A LOW-COST/HIGH PERFORMANCE SENSOR SIMULATION: THE NEXT GENERATION

John Burkley
Andrew Gurcak
Loral Defense Systems-Akron
Akron, Ohio

ABSTRACT

This paper describes a real time Sensor Simulation system that utilizes an array of processors organized in a fine-grain multi-instruction multi-data stream (MIMD) computer architecture. The application described here is for a multi-mode radar simulation. The software was developed using the structural model with coding in Ada. The methodology for implementing a radar simulation on this architecture, database storage, and software development approaches will be discussed.

INTRODUCTION

Simulation of new sensors, including synthetic aperture radar and imaging sensors, demands high fidelity imagery at increased resolution. Attempting to meet the requirements of these systems simply by brute force increase of processing power and data storage leads to unwieldy increases in database storage and real-time computing requirements. Future training systems will be both deficient and more expensive unless affordable alternative solutions are found. These new systems must also be highly reliable and offer reduced lifecycle costs. To meet these challenges the following must be addressed:

1. New computer architectures offering increased performance at lower cost.

2. New sensor simulation architectures that take advantage of the latest processor and memory technology to meet computation and on-line storage requirements in a cost-effective manner.

3. New database concepts and effective data compression/expansion techniques.

4. Innovative algorithms that allow processing of all sensors on a common platform with a common database.

5. Software implementation using object-oriented design techniques and coded in Ada.

The Loral NOdal Sensor Simulator (LNOSS) described in this paper meets these challenges. It consists of "n" RISC processors (Intel 80960MC) organized in a linear array and programmed using Ada. The radar simulation problem was chosen as a test case to prove this design's applicability to generic sensor problems. The focus of this paper is a description of this design and a solution to the radar simulation problem. The algorithm supports data compression while maintaining full database information and fidelity. The software was designed using the structural model with all code in Ada. We believe the techniques described in this example are also applicable to higher rate EO/IR sensors and 3D laser sensors.

ARCHITECTURE

We performed an architecture study to recommend advanced computer architectures for sensor processing. This study found that the radar simulation problem has inherent parallelism and that only minimal computational node interaction was required. It found that the work function per database element varied significantly. Finally, it found that a global communication between nodes would be required. These results suggested that a fine-grain multi-instruction, multi-data stream (MIMD) computer architecture organized as a linear array would be a good fit for this class of problem. Each node in the array consists of a RISC processor with sufficient processing and memory capacity to support execution of the full sensor problem on a limited set of data in real time. For inter-node communication we chose a Time Multiplexed bus using a token passing approach. This minimizes hardware and has sufficient bandwidth for medium size networks. A VME bus I/F is also provided for global communication with the system controller and database update from disk.

DATABASE CONSIDERATIONS

The foundation of the imaging system is the gridded database whose spacings are tuned to the resolution requirements of the display device itself. The spatial frequency of the grid sampling is dictated by the "information content" (i.e., pixel) on the screen. The minimum information content is that required to just distinguish or, in visual terms, resolve two closely spaced high contrast point radar targets on a pixelized display. The thrust then becomes arriving at that minimal set of informational parameters for an area of terrain and features which, when formed into a grid unit (datapost), can be processed by the real-time software into the range-azimuth display area covered by a single pixel. Each radar display represents a collation of pixels for a particular range, thus implying the formation of a set of databases, each designed for a particular range. The on-line digital database is a collection of radar information that is stored in logical planes corresponding to each range (resolution). For the entire gaming area, each plane is a set of disk storage blocks, modified appropriately for the sampling integerizations induced by latitudinal variations in geometry. A critical feature of this form of on-line database organization is that processing is automatically normalized for all imaging work, i.e., the real-time system is always processing the same number of dataposts for any resolution, thus reducing the processing capacity required.

DATAPOST CODING

The radar-significant elements of each individual sampling square are encoded into the digital radar landmass system (DRLMS) datapost. The function of the datapost is to convey to the radar processing system the amount of radar energy that a terrain (including features) square reflects back to the aircraft's antenna. A single average reflectivity assigned to a terrain square is not sufficient to do this because the actual reflectivity is dependent on many terrain and aircraft sensor related factors. The primary topographical factors governing radar behavior are the physical composition of the terrain square and its orientation with respect to the radar source. Physical composition includes the general topographical makeup and the makeup of any structures that may be located within that square. For any (instantaneous) fixed position of the source, the terrain height and slant range, with respect to the source, determine the relative radar illumination angle. If there are structures on the terrain square, their height, shape, distribution over the square, and orientation all determine the relative illumination angle. Structures frequently have flat surfaces that exhibit very strong reflectivity at nearly perpendicular illumination angles, and little at other angles. For this reason, the model operates with base reflectivities over the broad range of grazing angles, with special augmentation for those reflectors that exhibit pronounced specular or low-grazing-angle reversal effects. Each datapost is divided into descriptive fields, including terrain and feature height, reflectivity, feature character and directivity, and special orientation, shadow, and coverage codes.

ALGORITHM DESCRIPTION

The basic algorithm used to implement a radar simulation is based on a pipelined station concept developed for the F-15E DRLMS in which the problem is broken into nine steps or stations (see Figure 1). The following paragraphs describe each stage of the software pipeline:

1) System Controller: Although not part of the radar sequence proper, this task determines whether the node owns a particular radial, and monitors the loading of the database memory needed for the picture.

2) Radar-Polar: This task forms the string of dataposts from the nadir to range from the appropriate resolution plane into a "radial," which becomes the basic processing unit for the remainder of the algorithms.

3,4) Radterp/Crossterp: This process unpacks

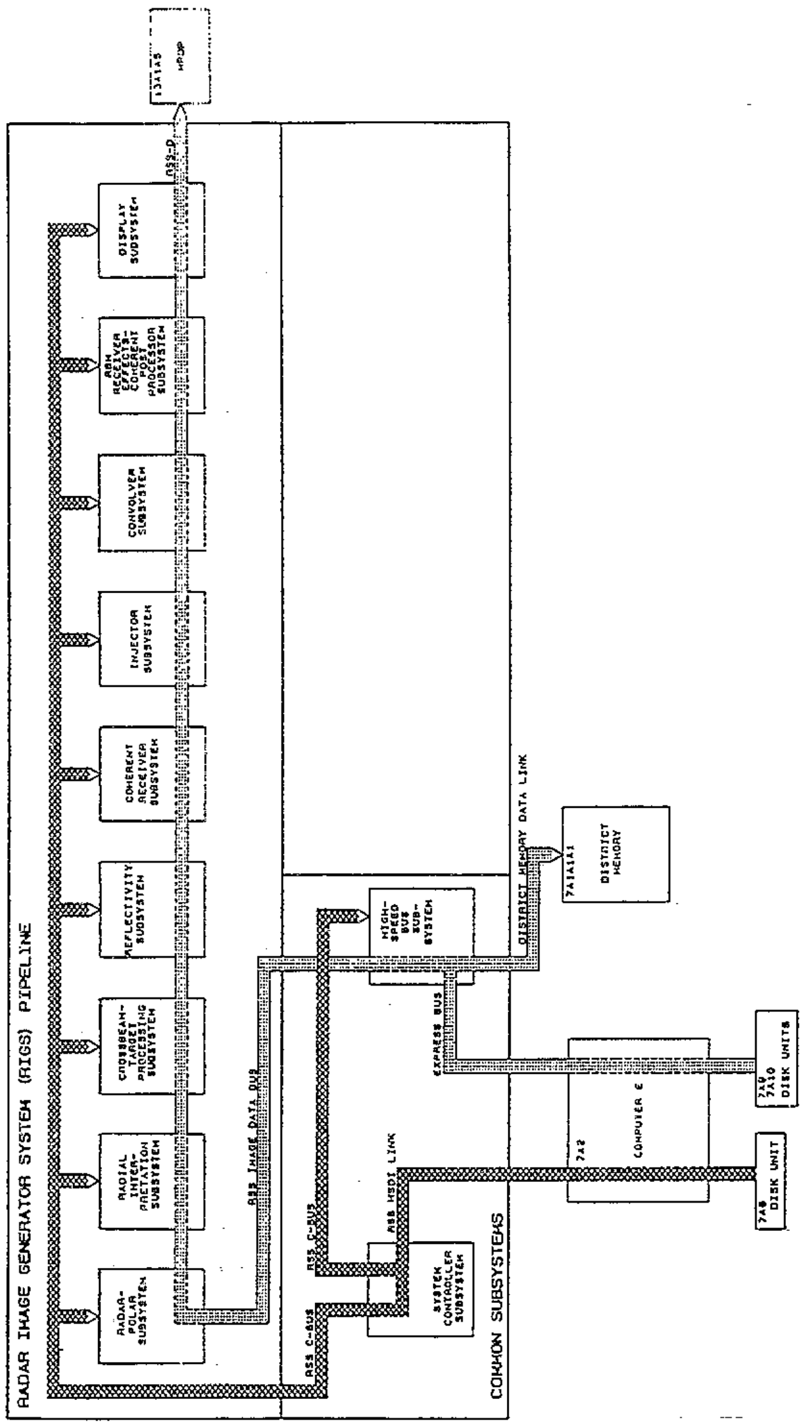
RADAR IMAGE GENERATOR SYSTEM (RIGS) PIPELINE
RADAR-POLAR SUBSYSTEM
RADIAL INTER-PRETATION SUBSYSTEM
CROSSBEAM-TARGET PROCESSING SUBSYSTEM
REFLECTIVITY SUBSYSTEM
COHERENT RECEIVER SUBSYSTEM
INJECTOR SUBSYSTEM
CONVOLVER SUBSYSTEM
RBM RECEIVER EFFECTS-COHERENT POST PROCESSOR SUBSYSTEM
DISPLAY SUBSYSTEM
RSS-D
13A1A5
HPDP
RSS IMAGE DATA BUS
RSS C-BUS
RSS C-BUS
RSS HSDI LINK
SYSTEM CONTROLLER SUBSYSTEM
HIGH-SPEED BUS SUB-SYSTEM
EXPRESS BUS
DISTRICT MEMORY DATA LINK
7A1A1A1
DISTRICT MEMORY
7A2
COMPUTER E
7A9
7A10
DISK UNITS
7A8
DISK UNIT
COMMON SUBSYSTEMS

Figure 1 — F-15E DRLMS Pipeline Basic Block Diagram

the datapost words into fields for further use. It expands the feature height data and generates various along-range effects, as well as signalling the possibility of low-grazing-angle effects processing. It also buffers several radials to discover cross-range effects. It processes cross-range terrain-tilt, presence and orientation of cardinal reflectivity effects, and the interpolation of cross beam terrain height.

5) Reflectivity: This process induces a multitude of radar effects into the image. It determines the radar backscatter of landmass elements for terrain, and feature tops and sides. By use of the separability of return calculations, it computes layover, moving target displacements, and height displacements as required. In terms of aspect effects, it performs horizontal and vertical aspect calculations for terrain and features, taking into consideration such non-Lambertian variations as low-level/grazing angle effects, plateau/ medium angle effects, and specular/high grazing angle effects so that basic slope, onset, leading edge, and low-level effects are directly calculated. In addition to setting driving mechanisms for INJECTOR, it also handles imaging geometry attenuations with respect to range and vertical antenna patterns.

6) Injector: The injector implements efficiency in spatial domain processing, the system impulse response function for each significant elementary return. This process begins by calculating the centroid displacements of features resulting from datapost expansion. From the expanded dataposts, such map distortions as range, range rate, and cumulative system mapping errors are determined and placement adjustments made. The spread function yields the basic aberration for range and azimuth focus, the range and azimuth displacement from sidelobe levels, and the mainlobe loss and sidelobe increase from energy conservation and amplitude management. Trace aberrations involving range and azimuth walk and layover are also calculated. Extended special forms for range and azimuth ambiguities and systematic phase modulations are also performed. Speckle, clutter, and return fluctuations for PRF simulation are developed by partially random spread addressing formulations.

7,8) Convolver/Receiver: Beamwidth convolution and effects due to receiver noise, jamming and ECM/ECCM and feedback sampling are performed here.

9) Display Processing: Display formatting and processing are performed both on individual radials and on the map ensemble.

The new approach was not to redesign this basic algorithm, but rather to determine how it could be applied to an array of processing nodes. In a pipeline processor all the data is passed between the processors, and each performs a part of the problem. In parallel processing each processor solves the whole problem for a portion of the data. The number of processors required depends on the amount of data needed to be processed rather than the number of steps in the problem. The task at hand is to devise a method for distributing the data so that all the processors are kept busy. We solved this problem by applying the following techniques:

a. Radial Processing Distribution

We determined that the best way to share the work function was to operate on a number of radials at once so that each processor would have the same number of radials on which to operate independent of aircraft position. This was done by eliminating the database "District Memory" as a separate module and distributing the database storage among the processors.

b. Common Node Software

Each node contains the same software for the whole radar problem. Each node runs the same software on its set of dataposts and a broadcast set of initial conditions. Each node processes asynchronously approximating a MIMD processor; this tends to equalize the loading since the work function required to process a given post may vary.

c. Tier I/O

The radar problem has inherent parallelism but also has sequential processing dependent on intermediate values. In the pipeline technique, this intermediate data was added to the data post as it was passed between stages in the pipeline. We solved this problem in parallel by establishing tiers of processing after which each processor would exchange datapost with all the other processors. We determined that three such tiers were required to solve this problem. The processor interface design was optimized to allow this data interchange at a high rate so that the total I/O time was less that 20% of available real time.

d. Display Processing

In the F-15E radar simulator, all display functions were handled by a custom display processor which took the output of the pipeline and performed coordinate conversion display formatting and storage in frame buffers for display. This operation can be handled much more efficiently in parallel. The video memory is partitioned across the processors. Also, the required video sync is imbedded in the video memory, allowing a generic video design to interface with a number of video standards simply by changing software. The display formatting and coordinate conversion is done within the processing node. Video memory is dual ported and scanned as needed by the video display controller.

LNOSS HARDWARE DESCRIPTION

The LNOSS system is a MIMD processor array of RISC processors. A detailed block diagram is shown in Figure 2. The heart of the system is a processor board which integrates an array of four RISC processors, supports a 64Mbyte/second interface bus, and provides VME bus compatibility as shown in Figure 3. The system assumes a VME form factor and has an open architecture which supports commercial off-the-shelf equipment.

DRLMS System Controller - A VME 68030 processor card will perform the GEO/Disk control, target processing, and system controller (SYSCON) functions. This card serves as VME bus master and has a serial RS232 interface for diagnostics and stand-alone operation.

LNOSS Processor - The LNOSS processor board performs the radar imaging tasks in the radar simulation pipeline. It consists of an array of four loosely coupled RISC-based 32-bit processors. The Intel 80960MC processor family was selected as the best of twelve candidates for serving as a node processor in this design because it was a single chip processor and required a minimum number of chips for a node, it had a verified Ada compiler as well as an impressive set of software tools available, and it is one of the JIAWG-approved 32-bit architectures. The LNOSS is built to a VME 6U 220mm form-factor with connectors P1 and P2 compliant to the IEEE P1014 VME standard. Each node provides a standard 32-bit VME port and an AP-port for highspeed (64Mbytes per second) interprocessor communication on available P2 connector pins. Each node includes 512Kbytes of SRAM for temporary storage and zero-wait state program storage, 1Mbyte of Flash EEPROM for processor boot, test programs, and other noncritical program and table storage, and 4Mbytes of DRAM for database, video buffer, and other data and programs as required. The design supports zero-wait state operation from SRAM. The DRAM memory is dual ported to both the nodal processor and the VME, allowing update of global parameters and database in parallel with nodal processing. A special 8-bit microcontroller is included on each PCB for I/O control and test functions. The number of LNOSS processor cards needed varies with the simulation requirements.

Disk Subsystem - The GEO/Disk subsystem provides geographical data to the radar pipelines from the on-line GEO data disk. The VME 68030 processor card prefetches database updates as required by the simulation problem. The GEO/Disk subsystem can support up to eight removable winchester disk drives on a SCSI bus. An 8mm tape drive is provided for database loading and system backup.

Display Interface - The display interface board is a simple frame buffer with a built-in scanner to perform the digital scan conversion. The display board itself has no processing capability; all display functions are done in the processor. The design is capable of supporting multiple images at once. One innovation is the use of imbedded sync in the frame buffer which the scanner reads out as data. This allows the output to be easily changed from one display format to another without changing hardware.

Reliability/Fault Tolerance - Because of its small size and high level of integration, the LNOSS processor has a predicted mission critical MTBF

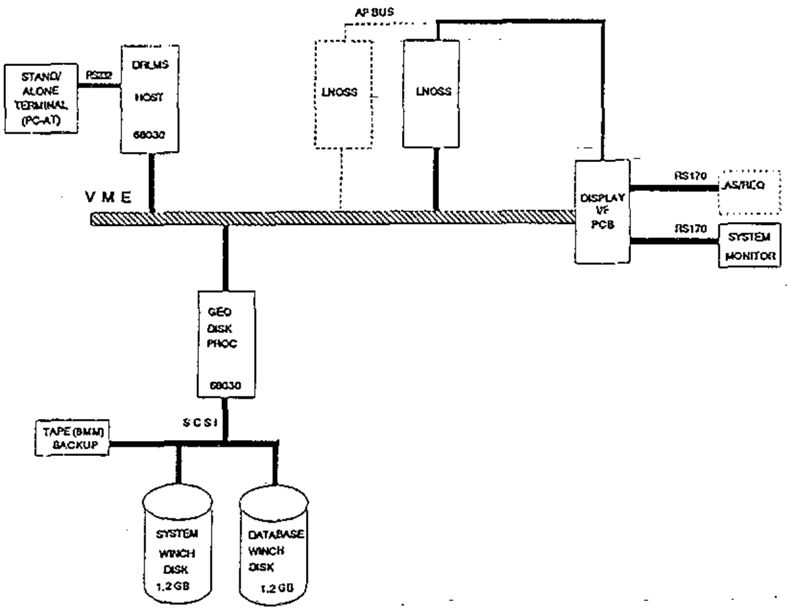

Figure 2 — LNOSS System Block Diagram

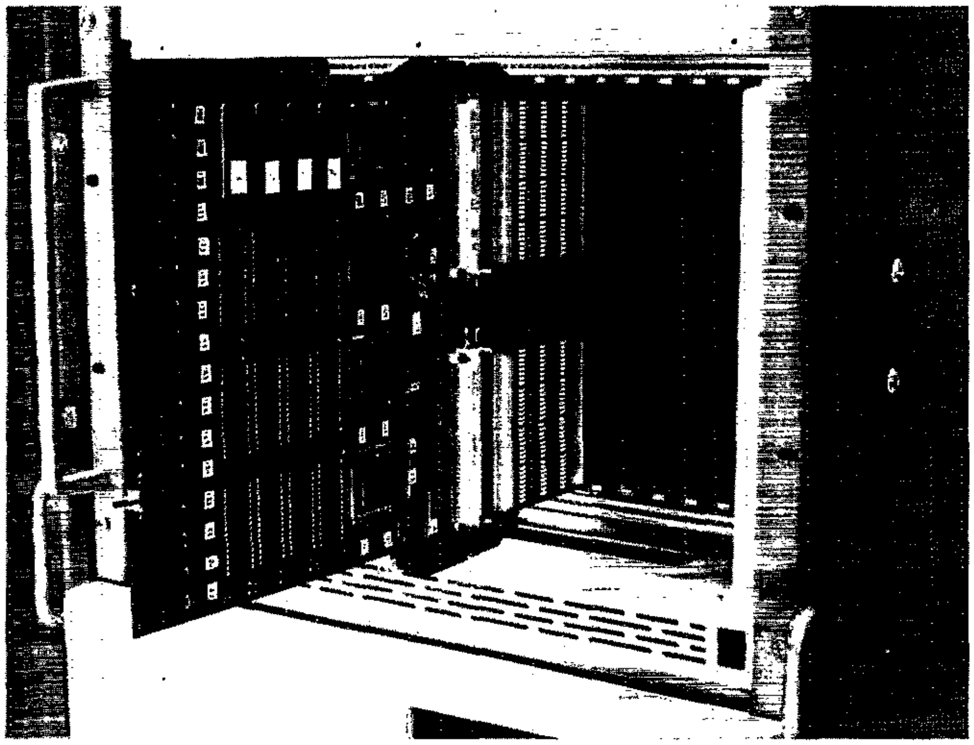

Figure 3 — LNOSS Processor Board

of over 4000 hours for a four-board system. The nature of radar simulation is that it is inherently fault tolerant to data errors. Because of the large amount of data and inherent smoothing operations in solving the radar equation, single bit errors from the disks and data memory do not constitute mission failures. The LNOSS design takes advantage of this characteristic of radar processing in its parallel architecture. The processing of the radar picture is scattered across the processor so that the failure of a single node is analogous to a data fault and does not cause the picture to degrade noticeably. The design of the system uses an on-board 8-bit microcontroller for rapid on-line identification and disabling of failed processors, and thus provides a significant fault tolerant capability.

SOFTWARE DESCRIPTION

The software has been designed to meet the goals of high-precision, real-time radar simulation and the DOD-mandated use of Ada for all real-time applications. These objectives have been met by keeping the fast, image-making portion of the real-time system generic and parameter-driven; all the specific details of beam-shaping, antenna characteristics, and processing effects can be driven by relatively few parameters calculated at a "slow" rate in the system controller. The real-time image-making software runs at maximum speed and efficiency, using Ada code for the LNOSS's Intel 80690MC microprocessors.

The radar simulation is organized along the structural model developed by the Software Engineering Institute under Air Force contract specifically for simulation. A structure diagram is shown in Figure 4. Its basic strategy is to employ a series of generic and specific software objects, with coupling among them minimized. The aim is to produce a "flat" system with as shallow a calling-tree as possible. Objects, both internally and in their interfaces, are based on a common structure or template, preventing software designers from arriving at conflicting decisions on the nature of their programs. It also ensures a uniform ease of documentation both for initial production and for subsequent modifications. It is frame-oriented in the system controller, but runs asynchronously in the LNOSS processor.

The radar image is formed by a multi-step process. The basic strategy is to direct the LNOSS processors to perform a series of ray traces from the instantaneous antenna position to the intersecting point on the earth along an arc from the aircraft nadir to the range limit, with the arcs (radials) mosaicked to form the picture following the antenna's motion. The basic geometry of the map is laid down by the DRLMS host at map initiation, and is continually corrected during the process to ensure natural tracking of non-linear scan rates for shearing and Doppler shift effects. During the map initiation process, coarse target screening is performed to see if any moving targets may be present in, or move into, the current map. The function of the system controller is to supervise the I/O transfers between the trainer and itself, calculate "slow" parameters, and prepare information buffers for the image-making system of distributed microprocessors. The LNOSS scheduler is a locally data-driven format. As the DRLMS host data is broadcast to all LNOSS nodes, each of them selects the radials it "owns" and works on the imaging process, bringing it to completion as the displayed radial.

CONCLUSIONS

The results to date show the capability to provide a demonstratable radar image on a single LNOSS card. A system of multiple LNOSS boards will provide a fully compliant digital radar landmass system programmed in Ada. This system equals the performance of the F-15E DRLMS at an order of magnitude lower cost. We are developing a next-generation sensor processor which has the capability to provide high fidelity sensor simulation at an acceptable cost. We feel this design can be extended beyond radar sensors to EO and IR type sensors.

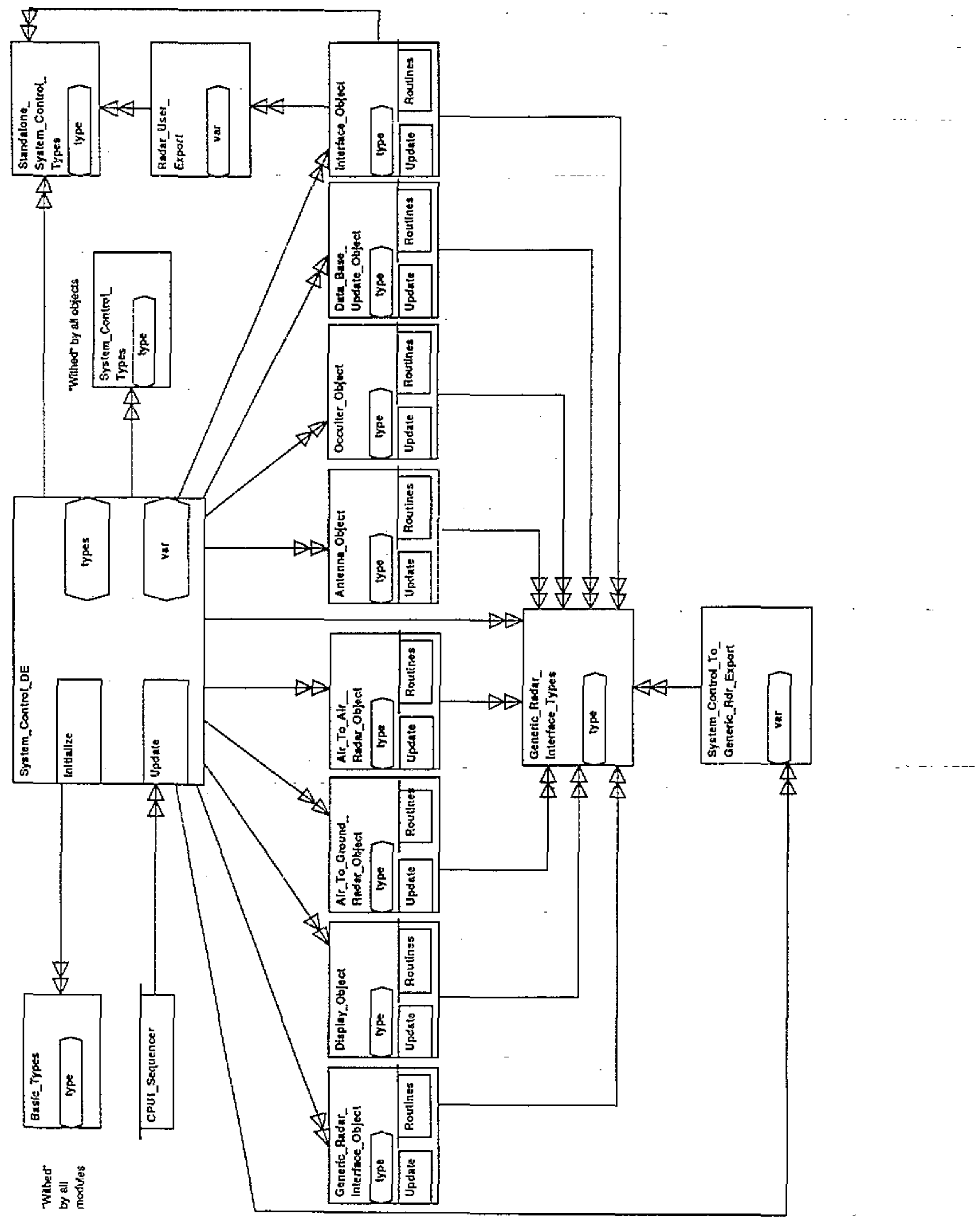

Figure 4 — DRLMS Host Real-Time Software
Top-Level Diagram

ACKNOWLEDGMENTS

The authors sincerely thank all those who have toiled lo, these many years in all incarnations of the Loral DRLMS. The authors especially wish to acknowledge the guidance of Lowell Stoner, without whom none of this would exist.

ABOUT THE AUTHORS

John Burkley is currently a member of the Senior Technical Staff, responsible for Sensor Simulation in the Simulation and Training Department at Loral Defense Systems - Akron. Prior to that, he was the Lead Hardware Engineer for sensor systems on the F-15E WST. He also worked on the Massively Parallel Processor delivered to NASA in 1983. Mr Burkley has a BSEE and MBA from the University of Akron and an MSEE from The University of Maryland.

Andrew Gurcak is the Responsible Engineer for IR&D in Training and Simulation at Loral Defense Systems - Akron. Prior to his current job, he was the Lead Software Engineer for sensor systems on the F-15E WST. Mr. Gurcak has a BS from MIT, an MA from the University of Virginia, and an MBA from the University of Pittsburgh.