ESC-TR-96-118

Project Report
RASSP-5

RASSP Benchmark 4 Technical Description

G.A. Shaw
A.H. Anderson
J.C. Anderson

9 January 1998

Lincoln Laboratory
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LEXINGTON, MASSACHUSETTS

Prepared for the Defense Advanced Research Projects Agency
under Air Force Contract F19628-95-C-0002.

This report is based on studies performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology. The work was sponsored by the Advanced Research Projects Agency under Air Force Contract F19628-95-C-0002.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER

Gary Tutungian

Gary Tutungian
Administrative Contracting Officer
Contracted Support Management

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LINCOLN LABORATORY

RASSP BENCHMARK 4 TECHNICAL DESCRIPTION

G.A. SHAW
A.H. ANDERSON
J.C. ANDERSON
Group 97

PROJECT REPORT RASSP-5

9 JANUARY 1998

LEXINGTON MASSACHUSETTS

ABSTRACT

This document describes the fourth in a series of application tasks designed to measure performance of a process for the rapid prototyping of embedded digital signal processors. The rapid prototyping process is being developed for the DARPA/Tri-Services rapid prototyping of application specific signal processors (RASSP) program. The benchmark task described here involves virtual and physical prototype development for portions of a real-time digital signal processing image intelligence (IMINT) system that performs semi-automated IMINT processing (SAIP). The task addresses hardware and software development issues relating to the high-definition imaging (HDI) and minimum mean square error (MSE) target classifier portions of SAIP. Application details and design constraints are provided in this document, as well as a description of product and process metrics collected to derive measures of product and process performance. The application task and associated performance metrics comprise what is termed a benchmark technical description (BTD).

Note: This is not the official Benchmark-4 Technical Description since material about the high definition imaging algorithm and SAIP scenarios has been removed from the BTD which was given to the RASSP developer. Nevertheless, it should give the reader an accurate understanding of the benchmark task.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. GENERAL

This document describes the technical requirements and deliverables for RASSP Benchmark-4. Benchmark-4 will generate case studies to aid in proliferation of RASSP methodology and tool development. This benchmark will test the RASSP design environment via development of a virtual prototype, hardware and software for an embedded processor capable of processing real time synthetic aperture radar (SAR) images for target detection. The processor is part of an image intelligence (IMINT) system for semi-automated IMINT processing (SAIP).

Benchmark-4 shall be done in two phases. In the first phase, a study of alternative designs shall be carried out. Based on results of the study, an implementation of all, or part, of the system described in this benchmark technical description (BTD) will be undertaken as determined by time and funding constraints. Material for case studies shall be generated in both phases. The Developer's response to this BTD shall focus on presenting a plan for the first phase, and proposals for case study development in both phases.

In this document, Section 2 sets forth the requirements for portions of a SAIP signal processor, including hardware and software development issues relating to high definition imaging (HDI) and mean square error (MSE) target classification. Section 3 describes the SAIP testbed, in which the prototype processor is to be inserted. Section 4 identifies the constituent elements of an associated executable requirement. The executable requirement is described in detail in a limited distribution document [15]. Section 5 describes the metrics that must be collected to evaluate the performance of the RASSP process (e.g., work flow and development steps) and products (e.g., hardware and software) associated with the development of the prototype. In addition to metrics, there is a requirement to document RASSP innovations employed during benchmark execution in the form of case studies. The cost of any case study documentation is not to be counted or charged against the benchmark cost, and similarly for any schedule impact. Deliverables, including process and product documentation requirements, are discussed in Section 6. The form of the response by the Developer to this BTD is described in Section 7.

1.1 INTRODUCTION AND OBJECTIVES

One component of the DARPA rapid prototyping of application specific signal processors (RASSP) program is the execution of application benchmarks by a RASSP Developer; in this case the Advanced Technology Laboratories (ATL) of Lockheed Martin Corporation. This fourth RASSP application benchmark is directed toward the development of a virtual prototype, hardware and software for an embedded processor to replace part of the DARPA SAIP system. The benchmark task is primarily intended to serve as a vehicle for providing a set of case studies that will document elements of the RASSP design environment.

The developer is responsible for establishing cost-effective methodologies and tools for the creation and application of virtual prototypes to the development of embedded digital signal processing systems. The level of detail for function and timing incorporated in the virtual prototype is not expected to extend beyond that of an instruction set architecture (ISA) model of programmable devices running application code written in a high-level source language such as C, except that more detailed models may be required to validate interface and timing constraints. The ISA level of modeling defines the limit of detail expected in the VHDL virtual prototype. It does not prohibit the developer from incorporating more detailed models to aid in risk reduction. However, prior to developing a detailed virtual prototype for a preferred architecture, less detailed modeling and evaluation of alternative architectures must be performed to select the architecture that best meets one of the several sets of performance goals described in this BTD.

This document establishes the requirements for development and delivery of a prototype, real-time SAIP subsystem processor. The RASSP process will be applied to convert a detailed virtual prototype into a physical prototype processor conforming to the requirements of this BTD. An initial step in this process is to develop a set of candidate architectures and perform an architecture trade-off study. Candidate processor implementations must then be examined for estimated development cost, development schedule and life cycle cost. For example, an MSE target classifier implementation using commercial off-the-shelf (COTS) processor boards may be compared with an accelerator implementation that uses custom application-specific hardware. The development and life cycle cost impact of designs involving a Standard Virtual Interface and full compliance with the design-for-test requirements described in Section 2.11 shall be examined. Various configurations must be described that optimize size, weight and power.

Benchmark-4 should be completed within the 10 month schedule and 80 man-month effort constraints shown in Figure 1. Therefore, at least one candidate design must be developed that can be produced in unit quantity within the cost and time constraints of the benchmark. Constraints on size, weight and power are described later in this BTD.

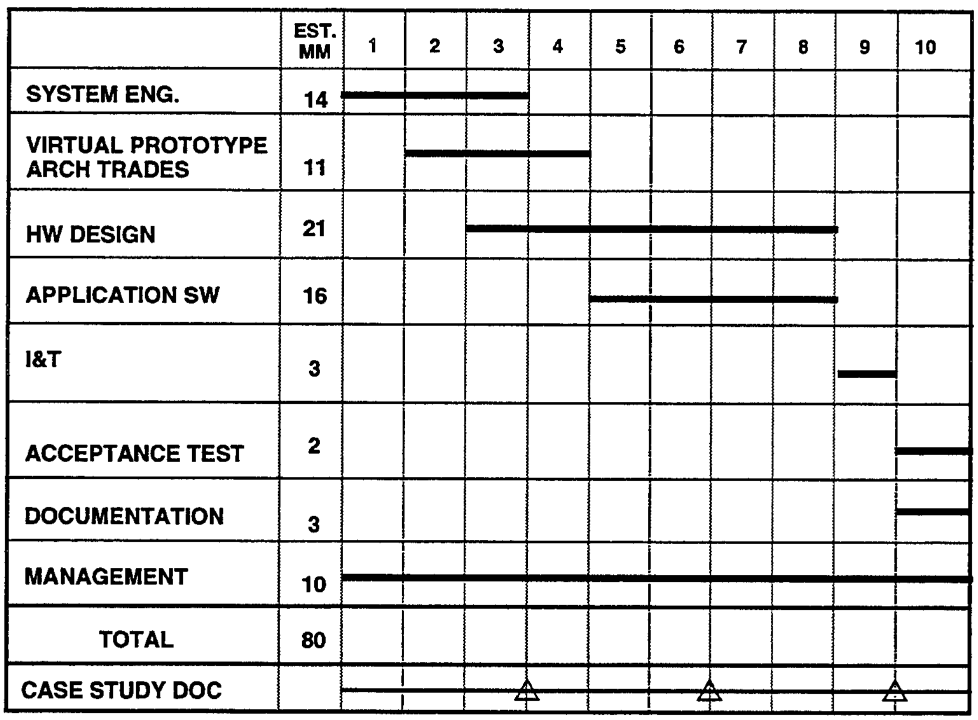

Figure 1. Benchmark-4 effort and schedule estimate.

Any required deliverables or activities not considered to be part of a normal RASSP development cycle must be identified by the Developer and reported separately. For example, the effort estimates in Figure 1 exclude any case study definition and documentation activities. The cost and schedule impact of such deliverables or activities, which shall be reported separately by the Developer, are considered to be outside the benchmark scope for purposes of comparing RASSP to standard practice development. Furthermore, case study definition and documentation activities are a proliferation activity and shall not be charged to the benchmark activity.

Benchmark-4 shall be carried out in two phases. Phase I consists of the first two months of the system engineering task shown in Figure 1, as well as development of VHDL token-based performance models that may be necessary for the initial virtual prototype and architecture trades tasks. In Phase I, the system shall be designed to satisfy the performance requirements and minimize development cost and the cost to build 100 systems. Low-cost maintenance over a life of 15 years and the ability to reuse the design for building systems with higher performance shall be important considerations. An architecture that supports model year upgrades is required. A non-software (i.e., other than a programmable multi-processor) implementation of at least a portion of the system (e.g., the MSE classifier) shall be considered. The level of design detail shall be adequate to support cost and performance comparisons of several designs, and to support detailed planning of the full development of a prototype of the selected system. Case study documentation of Phase I must describe the tools and processes used to develop the architecture trades and cost analyses, as well as the results of the architecture trade and cost analysis studies. Sufficient detail must be presented so that Phase I serves as a stand-alone case study. The design document produced during Phase I shall also include a proposal to prototype the entire system, or a part of it, as determined by Benchmark-4 goals, time and cost constraints. Construction of performance models may be appropriate, but the construction of a complete virtual prototype is not appropriate in Phase I. In Phase II (Months 3 or 4 through 10 in Figure 1), any required hardware and software are developed using a RASSP co-design methodology, I&T and acceptance tests are performed, and final documentation is prepared.

Process and product measurement is a critical part of this benchmark. The Developer shall identify an individual who will have the responsibility for collection of the data described in Section 5. The cost of metric collection shall be separately accounted for, and will not be counted as a development cost for purposes of comparing RASSP to standard practice, although metric collection may be charged against the benchmark.

1.2 SAIP OVERVIEW

An overview of the semi-automated IMINT processing (SAIP) demonstration concept is shown in Figure 2 {p. 10}. The SAIP system resides in the SAIP demonstration van.

SAIP is structured as an advanced concept technology demonstration (ACTD) with goals to provide state-of-the-art battle awareness software to the tactical field commander [1]. The operational goal of the SAIP program is to make imagery a responsive source in providing the commander with dominant battlefield awareness. This information will be derived from the high altitude endurance unmanned aerial vehicles and existing U-2 based advanced synthetic aperture radar system (ASARS) platforms. The gathering phenomenology includes radar and electro-optical (EO) sensors. Synthetic aperture radar will be the first technology to be brought on-line, followed by EO. The program is aimed at the tactical user, providing tactical information quickly to the commander regarding the opposition ground order of battle and missile order of battle. This will be attained while maintaining tactical surveillance capability of other targets of interest. Site monitoring will also be included in the suite of information to be provided to the commander. Site change detection and occupancy will both be available for decision making purposes using the airborne platforms and the sensors noted above.

The technical goals aim at increased image analysis efficiency. This will allow the battlefield commander to have information regarding more territory than ever before, while maintaining the same staffing levels for image analysis. Image analysts will be cued as to where isolated targets are located to allow swift human evaluation of critical targets. To do this effectively, the probability of automatically, accurately detecting a target must be greater than 90% and the false alarm rate must be less than one false alarm for every hundred square kilometers being assessed. The image analyst will receive cueing from search imagery regarding areas that should be examined more closely with higher resolution "spot modes." The successful completion of these goals will enable the battlefield commander to have real-time battlefield visualization at his fingertips for decision making processes critical to winning battles and outmaneuvering adversaries.

The SAIP program brings together many existing technologies into an integrated package. Key programs use interactive target recognition, cluster analysis, template based automatic target recognition, terrain analysis, object level change detection and image-to-image registration to garner the high probability of detection and low false alarm rate required to effectively supplement the image analyst (IA). This is presented to the operator by a robust human/computer interface that allows the analyst to look at full resolution data, a set of viewing and confirmation tools, and interactive model-based target recognition to provide rapid report generation to the battlefield commander.

The SAIP architecture provides state-of-the-art automated exploitation while connecting to existing advanced sensors with existing image processing systems. By enabling copious data to be supplied to image analysts through the SAIP demonstration van, the commander is provided with dramatically increased surveillance capability. The SAIP van supplements image exploitation systems such as the contingency airborne reconnaissance system (CARS) and enhanced tactical radar correlator (ETRAC). The net result is an IA force multiplier for analyzing wide-area imagery in real-time, using existing collection and processing assets. The SAIP van can be located in the theater or configured for remote operations.

As part of the overall United States Imagery System community of exploiters and producers, this ACTD testbed equipment suite will be capable of supporting imagery exploitation systems to afford increased flexibility and capability in satisfying multiple time-sensitive user needs. SAIP applications will be compliant with community processing, storage, retrieval and dissemination standards. SAIP technology will enable the Theater, Joint Task Force and/or Components/Commanders to employ interactive capabilities to meet the anticipated imagery demands of local regional conflicts and military operations other than war. Deployments of SAIP technology will be tailored to meet the imagery requirements in peace, crisis and war. At the conclusion of the ACTD, the testbed will provide residual capability that could be integrated into CARS and ETRAC to support the image analyst through wide area search (WAS) for specific targets and formations of ground forces, identification and characterization of target vehicles such as tactical ballistic missile launchers, and monitoring of activity at fixed sites or small scenes over time. Such capabilities are of interest to a wide range of sponsors, including the US Atlantic Command, Army, Air Force and Naval Reconnaissance Office.

In support of WAS requirements, the system screens strip-map data in real-time. The system will perform automatic target cueing on spotlight images with the same functional objectives as for strip-map. The system will perform object level change detection (OLCD) between the current imagery and the previous imagery from the area being covered. OLCD will be used for both false alarm mitigation and determining evidence of activity. The system will be able to group individual vehicle cues that are close together into a single region-of-interest image with multiple targets.

The system will be able to construct 2-D and/or 3-D geometric models of fixed installations. This feature will assist the IA with change detection, image registration, and monitoring trends and will maintain a history of the target area(s) being examined.

SAIP will be developed in three phases: baseline, enhanced and transition as shown in Table 1. The baseline configuration will be the initial delivery of testbed hardware incorporating SAIP technologies at the commencement of the baseline phase of the ACTD. In the second phase, the technologies will be upgraded, debugged and will constitute the enhanced configuration. The military assessment will commence with delivery of the enhanced configuration to the operational user site. Further technology upgrades will be made before delivery of the transition ACTD configuration. Final evaluation will be completed during the transition phase.

TABLE 1 SAIP Development Phases

BASELINE	ENHANCED	POSSIBLE TRANSITION TECHNOLOGIES
SAR Image/Image Registration	SAR Image/Map Registration	Same as Enhanced
Object-Level Change Detection	Same as Baseline	Coherent Change
Human/Computer Interface	Same as Baseline	Same as Enhanced
SAR	EO/IR	MSI
Interactive Target Recognition	Rapid Target Insertion	On-the-Fly Training
Cluster analysis	Force Structure Analysis	Flexible FSA
Template ATR	Model-Based ATR	Multi-Spectral ATR

The SAIP testbed will be an integrated set of technologies currently being developed by DARPA. The testbed architecture will allow the subsequent insertion of other technologies and modules. The modules to be integrated include force structure analysis and terrain reasoning, detection and classification of equipment/vehicles, and site monitoring. Each module brings a different set of tools to the overall problem of analyzing wide-area imagery data. The tools exchange information about the data so that best evidential reasoning can be applied to potential targets. The tools will focus on the automation of the following key functions: terrain analysis and area delimitation, object detection and classification, elimination of uninteresting objects, detection of changes between images, recognition and identification of specific objects/targets, detection and assessment of groups of objects/targets, recognition of detailed changes at fixed sites or small scenes, advanced methods for IA interaction, automated registration, and traditional analyst tools (image registration, recall of previous results, image manipulation, mensuration and assisted report writing).

Automatic target recognition (ATR) is the cornerstone of the SAIP system, with the baseline system initially incorporating a template-based SAR ATR. The SAIP performance goal of cueing IAs to isolated targets is being achieved through high definition imaging (HDI) techniques. DARPA's moving and stationary target acquisition and recognition (MSTAR) program is developing a model-based ATR as the next generation target recognition system. MSTAR will increase ATR robustness with its ability to address target occlusion and articulation and further reduce the SAIP false alarm rate. The addition of EO processing in 1997 will be addressed by an evolving multi-spectral ATR program currently under development.

A block diagram of the SAIP baseline system ATR processing is shown in Figure 3 {p. 11}. Image registration in real-time for large surveillance images is required to support several SAIP functions. The baseline system will initially incorporate a SAR image-to-image registration capability in support of object level change detection. The enhanced system will provide a SAR image-to-map registration capability to aid in terrain analysis, site monitoring, and force structure analysis. These capabilities will also be expanded to incorporate EO processing in the enhanced system.

Area delimitation provides the ATR with the false alarm mitigation required to achieve the SAIP performance goals. Terrain analysis in the baseline system will identify areas of target delimitation (e.g., tanks will not be found in a lake) in rural areas, with site monitoring addressing delimitation (e.g., tanks will not be found on the tops of buildings) in urban areas. A capability to automate the construction of urban sites is being developed for future inclusion and will be an asset in rapid deployments.

Object-level change detection (OLCD) is a valuable facet of SAIP's false alarm mitigation process. for example, the first time an IA analyzes an image, the resulting annotations may identify detected objects that are not targets (e.g., telephone poles along a road) which were detected by the ATR. Subsequent passes over the same area can be registered to the annotated image and detections initially discarded by the IA can now be suppressed. In the future, coherent change detection will provide information on movements through an area for exploitation by the ATR.

Cluster analysis provides the IA with an initial force level view of the battlefield by clustering target patterns to identify military formations (e.g., tank column, company). Force structure analysis (FSA) in the enhanced SAIP system will apply known military doctrine to further assess force structures and movements on the battlefield, while flexible FSA may provide future commanders with broader interpretations of the force structures encountered.

The human/computer interface (HCI) provides the IA with the interface tools for exploiting imagery through the use of interactive tools for analysis and annotation. The SAIP HCI is being developed through the participation of military image analysts. IAs have been a valuable asset in critiquing HCI designs through interactive experimentation and technical reviews. This role will continue throughout the course of the SAIP program as well as the incorporation of new techniques and designs into the SAIP HCI.

An important aspect of the HCI is the ability to provide IAs with the interactive tools required to exploit target detections. An interactive target recognition system developed by the Air Force will provide IAs with a tool to compare imagery against a database of existing target models. IAs will be able to interactively alter target model poses to match detected data, designate areas of the target model to be highlighted on the detected target, and provide additional models of related targets ranked by a probability match for further IA analysis. The enhanced system will provide users with the ability to rapidly add new target models to the SAIP system given 360 degrees of target model information. A research effort in on-the-fly (OTF) training may eventually provide IAs with the ability to construct target templates or models given limited views (e.g., three) of targets encountered on the battlefield for the first time.

1.3 SAIP HARDWARE ARCHITECTURE

The SAIP testbed hardware is composed of a networked collection of processors (subsystems) performing different functions (detection and feature extraction, object-level change detection, terrain delimitation, force-level grouping and discrimination, high-definition imaging and model-supported exploitation). These subsystems are interconnected by an asynchronous transfer mode (ATM) network, as shown in

Figure 4 {p. 12}. SAIP requires complex imagery data from an image formation process. The input imagery comes into the van on a high performance parallel interface (HIPPI) channel. This imagery is stored temporarily on disk prior to processing by SAIP processing algorithms. Once the SAIP algorithms have been applied, the imagery is displayed on an imagery analyst workstation. The testbed van contains four workstations. Three are IA workstations, each with two screens for operations by the editor/image analyst. The fourth terminal has one screen and will be used by the supervisor. The supervisor terminal is configured with image product archive (IPA) software for secondary imagery dissemination, and will have access to computer aided tactical information system/imagery exploitation support system (CATIS/IESS). The displays use a standard X-Windows/Motif interface with functions being made available through the click of a mouse. The functions include the standard light table functions such as roam, zoom, and mensuration, as well as the functions being provided by the SAIP technology.

The SAIP testbed is configured in a single van, as shown in Figure 5 {p. 13}. The van requires external power services, and will normally be co-located with a host imagery reception ground station (e.g., CARS, ETRAC, etc.) These host ground stations will provide the communications and imagery formation processing required for SAIP integration into the operational environment. However, the SAIP testbed is also capable of being operated in a split-based configuration where the host imagery ground station is not co-located and the image data is passed to the testbed by means of communication circuits via a HIPPI connection.

1.4 SAIP RASSP BENCHMARK-4

The SAIP Benchmark-4 is designed to exercise a significant portion of RASSP tool and methodology innovations within the allowable schedule and effort constraints. This BTD, combined with an executable requirement, is intended to provide stable and well-documented design information for the Benchmark-4 task.

Development of a SAIP accelerator represents an unclassified task amenable to inclusion of application-specific hardware as well as use of autocoded software. Algorithm descriptions are available in several documents (referenced later), and non real-time, single-thread C code is available with unclassified data to serve as part of an executable requirement. As shown in Figure 3, the SAIP accelerator will implement high-definition (super-resolution) imaging (HDI), and target classification using a minimum mean-square error (MSE) template matching algorithm. There is an opportunity to perform architecture trades comparing programmable versus application-specific implementations of the MSE classifier. An application-specific implementation would provide a vehicle for exercising the trade-off methodologies, virtual prototyping and synthesis. The strengths of the benchmark are that it requires high throughput with relatively simple algorithms, has a well-defined external interface, and can be compared to hand-coded workstation based software running on COTS hardware consisting of 25 cubic feet (19 cubic feet for the ruggedized version) of SGI (Silicon Graphics, Inc.) servers, as well as to future upgrades of the SGI hardware.

Sources of test data currently under consideration include simulated data, actual data with and without embedded targets, ASARS, ADTS and MSTARS data. Some of the test data is unclassified with distribution unlimited, some is unclassified but limited distribution (ITAR restricted), and classified data sets are also available.

Inputs to the Benchmark-4 system are complex SAR image chips, where a chip is a small subset of a larger SAR image. A sustained input rate of 30 chips/s is desired. The HDI output is real data at 30 chips/s. Note that the existing SAIP processors read parameter files from system disks for initialization.

The existing SAIP software runs on a ruggedized system of 14 processors (MIPS R8000 at 90 MHz) configured as three servers. The existing hardware in the SAIP van provides a peak computational capacity greater than the required theoretical throughput. However, as a consequence of inefficiencies in the hardware and high-order language implementation of the algorithm, only a fraction of the peak capacity is effectively available, leading to a maximum sustained input rate of less than 3.6 chips/s, or only 12% of the 30 chips/s goal.

Although insertion of Benchmark-4 products into the SAIP testbed (see Section 3 for details) is desirable for a final demonstration, all development work will be performed using a stand-alone Unix workstation. The software interfaces may be handled via the existing SAIP standard interface protocol based on a Unix socket interface specification (C++ code). It may be possible to support such development on either Sun or SGI workstations.

1.5 SAIP SOFTWARE DEVELOPMENT SYSTEM

The MIT Lincoln Laboratory SAIP software development system includes a 6 processor, R8000 SGI system at 75 MHz as well as a 4 processor R10000 SGI system (D-97604-S2-PWR Power Challenge L Deskside Server). It is likely that the Benchmark-4 tasks could be performed by the RASSP developer on a less capable and less costly SGI or Sun workstation.

The actual SGI servers used for SAIP software development are labeled "Model CMN A011, Iris L Series," on the chassis, regardless of whether the internal processor is R10000 or R8000. Each server is approx. 28.5" deep x 20.5" wide x 24.5" high (8.3 cu ft). There is room for a total of 14 9U VME boards (but only 11 labeled slots, so some may be double-wide), and the card cage is 12" wide and 16" high. The server contains power supplies, disks, control panels, tape drives, etc. There are 7 slots for various plug-ins, with each occupying a front panel space approx. 2" high x 5" wide. Depth would depend on the actual plug-in chosen. The card cage is labeled as follows: slots 1-5 are EBus (1 says "MC Only" and 5 says "1st I/O"), 6 is VCAM, and 7-11 are VME. In the R10000 server, the card in slot 1 is labeled P/N 050-0071-001 Rev. A. A variety of console connectors line the bottom of the front panel (SCSI, RS-232 terminal, etc.). The power input is labeled 115-120 VAC/16A or 208-230 VAC/13A, 50/60 Hz.

The Developer may require a similar software development system, or a development system capable of supporting both hardware and software development.

1.6 OTHER

In the area of prototyping and design, a hierarchical flow graph representation shall be employed with a non real-time performance model, virtual prototype and architecture trade study. A rapid real-time implementation will be made possible through the use of concurrent development (workstation and target libraries), as well as a quick (autocode) port to a real-time test bed. Reduced life cycle cost, size, weight, power and improved throughput for architecture options are expected, as well as legacy design documentation. The benchmark deliverables shall include details of process steps and work flow.

By providing an efficient implementation of high-definition imaging, the benchmark task addresses user requirements for a smaller system that may ultimately lead to processor insertion on a UAV platform. An HDI/MSE accelerator enables new algorithm architectures, new SAR applications such as site monitoring, and facilitates split-basing of the SAIP system. The benchmark objectives are responsive to the US Army's night vision electronic systems division (NVESD) tasking to evaluate ATR algorithm performance, reduced form-factor architectures, and reduced life cycle costs.

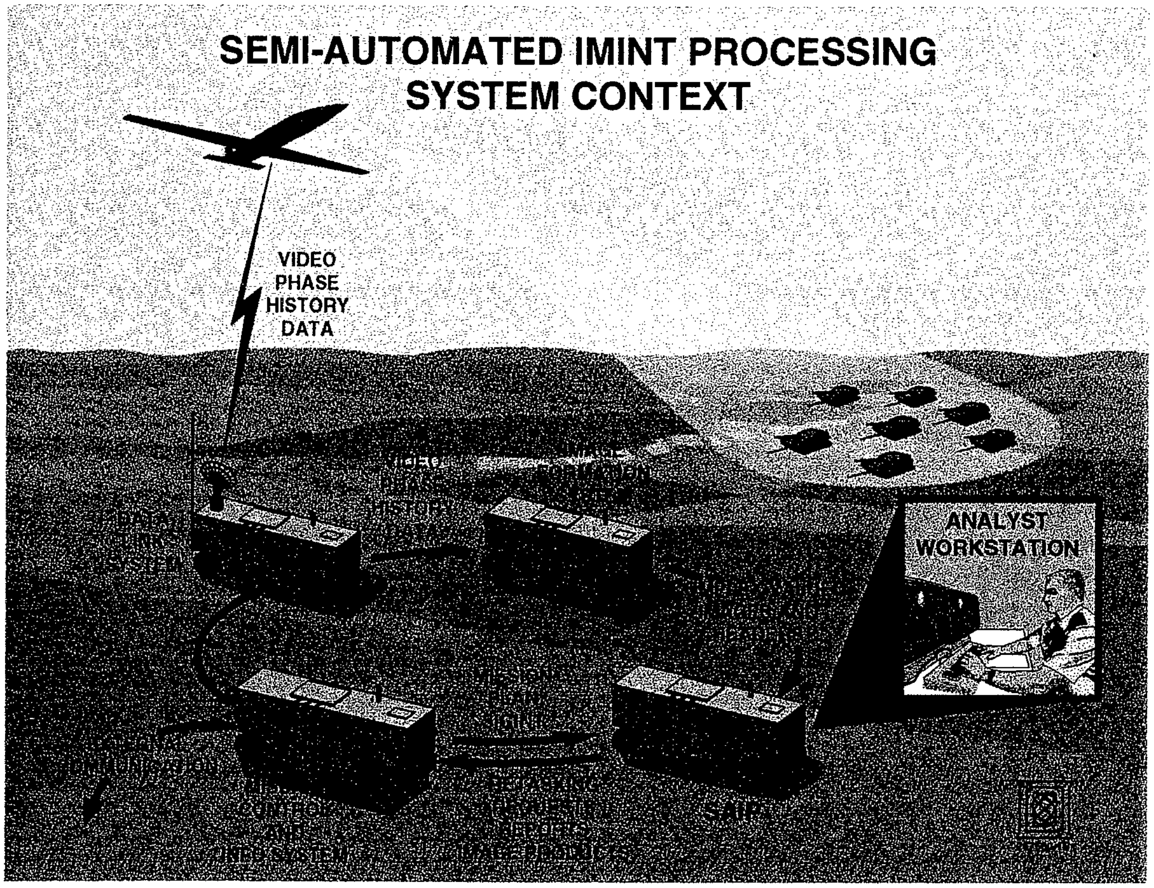

Figure 2. Semi-automated IMINT processing demonstration concept.

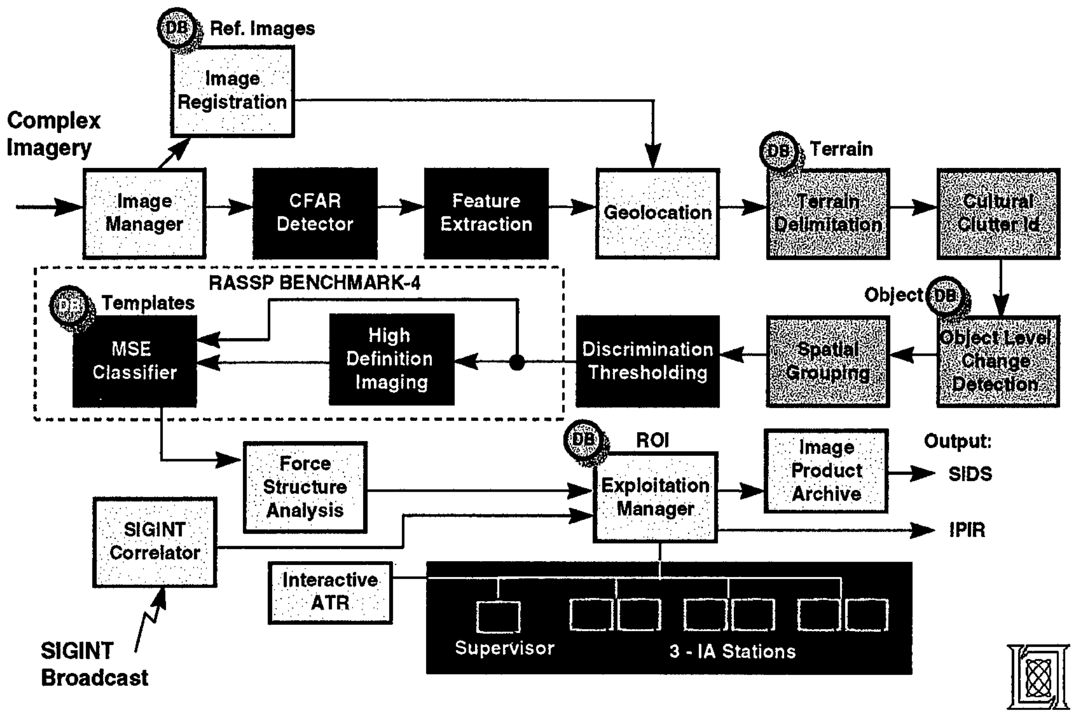

Figure 3. Block diagram of baseline SAIP system.

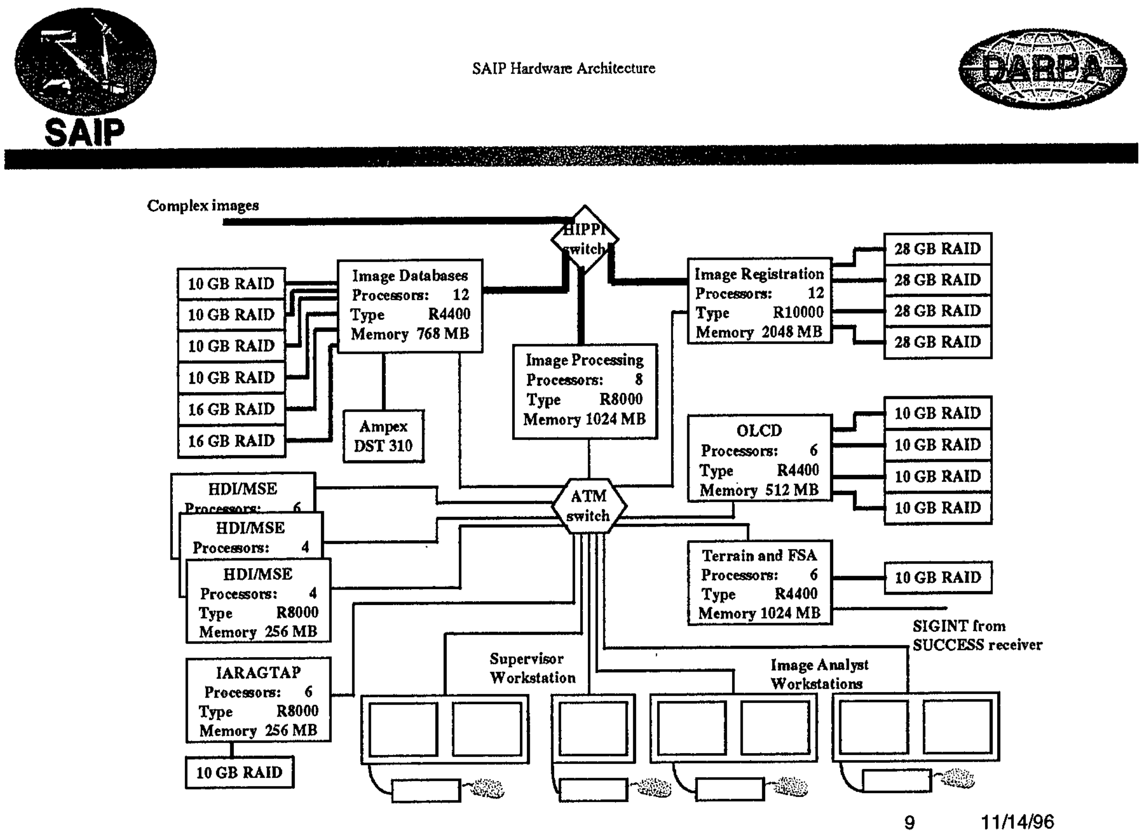

Figure 4. SAIP hardware architecture diagram.

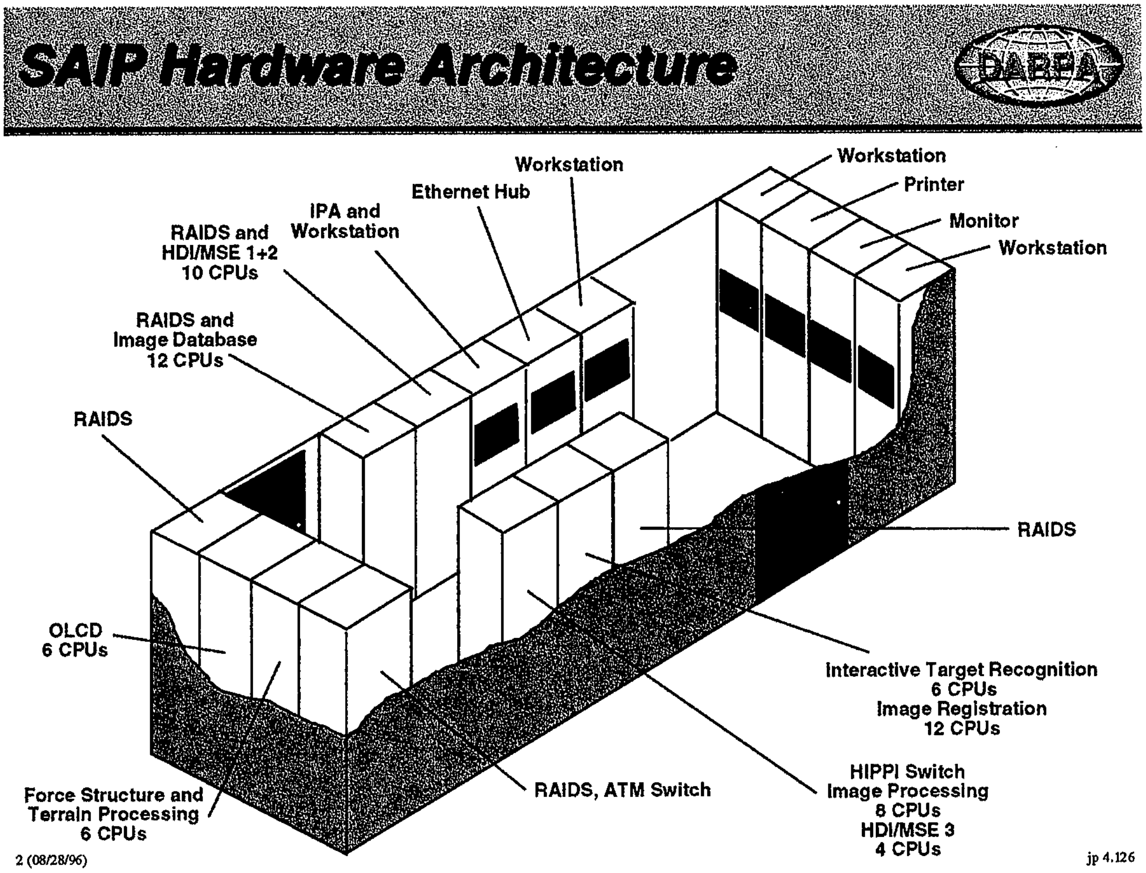

Figure 5. SAIP hardware in van.

2. PROCESSOR REQUIREMENTS

2.1 GENERAL

Benchmark-4 encompasses the two functional blocks enclosed by the dashed-line box in Figure 3 {p. 11}and shown in Figure 6.

The output of the Discrimination Thresholding module is a set of excised squares, or chips, from each image frame that contain interesting radar responses. Within each frame the chips are ordered by some criteria so that the most interesting ones are processed first. The High Definition Imaging (HDI) and Mean Square Error Classifier (MSE) blocks, which are the subject of this benchmark, improve the chip resolution and match the resulting pattern of signals against a library of templates. The following sections describe the HDI and MSE functionality and the processing capability of the existing SAIP implementation. The required capability for the BM-4 implementation, and a future growth capability are specified.

Figure 6 shows the data formats at the interfaces of the HDI and MSE functions. Data from the Discrimination Thresholding module must be converted from complex amplitude to real magnitude in dB before being used by the classifier. The data paths shown in the figure actually are implemented in the control software described below, which also performs the complex-to-dB magnitude conversion. All data paths shown involve single-precision floating point real or complex numbers.

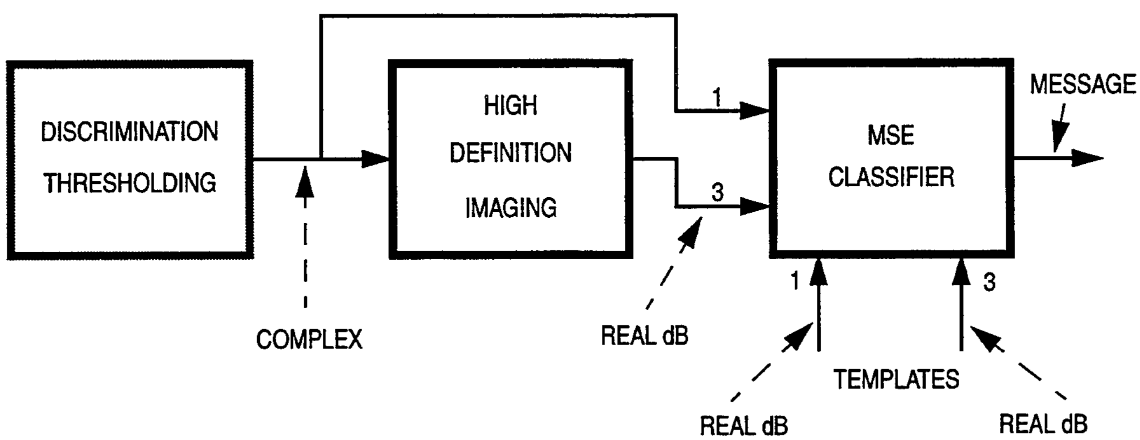

Figure 6. Data Formats at HDI and MSE interfaces.

2.2 CONTROL

2.2.1 Interface to SAIP

The SAIP system modules are controlled by transmission and reception of messages over an ATM network. Figure 7 is a representation of process distribution and flow of control and data information in the current SAIP implementation, which uses three SGI servers to perform the HDI and MSE process. Server

S1 contains six processors and servers S2 and S3 four each. A thresholding function is assigned to processor S1-5. (S1 is the Pserver 5 described in Section 3 while S2 and S3 are Pserver 6 and 7. The assignment of functions to processors shown here may differ from the SAIP system.)

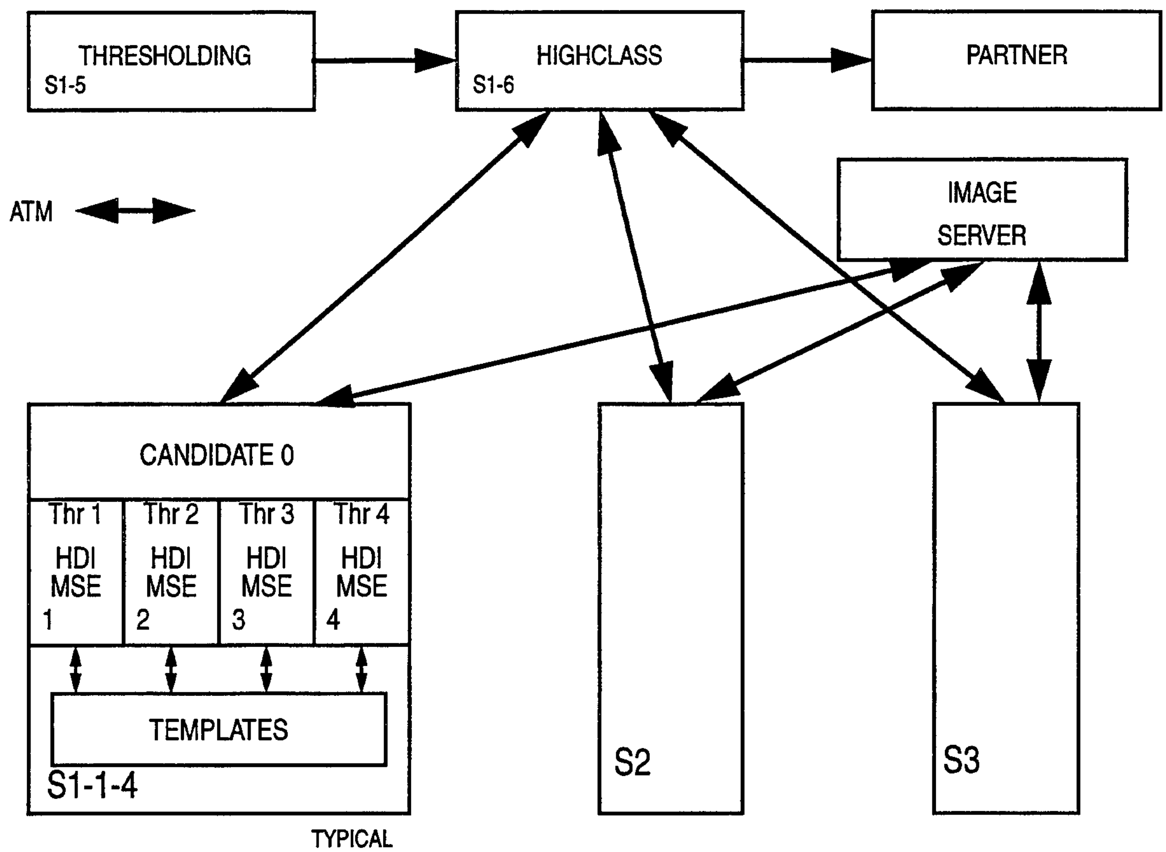

Figure 7. Processes and control and data flow for SAIP HDI and MSE.

The control process for HDI and MSE, named *HighClass*, is assigned to processor S1-6. It receives from the thresholding process a frame of data which contains header information which identifies Radar Mode and provides other real-time information. The frame also contains references for an ordered list of chips, where the references are to the position of the chip on a certain frame of image data. Two settable parameters in *HighClass*, NDet and MaxAge, determine how many of the chips in a frame will be processed: processing on the current frame will stop if the number of chips processed is greater than NDet, or if the age of the frame is greater than MaxAge.

HighClass assigns each chip, in sequence, to a server with an idle processor. Each server has a control process named *candidate*. They are referenced here with the generic *candidate*. The *candidate* process requests a chip from Image Server and passes the chip data it receives to one of several parallel processes spawned by *candidate* where some pre-processing is done. The chip is passed to an HDI process and the result is passed back. The high-resolution chip is passed to an MSE process which passes back the classification results. *Candidate* passes the high-resolution image and classifier results in a message to *HighClass* which passes them (the image, not a reference to it) on in a new frame to the *Partner* process.

The response time of the image server to the first chip request from a frame may be as long as one second. Response time to subsequent requests in the same frame are not longer than ten milliseconds. The image server services only one image request at a time with the current SAIP implementation.

In this SAIP implementation, each of the three servers has a copy of the in-memory representation of the template database.

The Benchmark-4 project shall include all of the functionality of *HighClass*, *candidate* and the HDI and MSE processes. The interface between the BM-4 subsystem and the SAIP system shall be passing of messages between the BM-4 subsystem and Thresholding, Partner and ImageServer in Figure 7. The messages which are passed on these interfaces are listed in Table 2, using the names for the current SAIP processes. Messages are passed on the ATM connections shown in Figure 4 {p. 12}using TCP/IP and, for these interfaces, the Classical-IP ATM format. There is not a one-to-one correspondence between message interfaces and ATM connections. For instance, in the current SAIP system shown in Figure 7 the interface between *HighClass* and *candidate* 0 does not use ATM, and the other seven message interfaces are mapped to one ATM interface on each HDI/MSE workstation. The SAIP message data structures are defined in a C++ class library [16] which is layered on the SAIP IPC library [17]. Code for both libraries and all of the relevant application code will be made available to the developer.

Table 2: Messages Between BM-4 and SAIP

MESSAGE TYPE	SOURCE	DESTINATION
frame	threshold	highClass
shutdown	threshold	highClass
imageDatumReq	candidate	image server
complexFloatDatum	image server	candidate
frame	highClass	partner
shutdown	highClass	partner

The BM-4 subsystem shall include a control computer and a disk large enough to hold the template data base, application programs, operating system, etc. The control computer shall have an ethernet port and an RS-232 connection for a remote console. All programs on this subsystem shall be launchable from its console.

Lincoln Laboratory will provide a workstation emulation of the SAIP system with one or two Classical IP ATM connections, as required, for acceptance testing of the BM-4 subsystem.

2.2.2 Run-Time Parameters

A radar mode is identified with each chip and each radar mode has a set of HDI and MSE parameters which are specified in a setup file. Some of these parameters will directly affect the run-time and memory requirements for HDI and MSE. It is required that the HDI and MSE systems be able to process frames of chips from two different radar modes for one setup, where the two modes may be for radars with different resolution.

2.3 HIGH DEFINITION IMAGING

The High Definition Imaging (HDI) module creates a higher resolution image from the central part of each chip where the size of the entire chip and the center part are specified by size and oversampledSize. The high-resolution chip is larger than the central part of the low-resolution chip in each dimension by approximately oversampling. Figure 10 {p. 25} illustrates the image improvement produced by the HDI process. In this experiment with an M48 tank, a 0.5 m resolution image (lower left) was spoiled to a 1 m resolution (upper right) which was the input to the HDI algorithm. The resulting output image is shown in the lower right.

(The next two paragraphs are adapted from [2]. Results of the Lincoln Laboratory work with this algorithm can be found in [4] and [5]. DeGraaf presents a review of several methods for improving resolution in [6], where his reduced rank minimum variance method (RRMVM) is similar to what is used here.)

Adaptive HDI is a data-adaptive approach to SAR image reconstruction based on superresolution techniques originally developed for passive sensor arrays. Image reconstruction is the process of transforming measured SAR data into an intensity profile which provides a map of radar cross section, or energy density, as a function of location (range and cross-range). SAR data can be thought of as having two dimensions, one corresponding to antenna position and the other corresponding to range-frequency. At each position of the antenna, the reflection coefficient (amplitude and phase) of the entire scene is measured over a band of frequencies. Image reconstruction can be thought of as applying weighting coefficients to this data, summing, and taking the squared magnitude. In conventional processing, the weighting coefficients are chosen to be a matched filter (matching the response of a point scatterer at the desired location; one filter for each pixel in the output image). In many applications this is simply a 2-dimensional Fourier transform.

Adaptive processing acknowledges the presence of multiple scatterers and adjusts the estimated RCS accordingly. In this HDI algorithm it is assumed that the observed scene comprises discrete point scatterers. The Maximum Likelihood Method (MLM), or Capon method [3], which is an adaptive technique for the estimation of power spectra, is employed. It is adaptive in that the weighting coefficients are a function of the data. The weights are chosen to satisfy two basic criteria, namely 1) they preserve unity gain for a point scatterer at the desired location, and 2) they minimize the perceived energy in the output image. A 2-D MLM is employed, using one of several different constraints on the norm of the weighting coefficients. The 2-D MLM method requires the computation of a sample covariance matrix, which is usually computed as an average of many statistically independent looks at one position. Since the SAR data collection process provides only one look, in this HDI method the covariance matrix is generated via a smoothing technique.

In the HDI C code, computation is in single precision except in some segments where precision is controlled by the parameter DOUBLE_PRECISION, which is set true. Double precision may not be necessary in all of those places but it is believed to be necessary, at least, for the FFTs in the invert_image and filt_and_dec2 functions, the entire SVD (singular value decomposition) function and for the matrix multiply which occurs right after SVD.

Further details of the HDI algorithm are presented in the RASSP Benchmark-4 Executable Requirement User Manual [15].

2.4 MEAN SQUARE ERROR CLASSIFIER

The Mean Square Error (MSE) classifier module receives both the low resolution and high resolution chips as shown in Figure 6. Each chip is compared with templates from a library to choose the most likely identification for the target. The classification results, along with the high definition chip, are made available to subsequent stages in the processing chain of Figure 3.

2.4.1 Templates

Each template is an n x n pixel square in which only some of the data is valid, as shown in Figure 8; the valid region may be unique for each template. Valid data is a single precision floating point representation of the pixel magnitude in dB. Invalid data is indicated by a pixel value of -999.

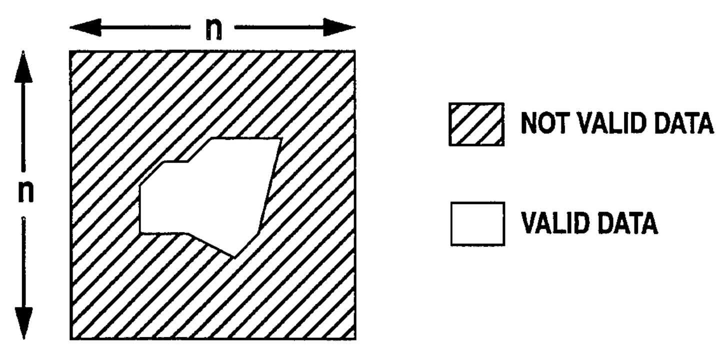

Figure 8. Template format.

Templates are organized in the tree structure of Figure 9, where the labels represent the organization of the template directory included in the executable requirement. Descriptions of each level are shown in Table 3.

2.4.2 Algorithm

The classifier implementation employs a two-stage process which has higher throughput and performance (higher probability of correct detection) than a single stage process. In the tree of Figure 9, the squint angle and depression angle branches which are closest to the squint and depression for the chip are followed. At the Oversample level, the MSE process compares the low resolution chip (input to the HDI process) with the low resolution templates for all target classes. This first stage identifies the five most likely target classes and the pose angles of the best matching templates. A second stage compares the high definition chip against the high definition templates in the five classes identified in the first stage and for a limited number of poses around and 180^0 away from the angle of the best template. The final result is a probability-of-match measure for each of the five classes.

In both stages, an estimate of the mean is subtracted from the chip data. The match measure is a normalized sum of squares of differences between the pixels in the chip and template, where the calculation is

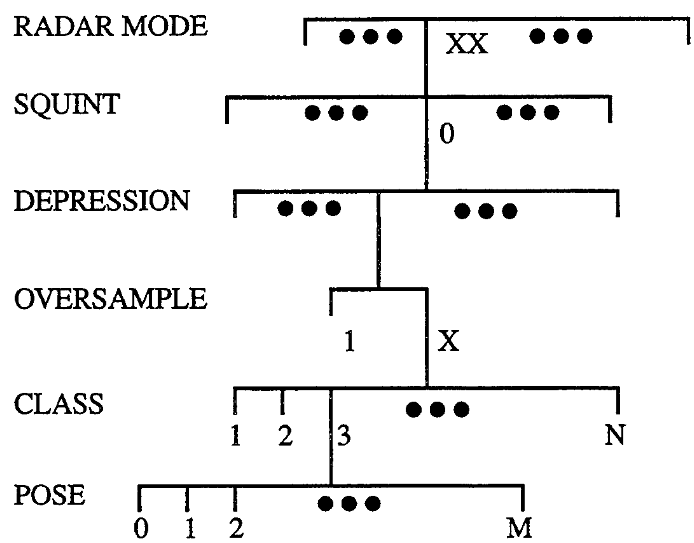

Figure 9. Tree structure for template library.

Table 3: Tree Structure Parameters

Level	Description
Radar	Type of radar (Mode)
Squint angle	Angle between radar centerline and direction vector of platform
Depression angle	Angle of radar beam to ground
Oversample	1 - no resolution enhancement x - resolution enhancement by a factor of x
Class	Target class
Pose angle	Angle of repose of target

done only for valid pixels in the templates (see Figure 8) and the pixel magnitude representations are in dB. The bright returns in the chip and template are only approximately centered, so the template must be translated over the chip by + and - X and + and - Y.

Note that the valid-data areas in the templates are not rectangular and are not the same for each template, so the processing time will also depend on how the templates are stored and accessed. In the SAIP implementation, the MSE algorithm is performed in single precision floating point which is adequate; lesser precision is probably adequate but has not been investigated.

Further details of the MSE algorithm are presented in the Executable Requirement User Manual.

2.5 ALGORITHM REQUIREMENT

The Benchmark-4 HDI processor shall implement the algorithm which is described by the code delivered with the executable requirement and associated user manual.

The Benchmark-4 MSE processor shall implement the algorithm which is described by the code delivered with the executable requirement with the following exception. The developer may replace the match measure of a normalized sum of squares of chip-template differences with a normalized sum of absolute magnitude of chip-template differences. If this change is made, Lincoln Laboratory will redefine certain MSE Run-Time parameters.

Design tradeoff shall be done using parameters for two radar modes identified as spot_a and spot_b. The data delivered with the executable requirement is from a spot_b system. The design shall be capable of supporting a more capable radar in a mode designated as enhanced.

The HDI and MSE processors shall be capable of processing data from two different radar systems (two radar modes) with no set-up delay between chips from the two modes.

2.6 PERFORMANCE REQUIREMENT

It is required that the HDI and MSE processors be designed to process 30 chips per second with the parameters and the data delivered with the executable requirement, after the latency of the first chip request from a frame. The performance shall be estimated for the enhanced cases.

2.7 ACCURACY REQUIREMENT

The accuracy requirement on the implementation of both the HDI and MSE subsystems is defined in terms of MSE scores and classification results. MSE results will be generated for all the target and clutter chips and the templates delivered to the developer. Comparison results will be generated by the C code for HDI and MSE delivered with the Executable Requirement except that if the developer uses a mean absolute difference measure instead of mean squared, then the C code will be likewise changed, but still using the same C-code arithmetic precision. The same chips will be processed by the developer's system. For the target chips, the difference in score for each of the five reported results for each chip shall be not larger than some specified value and the first ranked template matches shall be the same for at least a specified percentage of the chips. For the clutter chips, the difference in score for each of the five reported results for each chip shall not be larger than some specified value.

2.8 FORM FACTOR CONSTRAINTS

2.8.1 Size

Dimensions of the HDI/MSE processor node developed for the enhanced properties of Section 2.5 shall not exceed those of a single existing SAIP HDI/MSE subsystem as described in Section 3.1.

2.8.2 Scalability

The functional requirements included in this BTD only encompass the baseline SAIP testbed algorithms. In the future, additional functionality may be required for the enhanced SAIP configuration. The processor architecture must therefore be scalable to support at least the requirements of the enhanced configuration of Section 2.5.

2.8.3 Weight

The weight for a fully-loaded chassis, including a weight estimate for future expansion, shall be less than 250 pounds.

2.9 ENVIRONMENTAL

Air cooling in a non-condensing environment is assumed. The temperature range of the ambient air will be 0^{o} C to 40^{o} C.

As the SAIP van has built-in shock and vibration resistant equipment mounts, shock and vibration tests need not be performed. The equipment provided, however, should at a minimum meet the operational vibration specifications for commercial equipment (e.g., 0.35 G RMS, 19-500 Hz for SGI).

2.10 POWER SUPPLY

The power supply shall operate with an RMS input voltage from 105 to 125 VAC, over a frequency range of 47-73 Hz. The average input power, as computed over any 0.5 second interval, shall not exceed 3000 watts in the baseline system.

Provision to handle the enhanced configuration of Section 2.5 shall be made with the input power not to exceed 4500 watts. The power supply need not be sized to handle the fully populated chassis (4500 watts input) provided that modules may be added incrementally to the initial supply. The power supply system shall conform to the requirements of MIL-STD-704 for input transients (180 VAC for 0.1 second), and MIL-STD-461 for generation of and susceptibility to radiated and conducted interference.

2.11 FAULT DETECTION, ISOLATION AND TESTING

Fault detection, isolation and testing are desirable for COTS hardware and required for custom hardware. Any online reporting of faults or testing results shall be done to the monitor and recorded in a disk file. The developer shall provide a cost/benefit analysis for the capability described in this section during Phase I.

Built-In Test (BIT). The system shall have BIT for monitoring mission critical functions and isolating detected faults. BIT shall provide the primary means of fault detection, fault isolation, and system verification without the need for external equipment. BIT shall monitor the system functions and shall indicate where a failure has been detected. There shall be no operator participation beyond the initiation of BIT and observing results. IEEE 1149 buses shall be used to develop the BIT capability. Components that are selected containing IEEE 1149.1 interfaces shall be connected to allow external access to the IEEE 1149.1 system interface.

On-Line BIT. The BIT shall have on-line BIT to allow detection of 95% faults, including cable faults and network faults if a network/LAN is used.

Off-Line BIT. The BIT shall have an off-line BIT capability to allow detection of 97% faults, including cable faults and network faults if a network/LAN is used.

BIT Fault Isolation. 99% of the faults detected by on-line and off-line BIT shall be isolated to a single failing circuit card assembly or cable without the use of external test equipment other than a multimeter.

BIT results display. The BIT shall display results of BIT testing in two methods selectable by the initiator: maintenance BIT results and operator BIT results. The operator BIT results shall display failure and fault isolation information that would allow maintenance action by the operator. The maintenance BIT results shall display detailed failure and fault isolation information that would be appropriate to the maintainer to perform more skilled maintenance.

BIT and Connected Equipment. The BIT shall communicate with connected equipment that contains self-test to cause that equipment to initiate the connected equipment's self-test without operator intervention.

Diagnostic/Fault Log. The equipment shall maintain a diagnostic/fault log on a removable, non-volatile storage media. The media shall be capable of being processed at a location remote from the system as a diagnostic and maintenance aid for the system.

BIT and Embedded Training. A mechanism to allow maintenance training, e.g., fault insertion, shall be designed into the system.

Other requirements. The mean time between failures (MTBF) and mean time to repair (MTTR) shall not be degraded by inclusion of BIT capability. A detailed BIT plan shall be available at the system requirements review milestone detailing how BIT requirements will be met. This plan shall initially be a high-level plan, with details added as the design progresses. This plan shall include diagnostic methods to be used, and detailed diagnostic information (e.g., fault trees) as the design is completed.

2.12 DOCUMENTATION

2.12.1 Hardware

A complete set of drawings shall be provided with the prototype of the processor. For parts of the prototype processors that are COTS (commercial off the shelf), some of these requirements may be waived. At a minimum, the drawings must include both simplified and detailed block diagrams. Additional drawings shall include, but not be limited to, the following:

- Individual mechanical drawings of chassis, boards, backplanes and connectors.
- Detailed schematics and/or source files for all non-COTS printed circuit boards, MCMs, ASICS, PALs, FPGAs and PLDs.
- All source files and/or schematics for any programmable devices incorporated in the signal processor, including PALs, FPGAs, and complex PLDs. This requirement is for the lowest level description that was used in the course of designing the device.

- Parts list and cost.
- Net list of all non-COTS printed circuit boards.
- Full specifications for any non-standard or proprietary components.

The theory of operation shall be documented including,

- Modes of operation supported and the protocols for the test and diagnostic defined in X.
- All critical timing information.
- All non-standard interfaces.

2.12.2 Software

All non-COTS application software (i.e., software developed specifically for the benchmark by the Developer) shall be provided. Hard copy of all application source code shall also be provided.

Software documentation shall conform to best commercial standards and practices.

2.13 REPORTING

Progress reports shall be provided with each milestone as discussed in Section 6.

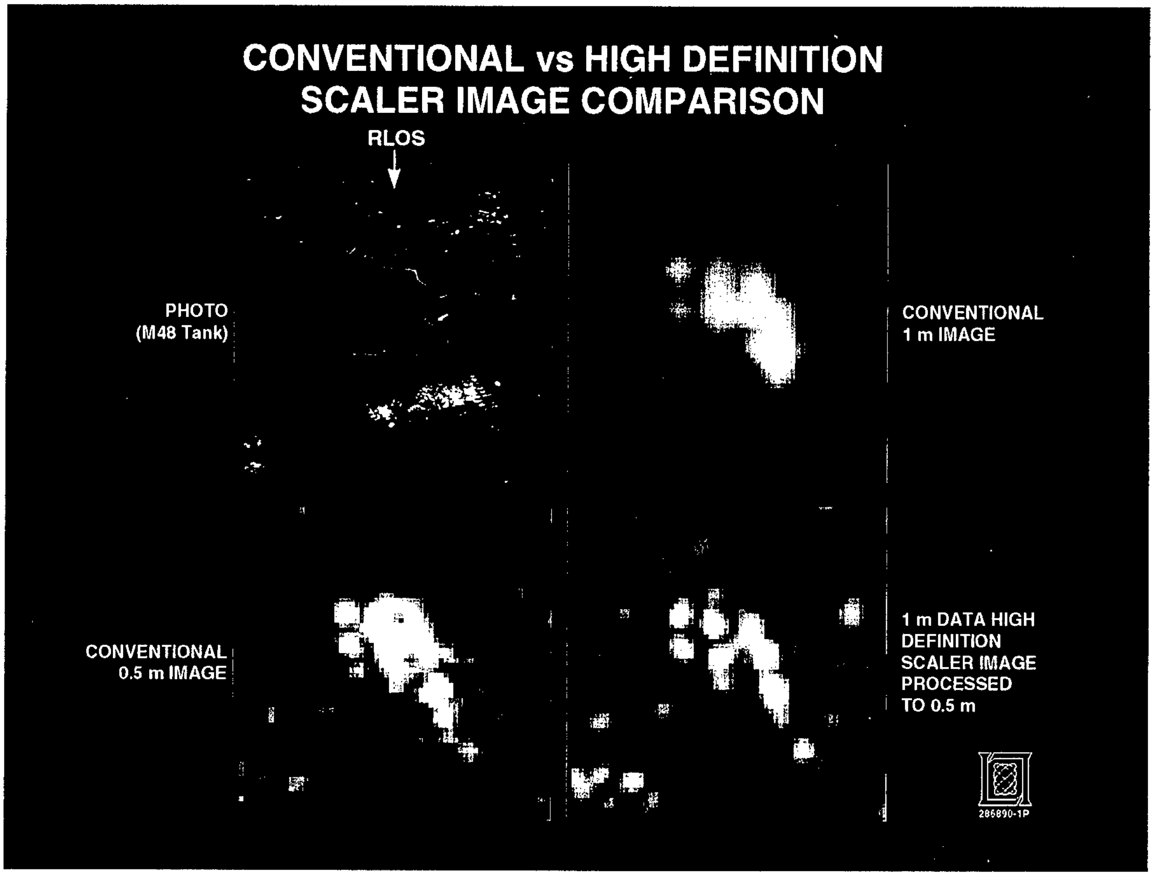

Figure 10. Demonstration of High Definition Imaging.

3. SAIP TESTBED

3.1 HARDWARE CONFIGURATION

The SAIP van contains 3 ruggedized SGI servers, designated Pservers, in two racks as shown in Figure 11. Two HDI/MSE subsystems are housed in Rack 6 and one is housed in Rack 16. Pserver 5 (HDI/MSE 1, A17, Rack 6) contains six 90 MHz R8000 processors, while Pserver 6 (HDI/MSE 2, A18, Rack 6) and Pserver 7 (HDI/MSE 3, A53, Rack 16) each contain four 90 MHz R8000 processors. Note that only four of the processors in each Pserver are used for the HDI/MSE processing, and two of the processors in Pserver 5 are assigned to other tasks. Presently, each Pserver has two memory cards (SGI P/N 030-0604-106), one I/O card (SGI P/N 030-0646-105), two or three processor cards (SGI P/N 030-0702-002 or -003), 2 or 3 empty EBus slots, one 9U VME VCAM card (EBus to VME interface, SGI P/N 030-0500-305), one empty 9U VME slot, one 6U VME ATM interface card (Fore VMA200) and three empty 6U VME slots. Details of the contents of each Pserver are given in Tables 4, 5 and 6.

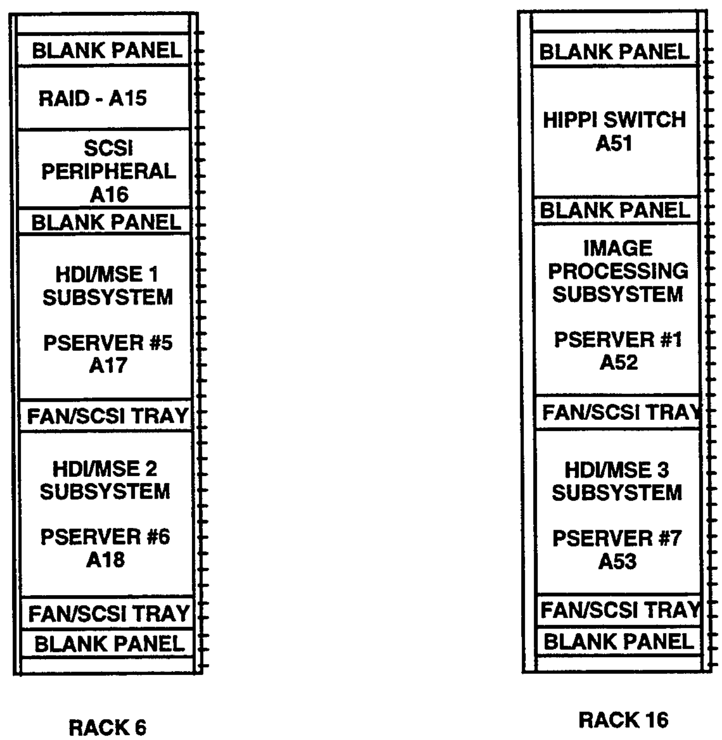

Figure 11. Ruggedized SGI server racks.

TABLE 4 Pserver 5 (HDI/MSE 1)

Slot Type	Slot contents	Mezzanine	I/O channels	Peripherals
EBus	Memory control	128 MB RAM		
EBus	Memory control	128 MB RAM		
EBus	IO4	VCAM	FWSCSI	4 GB system disk; SCSI peripherals chassis (CDROM, 8 mm MO drive)
EBus	2 ea. 90 MHz R8000			
EBus	2 ea. 90 MHz R8000			
EBus	2 ea. 90 MHz R8000			
EBus				
EBus				
9U VME	VCAM			
9U VME				
6U VME	ATM (Fore)			
6U VME				
6U VME				
6U VME				

TABLE 5 Pserver 6 (HDI/MSE 2)

Slot Type	Slot contents	Mezzanine	I/O channels	Peripherals
EBus	Memory control	128 MB RAM		
EBus	Memory control	128 MB RAM		
EBus	IO4	VCAM	FWSCSI	4 GB system disk
EBus	2 ea. 90 MHz R8000			
EBus	2 ea. 90 MHz R8000			
EBus				
EBus				
EBus				
9U VME	VCAM			
9U VME				
6U VME	ATM (Fore)			
6U VME				

TABLE 5 Pserver 6 (HDI/MSE 2)

Slot Type	Slot contents	Mezzanine	I/O channels	Peripherals
6U VME				
6U VME				

TABLE 6 Pserver 7 (HDI/MSE 3)

Slot Type	Slot contents	Mezzanine	I/O channels	Peripherals
EBus	Memory control	128 MB RAM		
EBus	Memory control	128 MB RAM		
EBus	IO4	VCAM	FWSCSI	4 GB system disk
EBus	2 ea. 90 MHz R8000			
EBus	2 ea. 90 MHz R8000			
EBus				
EBus				
EBus				
9U VME	VCAM			
9U VME				
6U VME	ATM (Fore)			
6U VME				
6U VME				
6U VME				

All three Pservers have cabling for ATM (multimode fiber), ethernet (10BaseT) and a console serial line. In addition, Pserver 5 is wired for differential SCSI.

Each SAIP van rack is custom-made by Northrop Grumman Corp. (formerly Westinghouse Electric Corp., Baltimore, Md.). Various components, e.g. the status panel, are removed from a SGI Deskside Challenge system and installed in an Equipto Electronics Corp. (Aurora, Il.) model 40-4722-03 rack. Overall dimensions of the rack are roughly 22" wide x 25" deep x 6'1" high. Each of the HDI/MSE subsystems occupies a height of approximately 20" within the rack, for a volume of approximately 6.37 cu ft each, or a total of 19.1 cu ft for all three subsystems.

Although the SAIP racks use custom power supplies connected to 208 VAC (60 Hz), power is also available in the van for racks that require 110 VAC (60 Hz). The van's cooling system is adequate to remove the resulting heat dissipated by all racks.

3.2 SAIP COTS HARDWARE UPGRADE

A simple upgrade to the existing baseline SAIP system could be performed by removing the seven dual-R8000 cards and returning them to SGI for credit towards the purchase of 15 quad-R10000 cards (the maximum number that could be placed in the present enclosures). The cost of performing such an upgrade would be $807K with GSA discounts. Note that neither the R8000 nor R10000 cards for SAIP will be supported by SGI after June 1997, as the Challenge series of processors is incompatible with the new Origin series.

Recent experiments indicate that the 190 MHz R10000 performs the required HDI/MSE calculations a factor of 1.6 faster than the 90 MHz R8000. Assuming that the existing baseline system uses 12 of the total 14 R8000 processors, and the upgraded system would use 58 of the total 60 R10000 processors, the upgraded system would have a throughput improvement factor of (58/12)x1.6 = 7.73. Although such an improvement factor for COTS is of the same order expected for the RASSP upgrade, the COTS upgrade would provide no overall size reduction nor future upgrade path.

4. EXECUTABLE REQUIREMENT

MIT Lincoln Laboratory, at the direction of the Government, will deliver to the Developer an executable requirement as a supplement to this written requirement. It comprises the following elements.

4.1 DATA

A collection of clutter-with-target image files. A set of templates in low and high-resolution for ten classes will also be included.

4.2 MATLAB

A Matlab implementation of the HDI processing.

Matlab scripts which display the different data files. The developer shall provide a Matlab tool to execute these programs.

4.3 SOFTWARE

The following software will be provided in source code form. It will be executable on Sun or Silicon Graphics UNIX workstations. The developer shall supply the necessary compilers and workstation platform.

4.3.1 High Definition Imaging

The software which executes the HDI algorithm is written in C++ and C with calls to Fortran functions. The Fortran code was obtained from the depositories at http://gams.nist.gov.

4.3.2 Mean Square Error Classifier

The software which executes the MSE algorithm is written in C++.

4.4 USER MANUAL

A User Manual which describes the data and software will be supplied. It will have instructions on how to use the programs and data.

5. METRICS

5.1 INTRODUCTION

All metrics associated with the benchmark are described in this section. The metrics believed to be essential for developing a comprehensive evaluation report are identified in Section 6 as deliverables, which the Developer must collect and supply to the Benchmarker during the course of benchmark execution. In some cases, only estimates of the required metrics or parameters will be available. In such cases, the Developer will supply a best estimate with a rationale (basis) for the estimate.

Two approaches will be applied to evaluate the RASSP process and products. In the first approach, commercially available parametric cost estimation (PCE) packages will be utilized, primarily to obtain estimates of cost and schedule that serve as a standard practice reference. Development of the standard practice reference is a cooperative, iterative effort involving both the Benchmarker and the Developer. The most comprehensive PCE packages are the Parametric Review of Information for Costing and Evaluation (PRICE) [11] and System Evaluation and Estimation of Resources (SEER) [12]. These packages are discussed in Section 5.2. The Benchmarker will compare the standard practice estimates with actual cost and schedule which shall be reported in detail as described in Section 6.2.2.1 and 6.2.2.2. In a second approach, metrics derived from basic principles will be collected and utilized as a basis for evaluating specific areas of RASSP product and process development. Such development areas include productivity measures of the RASSP process such as lines of code per day produced, ease of use of the design environment, performance and complexity of the product, quality of the product, cost of the process and the product. The metrics formulated for these and other areas are discussed in Section 5.3.

Since a primary goal of the benchmark activity is quantitative measurement of RASSP related improvements to design, it is anticipated that the collection and analysis of metrics for this purpose will require a non-trivial effort on the part of the Developer and the Benchmarker.

5.2 PARAMETRIC COST ESTIMATOR METRICS

In a cost estimation technique known as "parametric estimating," a cost estimating relationship (equation, table or graph) is used to predict cost as a function of design size, performance variables, applicable technology and other parameters. The Air Force provides a free program called REVIC which performs software cost estimates based on the Constructive Cost Model (COCOMO) [8]. In addition, there are at least 18 commercial companies which provide parametric cost estimation products for software [9]. Two product lines (PRICE from Martin Marietta PRICE Systems and SEER from Galorath Associates Inc.) are of particular interest as they also provide hardware cost estimation capabilities. These programs require a variety of inputs to perform their cost estimation function. The inputs to these various cost estimation programs form a basic set of metrics that can be used to track the progress of RASSP, and other metrics can be added as necessary. Note that actual benchmark measurements, not the predictive cost estimates produced by the programs, will ultimately measure the progress. The cost estimates produced by the programs can, however, be used to compare the complexity of one benchmark task relative to another benchmark task. In addition, the cost estimates can be used to identify areas in which progress is being made (e.g., a measured cost which is less than the standard practice-based predictive cost estimates by a factor of 4 indicates potential achievement of a RASSP goal in a particular area).

The benchmarker will use the PRICE and SEER families of parametric cost estimation tools to produce standard practice baselines. While the Developers are not required to use these tools (they may use internal tools which they intend to sell as part of the RASSP design environment, or simply maintain a rapid detailed bottom-up estimating approach), they are required to submit input data (metrics) to the benchmark team so that all applicable PRICE and SEER estimating models can perform accurate parametric cost estimates. Data must be provided in a format convenient for entry into the estimating models running on an IBM PC-compatible computer. The establishment of a standard practice baseline is a cooperative effort between the benchmarker and the Developers. This document describes the PCE tools to be used but does not provide a list of the input data required.

5.2.1 PRICE S Software Model

The PRICE S software model applies parametric modeling methods to estimate the acquisition cost, software sizing cost, and operating and support costs for computer software. The acquisition cost estimates the software development acquisition process in each of the following phases:

1. System concept
2. System software requirements
3. Software requirements analysis
4. Preliminary design
5. Detailed design
6. Computer software configuration item (CSCI) test
7. System test
8. Operational test and evaluation (OTE)
9. System integrate and test.

The software sizing cost estimates the number of instructions in terms of source lines of code for both commercial and military applications. The operating and support costs estimate the life cycle costs for the maintenance phase, including software maintenance, enhancement, growth, and modification.

5.2.2 PRICE M Microcircuits and Electronic Assemblies Model

The PRICE M microcircuits and electronic assemblies model consists of three modes: microcircuit, module, and database. The microcircuit mode emulates the procedures and processes involved in the design and fabrication of microcircuits. The module mode represents a computerized modeling technique designed to produce cost and schedule estimates associated with the design and production of modules, boards, or hybrids. The database mode allows the user to place frequently used components into files which are then specified as extra input files in the module mode. Indices are derived from the calibration process based on cost history.

5.2.3 PRICE H Hardware Systems Model

Cost estimates in the PRICE H hardware model are made via an Estimating Breakdown Structure (EBS). The EBS is a sideways tree structure which provides a graphical depiction of the system to be estimated. Fourteen items called elements can be selected from the EBS for editing, copying, moving, deleting or processing. The 6 primary hardware operation elements are: system, assembly, electro/mechanical, structural/mechanical, modified and calibration. The 3 integrating operation elements are: design integration, hardware/software integration and hardware integration & test. The 5 specialized elements are: purchased, given cost, furnished, thru-put and multiple lot production. Four different types of data or operations may be associated with each element: input, output, global and escalation. Input variables, or metrics, may have a different definition and value for each element of the EBS.

5.2.4 PRICE HL Hardware Life Cycle Cost Model

The PRICE HL hardware life cycle cost model is a supplement to, and operates in conjunction with, the PRICE H model to compute maintenance costs for a variety of systems types. Through its association with PRICE H, PRICE HL eliminates the difficulty of having to develop a large quantity of cost factors for different equipment types in order to estimate life cycle costs. The model also provides ways of tailoring analyses to fit a wide variety of maintenance concepts and supply systems that can be custom designed for specific projects and user organizations.

5.2.5 SEER-SEM Software Estimation Model

The SEER-SEM software estimation model creates cost, schedule, risk, and maintenance estimations for software development. In SEER-SEM, software volume is the primary driver. It can be entered as functions, as lines of code, or as both.

The WBS (Work Breakdown Structure) divides the overall project into computer programs or Computer Software Configuration Items (CSCIs)--the highest unit of a software application--which can be further subdivided into Computer Software Components (CSCs), which can be further subdivided into Computer Software Units (CSUs). SEER-SEM provides cost estimates for each of the following project phases:

1. System concept
2. System requirements design
3. Software requirements analysis
4. Preliminary design
5. Detailed design
6. Code and CSU test
7. CSC integrate and test
8. CSCI test
9. System integrate through operational test and evaluation
10. Maintenance and operation support.

These phases correspond to the traditional waterfall model of development which may not apply to the RASSP design methodology (which may use, e.g., a spiral development model [13], but is appropriate for representing standard practice.

Built-in knowledge bases are chosen as a function of four characteristics--platform (avionics, business, ground, manned space, missile, mobile, ship, unmanned space), application (CAD, command/control, data base, diagnostics, flight, message switching, MIS, mission planning, MMI/graphics, office automation, OS/executive, process control, radar, report generation, simulation, software development tools, test, training, utilities, other), development method (Ada development, Ada development with incremental methods, Ada full use, prototype, spiral, traditional incremental, traditional waterfall), and development standard (commercial, 2167A, 2167, 2167A minimal set, 2167A full set, 1703, 483-490, 1679 with IV&V.)

The values of the aforementioned four characteristics define a specific type of WBS item which SEER-SEM uses to generate the most likely values and ranges for an extensive list of input parameters. These parameters can then be modified by the user to further customize and refine the model of the overall project environment.

5.2.6 SEER-SSM Software Sizing Model

The SEER-SSM software sizing model estimates the expected size of a software project based on qualitative/relative inputs without the use of databases.

As in SEER-SEM, the WBS (Work Breakdown Structure) partitions the overall project into modules--CSCIs which can be further divided into CSCs which can be further divided into CSUs--whose operational and functional characteristics are defined. SEER-SSM customizes the requirements for user-provided input after the partitioned modules to the model have been designated.

SEER-SSM requires project information (company/organization, project name, file name), module data (name of software unit and at least two reference modules of known size with their size expressed as in DSI, DEMI, or function point count), and four user-provided input data sets (DSXs)--pairwise data, PERT sizing data, sorting data, and ranking data--for execution.

5.2.7 SEER-IC Integrated Circuit Model

SEER-IC uses a Work Breakdown Structure (WBS) to create cost estimates for integrated circuits (chips), multi-chip modules (MCMs) and chips on MCMs. Built-in and customized knowledge bases may be used to provide information for estimates. Built-in knowledge bases are selected as a function of project type (MCM, complex gate array, custom chip, monolithic microwave integrated circuit, "none," semi-custom chip or simple gate array), platform standard (industrial, commercial, military airborne, military ground, military ground mobile, military sea, "none," manned space or unmanned space) and acquisition category (buy and integrate, customer furnished equipment, make, "none," or subcontracted item). User created knowledge bases (class) can be created if desired. Adjustment factors can be applied for specification generation, design, prototype hardware and average unit production in each of the class, platform standard and acquisition category knowledge bases. Such adjustments are used to accommodate variations due to fees or discounts. Once the applicable knowledge bases have been invoked and adjustments applied, information is entered to perform estimates. Most input variables have an optional associated range such as "least, likely, most," or "low, nominal, high." Application ranges for all required inputs (except production quantity) are loaded by the knowledge bases. Users narrow the input ranges when actual values are known.

5.2.8 SEER-H Hardware Estimation Model

SEER-H uses a Work Breakdown Structure (WBS) to create cost estimates for hardware elements. Built-in and customized knowledge bases may be used to provide information for estimates. Built-in knowledge bases are selected as a function of element type (mechanical or electronic), application (hydraulics, signal processor, communications, etc.), platform (ground, air, space, fixed or mobile, manned or unmanned), development standard (commercial, military specification), and acquisition category (buy and integrate, customer furnished equipment, make, subcontracted, or "none"). User created knowledge bases (class) can be created if desired. Adjustment factors can be applied for specific generation, design, prototype hardware, and average unit production in each of the class, platform standard, and acquisition category knowledge bases. Such adjustments are used to accommodate variations due to fees or discounts. Once the applicable knowledge bases have been invoked and adjustments applied, information is entered to perform estimates. Most input variables have an optional associated range such as "least, likely, most," or "low, nominal, high."

5.2.9 SEER-HLC Hardware Life Cycle Model

SEER-HLC is the operations and support option to SEER-H which can be used as an accessory to a stand-alone life cycle cost estimation tool. Outputs from other SEER models can provide many SEER-HLC inputs if desired, and other sources may also be used. SEER-HLC allows evaluation and tradeoffs between various prime and support equipment design philosophies and support concepts. Life cycle tradeoffs of reliability and reliability maturation, mean time to repair, repair turnaround times, and unique and shared support resources may be evaluated using SEER-HLC. Variances in operational scenarios may also be evaluated in terms of impact on cost and performance. SEER-HLC allows the evaluation of multiple support capabilities including organizational, intermediate, and depot maintenance, as applicable for the system under estimation.

5.3 PROCESS AND PRODUCT METRICS

The metrics in this section are directed toward specific issues of performance of both the RASSP process and products, and complexity of the application and products. Metrics for the complexity of the RASSP process, such as the total number of source lines of code in the RDE, are not required. The complexity of the RASSP process is measured indirectly through productivity metrics, cost of the tools and ease of use.

5.3.1 Design Process

The different tools and procedures that are used in all segments of the benchmark execution are considered in the evaluation. Metrics must be collected to quantify the value of both the tools and the underlying methodology. Although the ultimate measure of success is the reduction in the design cycle time, analysis of progress during the RASSP program requires an understanding of which steps in the RASSP process consume the majority of the time, and where improvements in the time required to execute the process are occurring.

5.3.1.1 Tool Evaluation

For each major tool used during execution of the benchmark, the following metric shall be reported.

TOOL_USAGE The number of hours the tool was used. Depending upon the Developer's design environment this metric may be obtained by direct measurement or estimated by users. Personal logs may be used for this purpose; it is then imperative that personal logs be reviewed periodically.

5.3.1.2 Reuse

Metrics shall be reported for each hardware and software module or element, including VHDL code, which is reused from a previous project. The metrics required to evaluate reuse relate to the time saved through use of this technique. This requires meaningful estimates of the time that would have been spent in creating an original design, the time spent in evaluating the module and the time spent in incorporating it into the applicable design. The specific measurements and metrics in both time (person-hours) and cost are listed in Table 7.

Table 7: Reuse Measurements and Metrics

Element	Description
REUSE_ORIG_T, REUSE_ORIG_C	Estimated (time or cost) for original design and implementation
REUSE_EVAL_T, REUSE_EVAL_C	Time (or cost) expended in evaluating the module
REUSE_T, REUSE_C	Time (or cost) spent incorporating module
REUSE_TRATIO	original implementation time / reuse incorporation time
REUSE_CRATIO	original implementation cost / reuse incorporation cost

5.3.1.3 Defects

For purposes of benchmarking and improving the RASSP development process the Developer is encouraged to document defects which are created in one phase of a development and found and corrected later. An example would be an omission in a Requirements Document which is discovered during software development or an incomplete simulation which allowed an error to continue through to integration and test. The use of configuration control for source files of application software and VHDL as well as all documents is strongly encouraged as an aid in determining source of defects. The measurements and metrics for defects shall be reported as described in Table 8.

5.3.2 Application Complexity Metrics

Application complexity metrics (ACMs) endeavor to capture the inherent complexity of a given benchmark application, independent of the particular hardware and software implementation. They form the

Table 8: Defect Measurements and Metric

Element	Description
DEFECT_FIND	TIme of defect identification
DEFECT_SRC	Source of defect
DEFECT_UNDO_T, DEFECT_UNDO_C	Time (and cost) consumed in fixing defect after existence was recognized
DEFECT_LOST_T, DEFECT_LOST_C	Estimated time (and cost) lost because defect existed

basis for comparing the difficulty of successive benchmarks. The ACMs will also serve as a reference for determining efficiency of the hardware and software realizations produced by the Developer.

ACMs consist of three components: application requirements, external constraints, and "ility" requirements. The following ACMs shall be computed by the Developer with reference to the processor requirements of Section 2.

5.3.2.1 Requirements Complexity

The complexity of any embedded signal processor is determined by the inherent complexity of the application being implemented. The complexity of the application is determined by its function, computational requirements, control flow, external interfaces, and dynamic range or precision.

A signal processing application can generally be described as a flow graph where data is mapped from one form to another, or transformed, by a succession of system operations. The number of different system operations shall be reported as UNISYSOP with reference to a figure or table which identifies them in the signal flow of the application. As a guideline, it is suggested that the count be application specific, that is, two vector multiplies of different length should be counted separately as would two FFTs of equal length in different parts of the flow.

For a defined output datum, such as an output vector or a frame of image data, the number of times each system operation is executed shall be reported in the figure or table referred to in the preceding paragraph. The total of these shall be reported as TOTSYSOP.

The maximum number of system operations per second required by the application shall be reported as SYSOPS.

Each UNISYSOP shall be defined in terms of the number of primitive operations it encompasses where a primitive operation is an add, subtract, shift, compare, etc. with no distinction between integer and real variables or single and double precision. (If there are a significant number of complex primitive operations, such as divide, they shall be counted as an equivalent number of simple primitives.) The required rate of primitive operations per second shall be reported as PROPS.

Control flow (CONFLOW) complexity is a measure of the number of user commanded modes of operation and degree of data dependent branching. It is rated Low, Medium or High.

The maximum number of system data, including constants, coefficients, etc., required to be resident within the application process at any time shall be reported as SYSRES in bytes. In addition, DATARES is the maximum amount of data required to be resident in the process as measured in bytes. No distinctions are made as to where the data may reside in a possible storage hierarchy.

The total number of external interfaces shall be recorded (TOTEXTINT), together with the number and type of unique (UNIEXTINT) and non-standard interfaces (NSTDEXTINT). For each data port, input and output, average and peak data rates shall be recorded (DAVG, DPK) as well as the dynamic range (DYNAMIC) and precision (PRECIS).

The required minimum latency (LATENT) between input data and resultant output data shall be reported.

5.3.2.2 External Constraints

External constraints affect the complexity of embedded signal processors. Such constraints include the physical constraints imposed by the system into which the processor is imbedded, as well as environmental and cost constraints and imposed mil-standards.

The physical constraints of the processor shall be recorded. This shall include maximum allowable size (MAXSIZE) and weight (MAXWGT), maximum allowable values of peak and average power (MAXPKPOW, MAXAVGPOW), and the source of prime power (PRMPOW); e.g., 110VAC, 28VDC, etc.

The environmental constraints of the processor shall be recorded. This shall include the allowable ranges for temperature, humidity, altitude, corrosion resistance, and shock and vibration (TEMP, HUMID, ALT, CORRES, SHOCK). Allowable values of all constraints for both operational use and storage shall be recorded.

All cost constraints shall be recorded. This includes total cost (TOTCOST) as well as non-recurring engineering costs (NRECOST).

All required mil-standards shall be recorded as well as any modifications, tailoring, or exemptions to required standards.

5.3.2.3 Ility Requirements

Requirements for testability, reliability, and maintainability increase the complexity of the embedded signal processor. The required degree of fault coverage shall be recorded (FLTCOV) together with the maximum allowable latency in detecting faults (FLTLAT) and the required level of fault isolation (FLTISO). The maximum allowable fault rate (MAXFLTRT) shall be recorded together with the maximum allowable time for fault recovery (MAXFLTREC).

5.3.3 Hardware Products

The primary concern of the hardware metrics is the RASSP product. Hardware performance metrics measure the performance of the processor and they mirror the application process requirements metrics of Section 5.3.2.1. Hardware complexity metrics measure characteristics of the hardware implementation.

5.3.3.1 Hardware Performance Metrics.

The Developers shall provide measurements for the following metrics.

Execution Rate. The execution rate realized in primitive operations per second (as defined in Section 5.3.2.1) while executing the Benchmark application shall be reported as PROPS. If the processor can operate in several different modes then PROPS shall be reported for each mode. If the entire processor can support a higher throughput than required by the application then the maximum possible rate shall be reported as MPROPS. The subsystem or interface which determines this maximum shall be described.

I/O and Dynamic Range. The peak and sustained data transfer rate (IOPEAK, IOSUS) at each system interface shall be reported.

Power. Peak power (PKPOW) and average power (AVGPOW) when operating at the rate for which PROPS is reported.

Size and Weight. The dimensions of the system box(es) and their weight (SIZE, WEIGHT).

Cost. Both real costs of the prototype and projected manufacturing costs are desired (PCOST, MCOST). Prototype costs shall include the total small-quantity cost of all components in the system and NRE incurred. The estimate of production cost for producing N units over a period of Y years shall include component, NRE, manufacturing, testing and documentation costs. Unit Life Cycle cost for an assumed total number of N units over a period of LC years shall be reported (N, Y, LCCOST).

Testability. The level of conformance to testability specifications as well as any additional capability added by the developer shall be described. The data may represent both estimates and results of experiments and shall include: time to execute routine diagnostics, test coverage, level of fault isolation and mean time to detect faults (TEST).

Reliability/Availability. The level of adherence to reliability/availability specifications shall be described. Data to be presented includes predicted mean time to failure and time to recover from or repair a fault (RELY, AVAIL).

Environment. For both operational and storage environments the design goals and measurement results for temperature, altitude, humidity, and shock and vibration resistance shall be reported (ENVIRONMENT).

5.3.3.2 Hardware Complexity Metrics

Hardware Complexity Metrics (HCM) capture the complexity of the benchmark application hardware through measures of degree of integration, COTS vs. custom, number of elements, clock rate, etc. They also give a measure of the level of technology employed.

- Size Storage: For each of the storage levels: cache (off processor-chip), main and second level, report the total number of bytes of storage(STORAGE).
- Technology Speed: Report the clock rate of the system and identify any asynchronous circuits and interfaces. Identify any circuits which use higher-speed internal clocks (TECHNOLOGY).

- Buses: Identify all internal system buses, their size, protocol and peak and average data transfer rate in this application (BUSES).
- Implementation style: List each unique circuit and the number used in the following classes of circuits: COTS, FPGA, gate array, standard cell and full custom (ICLIST).
- Packaging Levels: Identify and describe the levels of packaging. For example: wirewrap backplane, PCB pluggable module with surface mount devices, thin film MCM with ball grid array chips, ICs with various packages (PKGLIST).
- Heat: For the highest dissipation IC give the expected maximum junction temperature under the most severe operating condition specified in the benchmark (HEAT).
- Interfaces: Identify and describe system interfaces (INTERFACE).

5.3.4 Software Products

Software product metrics record the size and measures of quality of both application and VHDL code.

5.3.4.1 Lines of Code.

The lines of code for each application and VHDL source file shall be reported. This measurement shall include at least executable lines (LOC_EXEC) and total lines (TOC_TOTAL). COCOMO models use *non-comment source statements* (LOC_COCOMO) which shall also be measured or calculated from other measurements. At the beginning of the benchmark there must be agreement between Lincoln Laboratory and the Developer on the LOC counting programs.

All software code generated by automatic methods shall be identified and the same measurements made.

Lines of code shall be measured at several well defined times in the development process. The Developer is encouraged to establish source directories, preferably by a configuration management approach, and automatically create lines_of_code counts at established times, daily or weekly.

5.3.4.2 Application Code Complexity Metrics

The Developers shall provide the graphs used as input to the autocoding process. The Benchmarker will asses the algorithm complexity and compare it with the complexity of the autocoded source code.

5.3.4.3 Application Code Quality Metrics

The Developers shall provide detailed source files for the application code (manual- and auto-coded) such that the Benchmarker can calculate software complexity metrics. These metrics include, at a minimum, the McCabe metrics.

5.3.4.4 VHDL Code Quality Metrics

The Developer shall provide source files for the VHDL files used in defining the Virtual Prototype as well as in programming the FPGAs.

6. DELIVERABLES

This section provides information on the deliverables required for Benchmark-4.

6.1 PROCESSOR

6.1.1 Virtual Prototype Designs

The RASSP Developer shall investigate at least two prototype processor designs. Both designs will be developed to the point where realistic estimates of performance can be made. Use of VHDL performance modeling to substantiate performance estimates is desired.

One of the architectural concepts investigated at the performance model level shall be selected by the Developer for evolution to a virtual prototype as described in Section 3. Insofar as possible, subject to the calendar and labor hour limits established for this Benchmark, the virtual prototype shall emulate the critical behavior and timing of the selected design.

6.1.2 Accuracy Requirements

The RASSP developer must demonstrate a prototype processor design that meets the accuracy requirements described in Section 2.7. During development, however, other processor designs may be found to satisfy the specified performance requirements for the processor, but may not meet the specified accuracy requirements for the processor. In these cases, the RASSP Developer is encouraged to propose an alternative accuracy requirement.

The RASSP Developer must fully describe each alternative accuracy requirement and demonstrate that the alternative requirement adequately characterizes the performance of the processor.

6.1.3 Product Acceptance

Acceptance testing of RASSP processor prototype designs shall be performed at the Developer's site with the BM-4 subsystem operating with a SAIP emulator supplied by MIT Lincoln Laboratory. The BM-4 subsytem shall connect to the SAIP emulator with one or more ATM connections, a console line, and an ethernet connection. The prototype shall satisfy the performance and accuracy requirements of Section 2.6 {p. 21} and Section 2.7 {p. 21}. The Developer shall demonstrate whatever fault detection and isolation features have been implemented.

The Government and the Benchmarker shall designate witnesses for the acceptance testing, and the Government shall decide whether to accept delivery of benchmark prototype test reports based on the outcome of the acceptance testing.

In addition to the final acceptance test, an opportunity to audit the design will be provided upon completion of the virtual prototype. This audit may include a detailed examination and demonstration of the VHDL design data and VHDL testbench.

6.2 PROCESS REPORTING

Benchmarking requires a level of reporting beyond that which is normal in a development project. This section describes the timing and expected content of benchmark reporting. Table 9 shows the expected initial delivery and updates for the data.

Table 9: Schedule for Process Reporting (X) With Updates (U)

Data Item	Benchmark Time				
	Proposal	Kickoff	Monthly	Milestone	Final
Schedule	X	U		U	U
Costs			X		X
Process Plan	X	U	U		
Process Followed					X
Standards	X	U		U	U
Tools Available	X	U		U	
Tools Used					X
Personnel	X	U	U		U
Logs			U		
Metrics			U	U	U

6.2.1 Reporting Schedule

6.2.1.1 Proposal

Before initiation of the benchmark a work description, schedule, process plan, list of standards, tool list and list of personnel who will work on the project shall be provided.

6.2.1.2 Kickoff

Not later than one month after start of work there shall be a Kickoff Review.

6.2.1.3 Monthly

There shall be a monthly written progress report. This requirement may be satisfied by inclusion in the RASSP Monthly Report. There shall be a monthly report of charges made to the Benchmark Project with breakdown to the lowest level used internally by the Developer There shall be a monthly delivery of logs and any updates to Lines_of_code counts. Delivery of these items shall be by electronic means.

6.2.1.4 Milestones

The Developer shall define several Milestones keyed to significant transition points in the Benchmark tasks but with at least one Milestone each three months. There shall be a review at each milestone with presentations with viewgraphs, delivery of completed or in-progress documentation as requested by Lincoln Laboratory and delivery of source files of software and hardware tasks as requested by Lincoln Laboratory. No formal written report will be required for the Benchmark Milestone but source files of the viewgraphs used in presentations shall be delivered.

6.2.1.5 Final

At the completion of the Benchmark there shall be a Review with delivery of final versions of all documentation.

6.2.1.6 Other Reviews

Lincoln Laboratory should be made aware of other Review meetings which may be requested by sponsors and internal meetings and be given the opportunity to attend. Lincoln Laboratory staff should be able to interview individuals working on the benchmark to aid in the evaluation of process and tools. In order to support geographically distributed benchmarking, the Developers may be required to support teleconference and video conference meetings.

6.2.2 Process Reporting Data Items

6.2.2.1 Schedule

A schedule with a Work Breakdown three or four levels deep in the form of a Gantt chart with personnel hours and costs shall be maintained.

6.2.2.2 Costs

Project costs according to the Schedule WBS shall be reported monthly and at the end of the project.

6.2.2.3 Process Plan

At the beginning of the Benchmark the Developer shall describe the process to be followed. This plan shall clearly indicate those elements which are new to the organization and personnel. The process plan description shall make reference to the Developer's proposed RASSP methodology. The principal RASSP capabilities intended for demonstration on the benchmark shall be enumerated and described in sufficient detail to allow appropriate performance metrics to be identified. Transition points at which source files and other documentation are put under configuration control shall be shown as well as the review procedures followed at those points. If a workflow manager is to be used it shall be described. Any significant changes to the Plan shall be described in monthly reports and milestone reviews.

6.2.2.4 Process Followed

At the conclusion of the Benchmark, the process which was followed shall be described in the same detail as specified in Section 6.2.2.3. All process steps and workflow paths shall be defined, and their overall effect shall be described in sufficient detail to determine relative merits, if any.

6.2.2.5 Standards

Any Standards and Style Manuals which are mandated for use in the project shall be specified. Copies of internal standards and style manuals shall be delivered.

6.2.2.6 Tools Available

At the outset of the Benchmark the Developer shall provide a list of all of the electronic design automation (EDA) tools available in the RASSP system and an indication of which tools are likely to be used. The description shall include, as a minimum, the following information:

- The association between the tools and the RASSP process steps
- The integration status of each tool including:
  - Revision number of the tool
  - Interfaces to other tools
  - Level of integration with RASSP Design Environment
- Minimum host machine resources required to effectively use each tool including:
  - Minimum host memory configuration for executable
  - Disk resources required
  - Representation of the minimum acceptable CPU performance (e.g. Specmarks)
- Platforms on which each tool is supported
- Purchase and maintenance costs for each tool
- The minimum skill category or area of specialization required to effectively use each tool.

The addition of tools during the Benchmark shall be noted in the Milestone reviews.

6.2.2.1 Tools Used

At the conclusion of the Benchmark the tools which were used for each WBS element shall be described and the data of Section 6.2.2.6 updated as necessary.

6.2.2.2 Personnel

A list of all the individuals projected to work on the benchmark an average of one or more days a week must be provided at the initiation of a benchmark. The list should indicate the title and job category of each of the individuals, along with a description of their familiarity with both the benchmark application and the RASSP tools and processes. Personnel changes made during the course of the benchmarking by the Developer shall be reported to the Benchmarker in the monthly reports.

6.2.2.3 Logs

We request that each participant in the benchmark make a daily personal log entry, by computer means, which includes, at least, task, tools used, tool problems, significant interactions, accomplishments and errors found. A common format shall be used by all participants and delivery of the logs shall be done weekly by electronic means.

6.2.2.4 Metrics

The Developer shall update the Metrics information described in Section 6.3 as data is available and report the complete information at the Final Review.

6.3 METRICS

The metrics must be applied in a framework which considers the mode of project development as well as phase of the project (see Section 5). Each developer must, therefore, supply development mode and project phase data with each delivered metric.

6.3.1 Metric Deliverable Lists (MDLs)

6.3.1.1 Design Process MDL

Table 10: Design Process Metric Deliverable List

DESCRIPTION	SECTION	METRICS
Design process	Section 5.3.1	TOOL_USAGE
Reuse	Section 5.3.1.2	REUSE_ORIG_T REUSE_ORIG_C REUSE_EVAL_T REUSE_EVAL_C REUSE_T REUSE_C REUSE_TRATIO REUSE_CRATIO
Defects	Section 5.3.1.3	DEFECT_FIND DEFECT_SRC DEFECT_UNDO_T DEFECT_UNDO_C DEFECT_LOST_T DEFECT_LOST_C

6.3.1.2 Application Complexity MDL.

Table 11: Application Complexity Metric Deliverable List

DESCRIPTION	SECTION	METRICS
Application requirements	Section 5.3.2.1	UNISYSOP TOTSYSOP SYSOPS PROPS CONFLOW SYSRES DATARES TOTEXTINT UNIEXTINT NSTDEXTINT DAVG DPK DYNAMIC PRECIS LATENT
External constraints	Section 5.3.2.2	MAXSIZE MAXWGT MAXPKPOW MAXAVGPOW PRMPOW TEMP HUMID ALT CORRES SHOCK TOTCOST NRECOST MIL_STD
Ility requirements	Section 5.3.2.3	FLTCOV FLTLAT FLTISO MAXFLTRT MAXFLTREC

6.3.1.3 Hardware Product MDL.

Table 12: Hardware Product Metric Deliverable List

DESCRIPTION	SECTION	METRICS
Performance	Section 5.3.3.1	EXRATE PROPS MPROPS IOPEAK IOSUS PKPOW AVGPOW SIZE WEIGHT PCOAT MCOST N Y LCCOST TEST RELY AVAIL ENVIRONMENT
Complexity	Section 5.3.3.2	STORAGE TECHNOLOGY BUSES CKTLIST PKGLIST HEAT INTERFACE

6.3.1.4 Software Product MDL.

TABLE 13: Software Product Metric Deliverable List

DESCRIPTION	SECTION	METRICS
Lines of code	Section 5.3.4.1	LOC_EXEC LOC_TOTAL LOC_COCO
Software code metrics	Section 5.3.4.3	HAL_N_OPTOR HAL_N_OPAND HAL_N_OCC_R HAL_N_OCC_D HAL_VOCAB HAL_OB_LEN HAL_ES_LEN HAL_VOL HAL_DIFF MCCABE_CCN FENTON

6.3.2 Metrics Delivery Formats

The Parametric Cost Estimation data shall be delivered in computer readable form in a method agreed on between the Developer and Lincoln Laboratory. The application complexity data and the overall hardware complexity data may be delivered in tabular form. Source code text files are deliverable as UNIX 'tar' files on 8 mm magnetic tape. Logs must be supplied as machine-readable text files.

Other hardware and software metric data is associated with each hardware or software module. A format which ties the metric data to other cross reference data (drawing numbers, file name, etc.) and historical and defect data is recommended. Such a format might be a separate page for each HW or SW element.

6.4 DELIVERY OF PRODUCT-IN-PROGRESS MATERIALS

During the course of each benchmark, software baselines shall be created by the Developer as a deliverable item and are due at the milestones. For the purposes of this benchmark, software specifically includes VHDL code. A baseline is not intended to be comprehensive or a final version but is intended to represent a working package for some subset of the overall task. As a result of experience gained from Benchmark-1, forms for tracking software development through all development phases will be used. The form may be the one described in Section 6.3.2 or one similar to Table 14. It is recommended that the Developer conform to the tracking and reporting specification of [14] for each type of software developed (i.e., Ada, C, and VHDL). The contents of code inspections and their results form part of the deliverables. This includes, of course, structure charts and flow diagrams.

Hardware schematics are deliverable items.

TABLE 14: Software Product Tracking Form

	Requirements Analysis	System Design	Detailed Design	Code Production	Unit Test	Integration Test	System Test
Original LOC Est.							
Revised LOC Est.							
LOC Produced							
LOC Released							
Est. Effort†							
Revised Effort†							
Actual Effort†							
Est. # Defects							
Act. # Defects							

† In units of person-month

6.5 REPORT FORMATS

Unless otherwise agreed upon in writing, the Developer shall supply all non-hardware deliverables and reports in the following electronic formats. Where multiple formats are noted, the Developer can select the format most appropriate to the data item. Style and format files must also be supplied whenever either is required to view or print a data item.

- Schedules - Microsoft Project or a compatible format
- Reports/Documentation - Microsoft Word or Framemaker
- Spreadsheet - Microsoft Excel or a compatible format
- Personal logs - Text files
- Project Data - Both native tool format and project-wide database format
- Application Source Code - ASCII text files
- HOL/HDL Source code - ASCII machine-readable format

The digital data may be provided via an Exabyte model 8200 or 8500 uncompressed tar format 8mm tape, or via an FTP site accessible over the Internet. Wherever requested in the BTD for a given benchmark, the deliverables shall also be supplied in printed form. Password protection may be used for security at the Developers' option.

7. DEVELOPER RESPONSE

This section provides additional detail regarding the response the Developer shall provide to BTD-4.

7.1 BENCHMARK EXECUTION CHECK LIST

For Benchmark-4, the Developer shall include in the response to the BTD a Benchmark Execution Check List (BECL). The BECL shall be based on the RASSP process steps which the Developer envisions applying to execute the benchmark. For each major process step, the Developer shall provide the following information:

1. Cost
2. Schedule
3. Tools utilized
4. Caliber of individual(s) required to execute the process

The BECL can also be organized according to the deliverables (products) required in BTD-4, but in this case, the process steps and cost associated with the development of each deliverable must be indicated where appropriate. For example, since application hardware and software are deliverables, the process steps and tools used to produce the hardware and software must be indicated. In the case of metric deliverables, the costs should be broken out on the basis of the metric categories defined in Section 5.2 through Section 5.3.

The BTD includes points of contact at the Benchmarker's organization for the purpose of addressing technical questions regarding the BTD, however, all questions submitted to the Benchmarker shall also be submitted simultaneously to the cognizant Government COTR or his designee.

The Developer shall respond with a comprehensive estimate of the cost to execute the first phase of BTD-4. The Developer shall include a WBS and associated schedule for the tasks in the WBS, along with a list of all the individuals assigned to work on the Benchmark more than an average of one day a week. The level of detail shown in the WBS and schedule shall be sufficient to identify and briefly describe the distinct steps in the Developer's RASSP design process, and shall conform to specific formats and reporting details called for in this BTD. For each entry in the BECL, the total estimated cost of executing that part of the RASSP process shall be required. An indication shall be provided of the cost and schedule impact on the remaining process steps of deleting a particular process step. The categories of impact are:

- None
- Modest
- Significant
- Essential

At the conclusion of the first phase the Developer shall provide cost estimates including life cycle costs for all hardware and software proposed to be implemented in the second phase. All cost estimates must provide appropriate background information for review. An identifiable database of information which is consistent, accurate, traceable and relevant must be used as a basis for the cost estimate. Appropriate and supportable adjustments can be made to the database as required. Factors such as inflation, production rate, quantity and required changes must be considered. Historical data may consist of hours for a similar completed task or task in progress, previous material or subcontract costs, departmental statistics and learning curve experience. The database need not be generally accessible to personnel who are not developer employees, but all estimates must be auditable (even if such an audit must be performed by DPRO). A suitably calibrated parametric cost estimate is considered a valid substitute for a detailed bottom-up approach. The Developer must use whatever approach is deemed appropriate for the future RASSP design environment. In the event the benchmark execution is distributed over more than one organization, the division of responsibility should be clearly indicated as part of the BECL.

The estimates at the end of Phase I shall include identification of procurement risks and fall-back plans in the event items in the proposed implementation can not be procured.

7.2 TOOL STATUS INFORMATION

At the outset of Benchmark-4, along with the schedule and cost estimate, the Developer shall provide a list of all of the electronic design automation (EDA) tools available in the RASSP system, an indication of which tools are likely to be used, and a description of the RASSP design process supported by the tools. The tool and process description shall include, as a minimum, the following information, and shall be provided in written form and in a common electronic format:

- The association between the tools and the RASSP process steps
- The integration status of each tool including:
  - Revision number of the tool
  - Interfaces to other tools
  - Level of integration
- Minimum host machine resources required to effectively use each tool including:
  - Minimum host memory configuration for executable
  - Disk resources required
  - Representation of the minimum acceptable CPU performance (e.g. Specmarks)
- Platforms on which each tool is supported
- Purchase and maintenance costs for each tool
- The minimum skill category or area of specialization required to effectively use each tool. Example tool and skill categories are given below:

TOOL	SKILL CATEGORY
Word Processor	Secretary/Technical Writer
Architecture Trade-off	System Analyst
Ada Compiler	Programmer
Thermal Design	Mechanical Engineer

VHDL Simulator	Digital Designer
Schematic Entry	Technician

In order to visualize the degree of tool integration within the RASSP design environment, the equivalent of a 2-D matrix (N^2 chart) of the available tools will be created and the level of integration which exists between all pairwise combinations of tools will be entered as a number at the row and column intersection of the tool pair. The level of tool integration shall be supplied by the Developers and verified by the Benchmarker.

REFERENCES

1. "ISO Battlefield Awareness," <http://yorktown.dc.isx.com/iso/battle/saip.html>.

2. Benitz, Gerald, "Adaptive High-Definition Imaging," SPIE Vol. 2230, pp. 106-119, 1994.

3. Capon, J., "High-Resolution Frequency-Wavenumber Spectrum Analysis," Proc. IEEE, no. 57, 1969, p.1408.

4. Benitz, G.R., "Preliminary Results in Adaptive High Definition Imaging for Stationary Targets," Project Report AST-34, 4 Nov. 1993, MIT Lincoln Laboratory, Lexington, MA.

5. Novak, L.M., et.al., "ATR Performance Using Enhanced Resolution SAR," SPIE Vol. 2757, pp. 332-337, 1996.

6. DeGraaf, S.R., "SAR Imaging via Modern 2-D Spectral Estimation Methods," to be published in IEEE Transactions on Image Processing.

7. Wong, R., "Cost modeling," Ch. 20 in *Space Mission Analysis and Design*, 2nd Edition, Ed. by W.J. Larson and J.R. Wertz, Kluwer Academic Publ., Norwood, MA, 1992.

8. Boehm, B.W., *Software Engineering Economics*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1981.

9. Davis, J.M., "Planning and estimating software," *CASE Outlook 89*, No. 2, pp. 23-34, 1989.

10. Fenton, N.E., *Software Metrics, A Rigorous Approach*, Chapman and Hall 1991.

11. Martin Marietta PRICE Systems, *PRICE Reference Manuals*, Moorestown, NJ, 1981.

12. Galorath Associates, Inc., SEER Technologies Division, *SEER User Manuals*, Los Angeles, CA, 1993.

13. Boehm, B.W., "A spiral model of software development and enhancement," *IEEE Computer*, pp. 61-72, May 1988.

14. Fulton, R., *Space and Missile Systems Center Software Database Collection Form and Dictionary*, U.S. Air Force SMC/FMS, El Segundo, CA, September 1993.

15. Anderson, A.H, Anderson, J.C, and Shaw, G.A., "RASSP Benchmark-4 Executable Requirement User Manual," MIT Lincoln Laboratory, Lexington MA, February 1997. Proprietary Information.

16. "SAIP Baseline System Data Structure Class Library Interface Control Document," MIT Lincoln Laboratory, Lexington MA, April 1997.

17. "SAIP Baseline System Inter-Process Communications Software," MIT Lincoln Laboratory, Lexington MA, April 1997.

GLOSSARY

Benchmark Cycle	A nominal six-month long period during which the Developer applies the current RASSP process to develop an application and meet the requirements defined in a Benchmark Technical Description.
Benchmark Technical Description	The BTD is a document and supporting technical information which defines each benchmark including system requirements, deliverables, and allowable duration.
COCOMO	A well-documented parametric cost estimation tool for software efforts. Many computer programs which implement different versions of the COCOMO (constructive cost model) equations are available.
PRICE	A suite of parametric cost estimation computer programs from Martin Marietta. The product line presently covers software and software life cycle (PRICE S), microcircuits and electronic assemblies (PRICE M), hardware systems (PRICE H) and hardware life cycle (PRICE HL).
Scalability	As applied to hardware and software architectures is the property of being expandable to address new requirements without substantially changing the design or the existing components.
SEER	A suite of parametric cost estimation computer programs from Galorath Associates. The product line presently consists of a software sizing model (SEER-SSM), software estimation model (SEER-SEM), integrated circuit model (SEER-IC), hardware estimation model (SEER-H) and hardware life cycle model (SEER-HLC).
Virtual prototyping	The process of simulating all applicable levels of hardware functionality (whether behavioral or register-transfer level) in a hardware description language such as VHDL.

ACRONYMS

ACTD	Advanced Concept Technology Demonstration
ADTS	Advanced Detection Technology Sensor
API	Application Programming Interface
ARCM	Application Requirement Complexity Metric
ASARS	Advanced Synthetic Aperture Radar System
ASIC	Application Specific Integrated Circuit
ATL	Lockheed Martin Advanced Technology Laboratories
ATR	Automatic Target Recognition
BECL	Benchmark Execution Check List
BTD	Benchmark Technical Description
COCOMO	Constructive Cost Model
COTS	Commercial Off-The-Shelf
CSC	Computer Software Component
CSCI	Computer Software Configuration Item
CSU	Computer Software Unit
DARPA	Defense Advanced Research Projects Agency
DSP	Digital Signal Processor
EBS	Electronic Breakdown Structure
EOF	End of File
FFT	Fast Fourier Transform
FPGA	Field-Programmable Gate Array
GOPS	Giga-Operations per Second
GUI	Graphical User Interface
HCM	Hardware Complexity Metric
HDI	High Definition Imaging
HDL	Hardware Definition Language
HOL	Higher Order Language
IMINT	Image Intelligence
ITAR	International Traffic in Arms Regulation

LL	Lincoln Laboratory
LRU	Line Replacement Unit
LSB	Least Significant Bit
Mbps	Megabits per second
MB	Megabyte
MCM	Multi-Chip Module
MDL	Metric Deliverable List
MFLOPS	Millions of Floating Point Operations per Second
MIPS (1)	Millions of Instructions per Second
MIPS (2)	MIPS, Inc.; a subsidiary of Silicon Graphics Inc.
MLM	Maximum Likelihood Method
MOPS	Millions of Operations per Second
MSB	Most Significant Bit
MSE	Mean Square Error
MSTAR	Moving and Stationary Target Acquisition and Recognition
MUSIC	MUltiple SIgnal Classification
MW	Megaword
PAL	Programmable Array Logic
PCB	Printed Circuit Board
PLD	Programmable Logic Device
PME	Prime Mission Equipment
PRICE	Parametric Review of Information for Costing and Evaluation
PSE	Peculiar Support Equipment
RAID	Redundant Array of Independent Disks
RASSP	Rapid Prototyping Application-Specific Signal Processors
RCS	Radar Cross Section
REVIC	Revised Intermediate COCOMO
SAIP	Semi-Automated IMINT Processing
SAR	Synthetic Aperture Radar
SEER	System Evaluation and Estimation of Resources

SEI	Software Engineering Institute
SGI	Silicon Graphics Inc.
SNR	Signal-to-Noise Ratio
SVD	Singular Value Decomposition
UAV	Unmanned Air Vehicle
VME	Versa Module Europe
WBS	Work Breakdown Structure

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	9 January 1998	Project Report

4. TITLE AND SUBTITLE

RASSP Benchmark 4 Technical Description

5. FUNDING NUMBERS

C — F19628-95-C-0002
PE — 63739E
PR — 394

6. AUTHOR(S)

Gary A. Shaw, Allan H. Anderson, and James C. Anderson

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Lincoln Laboratory, MIT
244 Wood Street
Lexington, MA 02173-9108

8. PERFORMING ORGANIZATION REPORT NUMBER

PR-RASSP-5

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

DARPA/ESTO
3701 N. Fairfax Dr.
Arlington, VA 22203-1714

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

ESC-TR-96-118

11. SUPPLEMENTARY NOTES

None

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

This document describes the fourth in a series of application tasks designed to measure performance of a process for the rapid prototyping of embedded digital signal processors. The rapid prototyping process is being developed for the DARPA/Tri-Services rapid prototyping of application specific signal processors (RASSP) program. The benchmark task described here involves virtual and physical prototype development for portions of a real-time digital signal processing image intelligence (IMINT) system that performs semi-automated IMINT processing (SAIP). The task addresses hardware and software development issues relating to the high-definition imaging (HDI) and minimum mean square error (MSE) target classifier portions of SAIP. Application details and design constraints are provided in this document, as well as a description of product and process metrics collected to derive measures of product and process performance. The application task and associated performance metrics comprise what is termed a benchmark technical description (BTD).

14. SUBJECT TERMS	15. NUMBER OF PAGES
	80
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	Same as Report