PULSE CODE MODULATION DATA COMPRESSION FOR AUTOMATED TEST EQUIPMENT

T. A. Navickas
S. G. Jones
Allied-Signal Inc.
Kansas City Division*
P.O. Box 419159
Kansas City, Missouri 64141

ABSTRACT

Development of automated test equipment for an advanced telemetry system requires continuous monitoring of PCM data while exercising telemetry inputs. This requirements leads to a large amount of data that needs to be stored and later analyzed. For example, a data stream of 4 Mbits/s and a test time of thirty minutes would yield 900 Mbytes of raw data. With this raw data, information needs to be stored to correlate the raw data to the test stimulus. This leads to a total of 1.8 Gb of data to be stored and analyzed. There is no method to analyze this amount of data in a reasonable time. A data compression method is needed to reduce the amount of data collected to a reasonable amount.

The solution to the problem was data reduction. Data reduction was accomplished by real time limit checking, time stamping, and smart software. Limit checking was accomplished by an eight state finite state machine and four compression algorithms. Time stamping was needed to correlate stimulus to the appropriate output for data reconstruction. The software was written in the C programming language with a DOS extender used to allow it to run in extended mode. A 94 - 98% compression in the amount of data gathered was accomplished using this method.

INTRODUCTION

Several commercial manufactures products were evaluated as a solution to this problem. Many of the products were found to come close or meet the requirements. However, none were found to be completely acceptable do to either cost or

---

*Operated for the U.S. Department of Energy by Allied-Signal Inc., Kansas City Division under contract number DE-ACO4-76-DP00613.

performance. A combination of commercial products and custom boards was needed to solve the problem. A commercial decommutator was purchased to decom the data and send it to this chassis. This chassis is to be called the PCM Data Compressor.

This paper will present a brief overview of the custom hardware and software that was developed to create the PCM Data Compressor. It will describe each custom board and the software design that was chosen to solve the problem.

PCM DATA COMPRESSOR

Theory of Operation

The data compressor is enclosed in a 13 slot VME bus with a three slot VSB bus. The data compressor is comprised of an 80486 embedded PC with 16 Mb of DRAM, a 32 Mb dual port DRAM memory board, and three custom designed boards numbered 781763, 781764, and 781765. 781763 interfaces the data compressor to the VME bus and performs the actual data compression. It then sends data and control signals to the 781764 board. The 781764 board interfaces the data compressor to the VSB bus. It is also where the timing circuits are that perform the time stamping features. The 781765 board records time for test stimulus which is needed for data analysis later.

781763 VME Interface, Data Compressor

VME Interface:
The PCM Data Compressor occupies the A32 address space on the VME bus. Each area of the compressor is memory mapped onto the VME bus by hardware located on this board. The embedded PC, BUS MASTER, presents the address on the bus along with the appropriate control signals for a read or write cycle. When this address matches the base address of the compressor the address cycle starts. The address cycle decodes the lower twenty four address lines and enables the appropriate hardware. The MASTER then presents the address on the address lines along with signals to tell the board the length of the data to be used. Data can be single, double, or quad bytes in length. The board responds by latching the data into the appropriate area. After the data has been latched the board drives the data acknowledge signal low telling the MASTER to end the address phase. If this was a read transfer the data acknowledge signal would be used to tell the MASTER that the data had been placed on the data lines.

Data Compressor:
The data compressor part of 781763 board is where the actual data analysis is performed. When a channel is presented to the board, decisions are made whether or

not to store the channel in memory. A channel is comprised of a tag and data associated with a particular word in a frame. This decision is based on information about each channel that is stored in two 4K X 16 ram's prior to testing. These ram's contain information on upper and lower limits, which algorithm is to be performed, and if the channel is to be time stamped. The four available algorithms are fixed limits, pass always, pass never, and delta limit. The first three algorithms perform exactly as they seem. The delta limit is a method of computing new limits based on the current data. This information is taken from the information the user enter on the channel information page. The hardware also has the ability to bypass the information for each channel and globally store each channel or never store any channel.

This board is controlled by an eight state finite state machine (FSM). The FSM cycles through all states every time a channel is presented to the board. Each state is 800 Ns long which produces a 6.4 Us cycle. The cycle period can be easily changed by increasing or decreasing the clock speed. When the FSM receives a word strobe from the decommutator the cycle starts. State 1 latches the tag and the data from the decommutator into the compressor. States 2 generates chip enable for the two ram's. State 3 outputs the specific information about the current channel to the compressor. Two processes occur at state 4. First, the data for the channel is limit checked against the limits generated in state 3 and a data error is generated if the data is outside the limits. The channel is then sent to the 781764 board to be written to memory if a data error was generated. If the pass always or pass never function was selected they take priority over the data error. The second process that occurs is new limits are created if the delta limit was selected. The limits are generated by the value of the channel and the tolerance bits that were stored in the RAM. State 5 causes the information about each channel to be removed from the compressor. State C6 allows the new limits to be written back into the RAM if the delta limit was selected and a data error occurred. State C7 resets the FSM and prepares it for the next word.

781764 VSB Bus Interface, Timer, Data Display Page

The 781764 board is configured as a VSB bus MASTER. The VSB interface is controlled by two Johnson counters. When a data error is detected on the 781764 board the first counter is started (control). This counter runs at 1.25 MHz and has 6 active states. State 1 increments the memory address counter and starts the second counter (interface). States C2 and C3 are wait states for the Interface counter. State C4 is used when time is to be stored with the channel. The control counter runs up to a 5.6 Us cycle depending on the speed of the memory card. The interface counter controls the actual bus traffic on the VSB bus. It is a nine state counter running at 10 MHz. The cycle time is 900 Ns if no waiting occurs, with an average cycle time of 1.90 Us. The reason for the variation in the cycle time is due to the memory cards response time.

State 1 enables the memory address buffers placing the address on the bus along with the appropriate address size and space signals. State 2 asserts the valid address signal. State 3 waits for the memory board to acknowledge it has received the address and then disables the address buffers. State 4 enables the data buffers placing data on the lines. The address and data lines are multiplexed. State 5 asserts the data valid signal telling the memory card data is on the bus. State 6 is a wait state. The wait varies from 100 to 400 Ns. State 7 is skipped because of the asychronous characteristic of the counter. State 8 resets the control signals and disables the buffers while state 9 resets the counter. Figure 1 shows how the data is written into memory when a data error occurs and when a channel is to be time stamped. Each block represents one byte of memory.

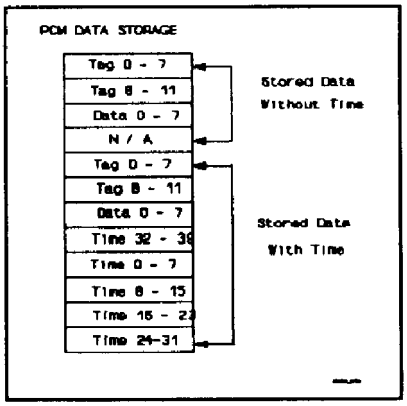

Figure 1

Timer

The second function of the 781764 board is to mark time. Two events can cause time to be recorded. The first occurs when a channel is to be time stamped and the second is when an external event occurs on board 781765. The timing circuits have a resolution of 2 Us which can easily be reduced by increasing the clock speed. When time is to be stored two processes occur. The first process blocks data from the timer from changing state while the data is being read. The second event places the time on the bus so it can be stored in memory.

Data Page

The data display page is used as a window into the data compressor. As a channel enters the compressor its data can be viewed on the crt. This viewing is done with two 2K X 8 dual port ram's. The channels tag is used as an index into the RAM. The first 2048 channels are written into the first RAM, while the last 2048 are written into the other. This decoding is done using the most significant bit and the tag. At the same time data is being written into the RAM it is being read out over the VME bus. Any

address conflict is handled by the internal circuits of the RAM. If any conflict still arises the data compressor side of the RAM has priority.

7861765 Event Card

Boards 781763 and 781764 compress, time stamp, and store data for each channel. The results are then stored in memory. Once all the data is compressed it needs to be analyzed to determine if the unit responded the way it should to a particular stimulus. This analysis requires a method to match the stimulus to the corresponding data from the unit. This matching is done with time. Each response from the unit has a time associated with it. This time is derived by the first channel in every frame always stored and time stamped. Every channel can have its time computed by taking its word location in the frame, multiplying by the word rate, and adding the result to the time obtained for the first channel. The stimulus has a similar method to record time when it was applied or removed. Both these times must be relative to the same source.

The 781765 board treats each stimulus as an interrupt. There are eight interrupt lines with each lines rising or falling edge being able to cause an interrupt. Each interrupt can be programmed for a rise, fall, or both to cause an interrupt. An example of this would be applying a dc voltage to a unit input. When the voltage is applied a rising edge interrupt occurs and when it is removed a failing edge interrupt occurs. When an interrupt occurs a signal is sent to the 781764 board telling it to latch the time. This time is then written into RAM located on the 781765 board. This entire process is approximately 1.0 Us in length. Any event occurring in this time will not cause time to be stored but could be recognized when the event is written into memory. There is a 200 Ns period where an interrupt could be missed.

Software

The software routines that control the PCM data compressor were written in C Language, along with three memory access routines written in assembly language, and linked with the C routines. The need for rapid access to all 32 Mb of data and extremely large data segments to hold the processed PCM data forced the embedded PC to use a compiler/linker that would run the software in extended DOS mode. WATCOM C was chosen for the source code compiler and Phar Lap Dos Extender was used to put DOS in extended mode.

The two software packages mated quite well and allowed the software to access computer memory above 1 Mb, which is the DOS real mode limit, without having to use any DOS real mode patches. This solved the large data segment problem. The protected mode environment allowed more rapid access to VME memory by not being

restricted to 64K windows with direct access to all memory locations in the 32 Mb block. Using the Phar Lap DOS extender the PC was able to initialize 32Mb of DRAM in approximately 100 seconds compared to 12 minutes using the conventional method.

The software processes for the PCM data compressor is divided into three areas, PCM channel setup, run time data display, and PCM data post processing.

A. PCM Channel setup.
The data compressor allows up to 4096 channels to be defined. Each channel has eight attributes associated with it. These channels and attributes are defined by the user through a channel information user and then stored on a disk. The eight attributes of a channel are:

1. Channel Name - Any alpha numeric name used to reference channel in the data stream.
2. Repeat Index - The number of channels between the first and second occurrence of a channel in the same frame. This is used for super commutated channels.
3. Compression Method - Selects one of the four available methods.
4. Time Stamp - Indicates if the channel is to be time stamped.
5. Real Time Limits - Indicates if a channel is to have interrupt capability.
6. Compression Limits - A limit from 0 to 255
7. Compression Tolerance - a range from $\pm$ 15 for delta limit.
8. Interrupt Limits - A limit range from 0 to 255.

B. Run Time Data Page.
The user issues a run command after the channel information is written to memory to setup the compressor and the data capture memory has been initialized. When this command is received the software will enter a window that displays channel names and the current data for that channel. The data for the channel is continually updated until the user issues a stop command.

The user sets up the data page using the names of the channels defined in the channel information page. Up to 95 channels can be entered on each data page. The data for each channel is gathered by reading a memory location using an offset equal to the tag number. The value is read using an assembly language interface and then displayed at specified coordinates on the crt.

The data page is also where the interrupt limits, defined on the channel information page, are used. If a channel has these limits enabled the computer checks the data read

for the data page and if out of limits, it sets up an interrupt. This information is used when a channel requires immediate attention if out of limits. An example of this would be a channel that monitors temperature of the unit.

C. PCM Data Post Processing.
After data capturing is complete the raw data has been stored in the 32Mb of memory. This raw data must then be further compressed and then converted into a user friendly format. The post processing compression is accomplished by removing unnecessary special channels. A special channel, typically the last channel of the frame, is utilized to count the number of captured frames and to record what time that frame occurred. A special channel is compressed if there is no other data between two consecutive special channels and the time between these two channels meets certain limits. The time requirement is to verify there is no data gap present. If these conditions are met the frame counter is incremented and the second special channel is compressed. This extra compression is done in seconds and compresses 2 - 3 % more data of the total data volume.

After the raw data has been post compressed it is then ready to be converted into a format the user can understand and use in analysis. Five elements comprise this format: frame number, tag number, data, time, and name of the channel. The data and tag are already in memory, the name is derived from the tag and the channel information page. The frame and time are obtained from the compressed special channels. The time is computed by the following two equations:
If the tag number is less than the special channel:
time = special channel time base+((special channel-(special channel - tag))*word time)
If the tag number is greater that the special tag:
time = special channel time base+((special channel + (special channel - tag))*word time)
where word time = 8.0/(bit rate *1000)

All the raw data is converted to this format and then sent on to the host computer for analysis.

The event data also has to be converted into a user friendly format. This format converts the digital time in a seconds reading. The conversion also determines what event occurred and whether or not it was a rising or failing occurrence. This process also sends back status information which aids the main computer in analysis.

CONCLUSION

The design of the PCM Data Compressor was extremely successful. The reasons for this success are cost, maintenance, data compression, and speed.

The cost of the PCM Data Compressor was $26,213. This includes $1500 for each custom board. The cost of the entire system was $38,213 taking into account the cost of the commercial decommutator. This price was comparable with the cost of commercial products. With the extra capability and customizing the cost of the pcm compressor is justified. Having designed and proved in the custom boards the expertise exist on site and any needed service is available immediately.

The most important reason for the success of the PCM Data Compressor is the excellent data compression ratio and speed that was obtained. The hardware achieved a 95% compression ratio. This was due to the flexibility of the compression methods. The delta limit method allows the user to compress out channels whose values fluctuated randomly during testing. The software was able to increase this ratio to 98% by deleting unnecessary special channels. This method was only possible because of the software running in extended DOS mode and the speed of the CPU.

In conclusion, the problem stated at the beginning showed the need to be able to store and analyze 1.8 Gb of data. The data compressor was able to reduce this quantity to under 32Mb. This compression allows the user to test a unit faster and more accurately than was possible before.