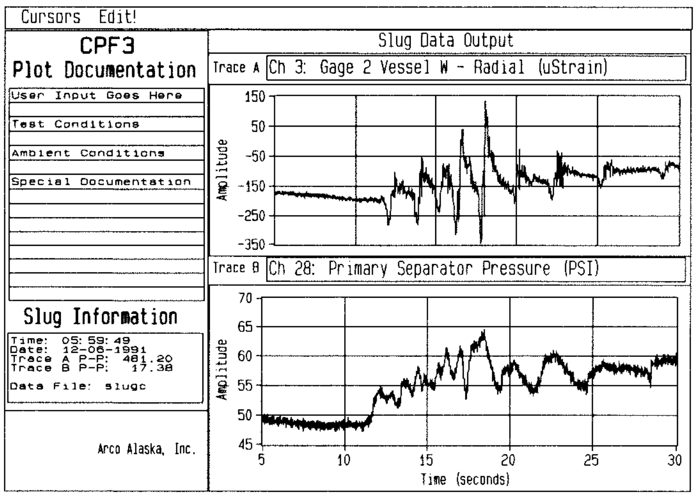


Figure 11

box in the lower left portion of the screen. Other lines are available for user-specified documentation (e.g., Ambient conditions, process conditions, special tests, etc.).

5.5.2 Statistical

Statistical information about the slug events as a group can be manipulated with the `Stats` interface (Figure 12). After a particular database of information is selected, a range of hours (fractions of an hour up to thousands) can be specified. Data within that range can be displayed as a Histogram and an Event History[2]. Hard-copy and documentation similar to the Event Analysis screen is also provided.

5.5.3 Tabular

It is sometimes useful to display the raw data in tabular formats. Four types of tabular displays were included: A **Per Event Summary** containing peak–peak values for every channel for a single event, an **Overall Summary** which contains a sequential summary table of the currently available slug events within the data range specified, a **Per Channel Summary** which is similar to the Overall Summary, except a specific channel can be selected, and a **Comparison** table that allows comparison of up to nine channels for every event.

[2]Note that because of the 64K array limitation, only 65,535 events may be displayed in the Event History plot

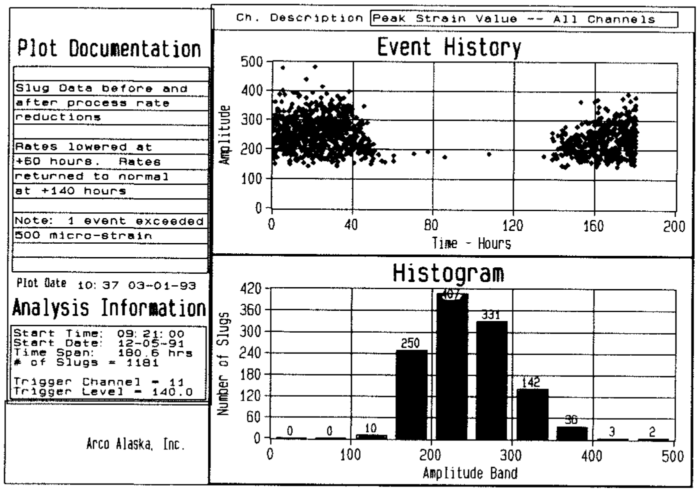


Figure 12

6. Slug Monitor Operation

After the initial development of the system was completed, the hardware was installed at three different sites. Results from slug-monitor operation were immediate and significant. However, because conditions in field operation are usually not as pristine as in a laboratory environment, a few unforseen problems were encountered that merit discussion.

6.1 Spike Noise

High frequency noise spikes appeared in the data from time to time. These spikes would sometimes cause errors in event detection and analysis. Efforts to determine the source of the noise spikes proved ineffectual. Electronic filters removed the spikes, but caused excessive distortion of the data. Since there were unused CPU cycles available during program operation, a spike detection/removal algorithm was developed to digitally remove the spikes as the data were acquired. This action proved to be quite successful.

6.2 Multiplexer Failure

As the real-time data were monitored, it was noticed that periodically, the data would seem to change abruptly. Close inspection revealed that the data channels seemed to be rotating in groups of four (i.e., channels 1,2,3,4 would suddenly become 2,3,4,1, then become 3,4,1,2, etc.). It was noticed that the condition could be generated at any of the installations when static electrical discharges occurred nearby. Because the *AMUX-64* board was a 4-to-1 multiplexor, it was thought the problem originated there. Efforts to correct the problem electrically (isolation, grounding, etc.) were unsuccessful. The multiplexor board manufacturer also was not able to offer a solution.

Since it was apparently not possible to solve the problem with the multiplexor in the field, a software "solution" was implemented. A group of four unused channels was selected in which one channel was tied high (connected to +5 volts) and the other three were tied to ground. The software monitored the channel that was high. When a multiplexor failure occurred, the high channel shifted, and the software detected this as an error condition. Appropriate action was then taken to correct the channel alignment. While this solution was somewhat undesirable, it did not significantly affect system performance. Without this action, the monitoring system would not have been usable.

6.3 Results

Prior to installation of the Slug Monitor, it was thought that it was not possible to alter process variables that could reduce slug forces without affecting production. However, with the monitor in place, plant personnel were able to experiment with the process and have immediate feedback about slug forces. Significant reductions in slug forces were achieved without affecting production.

The statistical data indicated that certain inlet flow lines were originators of the majority of slug events. This insight helped engineers understand the slug flow phenomenon.

The data provided accurate estimations of slug force amplitudes where previously, none were available. Combining this information with analytical models, design modifications could be optimized. Data obtained after modifications were installed helped close the design "loop" by validating or modifying analysis assumptions. Ultimately, these data resulted in better design at lower cost than could have otherwise been achieved.

Prior to the availability of a significant amount of statistical data obtained over a wide variety of operating conditions, management was required to make decisions concerning the slug-flow problem based on opinions of their staff, which varied widely. The data was able to help provide justifications for the significant expenditures of resources that were required to address the slug-flow problems.

6.4 Future Modifications

Subsequent to addressing short-term concerns, the long-term issues that are under consideration are improvements in monitor operation, and computation of the fatigue life of the piping and vessels.

Fatigue life calculations can be developed from the stored data and as future data is collected. Efforts are currently in progress to provide a capability to estimate current damage and the remaining life of structural components. Additionally, a "rate-of-use" indicator is being investigated for inclusion into the monitor.

Monitoring system improvements could include connection of the monitor into a central process computer. Such a connection could provide improved archive capability, as well as a more intimate interface with process conditions. Remote users could also access the acquired data.

7. Conclusions

The slug monitoring system provided the information necessary for plant personnel to improve the facilities response to forces generated by slug flow. Improvements in the ability to resist slug forces resulted in 300–500 % reductions in strain levels at vessel nozzles and piping.

Data from Slug Monitors were used to maximize production rates while minimizing slug forces. Strain levels were used to provide a comparison to ASME code criteria for fatigue life calculations. Therefore, management decisions optimizing production rates and resource expenditure were made more effectively.

Modifications made to date appear to be adequate at restraining slug forces, negating the need for additional, more costly design modifications. Future use of the system will provide a tool to assess operating considerations on a continuous basis as field and production requirements change.

The techniques utilized in development of the software provided a system that could simultaneously acquire, analyze, and display data, without losing information. Although this software was developed for a specialized application, the system could be easily adapted to a variety situations where intermittent transient monitoring is desired. Such situations include:

- evaluation of fatigue damage to structures from intermittent loads (such as damage to highway bridges from traffic, damage to aircraft structures due to high landing loads, upsets in chemical processes, etc.),
- capture of data prior to shutdown of rotating equipment due to high vibration (e.g., instability or surge), and
- noise monitoring where L_{10} noise (short duration, high level) is a problem.

Acknowledgements — The work presented in this paper could not have been completed without the efforts of Ken Atkins of EDI who provided invaluable insight and assistance with application programming. Additionally, the author would like to thank Don Smith for managing the project under which this work was performed, and both Don and and Bill Farnell for their work installing the instrumentation system. Appreciation is also directed to Mark Broom who provided many of the figures, and to the management of EDI for allowing development and presentation of this paper.

References

[1] *NI-DAQ Software Reference Manual for DOS/Windows/LabWindows.* Version 4.3, March 1992, pp 7-1 – 7-7. National Instruments Corporation, Austin, Texas.