DESIGNING A HIGH-SPEED DATA ARCHIVE SYSTEM

R.S. Bain
Design Engineer
Veda Systems Incorporated
California, MD 20619

KEY WORDS

Archive, Storage, Playback, Recording

ABSTRACT

Modern telemetry systems collect large quantities of data at high rates of speed. To support near real-time analysis of this data, a sophisticated data archival system is required. If the data is to be analyzed during the test, it must be available via computer-accessible peripheral devices. The use of computer-compatible media permits powerful “instant- replay-type” functions to be implemented, allowing the user to search for events, blow-up time segments, or alter playback rates. Using computer-compatible media also implies inexpensive COTS devices with an “industry standard” interface and direct media compatibility with host processing systems. This paper discusses the design and implementation of a board-level archive subsystem developed by Veda Systems, Incorporated.

DESIGN REQUIREMENTS

This system is designed to record processed telemetry data as output by PCM decommutators, 1553 bus monitors, etc. It is not intended to replace traditional instrumentation recorders, which are unsurpassed at recording raw unprocessed data streams. The requirement for this system is driven primarily by the flight test engineer who needs the capabilities to record slices of data (runs) during an in-flight test and to replay and evaluate the data during or immediately after the flight. Specific requirements imposed in this design include:

- High recording rates - up to 10 megabytes/second
- Reasonable storage capabilities - up to 4 gigabytes
- Low cost COTS peripheral devices
- Low error rates, error detection and correction

- Standardized computer-compatible interface formats
- Selective recording of bus traffic (data)
- Adjustable replay rates
- Multiple recording sessions
- Ability to easily transfer data to other systems
- Single-board design

To meet these requirements, low-cost Winchester disk drives were selected as the recording media. Current drive technology offers Small Computer Systems Interface (SCSI) disk drives with capacities in excess of two gigabytes and recording rates up to 33 megabits per second. By using multiple drives, it is possible to achieve both the rates and capacities requested. The use of a standardized peripheral interface (SCSI) not only offers a wide selection of peripherals but also provides a standardized link to host systems for subsequent data transfer. Since SCSI drives incorporate their own internal controllers, they perform all necessary formatting, error detection, and error correction. The SCSI host system is not responsible or concerned with any of these (now internal) functions.

To further reduce storage requirements (and increase bandwidth), data compaction and compression functions are required. To perform these functions in a real-time environment requires either complex dedicated hardware or an extremely fast processor. To keep physical size and material costs down, a microprocessor was selected to perform data compression functions.

DESIGN SPECIFICS

After reviewing the requirements, a basic hardware architecture was devised to include five major functional groups: a TM data interface, a processor section, a SCSI section, a host-bus interface, and a memory subsystem. Figure 1 depicts the hardware architecture.

The target system for this design employs a 32-bit telemetry data bus with a 16-bit tag (address) bus. This bus is collectively referenced as the Global Memory Interface (GMI) bus. The host bus in the target system is a 16-bit IBM PC/AT ISA bus. This archive subsystem design fits the PC/AT circuit card form factor.

The processor selected for the archive subsystem is an INTEL i860-XR. The i860 is a high-performance, RISC-based, 64-bit microprocessor. Internally, it contains separate integer and floating point units with separate pipelines, enabling it to execute both floating point and integer instructions within the same clock cycle. The i860 is clocked at 40 megahertz, yielding a 25-nanosecond instruction cycle.

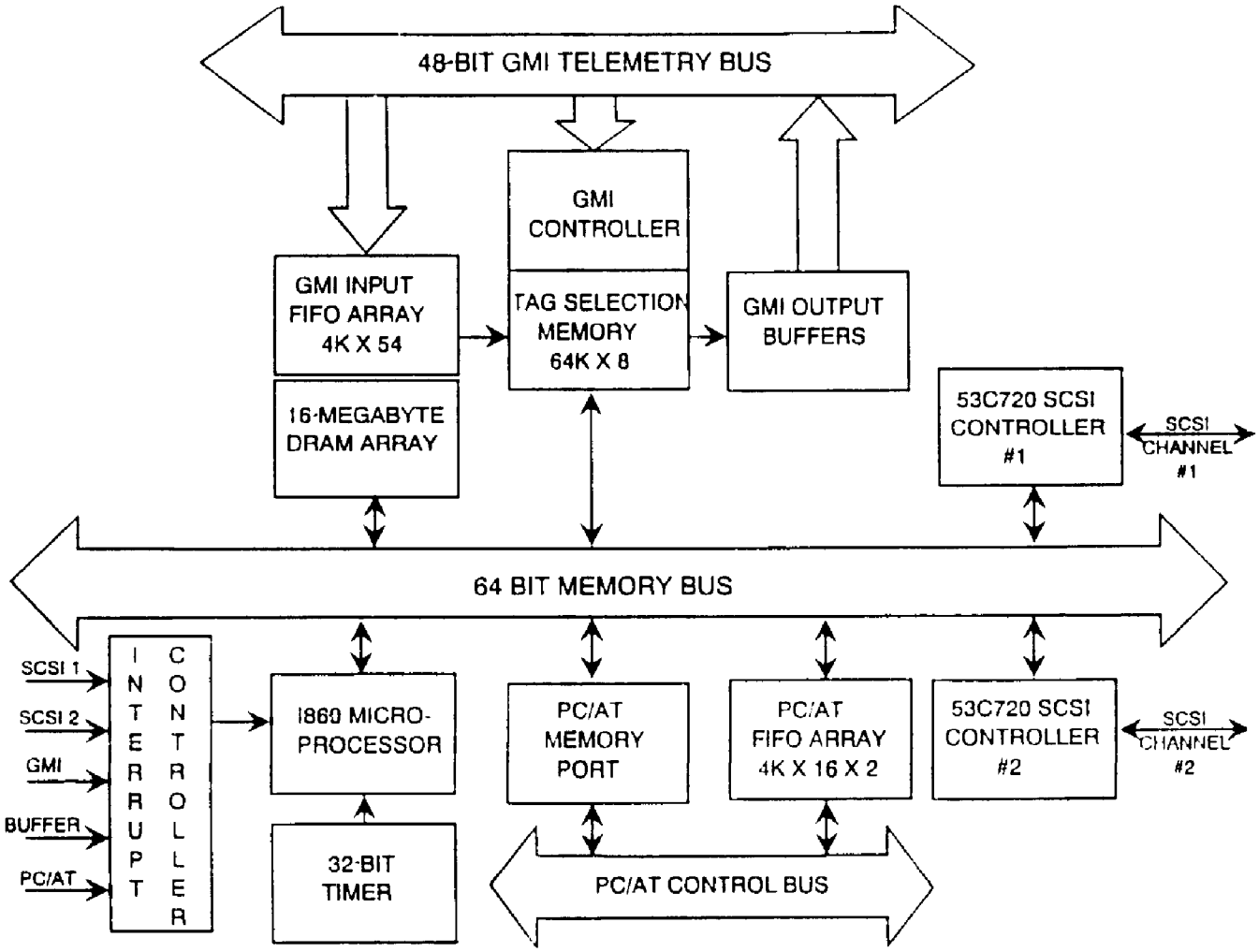

Figure 1 - Hardware Architecture

The resulting design employs three strategies to circumvent memory bottlenecks. First, memory access is performed in “fast page mode,” reducing memory cycle time to 75 nanoseconds (for a 64-bit word). Second, instruction and data streams are pipelined, allowing the next instruction to be pre-fetched while the last two instructions are decoded and executed. Third, the i860 employs two 8-kilobyte caches (instruction and data), which are accessible in a single clock cycle. When a cache miss occurs, the i860 performs a cache line load, pre-fetching instructions and data for subsequent use. At 40 megahertz the i860-XR yields peak performance ratings of 40 million instructions and 80 million floating point operations per second.

Like most RISC-based processors, the i860 has a limited interrupt architecture, so an interrupt controller became a design requirement. The interrupt controller accepts inputs from the host interface, each of the two SCSI controllers, and the GMI interface. The host interface generates an interrupt to signal the i860 that

communications services are required. The SCSI controllers generate interrupts when exception conditions occur that require host intervention (i.e. disk failure). The GMI interface generates interrupts for requested parameter detection and for 32k buffer boundary crossings. An additional interrupt is generated as a 50-millisecond heartbeat interrupt to support multitasking operations. A master interrupt bit signals that an interrupt request has occurred. After generating an i860 interrupt, subsequent interrupts are recorded but held off for 50 microseconds, allowing the current interrupts to be serviced. The interrupt controller may be programmed to accept or ignore any interrupt request and may also be operated in a polled mode, where i860 interrupts are not actually generated but interrupt request status is latched and read by the CPU. This last mode is the normal mode of operation. since it significantly reduces CPU overhead.

The GMI interface extracts 48-bit telemetry data from the telemetry bus and deposits the data in a 64-kilo-word circular buffer located in i860 DRAM. FIFOs are used to buffer incoming GMI traffic. A block of tag-selection memory is employed to identify parameters which have been selected for extraction (archival). As each parameter is received, its corresponding 16-bit tag is examined. If the parameter has been selected for extraction, it is written into FIFO memory for subsequent transfer into i860 memory. Two basic modes of operation, burst and demand, are provided. In demand mode, a memory access is generated every time a parameter is available in the FIFOs. This mode provides rapid memory updates when GMI bus traffic is minimal, but forces long memory access times because page mode operation is circumvented. In burst mode, memory access is not requested until the FIFOs are half full. Then FIFO data is bursted into memory using page-mode accesses until the FIFOs are empty. Burst mode is far more efficient than demand mode when high data rates are employed.

The GMI controller transfers data into a 64-kilo-parameter circular buffer in the i860 DRAM address space. As each 32-kilo-parameter boundary is crossed, an interrupt request is generated to the interrupt controller. To assist in processing the data, six bits of flag information are transferred from select memory to the circular buffer with each parameter. Two of these bits define the size of the parameter (in bytes), two others define the source of the parameter (PCM stream, 1553 stream, etc.), and the remaining two bits are application specific. The 64k x 8-bit tag-selection memory provides these six bits for each tagged parameter. The remaining two bits of selection memory identify the parameter for extraction and allow an interrupt to be generated upon detection of a specified tag. Tag selection memory may be modified “on-the-fly,” allowing dynamic reconfiguration. Burst and demand modes may also be selected “on-the-fly”, allowing any remaining parameters to be flushed from the FIFOs when input data stops. The base

address of the circular buffer is programmable, allowing it to be located anywhere in the i860 address space.

A 16-bit buffer index register is provided to identify the exact location of the last parameter transferred into the circular buffer. A single 32-bit timer allows the i860 to control data output onto the GMI bus in playback modes. The timer has an LSB resolution of one microsecond, allowing the i860 to regulate output to the GMI bus to simulate various data rates.

The High-Speed Archive System employs two NCR 53C720 high-performance SCSI controllers, which provide dual SCSI-2 data paths. The 53C720 contains an embedded processor to perform all SCSI operations without CPU intervention. It supports both “wide” and “fast” SCSI options, providing data transfer rates as high as 20 megabytes per second (synchronous burst per channel). Each controller has its own internal FIFOs for data and instructions. Memory accesses are performed in block mode, allowing the controller to transfer up to eight long words of data per memory access. The controller operates in a loosely coupled mode, executing an instruction script placed in memory by the CPU. Each controller has the ability to generate interrupts to the i860 to signal exceptions.

Two distinct interfaces are provided from the archive subsystem to the PC/AT bus (control bus). A direct memory interface is employed to allow the host (PC/AT) processor to download executable code to the i860 CPU. This interface allows the host processor to directly read from or write to i860 memory, regardless of whether the i860 is running or halted. All data transfers are performed as 64-bit words with 32-bit addresses. This interface allows the host system to download code, examine memory, provide debugging facilities, and supplement hardware integrity tests.

A FIFO-based data transfer interface is provided to transfer blocks of data to or from the host. This is a 4k-deep, 16-bit-wide, full-duplex interface with empty and half-full flags available to both the i860 and the host. An 8-bit-wide, bi-directional port is provided to coordinate data transfers. This interface is I/O mapped on both the host and i860 sides and provides zero-wait-state operation for all data transfers. System startup is accomplished by holding the i860 microprocessor in a reset condition, downloading the executable code modules via the host memory interface, and then releasing the i860 to execute the code.

To buffer incoming data during the various SCSI transfer cycles requires a large amount of memory. To meet the design goals and allow a twenty percent safety margin, at least six megabytes of memory are required. This design has eight megabytes of 55-nanosecond, fast-page-mode DRAM as main memory,

permitting a relaxed 75-nanosecond page-mode transfer rate. Memory is organized as one megaword by 64 bits, growing downward from the top of the i860 address space.

A sophisticated memory controller is employed to support 5-way memory porting and fast-page-mode accesses. The SCSI controllers have the highest priority in the memory request chain (followed by the GMI and host interfaces). The i860 has the lowest memory request priority, since it will fetch instructions and data from its internal cache memories - in most cases. Worst case memory arbitration latency is one clock cycle. Long memory accesses require 6 clock cycles to complete. Page-mode accesses are accomplished in 3 clock cycles.

OPERATIONAL DESCRIPTION

Prior to beginning data acquisition, the tag-selection memories are initialized to select the appropriate parameters to be archived. This initialization data is recorded in a header record in the front of the output data file to enable reconstruction of the data for playback. As telemetry data is received on the GMI bus, the parameter ID (tag) is fed into the tag-selection memories. If the tag has been selected (for archive), the memories will trigger a write into the parameter input FIFOs. When the FIFOs become half full, they will trigger a memory request to the memory controller. Once any previously active memory cycle is complete, the controller will begin to transfer data from the FIFOs directly into a 64-kilo-parameter circular buffer in i860 memory, using fast-page-mode memory accesses. Once 32 kilo-parameters have been transferred into the buffer, the controller will generate an interrupt, signaling a buffer-half-full condition. The processor responds to this condition by reading the input data, compacting it (modulo 8 packing), and placing the results in an output buffer. When the output buffer becomes full, the processor sets a semaphore, signaling the SCSI processor to output the data to disk.

The data compaction function performed by the processor is simple modulo-eight data packing. Embedded in the data word for the parameter is the length of the parameter in bytes, the destination buffer number, and the archive mode (tagged or tagless). The processor simply shifts the parameter the appropriate number of bytes and then merges it with the previous data in the output buffer. The idea is simply to minimize the number of bytes required to represent a parameter.

For data rates which do not exceed the bandwidth of the on-board processor, sophisticated data compression schemes may be implemented. These may be simple N-sample averaging, delta change, or even bit compression (hashing)

algorithms, depending on the nature of the data to be recorded, data rates, and the requirements of the ensuing analysis.

In cases of extremely high data rates, it may not be possible to utilize these processing intensive algorithms, but a synchronous, “tagless” form of archiving may be implemented. This mode only works if the data is synchronous in nature (most PCM streams are). The first parameter in the output buffer must be the sync word, since the data is now positional. Both tagged and tagless data may be employed in the same archiving session (for different parameters) through the use of separate output buffers. The output buffer is prefixed with a header detailing the recording mode, start time, and word count. The start time and word count are used to compute an averaged data rate across buffers, permitting regulation of the output rate on playback.

For extended recording sessions, the disk drives may be housed in a data shuttle and a “ping-pong” recording approach is used. This allows the user to swap full disk cartridges for empty ones without interrupting the archiving session.

For extremely high data rates, a data-spreading approach (much like track spreading on an HDDR) is used to interleave data across multiple drives. The 53C720 SCSI controller is capable of bursting data (in synchronous mode) at 10 megabytes per second. Data is burst through SCSI channel #1 to drive A, until a disconnect is issued by the drive. At the same time, a transfer is initiated through SCSI channel #2 to drive C. When drive A disconnects, SCSI channel #1 initiates a second transfer to drive B. When drive C disconnects, SCSI channel #2 initiates a transfer to drive D. With this overlapped, multithreaded operation, it is possible to utilize nearly 100 percent of the disk drive bandwidth (exception - seek time losses).

CONCLUSION

Conventional recording methods do not adequately support real-time flight-test analysis because they lack random access characteristics and are not directly computer compatible. COTS SCSI peripheral devices can record today’s high-speed telemetry streams when they are coupled to an intelligent controller that is capable of effective data management. COTS peripherals provide the best cost, performance ratios, and intersystem connectivity because of their widespread use in the personal computer marketplace. Modified recording techniques can decrease storage requirements and increase available recording bandwidth.