MICRO-COMPUTER/ARRAY PROCESSOR SYSTEM DESIGN FOR ACTIVE SONAR SIMULATION

T. P. Magnani
C. J. Paroskie
Link Simulation Operations, CAE-Link Corp.
Silver Spring, MD

ABSTRACT

The active acoustic simulation was implemented for the AN/SQS-53C Hull Mounted Sonar System using simulation models and signal generation algorithms completely implemented in software and executed on general purpose array and data processors. The software requirements are hosted by a VME-bus based system comprised entirely of commercial-off-the-shelf units featuring 60 MFLOPS/3 MIPS of processing power, over 21 Mbytes of memory, standard interfaces, a real-time multi-tasking operating system, and spare provisions for future growth. The system is a common modular design having potential for reuse in the simulation of other acoustic and non-acoustic sensors, of multiple contact/multiple sensor systems requiring relative positioning, and of many other computationally intensive software systems.

INTRODUCTION

The Active Sonar Data Generator (ASDG) is the most recent of a continuing series of signal generation devices used in the simulation of Navy sonar systems. This system is shown in Figure 1 as part of the complete AN/SQS-53C simulation system. It generates real-time, complex samples of active ambient noise, reverberation noise, and up to 50 contact echo processes, sums these components, normalizes the results, and sends them in message format to the AN/SQS-53C tactical processors for up to 125 independent receive beams (Figure 2).

It bears mention apart from its predecessors because of the method chosen for its implementation. This design approach represents a divergence from earlier dedicated hardware approaches in that its signal generation algorithms are all implemented in software executing in a real time multi-processor system. This software signal generation approach was chosen for two reasons.

The first reason is that the micro-computer system products currently on the market are excellent candidates for fulfilling the ASDG processing requirements. These products boast high processing speed, high memory density, low cost, backplane compatibility, and extensive development software.

The second reason is for life cycle costs considerations. Because of the evolutionary behavior seen in the operational system, the inherent maintainability and expandability of our "all software model" will reduce the cost and complexity associated with simulation enhancements resulting from operational system improvements.

The ASDG is dedicated to a single purpose; the AN/SQS-53C active simulation. The ASDG hardware, its accompanying operating system and its support software, however, form a basis on which other computationally intensive, software based systems can be built.

ASDG SYSTEM DESIGN

As a significant component of the overall trainer, the ASDG is integrated into the trainer between the main simulation computer complex (SIMCOM) and the tactical processors. The SIMCOM provides control and coordination information such as sonar configuration, instructor selected environmental data, platform dynamics/position keeping and event timing. The ASDG accepts this control data and uses it to develop the waveforms required to simulate the sonar's received signals.

The ASDG architecture is derived from specific AN/SQS-53C requirements. The AN/SQS-53C has three, possibly simultaneous active operating modes (two search, one track). The two search modes are independent of one another. The track mode is either slaved to one of the search modes or is operated in a stand-alone mode. The partitioning of signal generation tasks in the ASDG is matched to the operational modes. One search mode is handled by one array processor (AP), the other search mode is handled by a second AP, and the track modes are handled by both. A third AP provides spare processing capacity for expansion.

The initial task partitioning allocated one mode per AP. However, for the track modes slaved to search modes, search mode resources are required (e.g., their ping inventories). These portions of track are more efficiently handled by the search APs performing track processing in series with their search processing. The stand-alone track modes (i.e., those with their own unique ping inventories) are allocated to the first search AP because search processing is disabled during these times, thereby freeing all the AP resources for track processing. This repartitioning demonstrates the ease and flexibility of the software approach to accommodate change.

Two types of signal processing architectures were considered to implement the ASDG, one based on delay memory (DM) and another on data required at the receiving sensor. The DM approach generates echoes at the target and stores the echoes in a time DM to simulate propagation delay. Signals are then processed to match the geometry as they arrive at the receiver. There are two advantages to this approach. First, if multiple sensors are used, (e.g., bistatic hull array, towed array, or multiple sonobuoy systems), the target radiated/reflected signal need only be generated once, and stored in each sensors DM, rather than regenerated for each sensor. This is not the case for the AN/SQS-53C sonar. Second, the computational load to process the echo for ocean media and sensor aspect/directionality effects can be distributed over its propagation time.

The disadvantage of the DM approach is the amount of memory required. Consider for example a hypothetical second convergent zone (80 nmi.) system. Each target must have a 100 sec (one way propagation time) DM for each aspect related radiator (highlight). A single 5 highlight target would require 10 MBytes of memory at a 5 KHz sample rate

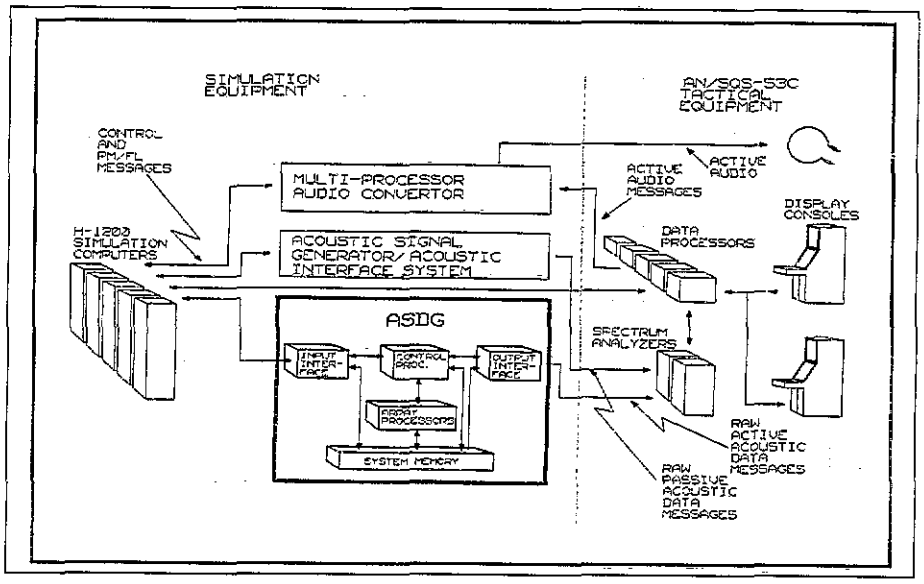

Figure 1. AN/SQS-53C Simulation System Block Diagram

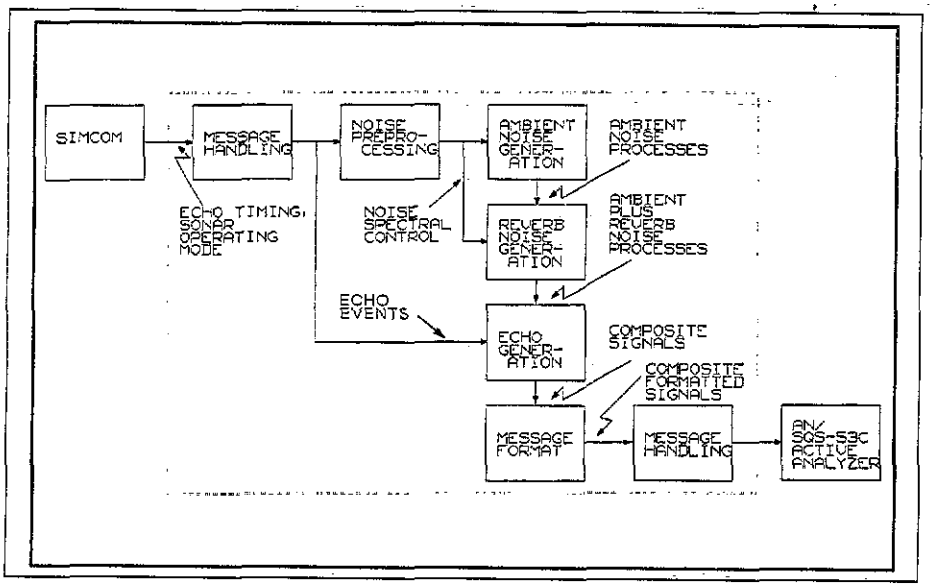

Figure 2. ASDG Function Block Diagram

using 32 bit words. A 50 target system would require 500 MBytes of memory, resulting in a large and expensive ASDG.

The sensor architecture (SA) approach forms signals directly as they are needed at the sensor, trading off memory for the additional processing power needed to regenerate the echoes for each receiving sensor. The advantage of the SA is that since the signal is at, or near, the receiver when it is formed, the geometry and receiver dynamics are known, allowing highlight sequencing and Doppler compensation during generation. The disadvantage is the processing requirements for worst case (simultaneous reception of target echoes) scenarios results in inefficient use of the APs; high peak loading, but significant overall idle time.

A hybrid DM-SA approach is utilized by the ASDG. A short DM, corresponding to a few seconds prior to arrival at the receiving sensor, was employed. Platform orientation and dynamics errors can be accurately estimated for surface ships several seconds in advance. Therefore the advantages of both the DM and SA approaches can be exploited, while minimizing their disadvantages.

The algorithms to implement this design utilize various combinations of time and frequency domain techniques to provide efficient high fidelity simulation signals. Processors that can efficiently execute FFT/IFFTs are used for these signal processing applications.

Just as the system design is driven by the sonar simulation requirements, the selection of processors and other hardware were selected to efficiently and effectively implement the design.

HARDWARE SYSTEM DESCRIPTION

The ASDG has the following hardware resources:

Primary/Control Computer System

A single board micro-computer containing the MC68020 CPU is the ASDG's primary computer; and is accompanied by two 512 kilobyte local memory boards and two two-Megabyte global memory boards. The MC68020 micro-computer executes the real-time multi-tasking operating system and system monitor (pSOS and pROBE, respectively), executes the active sonar applications models, services all interrupts, and initiates and controls signal generation operations in the APs, Input/Output (I/O) operations in the interface boards, and data acquisition operations in the Digital-to-Analog (D/A) board. A separate System Controller board performs prioritized bus arbitration to grant system bus access to the highest priority user.

Interfaces

Information enters and exits the ASDG via two interfaces, a DR11W Interface board, and an NTDS Interface board. The DR11W board links the ASDG to the SIMCOM for loading ASDG control information. The NTDS board links the ASDG to the AN/SQS-53C tactical equipment for transferring simulated active acoustic information out of the ASDG.

Arithmetic Processors

The ASDG has three APs which are its signal generation engines. They generate beamformed ambient noise, reverberation noise and contact echoes for the sonar mode of operation keyed in by the AN/SQS-53C operator. Each AP consists of a 3 board set plus memory (2 or 8 MBytes each). They each run at 10 MHz and use a Single Instruction-Dual Data architecture for an effective 20 million floating point operations per second (MFLOPS) per AP. Each AP can perform a 1024 point complex FFT in 2.2 msec.

Data Acquisition

A Digital to Analog (D/A) Converter provides a means for the simulated digital acoustic data to be monitored in its analog form. It receives, under software control, buffers of digital data and the data's sample rate, and performs a 12-bit two's complement digital to +/- 10 V p-p analog conversion.

Each component of the above ASDG has been chosen from readily available commercial off the shelf products that are sufficient to meet the computational, memory and I/O requirements for high fidelity signal stimulation of the operational equipment. This is also true of the system bus.

System Bus

The ASDG hardware resources center around the VME-bus, Rev C.1, backplane architecture. The VME-bus features a rated 40 Megabytes per second (MByte/sec) asynchronous data transfer bus, direct addressing of 4 Gigabytes (GBytes), multiple bus masters controlled by a centralized bus arbiter, and a seven level prioritized interrupt structure. As an accepted industry standard, the VME-bus is well supported with second and third source products. It will also be upward compatible with the proposed 3.2 GByte/sec Futurebus. These features make the VME-bus an excellent global/system bus choice for real-time, multi-processor applications!

Subsystem Busses

Three subsystem busses augment the VME-bus, reducing its workload. The MC68020 micro-computer uses one subsystem bus to make program fetches and local data accesses, and a second subsystem bus to make data accesses to global memory. These two busses allow the CPU to operate without relying on the VME-bus for its data and instructions. In fact, VME-bus usage by the CPU is only for control communications with the APs, interfaces, and D/A converter.

Similarly, the ASDG's APs each contain a subsystem bus to link their processors, memory, and interface controllers into stand-alone computer systems. They use the VME-bus only for initialization, control and status communications, and for outputting the simulated active acoustic signals.

Figure 3 shows a functional layout of the ASDG boards and their bus connections.

Packaging

The twenty-seven ASDG boards are packaged in two standard twenty slot drawers, each containing a VME-bus backplane. Fifteen of these boards communicate over the VME-bus, and these occupy the first drawer, with five slots available for expansion. They are grouped together to eliminate the need for drawer-to-drawer VME-bus communications (which requires additional hardware and software resources). The remaining twelve boards occupy the second drawer, with eight slots available for expansion. These are memory and arithmetic boards for the three APs (the control and interface boards for the three APs are in the first drawer), and draw power only from the VME-bus backplane. They are connected via their respective subsystem busses to their accompanying AP boards in the first drawer.

SOFTWARE SYSTEM DESCRIPTION

The ASDG software system uses a master/slave arrangement to control and synchronize programs in its processors. The MC68020 micro-computer is the master; the APs are the slaves.

Software in the MC68020 micro-computer is written in two languages; Fortran 77 for the simulation models, and MC68020 assembly language for the device drivers, interrupt handlers, and operating system. pSOS, the real-time, multi-tasking operating system kernel, is at the center of the MC68020 micro-computer. Accompanied by the real-time executive, it can schedule simulation programs to run at 20, 10, 5, 4, 2, and 1 Hz or On-Demand. The ASDG has fifteen simulation programs, three device drivers, three interrupt service procedures, and numerous configuration tables in addition to pSOS. Adding programs, drivers, and service routines is accomplished by loading them into available memory and making appropriate additions to the configuration tables.

The APs run completely asynchronously to the MC68020 micro-computer. They are loaded with their executive and their signal generation subroutines during initialization, and execute I/O and subroutine commands sent by the MC68020 micro-computer during real-time.

Program Description

Figure 4 provides a breakdown of the main software modules in the ASDG. Most programs execute in the primary control computer. Duplicate copies of some subroutines are stored in each of the two APs to accommodate parallel and independent sonar mode operations. Figure 5 demonstrates the functional flow and resource allocation for these modules. Computationally intensive tasks are executed in the APs, such as FFTs, while more generic tasks, such as table look-ups, are allocated to the primary computer. This allocation provides an effective match of tasks to each processors architecture/strengths. Each module is described briefly below.

- Initialization loads the APs simulation software, initializes the ASDG database and creates operating system software interfaces.

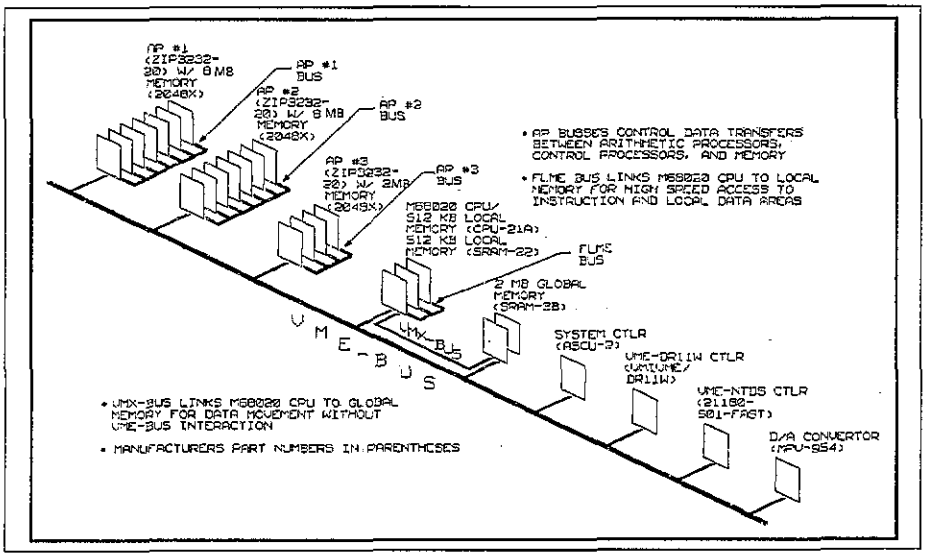

Figure 3. Active Sonar Data Generator Block Diagram

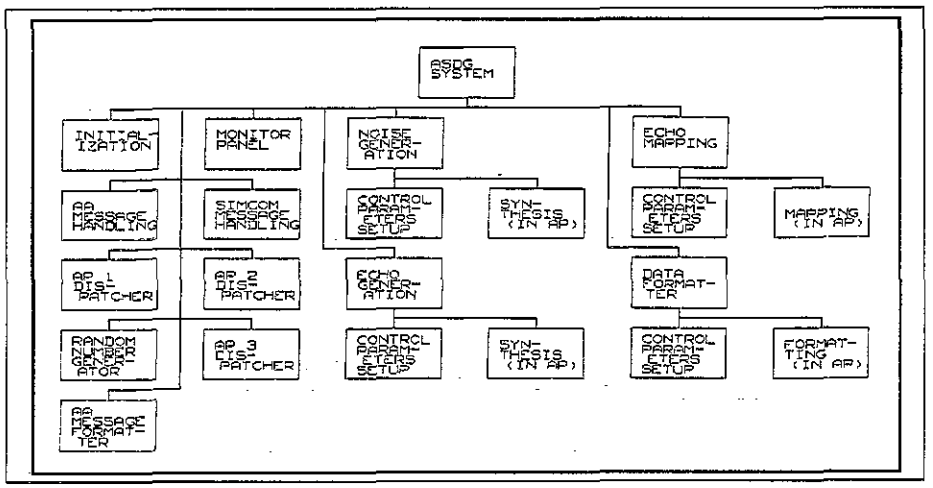

Figure 4. AN/SQS-53C ASDG Software Hierarchy

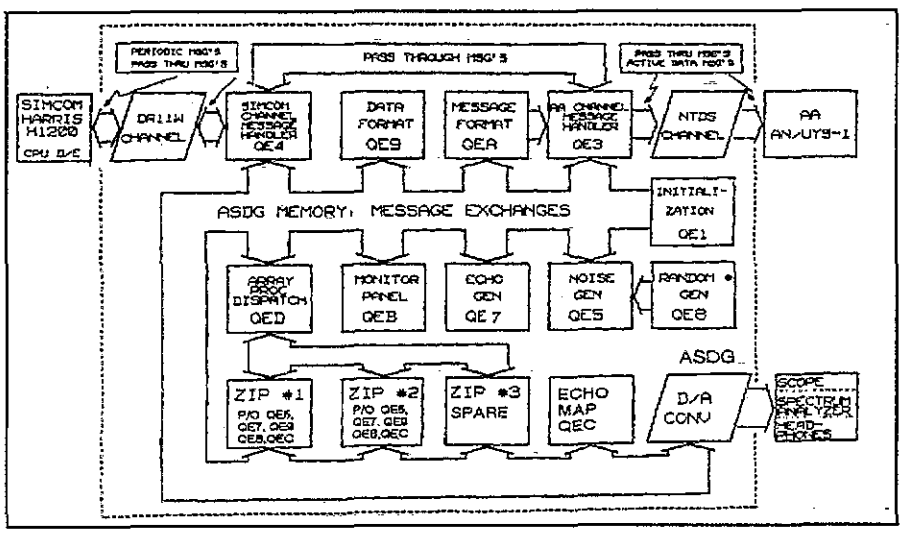

Figure 5. ASDG Functional Flow Digram

- The Active Analyzer (AA) Message Handler controls the output to the tactical equipment.
- The SIMCOM Message Handler controls input from/output to the SIMCOM and posts messages when certain buffers are received.
- Noise Generation creates ambient noise (ocean, own ship, contact broadband and modulated contact noises) and reverberation (surface, bottom, volume and pinnacle) for each active receive mode.
- Echo Generation forms echoes from up to five highlights per target, adjusts the echoes for target and sensor Doppler, and amplitude scales the echoes to match the sonar equation.
- Random Number Generator creates Gaussian distributed random numbers for the ambient and reverberation generators.
- Data Format converts each active receive mode's composite acoustic signal buffer to output message format.
- Message Formatting consolidates the formatted acoustic data, non-acoustic data and receive gate information into messages for output to the tactical equipment.
- Monitor Panel converts complex composite single beam single mode signals from floating point to fixed point format, and outputs the data to the D/A converter.
- Echo Mapping maps the echoes into appropriate temporal locations of the noise buffers.
- The AP Dispatchers control the input from/output to each AP.

The ASDG utilizes approximately 8000 lines of code (LOC); 6000 for modeling in the primary computer and 2000 in the APs. Although this represents a considerable coding effort, it is relatively small compared to the overall trainer software and is comparable to the initial cost of implementing the system design in hardware. Testing and correcting/modifying this code is considerably less expensive than altering hardware, due in part to the commercially available software development and support tools.

Synchronization and Data Flow

In a multi-processor system, inter-processor synchronization and data transfer require special considerations. We used interrupt driven synchronization methods to schedule AP operations and to perform I/O operations across the interface devices, eliminating unnecessary polling by these processors. This reduced both the processors' execution overhead, and the VME-bus's.

Our data transfer design utilized multiple-buffering techniques. Here, data areas a replicated two, three or more times their normal size, and the operations performed on each copy vary with time. This "pipelining" effectively permits multiple, simultaneous operations to be performed. Data integrity is assured because, at most, only one operation is being performed on each copy of the data area.

DEVELOPMENT SYSTEM DESCRIPTION

Numerous standard on-line and off-line development tools simplify the software design cycle. The off-line tools run on VAX computers and PC's, and include editors to create the source code, signal generation libraries, cross-compilers and cross-assemblers, cross-linkers, simulators, and loaders. Figure 6 shows the development facility and its link into the Simulation Computer Complex.

Programs for the MC68020 micro-computer are developed on a VAX. They are written in Fortran 77 or in M68020 assembly language, and are ultimately cross-compiled, cross-assembled, cross-linked, and loaded into the ASDG. Identical VAX and M68020 word sizes and similar compilers allow some module and subprogram level testing and integration of the MC68020 micro-computer's Fortran 77 programs to be performed on the VAX.

Programs for the APs are developed on a VAX or on PC. They are written in ZIP/C, a language unique to the ZIP APs, then cross-assembled and ultimately cross-linked and loaded into the ASDG. A special linking can be performed to allow

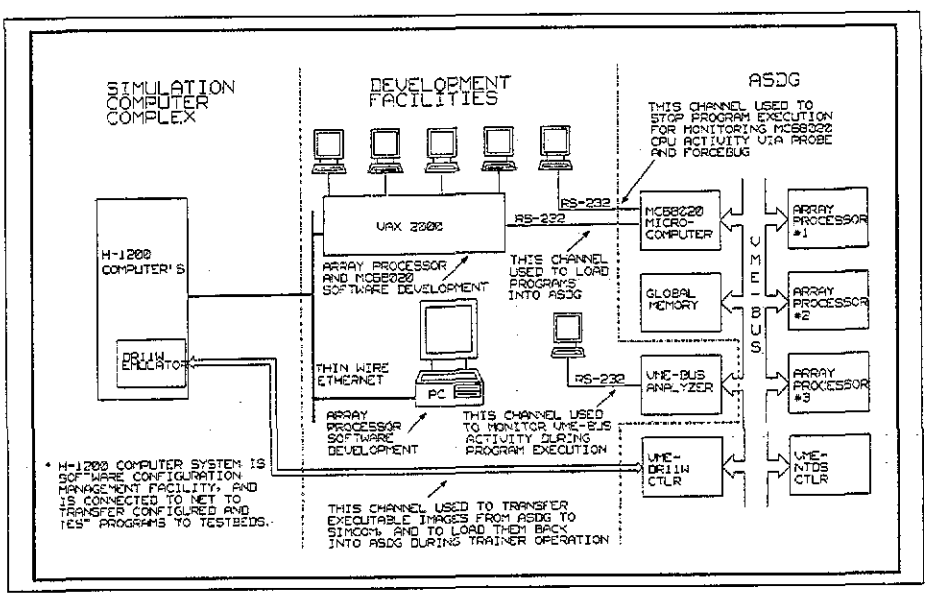

Figure 6. ASDG Software Development Facility

these programs to be debugged on the VAX or PC resident ZIP simulator before loading into the ASDG. The simulator features single-stepping, breakpoints, register and memory monitoring/modifying and indicators to determine if an algorithm's implementation makes the most efficient use of the AP resources.

The on-line tools are pROBE, which is a M68020 resident monitor, and a VME-bus analyzer that passively captures VME-bus activity for display on a maintenance terminal. pROBE provides access to system memory for display and modification, sets breakpoints, emulates certain pSOS commands, and profiles program execution. The VME-bus analyzer allows non-invasive, real-time monitoring of VME-bus communications and can be configured to start or stop analyzing on any VME-bus event (e.g., any address or data pattern, interrupt, bus request, bus grant).

An example of the benefits of the VME-bus monitor occurred as a result of bus priority assignments. The original design assigned bus arbitration levels with the AP I/O highest to prevent slowing these number crunchers down, followed by the primary computer I/O and external I/O. However, after analysis of the VME-bus loading utilizing the above monitor, it became apparent that, if trainer requirements evolved (i.e. additional beams were required) or during stressed situations, the external I/O could saturate and stall real time execution. Since the hardware is configurable via software control and board level jumpers, the priority levels were easily changed to place primary computer I/O and SIMCOM external I/O highest, followed by NTDS external I/O and AP I/O. This software development tool was responsible for identification and elimination of the external I/O bottleneck.

CONCLUSIONS AND COMMENTS

The ASDG demonstrated the feasibility of utilizing readily available COTS products on an industry standard bus to implement an all software acoustic signal generator (ASG). These techniques can be expanded to passive ASGs and potentially to non-acoustic generators. Software development tools are available that help speed initial software development, debugging, testing, performance optimization and bus loading analysis. The all software signal generator can now be efficiently and effectively implemented in a size and cost competitive with customized hardware systems.

Todays technology products from the same AP vendor are four times more powerful computationally, and are still software compatible with most of the original AP programming language; although code modifications would be required to match and maintain efficient use of the new processors architecture and bus structure. In fact, all 18 6U AP boards can be consolidated into a single 9U VME board controlled from an equivalent MC680X0 processor for about the same cost.

With the advantages obtained in development and life cycle cost, due to the versatility to adapt to changing requirements and system modifications, the all software ASDG is now a practical alternative to custom designed hardware generators, with potential for substantial long term savings.

ABOUT THE AUTHORS

Tim Magnani is a senior staff engineer at CAE-Link Corporation, Link Simulation Operations in Silver Spring, Md., where he specializes in underwater acoustic modelling.

He has been with Link for ten years, where he has designed and implemented several acoustic systems which model both underwater phenomena and specific sonar platforms using distributed processing architectures. His interests include the refinement of system modelling algorithms to exploit new advances in computer and software technology, and the design and implementation of large-scale digital signal processing systems.

Tim received his BSEE degree from Virginia Tech in 1980, and his MEE degree from Catholic University in 1986.

Chuck Paroskie is a staff scientist at CAE-Link Corp., Link Simulation Operations in Silver Spring, Md., where he specializes in advance software techniques for underwater acoustic signal generation.

He has been with Link for three years, where he has been the principle investigator for the all software acoustic signal generator (SASG) R&D program. Signal processing architecture and algorithm development, and processing performance optimization are among his specialties. Previously he was program manager at NUSC/NL for a technology based all software underwater communication system program.

Chuck received his BSEE degree from Northeastern University in 1973, and his MSEE degree from University of Connecticut in 1979.